**TALLER 1**

**OBJETIVO**

- Evaluar los conocimientos previos de arquitectura de un sistema.

**FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE**

Para realizar un correcto desarrollo de software, se requiere tener un análisis detallado y acorde a la infraestructura que se quiere implementar, es por este motivo que tener presente la arquitectura del sistema se hace indispensable antes de empezar el desarrollo de la aplicación deseada.

1. ¿Que es la Arquitectura tecnológica de un sistema?
2. ¿Qué criterios debo tener en cuenta a la hora de elegir una arquitectura ?
3. ¿Patrones de arquitectura utilizados en el diseño de sistemas de software?

**SOLUCIÓN DE EJERCICIO.**

Carlos Manuel Núñez López
Instructor

# Actividad de apropiación de conocimientos

Con base a la actividad desarrollada en el ambiente de aprendizaje, se plantea el siguiente cuestionario.

1. Mencione las principales arquitecturas de software en la actualidad.
2. Hexagonal
3. MVC
4. SOA
5. Microservicios

Para cada una de estas arquitecturas se requiere realizar una investigación mencionando ventajas y desventajas de cada una de ellas.

**TALLER 1**

**OBJETIVO**

- Evaluar los conocimientos previos.

**FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE**

Para realizar un correcto desarrollo de software, se requiere tener un análisis detallado y acorde a la infraestructura que se quiere implementar, es por este motivo que tener presente las fases del desarrollo y las responsabilidades del equipo de trabajo se hace indispensable.

1. Definición y objetivos de DevOps
2. Historia y evolución de DevOps
3. Principios fundamentales de DevOps
    a. ¿Qué es integración continua?
    b. ¿Qué es la automatización?

**SOLUCIÓN DE EJERCICIO.**

**TALLER 1**

**OBJETIVO**

- Evaluar los conocimientos previos.

**FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE**

Para realizar un correcto desarrollo de software, se requiere tener un análisis detallado y acorde a la infraestructura que se quiere implementar, es por este motivo que tener presente las fases del desarrollo y las responsabilidades del equipo de trabajo se hace indispensable.

1. ¿Qué es la administración de la configuración de un proyecto de software?
2. ¿Que es un código fuente de una aplicación?
3. ¿Qué plataformas tecnológicas existen para desarrollar un software?
4. ¿Que es una aplicación multiplataforma?
5. ¿Que es una herramienta de control de versiones? Mencionar las más populares.

**SOLUCIÓN DE EJERCICIO.**

GC-F-004v.08




# Consumo de API REST

@SENAComunica
www.sena.edu.co

Adalberto Cárcamo Alvarado

# Consumo de **API REST**

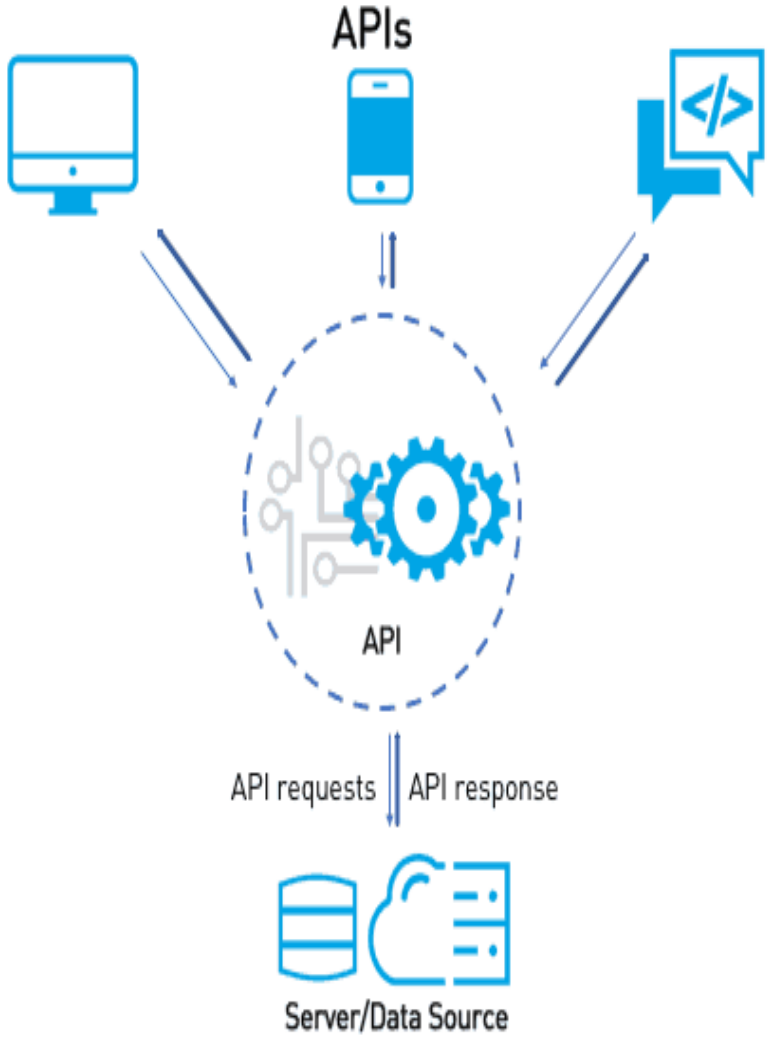
Consumo de API REST usando
JavaScript - Ajax

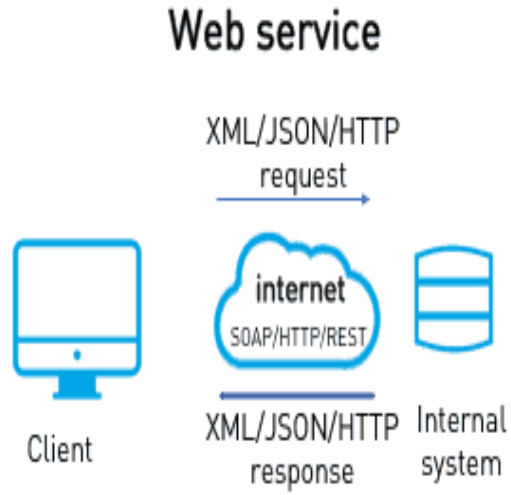
Adalberto Cárcamo Alvarado

# Uso de las APIS

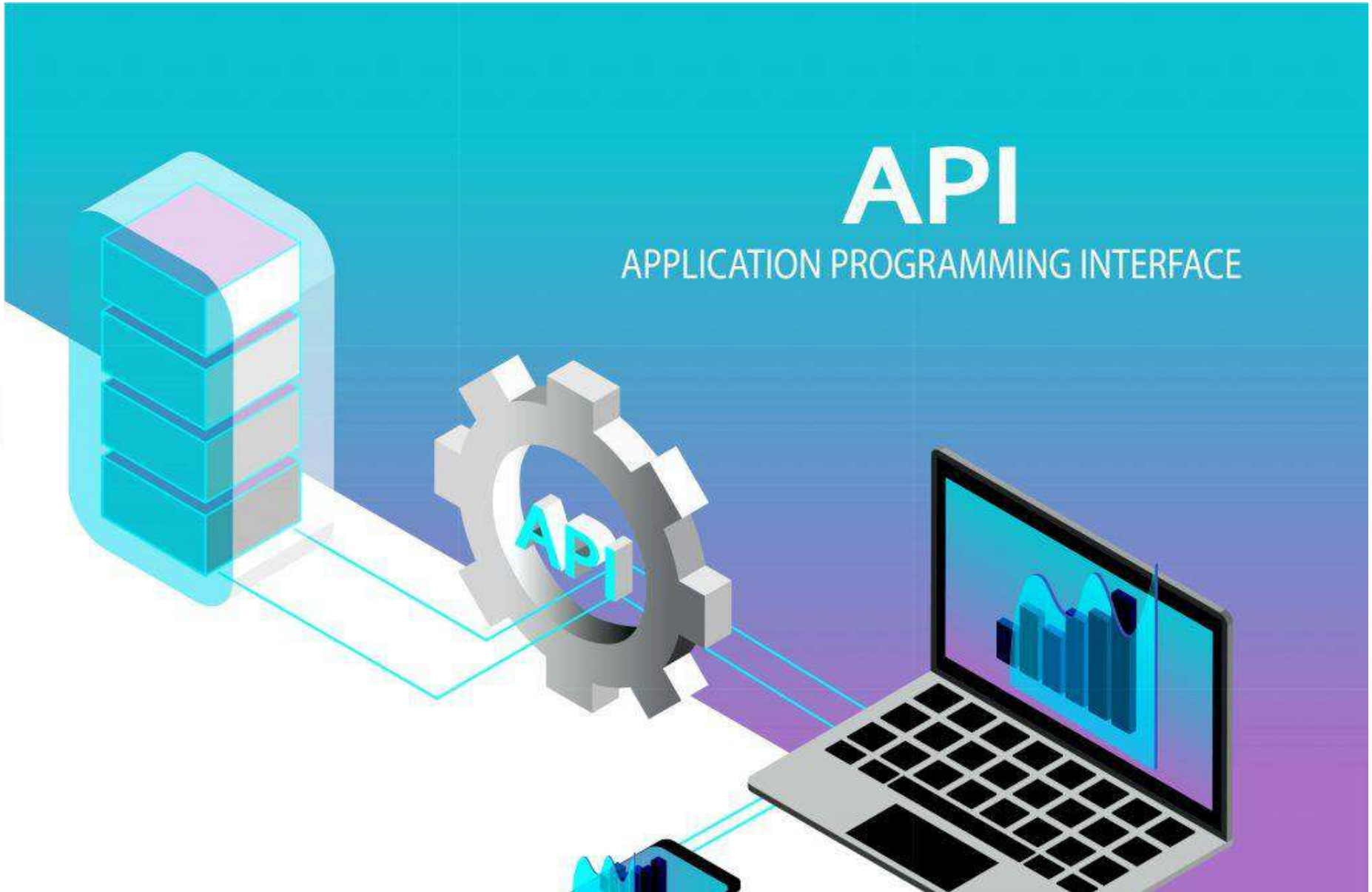
Las empresas consiguen grandes beneficios implementando APIs, ya que estas herramientas permiten ofrecer nuevos servicios y mejorar los existentes, además de que se resuelven problemas solamente con integrarlas en una plataforma, en lugar de tener que desarrollar desde cero una solución.

La principal ventaja de las APIs es que se pueden adaptar a las necesidades de tu negocio, proporcionando experiencias más completas para los consumidores y servir para la implementación de un modelo de servicio Omnicanal.



# Consumo de **API REST**




Adalberto Cárcamo Alvarado

# Consumo de API REST

Desarrollaremos una aplicación práctica para el consumo de la API usando Java Script y Ajax





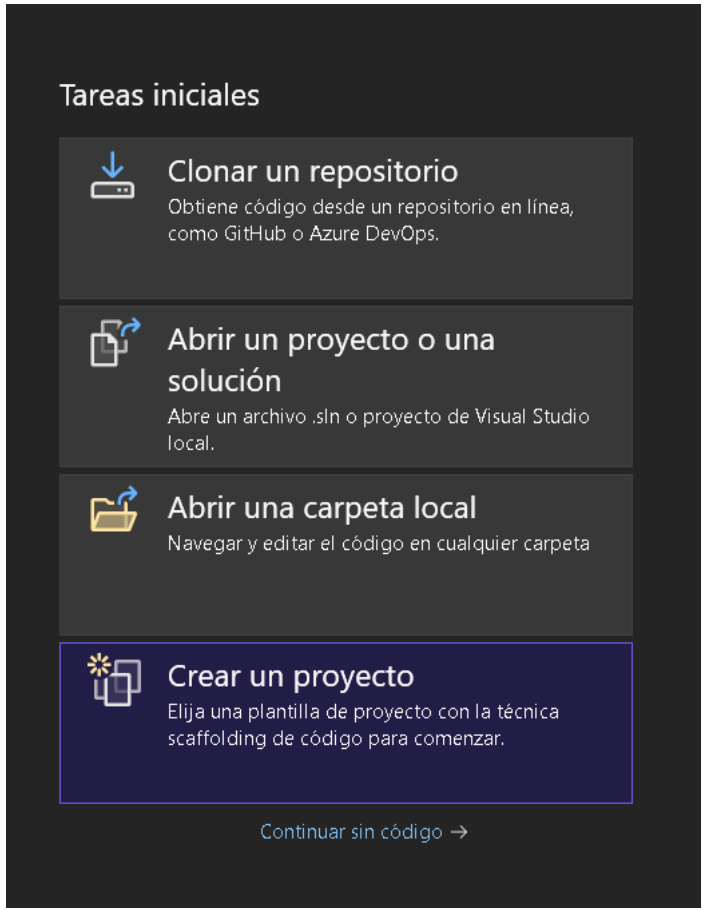


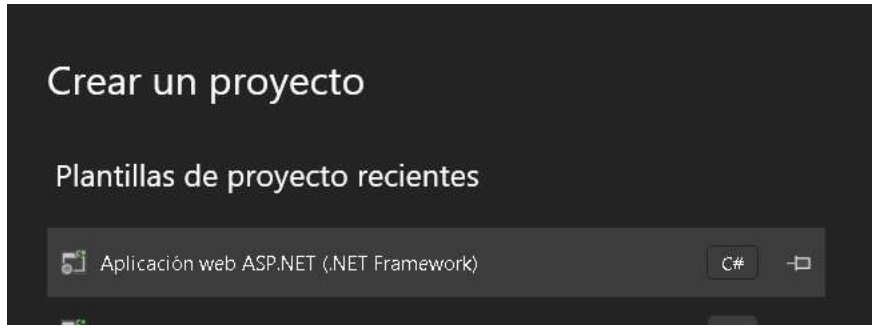
Ajax **permite que un usuario de la aplicación web interactúe con una página web sin la interrupción que implica volver a cargar la página web**. La interacción del sitio web ocurre rápidamente sólo con partes de la página de recarga y renovación.

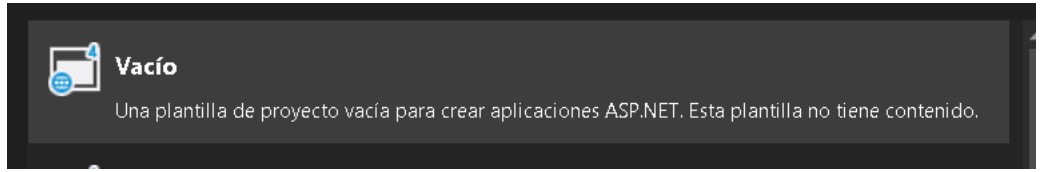
Adalberto Cárcamo Alvarado

# Consumo de API REST

1. Creamos una aplicación ASP desde la plataforma Visual Studio :

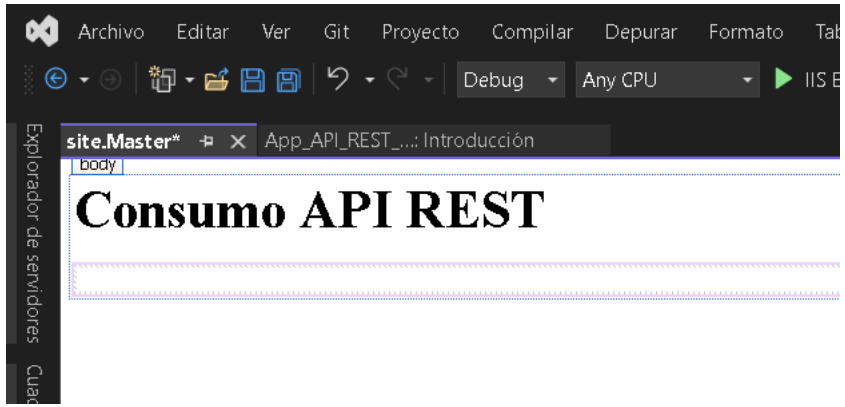


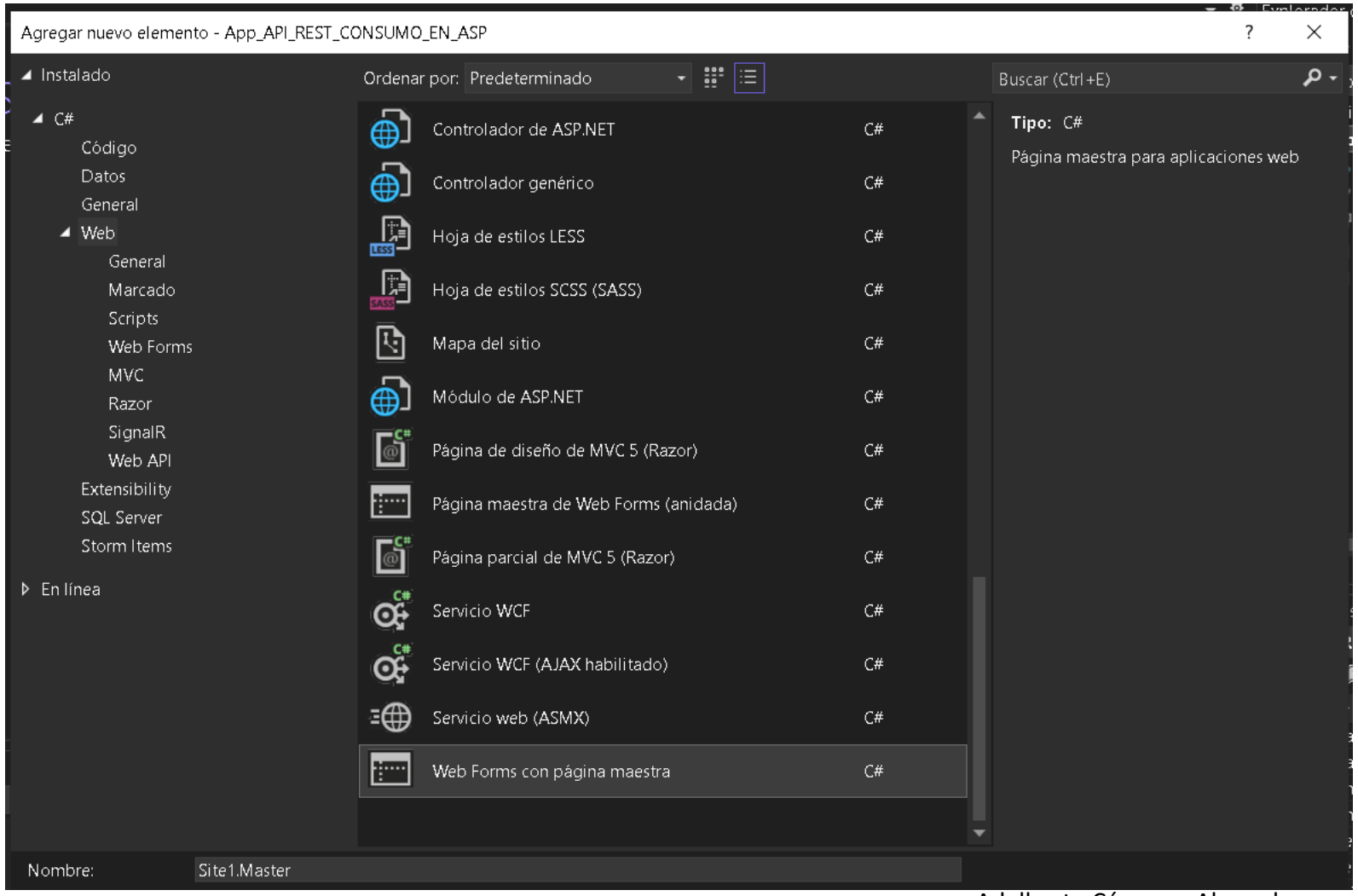



# Consumo de API REST

1. Creamos una página Site.master e ingresamos el titulo de la aplicación “Consumo API REST” :

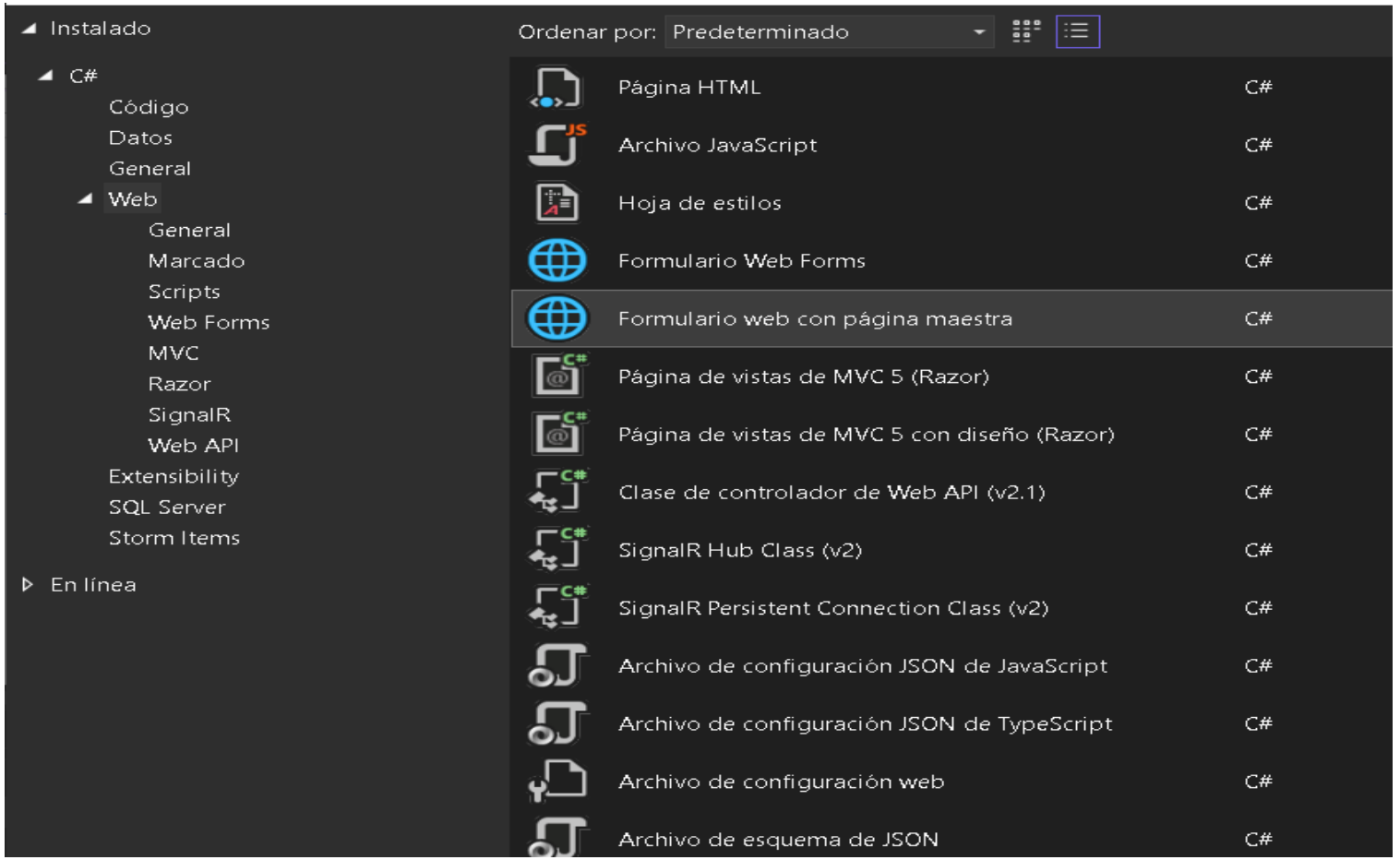


# Consumo de API REST

Agregamos un nuevo elemento en el proyecto y escogemos :
Formulario Web con Página Maestra

Le asignamos el nombre de :
Index.asp
Agregandolo en la página maestra
site.master

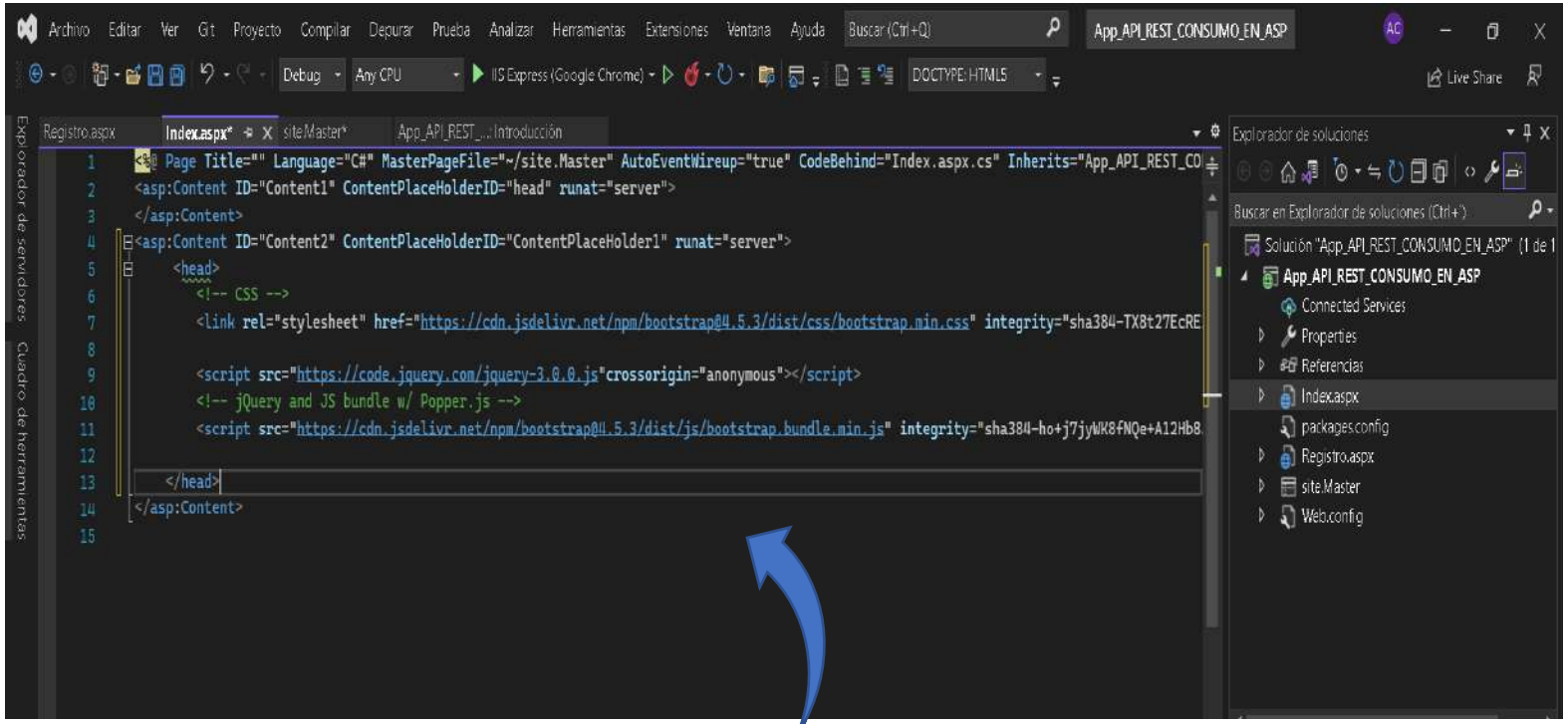


# Consumo de API REST

Agregamos un nuevo elemento en el proyecto y escogemos :
Formulario Web con Página Maestra

Le asignamos el nombre de :
Index.aspx
Agregándolo en la página maestra site.master.

Luego creamos otra página con el nombre de registro.aspx




En el archivo index.aspx agregamos el siguiente código que permite habilitar los diseños de bootstrap

Adalberto Cárcamo Alvarado

# Consumo de API REST

Código para asignación de  estilos de bootstrap :

```
<head>
        <!-- CSS -->
        <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-
TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2" crossorigin="anonymous">

        <script src="https://code.jquery.com/jquery-3.0.0.js"crossorigin="anonymous"></script>
        <!-- jQuery and JS bundle w/ Popper.js -->
        <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx"
crossorigin="anonymous"></script>

</head>
```

# Consumo de API REST

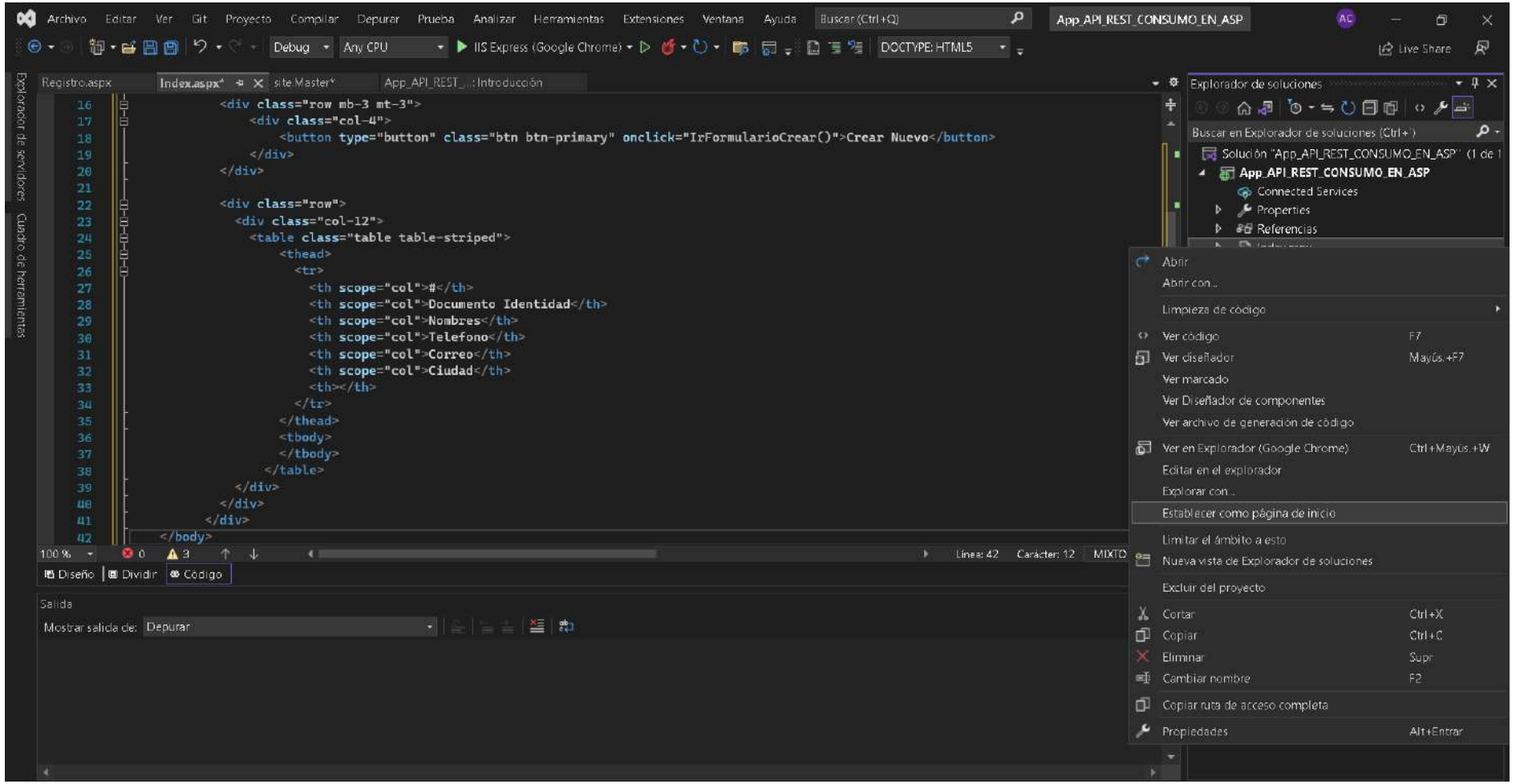
```
<body>
        <div class="container">
            <div class="row mb-3 mt-3">
                <div class="col-4">
                    <button type="button" class="btn btn-primary“ onclick="IrFormularioCrear()">Crear Nuevo</button>
                </div>
            </div>

            <div class="row">
              <div class="col-12">
                <table class="table table-striped">
                    <thead>
                      <tr>
                        <th scope="col">#</th>
                        <th scope="col">Documento Identidad</th>
                        <th scope="col">Nombres</th>
                        <th scope="col">Telefono</th>
                        <th scope="col">Correo</th>
                        <th scope="col">Ciudad</th>
                        <th></th>
                      </tr>
                    </thead>
                    <tbody>
                    </tbody>
                  </table>
              </div>
            </div>
          </div>
    </body>
```

En la etiqueta de body asignamos el siguiente código :

Se crea un botón “Crear nuevo”
Y una tabla que permita mostrar las columnas que mostraría del consumo de la API.

# Consumo de API REST

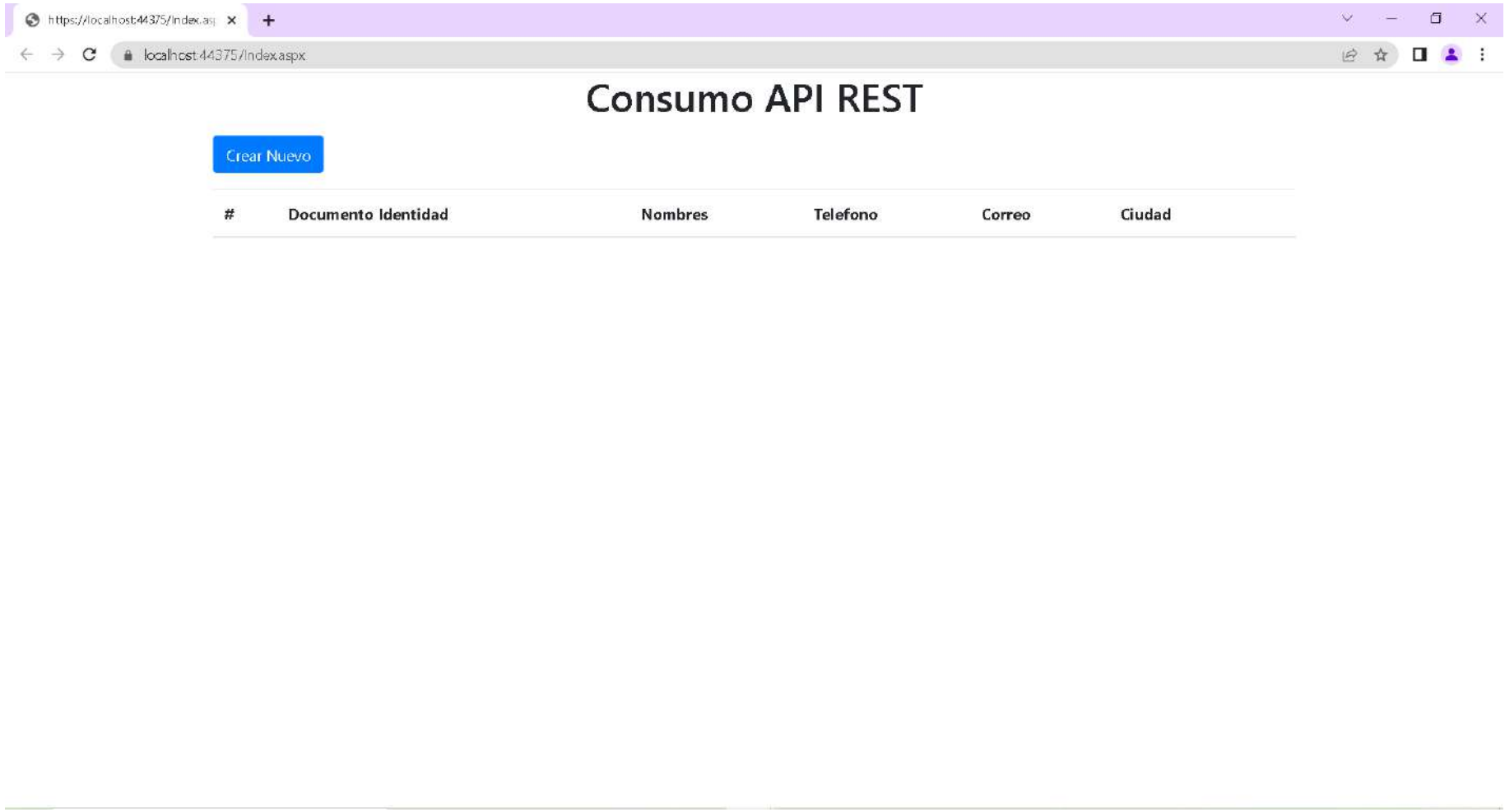



Establecemos como página de inicio :

Click derecho a la página Index.aspx y escogemos establecer como pagina de inicio.

Adalberto Cárcamo Alvarado

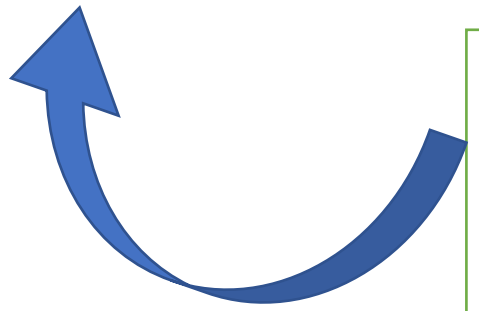
# Consumo de API REST




Ejecutamos la aplicación y nos muestra la siguiente página de inicio.

# Consumo de API REST

```
 <button type="button" class="btn btn-primary" onclick="IrFormularioCrear()">Crear
Nuevo</button>
```

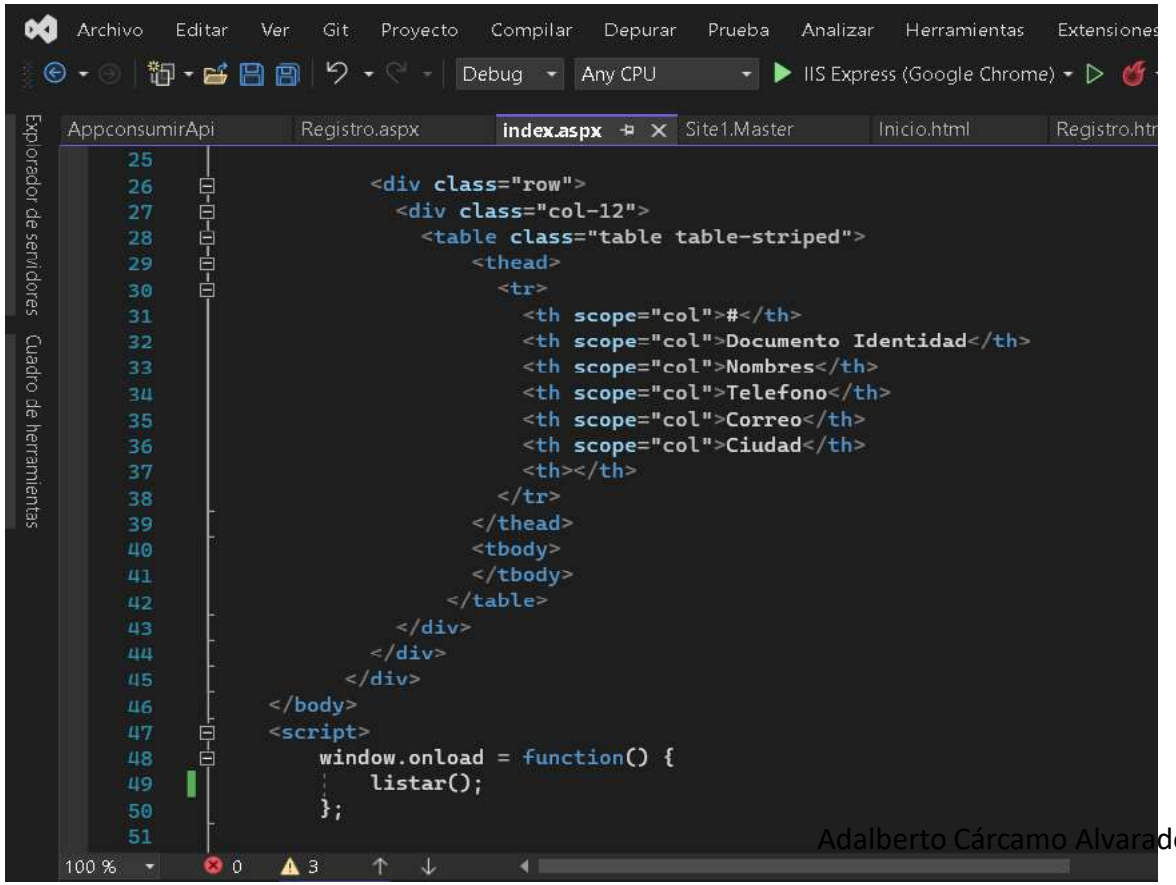


En el botón “Crear nuevo ” hacemos el llamado a una función irFormulario()

# Consumo de API REST

Debajo de la finalización de la etiqueta </body>  ingresamos el siguiente código de JavaScript  que permitira invocar el método listar() cada vez que inicie o cargue  la página

```
<script>
    window.onload = function () {
        listar();
    };
</script>
```



# Consumo de API REST

## Creamo s el método Listar()

```
function listar(){

        $(".table tbody").html("");

        $.get("http://localhost:58683/api/Usuario")
        .done(function( response ) {
            console.log(response);
            $.each( response, function( id, fila ) {
                $("<tr>").append(
                    $("<td>").text(fila.IdUsuario),
                    $("<td>").text(fila.DocumentoIdentidad),
                    $("<td>").text(fila.Nombres),
                    $("<td>").text(fila.Telefono),
                    $("<td>").text(fila.Correo),
                    $("<td>").text(fila.Ciudad),
                    $("<td>").append(
                        $("<button>").data("id",fila.IdUsuario).addClass("btn btn-success btn-sm mr-1 editar").text("Editar  ").attr({"type":"button"}),
                        $("<button>").data("id",fila.IdUsuario).addClass("btn btn-danger btn-sm eliminar").text("Eliminar").attr({"type":"button"})
                    )
                ).appendTo(".table");
            });
        });
    }
```

# Consumo de API REST

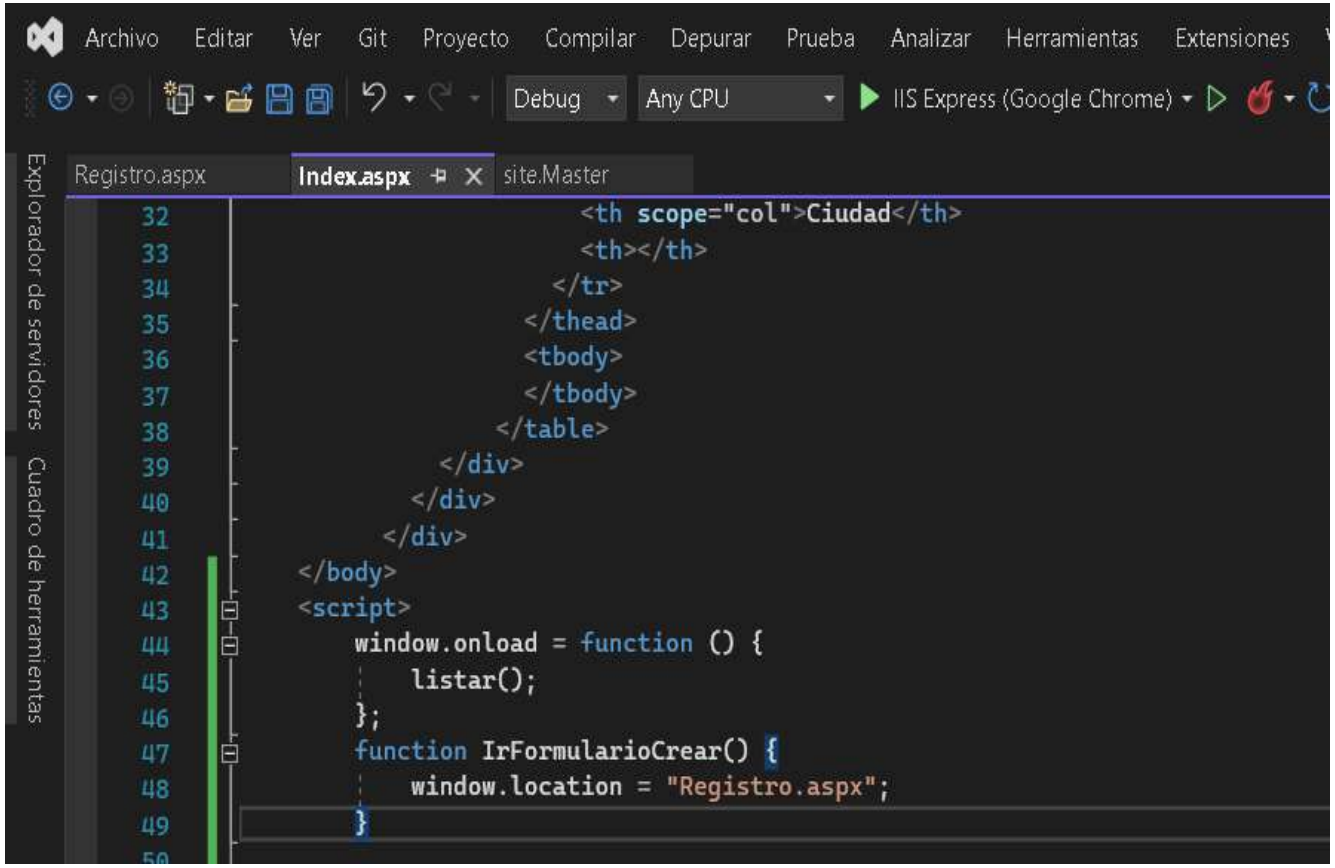
Ejcutamos la aplicación para verificar que el método listar() nos muestre la lista de registro de la base de datos



# Consumo de API REST

Ahora creamos el método `IrFormularioCrear() que permitirá ejecutar la página` registro.aspx

```
function IrFormularioCrear(){
        window.location = "Registro.aspx";
 }
```



# Consumo de API REST

Ahora creamos  el html de la pagina registro.aspx

Encabezado

```
<head>
    <!-- CSS -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css"
integrity="sha384-
TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"
crossorigin="anonymous">

    <script src="https://code.jquery.com/jquery-3.0.0.js" crossorigin="anonymous"></script>
    <!-- jQuery and JS bundle w/ Popper.js -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx"
crossorigin="anonymous"></script>
</head>
```

formulario

```
<body>
    <div class="container">
        <div class="row mt-3">
            <div class="col-12">
                <form>
                    <input type="hidden" id="txtidusuario" />
                    <div class="form-group">
                        <label for="exampleInputEmail1">Documento Identidad</label>
                        <input type="text" class="form-control" id="txtdocumento" placeholder="">
                    </div>
                    <div class="form-group">
                        <label for="exampleInputEmail1">Nombres</label>
                        <input type="text" class="form-control" id="txtnombres" placeholder="">
                    </div>
                    <div class="form-group">
                        <label for="exampleInputEmail1">Telefono</label>
                        <input type="text" class="form-control" id="txttelefono" placeholder="">
                    </div>
                    <div class="form-group">
                        <label for="exampleInputEmail1">Correo</label>
                        <input type="text" class="form-control" id="txtcorreo" placeholder="">
                    </div>
                    <div class="form-group">
                        <label for="exampleInputEmail1">Ciudad</label>
                        <input type="text" class="form-control" id="txtciudad" placeholder="">
                    </div>
                    <button type="button" class="btn btn-primary" onclick="GuardarUsuario()">Guardar</button>
                    <button type="button" class="btn btn-warning" onclick="IrFormularioInicio()">Volver</button>
                </form>
            </div>
        </div>
    </div>
</body>
```

# Consumo de API REST

Ahora ejecutamos nuevamente la aplicación y probamos  que el formulario inicie cuando se presiona el evento click de botón crear nuevo



# Consumo de API REST

Creamos  la función GuardarUsuario que es invocada en la pagina registro.aspx
Asignamos en una variable “data”  los valores obtenidos de cada caja de texto.

```
<script>
 function GuardarUsuario(){
            var data = {
                IdUsuario : $("#txtidusuario").val(),
                DocumentoIdentidad : $("#txtdocumento").val(),
                Nombres : $("#txtnombres").val(),
                Telefono : $("#txttelefono").val(),
                Correo : $("#txtcorreo").val(),
                Ciudad : $("#txtciudad").val()
             }

 }
</script>
```

# Consumo de API REST

Ahora creamos el método Post asignando la url del api **http://localhost:58683/api/Usuario**.

```
<script>
 function GuardarUsuario(){
            var data = {
                IdUsuario : $("#txtidusuario").val(),
                DocumentoIdentidad : $("#txtdocumento").val(),
                Nombres : $("#txtnombres").val(),
                Telefono : $("#txttelefono").val(),
                Correo : $("#txtcorreo").val(),
                Ciudad : $("#txtciudad").val()
              }
            $.post("http://localhost:58683/api/Usuario", data)
              .done(function(response) {
                console.log(response);
                if(response){
                    alert("Usuario Creado");
                    window.location = "Index.aspx";
                }else{
                    alert("Error al crear");
                }
```

# Consumo de API REST

Ejecutamos nuevamente la aplicación y probamos que ingrese un nuevo registro y permita grabar.



# Consumo de API REST

En registro.aspx  ejecutamos el evento onload que permite ejecutar cualdo se inicia la página
Permitirá capturar el id como parámetro y verificar que no esté null para  validar si deseamos  Editar

```
var editar = false;
window.onload = function() {
  var id = $.urlParam('id');
  console.log(id);
  if(id != null){
     editar = true;
     $("#txtidusuario").val(id);
        PintarUsuario(id);
      }
       };
```

```
$.urlParam = function(name){
        var results = new RegExp('[\?&]' + name + '=([^&#]*)').exec(window.location.href);
        if (results==null) {
        return null;
        }
        return decodeURI(results[1]) || 0;
     }
```

```
function PintarUsuario(idUsuario){

        $.get("http://localhost:58683/api/Usuario/" + idUsuario)
        .done(function( response ) {
            console.log(response);
            $("#txtdocumento").val(response.DocumentoIdentidad),
            $("#txtnombres").val(response.Nombres),
            $("#txttelefono").val(response.Telefono),
            $("#txtcorreo").val(response.Correo),
            $("#txtciudad").val(response.Ciudad)
        });
     }
```

Adalberto Cárcamo Alvarado

# Consumo de API REST

En la página Index.aspx creamos el siguiente código que permite direccionar a la página de registro llevando el id a editar

```
$(document).on('click', '.editar', function () {
        console.log($(this).data("id"));
        window.location = "Registro.aspx?id=" + $(this).data("id");

    });
```

# Consumo de API REST

Ejecutamos nuevamente el proyecto y verificamos que el botón editar trae el formulario con los datos a corregir



# Consumo de API REST

Se requiere ahora saber si la variable Editar es verdadero ejecutamos el método para actualizar de lo contario el método de ingresar.

Esto lo haríamos en la página registro.aspx

```
var data = {
            IdUsuario : $("#txtidusuario").val(),
            DocumentoIdentidad : $("#txtdocumento").val(),
            Nombres : $("#txtnombres").val(),
            Telefono : $("#txttelefono").val(),
            Correo : $("#txtcorreo").val(),
            Ciudad : $("#txtciudad").val()
        }

        $.ajax({
        method: "PUT",
        url: "http://localhost:58683/api/Usuario",
        contentType: 'application/json',
        data: JSON.stringify(data), // access in body
        })
        .done(function( response ) {
            console.log(response);
            if(response){
                alert("Se guardaron los cambios");
                window.location = "Index.aspx";
            }else{
                alert("Error al Modificar")
            }
        });
```

EDITAR

```
function GuardarUsuario(){
        if(editar){


        }else{

        }

    }
```

```
var data = {
            DocumentoIdentidad : $("#txtdocumento").val(),
            Nombres : $("#txtnombres").val(),
            Telefono : $("#txttelefono").val(),
            Correo : $("#txtcorreo").val(),
            Ciudad : $("#txtciudad").val()
        }

        $.post("http://localhost:58683/api/Usuario", data)
        .done(function(response) {
            console.log(response);
            if(response){
                alert("Usuario Creado");
                window.location = "Index.aspx";
            }else{
                alert("Error al crear");
            }
        });
```

GUARDAR

Adalberto Cárcamo Alvarado

# Consumo de API REST

```
$(document).on('click', '.eliminar', function () {
        console.log($(this).data("id"));

        $.ajax({
            method: "DELETE",
            url: "http://localhost:58683/api/Usuario/" + $(this).data("id")
        })
            .done(function (response) {
                console.log(response);
                if (response) {
                    listar();
                } else {
                    alert("Error al eliminar")
                }
            });

    });
```

Ahora programamos el evento click de eliminar
En la pagina de index.aspx

Ejecutamos nuevamente la aplicación y probamos

# Consumo de API REST

Por último en la página de registro.aspx  ingresamos el siguiente  método que permite volver a la pagina index.aspx

```
function IrFormularioInicio(){
        window.location = "Index.aspx";
    }
```

# Ejecutamos y probamos la aplicación



# G R A C I A S

Línea de atención al ciudadano: 01 8000 910270
Línea de atención al empresario: 01 8000 910682

@SENAComunica

www.sena.edu.co

Adalberto Cárcamo Alvarado

GC-F-004v.04

# Web Api - Rest

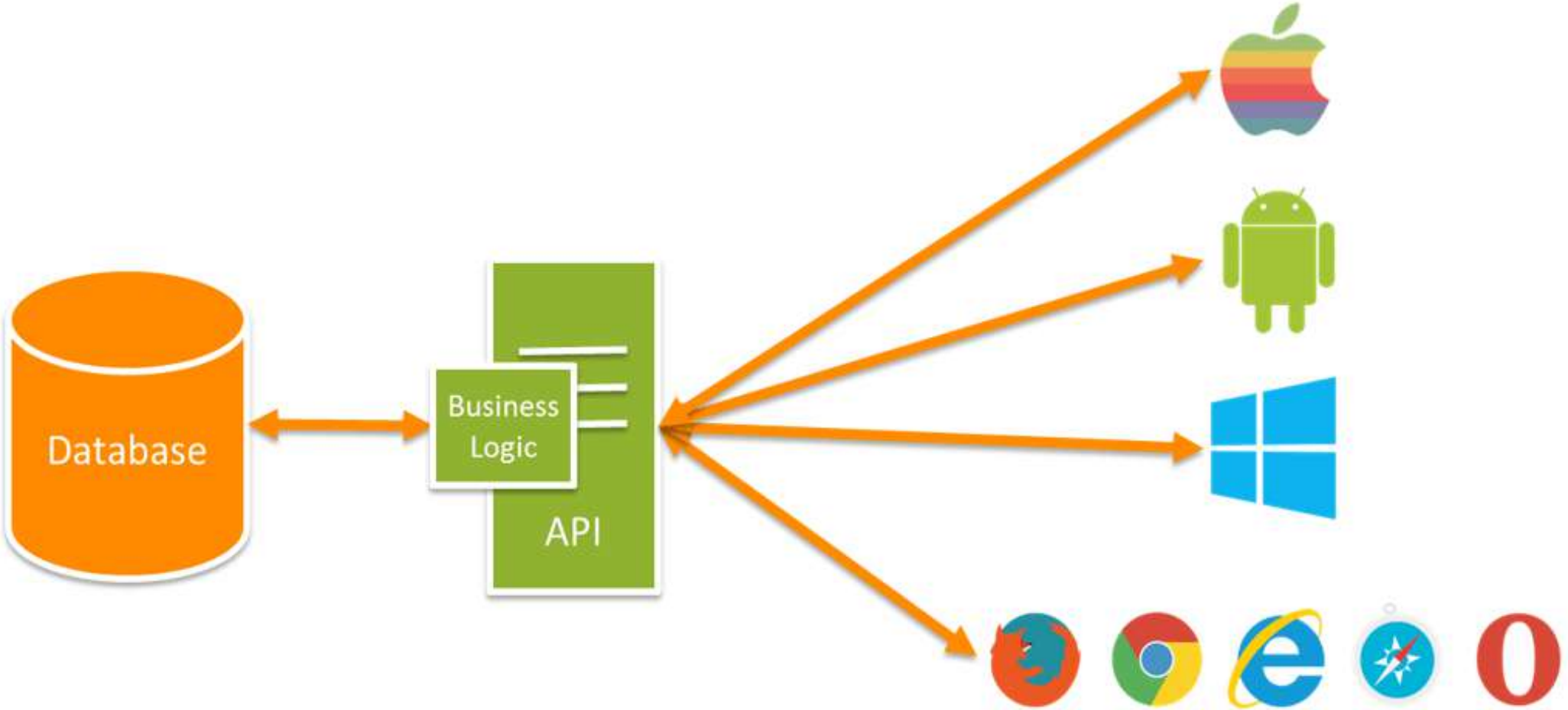



C#
Csharp

REST
API



Instructor: Adalberto Cárcamo A.


Database
Business Logic
API

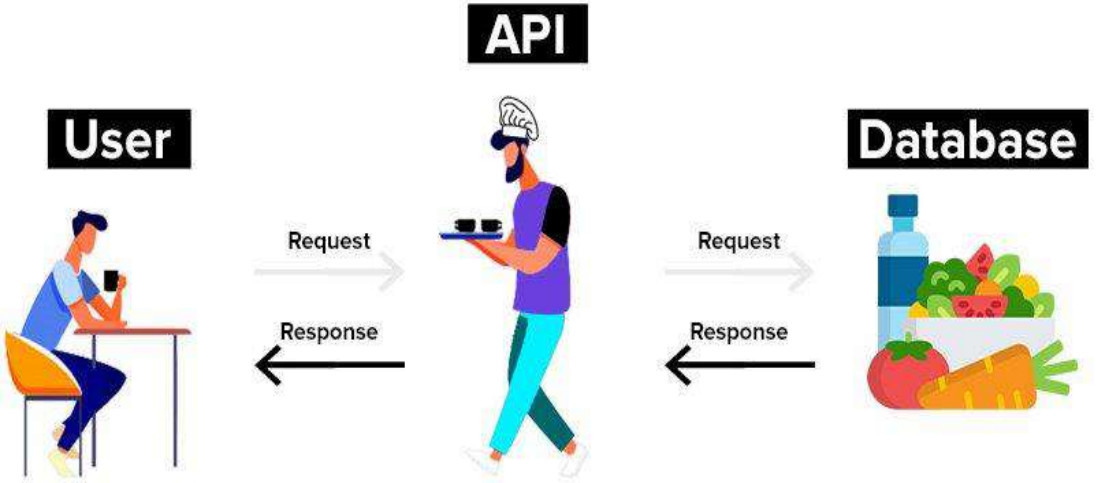



Instructor: Adalberto Cárcamo A.




## Qué es una API

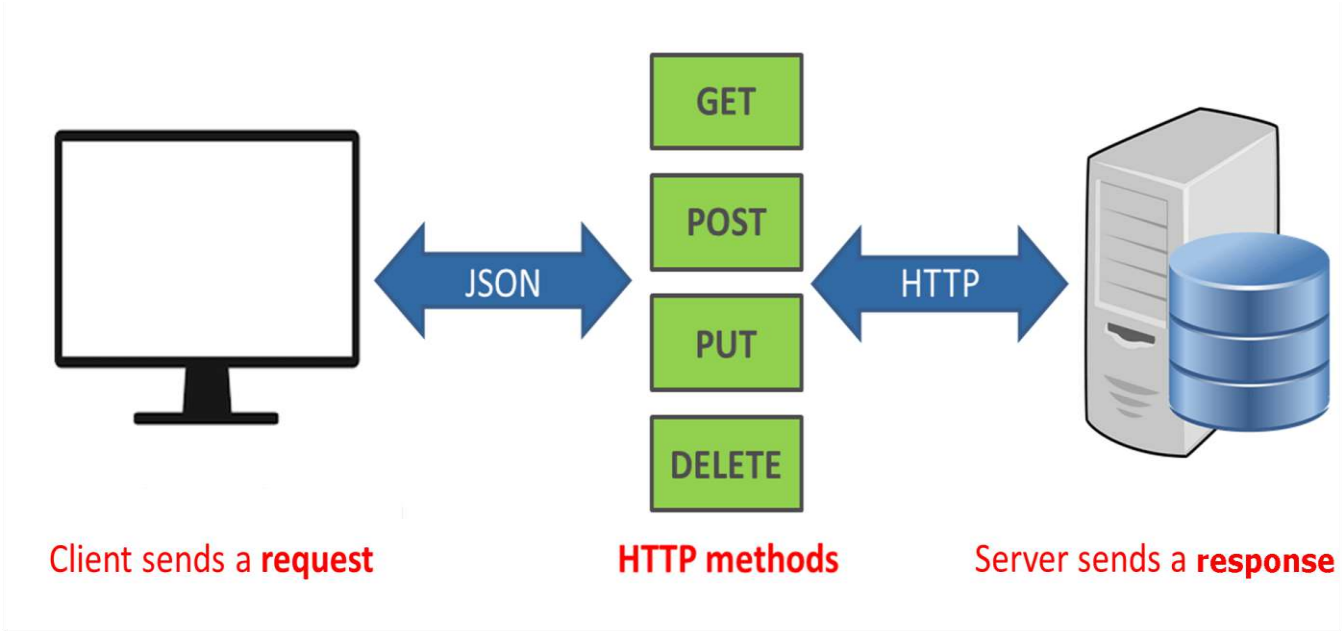
El término API es una abreviatura de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos aplicaciones de software a través de un conjunto de reglas.




Instructor: Adalberto Cárcamo A.




HTTP define métodos que son utilizados para diferentes circunstancias, por lo que trataremos de listar lo más relevantes y más utilizamos en la construcción de servicios REST, los métodos son los siguientes:




**GET**: Es utilizado únicamente para consultar información al servidor, muy parecidos a realizar un SELECT a la base de datos. No soporta el envío del payload

**POST**: Es utilizado para solicitar la creación de un nuevo registro, es decir, algo que no existía previamente, es decir, es equivalente a realizar un INSERT en la base de datos. Soporta el envío del payload.

**PUT**: Se utiliza para actualizar por completo un registro existente, es decir, es parecido a realizar un UPDATE a la base de datos. Soporta el envío del payload.

**PATCH**: Este método es similar al método PUT, pues permite actualizar un registro existente, sin embargo, este se utiliza cuando actualizar solo un fragmento del registro y no en su totalidad, es equivalente a realizar un UPDATE a la base de datos. Soporta el envío del payload

**DELETE**: Este método se utiliza para eliminar un registro existente, es similar a DELETE a la base de datos. No soporta el envío del payload.




# Métodos HTTP

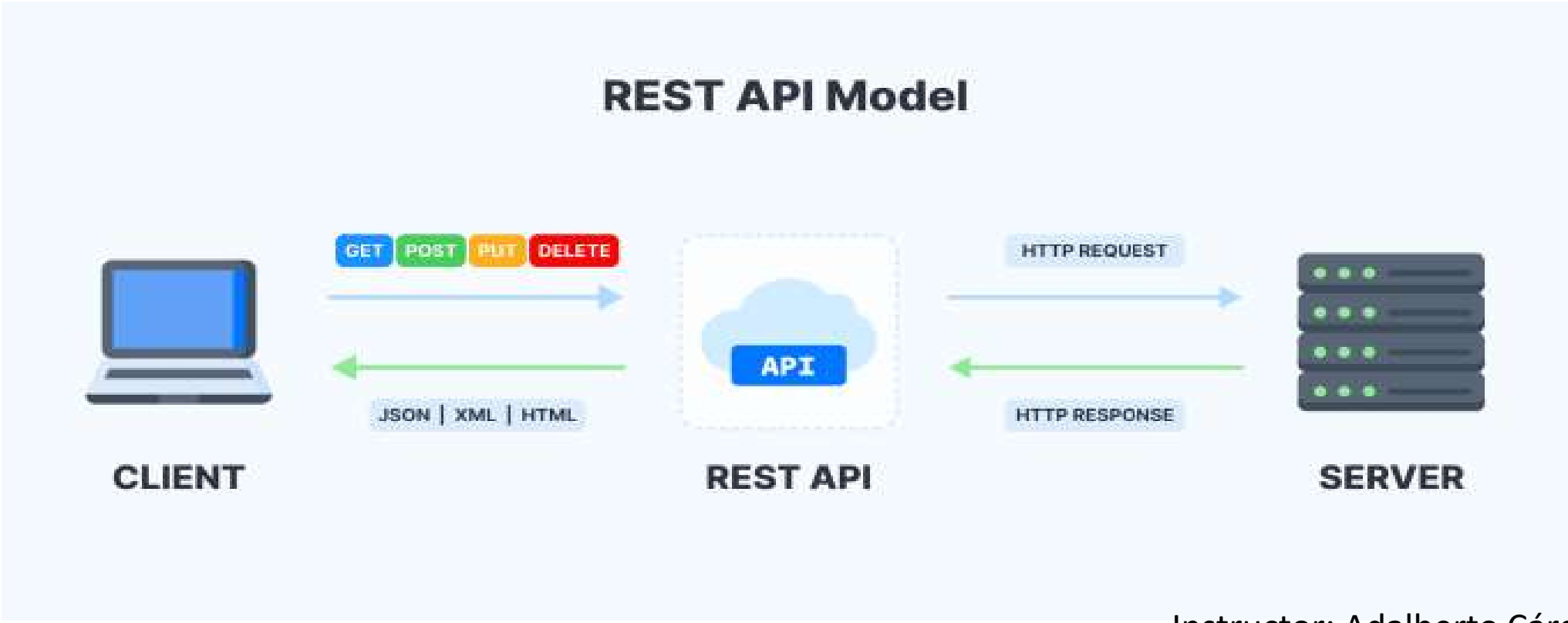
| | | |
|---|---|---|
| GET | → | Leer |
| POST | → | Crear |
| PUT | → | Modificar |
| PATCH | → | Modificar |
| DELETE | → | Eliminar |

Instructor: Adalberto Cárcamo A.




## Realizaremos un ejemplo práctico:

API Rest CRUD con el lenguaje de programación C# y SQL Server




Instructor: Adalberto Cárcamo A.




1. Creación de la base de datos:

```
CREATE    DATABASE    DB_API_DATOS

USE    DB_API_DATOS
```



BD

Instructor: Adalberto Cárcamo A.

2. Creamos  la siguiente tabla de usuario:

```
create table USUARIO(
IdU varchar(15) primary key,
Nombres varchar(80),
Telefono varchar(60),
Correo varchar(40),
Ciudad varchar(60),
FechaIngreso datetime default getdate()
)
```




3. Ingresamos 5 registros de prueba en la tabla de usuario:



```
INSERT INTO USUARIO (IdU, Nombres, Telefono, Correo, Ciudad)
VALUES ( ‘545454', 'Sara’, 3445654321', 'saracar@gmail.com','Cartagena')
```

4. **Creamos la capa de datos** :

Iniciamos con los procedimientos almacenados

a. Registrar
b. Actualizar
c. Eliminar
d. Consultar
e. Listar



Instructor: Adalberto Cárcamo A.




## a. Registrar Usuario



```
create procedure usp_registrar(
@idu varchar(15),
@nombres varchar(80),
@telefono varchar(60),
@correo varchar(40),
@ciudad varchar(60)
)
as
begin
insert into USUARIO(IdU,Nombres,Telefono,Correo,Ciudad)
values
(
@idu,
@nombres,
@telefono,
@correo,
@ciudad
)
end
```




## b. Actualizar Usuario



```
create procedure usp_actualizar (
@idu varchar(15),
@nombres varchar(80),
@telefono varchar(60),
@correo varchar(40),
@ciudad varchar(60)
)
as
begin
update USUARIO set
Nombres = @nombres,
Telefono = @telefono,
Correo = @correo,
Ciudad = @ciudad,
where IdU = @idu

end
```




## c. Eliminar Usuario



```
create procedure usp_eliminar(
@idu varchar(15)
)
as
begin

delete from usuario where IdU = @idusuario

end
```




## d. Consultar Usuario



```
create procedure usp_obtener
 @idu varchar(15)

as
begin
   select * from usuario where IdU = @idu
end
```




e. Listar Usuario



```
create procedure usp_listar
as
begin

select * from usuario
end
```

# Web Api - Rest

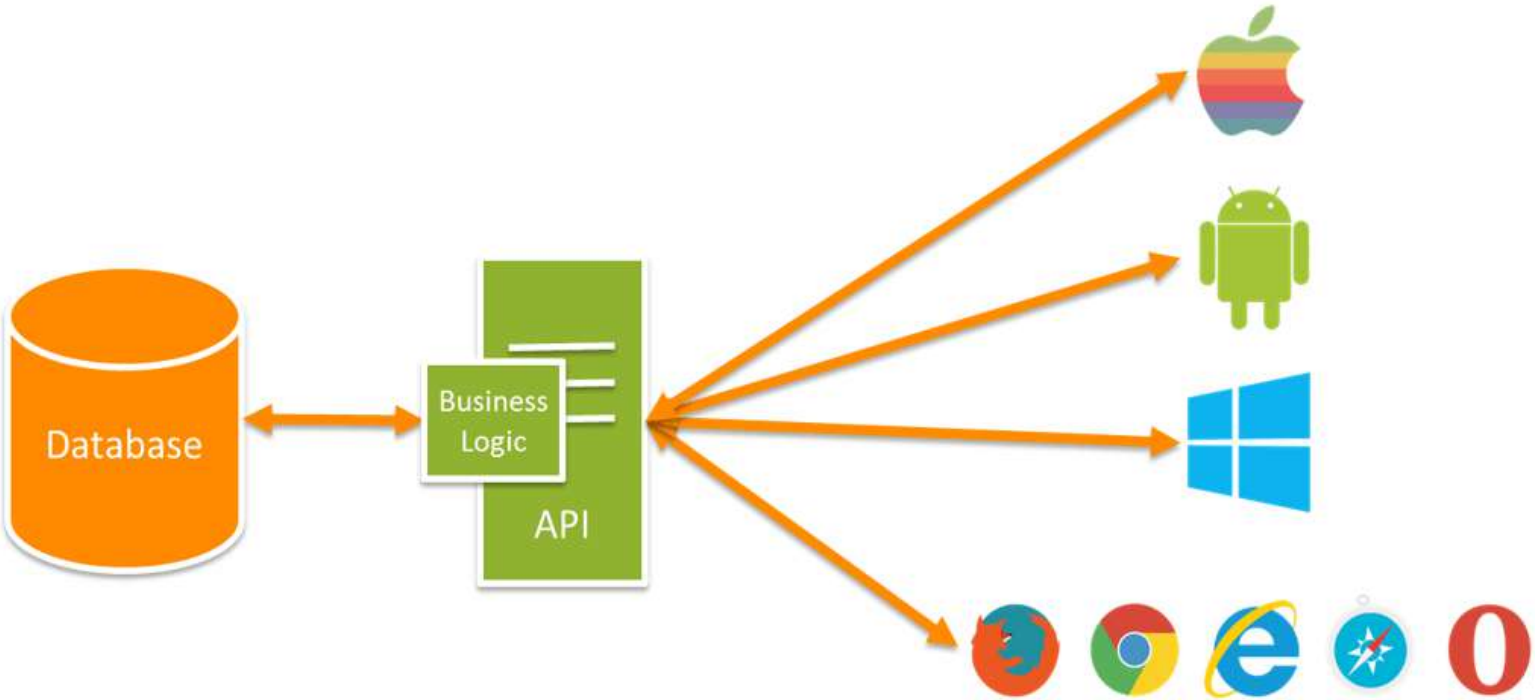



C#
Csharp

REST
API



Instructor: Adalberto Cárcamo A.

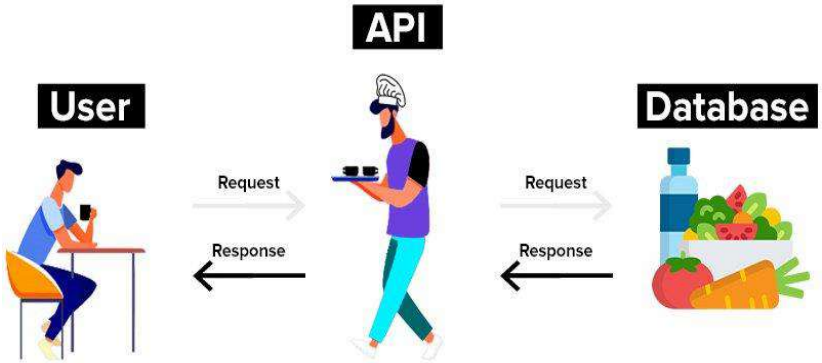

Database
Business Logic
API


API REST
DESARROLLADORES
DE FRONTEND
DESARROLLADORES
IOS/ANDROID
DESARROLLADORES
DE BACKEND




## Qué es una API

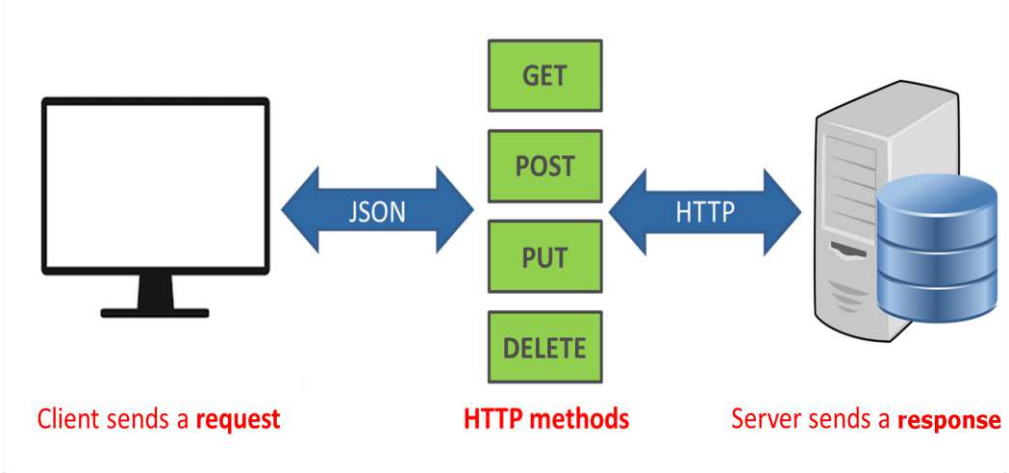
El término API es una abreviatura de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos aplicaciones de software a través de un conjunto de reglas.




Instructor: Adalberto Cárcamo A.




HTTP define métodos que son utilizados para diferentes circunstancias, por lo que trataremos de listar lo más relevantes y más utilizamos en la construcción de servicios REST, los métodos son los siguientes:




**GET**: Es utilizado únicamente para consultar información al servidor, muy parecidos a realizar un SELECT a la base de datos. No soporta el envío del payload

**POST**: Es utilizado para solicitar la creación de un nuevo registro, es decir, algo que no existía previamente, es decir, es equivalente a realizar un INSERT en la base de datos. Soporta el envío del payload.

**PUT**: Se utiliza para actualizar por completo un registro existente, es decir, es parecido a realizar un UPDATE a la base de datos. Soporta el envío del payload.

**PATCH**: Este método es similar al método PUT, pues permite actualizar un registro existente, sin embargo, este se utiliza cuando actualizar solo un fragmento del registro y no en su totalidad, es equivalente a realizar un UPDATE a la base de datos. Soporta el envío del payload

**DELETE**: Este método se utiliza para eliminar un registro existente, es similar a DELETE a la base de datos. No soporta el envío del payload.




# Métodos HTTP

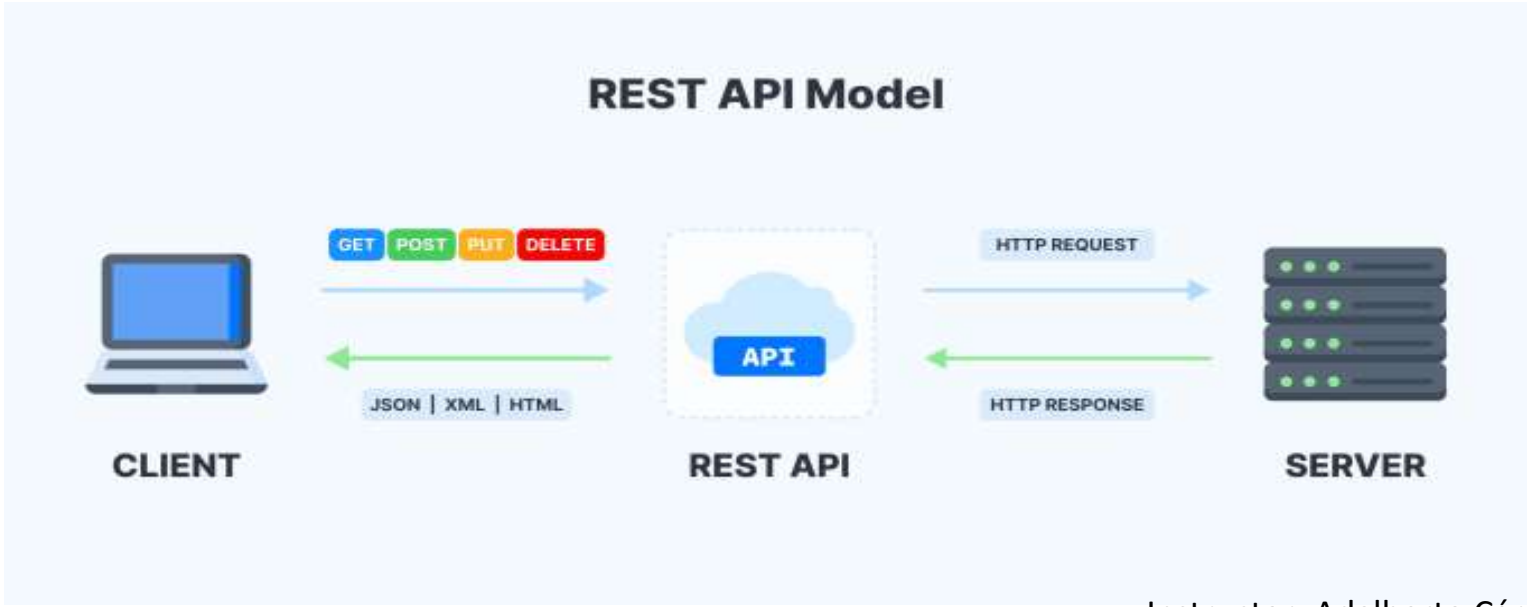
| | | |
|---|---|---|
| GET | → | Leer |
| POST | → | Crear |
| PUT | → | Modificar |
| PATCH | → | Modificar |
| DELETE | → | Eliminar |

Instructor: Adalberto Cárcamo A.




**Realizaremos un ejemplo práctico:**

API Rest  CRUD con el lenguaje de programación  C#  y  SQL Server




Instructor: Adalberto Cárcamo A.

1. Creación de la base de datos:

```
CREATE DATABASE DB_API_DATOS

USE DB_API_DATOS
```



BD

Instructor: Adalberto Cárcamo A.

2. Creamos  la siguiente tabla de usuario:

```
create table USUARIO(
IdU varchar(15) primary key,
Nombres varchar(80),
Telefono varchar(60),
Correo varchar(40),
Ciudad varchar(60),
FechaIngreso datetime default getdate()
)
```

3. Ingresamos 5 registros de prueba en la tabla de usuario:



```
INSERT USUARIO (IdU, Nombres, Telefono, Correo, Ciudad)
VALUES ( ‘545454', 'Sara’, 3445654321', 'saracar@gmail.com','Cartagena')
```

4. **Creamos la capa de datos** :

Iniciamos con los procedimientos almacenados

a. Registrar
b. Actualizar
c. Eliminar
d. Consultar
e. Listar






a. Registrar Usuario



```
create procedure usp_registrar(
@idu varchar(15),
@nombres varchar(80),
@telefono varchar(60),
@correo varchar(40),
@ciudad varchar(60)
)
as
begin
insert into USUARIO(IdU,Nombres,Telefono,Correo,Ciudad)
values
(
@idu,
@nombres,
@telefono,
@correo,
@ciudad
)
end
```




## b. Actualizar Usuario



```
create procedure usp_actualizar (
@idu varchar(15),
@nombres varchar(80),
@telefono varchar(60),
@correo varchar(40),
@ciudad varchar(60)
)
as
begin
update USUARIO set
Nombres = @nombres,
Telefono = @telefono,
Correo = @correo,
Ciudad = @ciudad
where IdU = @idu
)
end
```




c. Eliminar Usuario



```
create procedure usp_eliminar(
@idu varchar(15)
)
as
begin

delete from usuario where IdU = @idusuario

end
```




d. Consultar Usuario



```
create procedure usp_obtener
 @idu varchar(15)

as
begin
   select * from usuario where IdU = @idu
end
```




e. Listar Usuario



```
create procedure usp_listar
as
begin

select * from usuario
end
```

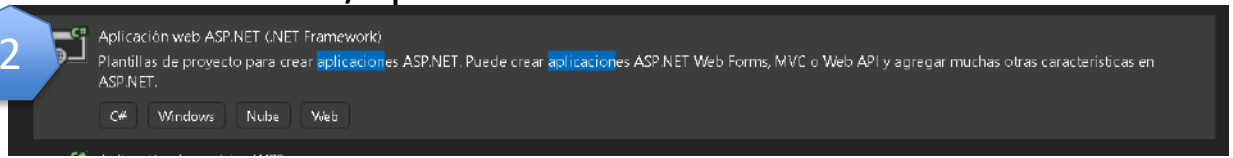
# API

Creamos ahora la aplicación Web vacía con la opción API web en Visual Studio /Aplicación Web

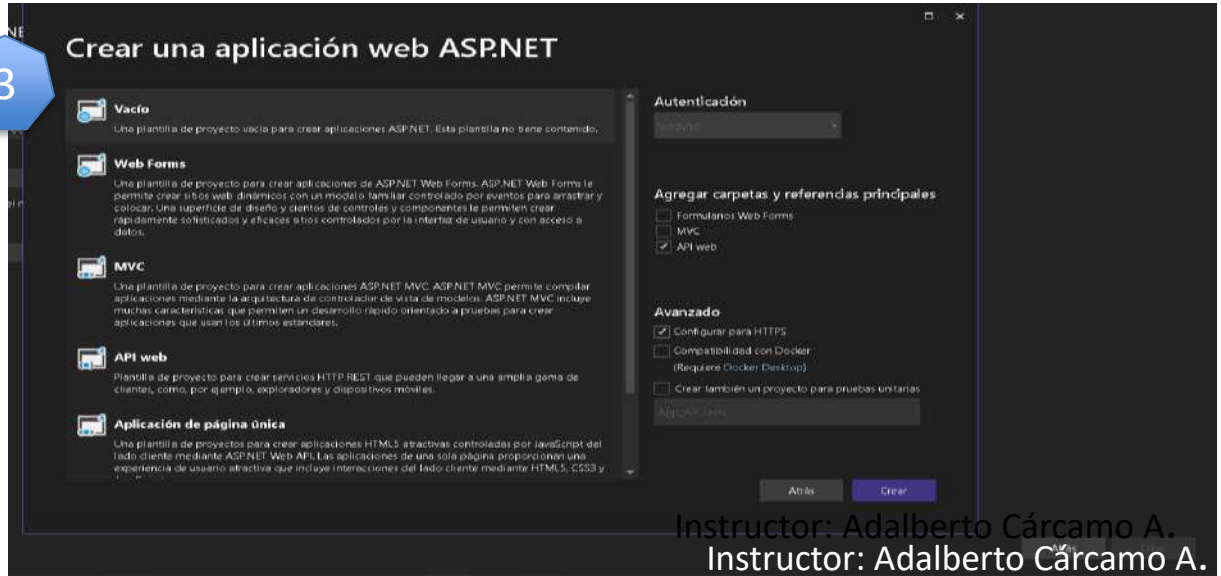
1




2



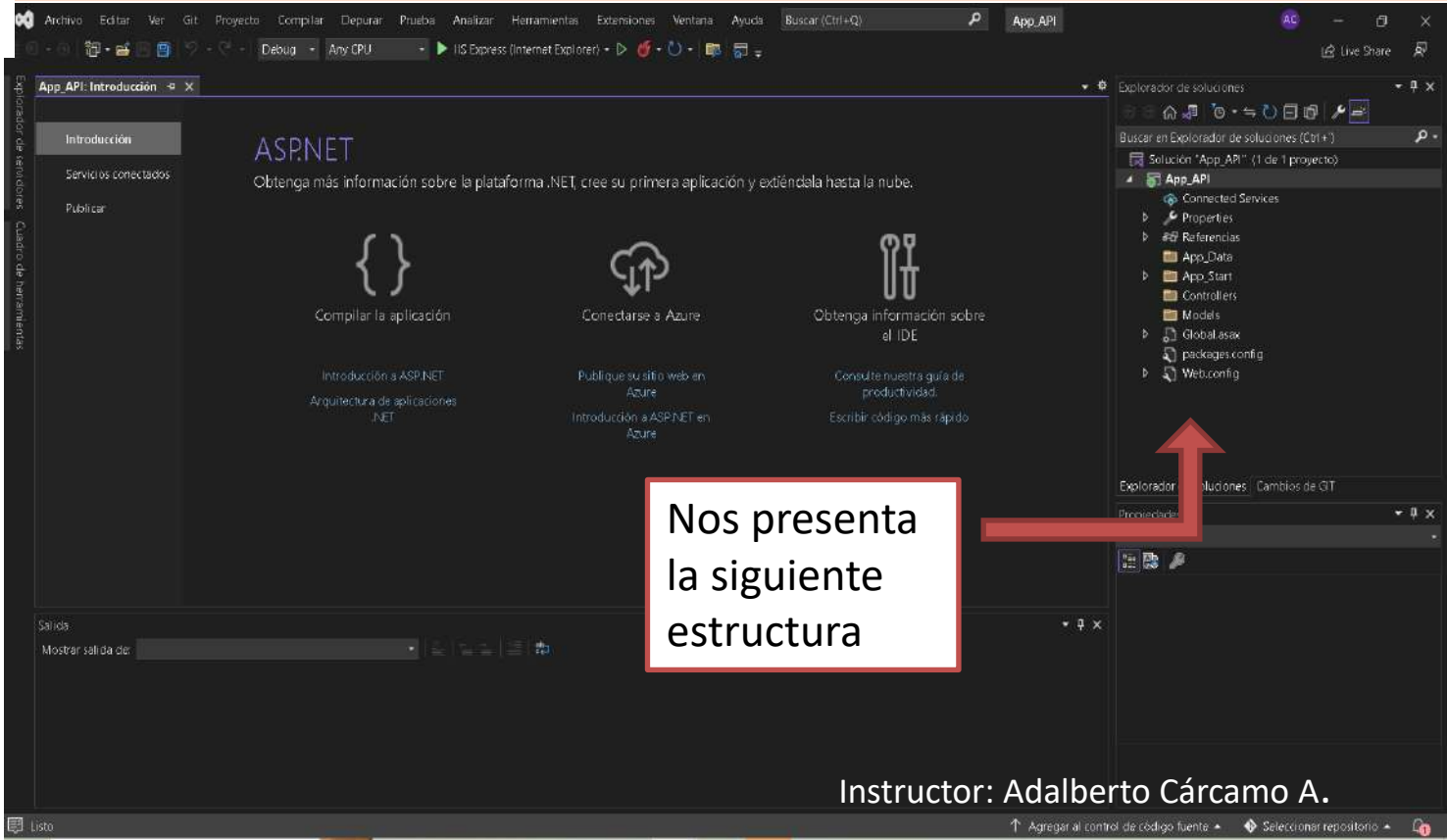

3





Instructor: Adalberto Cárcamo A.

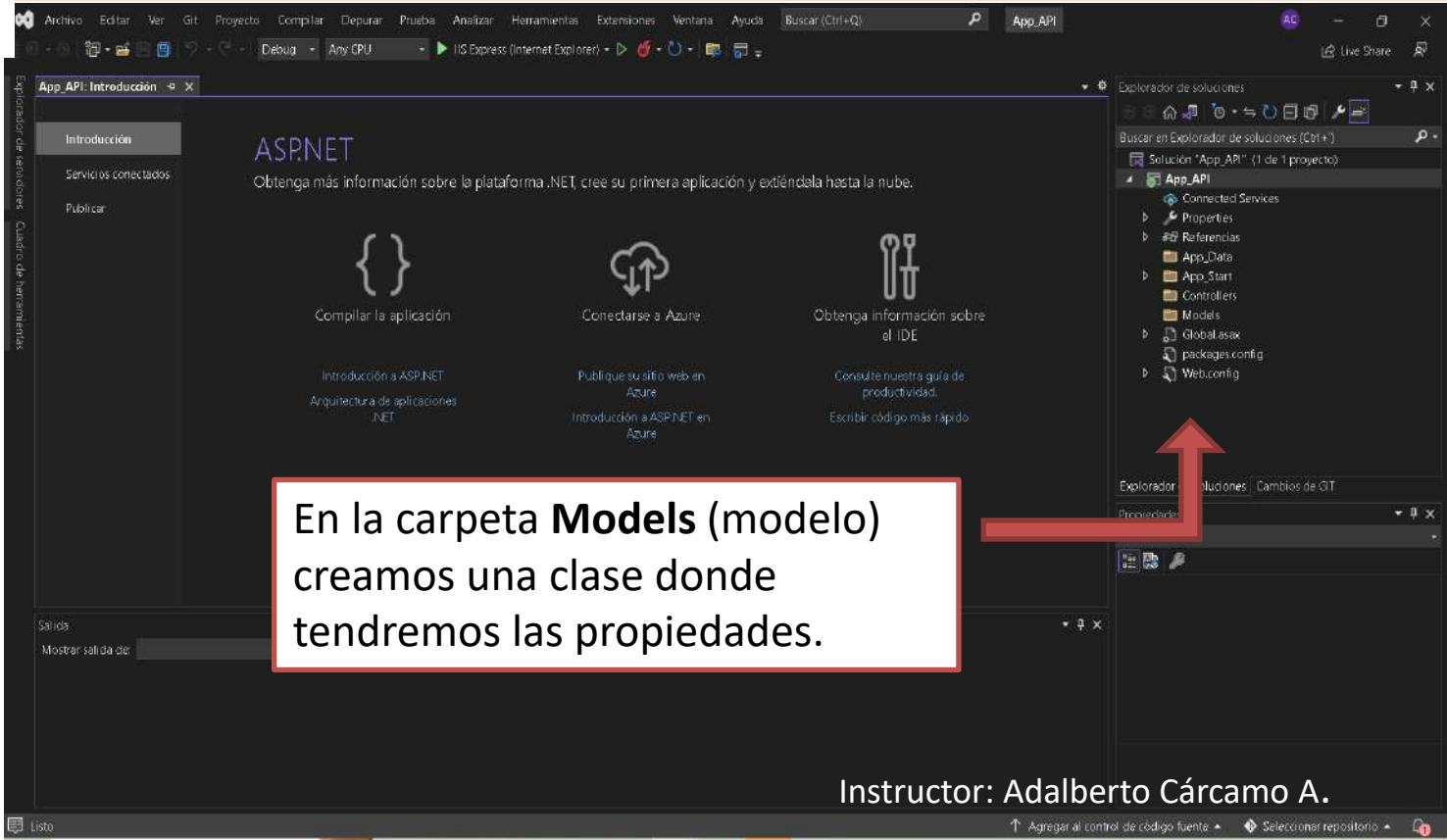



Nos presenta la siguiente estructura

Instructor: Adalberto Cárcamo A.



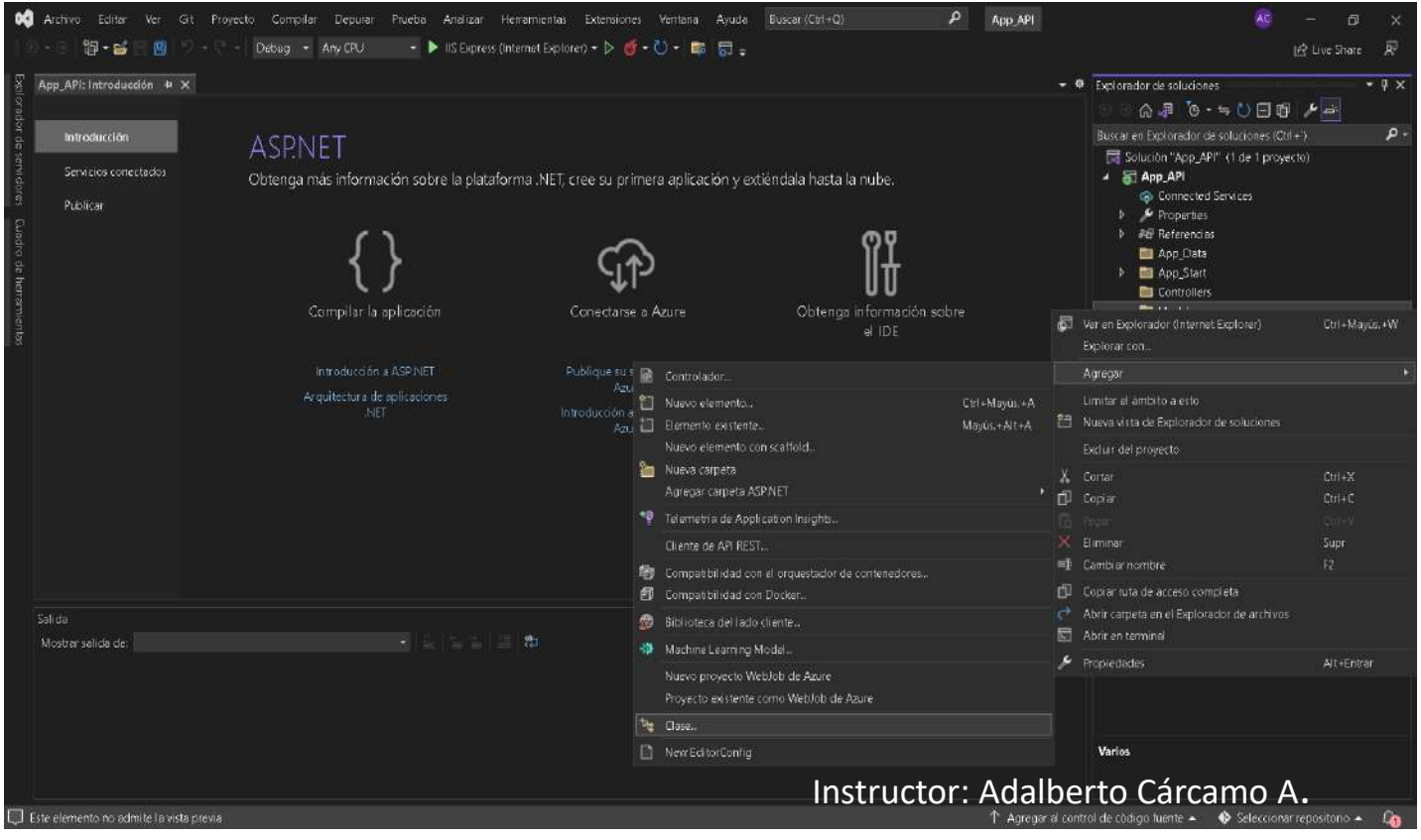




En la carpeta **Models** (modelo) creamos una clase donde tendremos las propiedades.

Instructor: Adalberto Cárcamo A.

**Para crear la clase:**

a. Click derecho en la carpeta Models
b. Agregar/Clase
c. Asignamos el nombre de la clase **Usuario**




Instructor: Adalberto Cárcamo A.




Nos presenta la siguiente estructura:



Usuario.cs | App_API: Introducción

App_API | App_API.Models.Usuario

```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Web;
5
6   namespace App_API.Models
7   {
8       public class Usuario
9       {
10      }
11  }
```

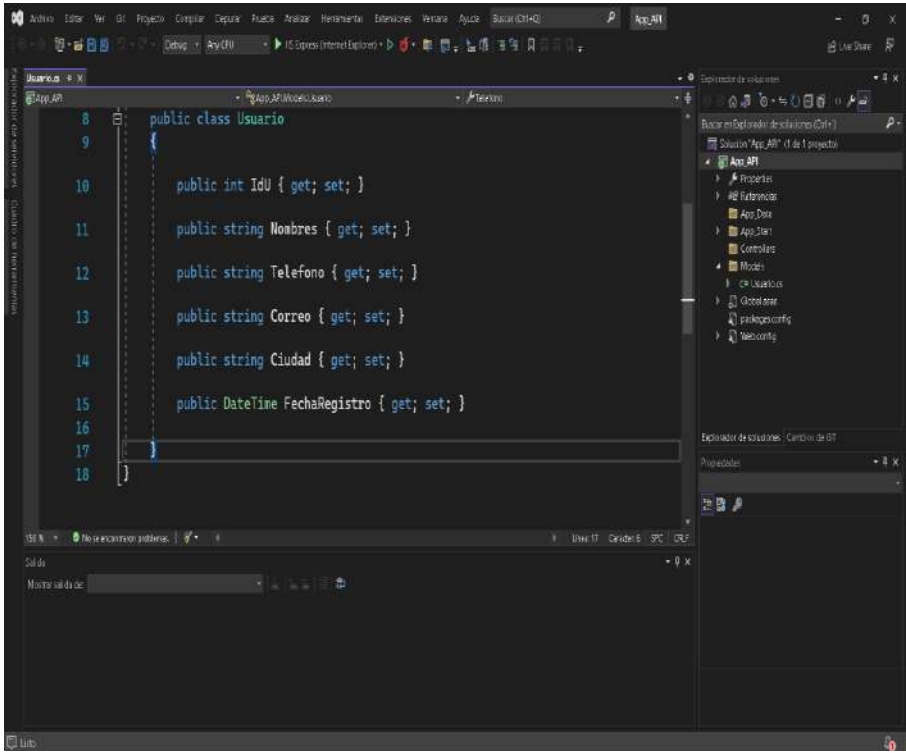
Instructor: Adalberto Cárcamo A.




```
public class Usuario
    {
        public int IdU { get; set; }
        public string Nombres { get; set; }
        public string Telefono { get; set; }
        public string Correo { get; set; }
        public string Ciudad { get; set; }
        public DateTime FechaIngreso { get; set; }

    }
```

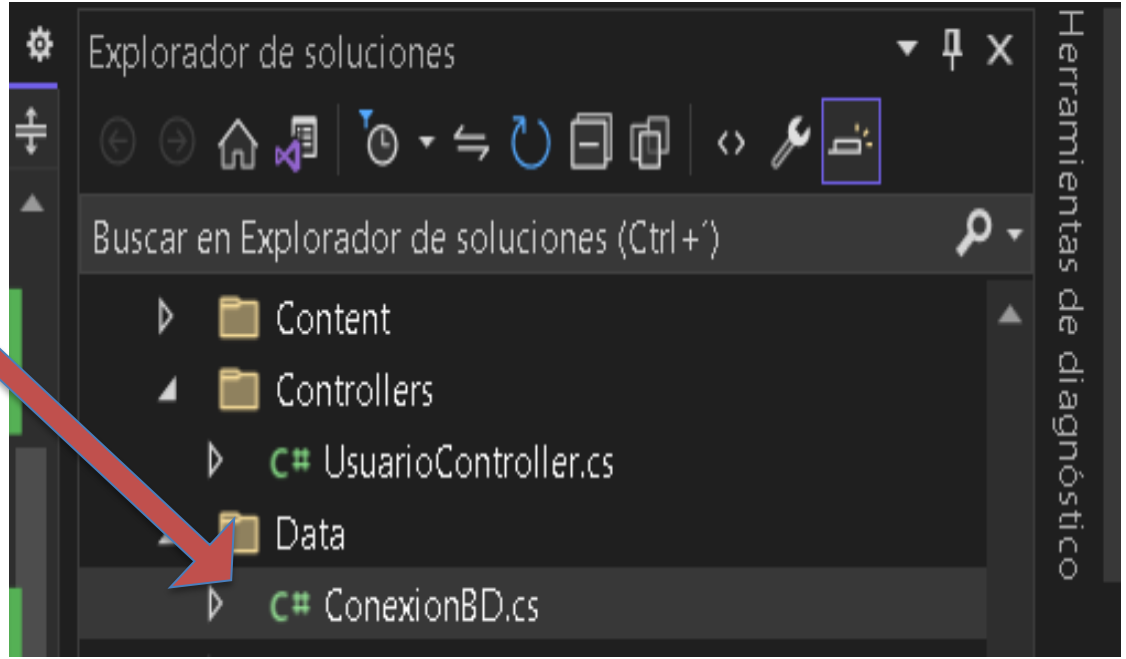





Instructor: Adalberto Cárcamo A.

Creamos una carpeta llamada Data y luego asignamos una clase llamada ConexionBD que será compartida por el instructor.





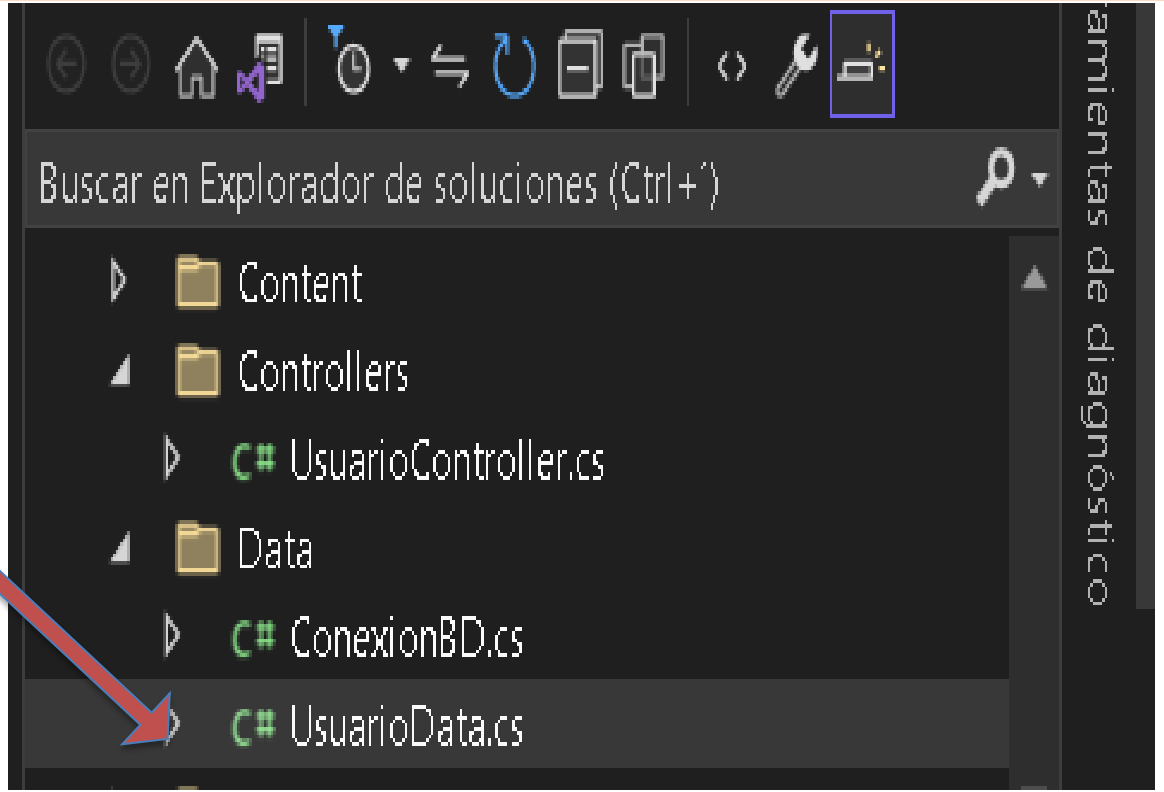

Instructor: Adalberto Cárcamo A.

Creamos una clase llamada UsuarioData en la carpeta Data.

En esta clase creamos la lógica de negocio de la Api

Donde desarrollaremos los métodos correspondientes.








Creamos el método Registrar Usuario

```
 public static bool registrarUsuario(Usuario oUsuario)
        {
            ConexionBD objEst = new ConexionBD();
            string sentencia;
            sentencia = "EXECUTE usp_registrar '" +
oUsuario.IdU + "','" + oUsuario.Nombres + "',’” +
oUsuario.Telefono + "','" + oUsuario.Correo + "','" +
oUsuario.Ciudad + "'";

            if (!objEst.EjecutarSentencia(sentencia, false))
            {

                objEst = null;
                return false;

            }
            else
            {
                objEst = null;
                return true;
            }
        }
```




Creamos el método Actualizar Usuario

```
public static bool actualizarUsuario(Usuario oUsuario)
        {
            ConexionBD objEst = new ConexionBD();
            string sentencia;
            sentencia = "EXECUTE usp_actualizar '" +
oUsuario.IdU + "','" + oUsuario.Nombres + "','" +
oUsuario.Telefono + "','" + oUsuario.Correo + "','" +
oUsuario.Ciudad + "'";

     if (!objEst.EjecutarSentencia(sentencia, false))
        {

                objEst = null;
                return false;

            }
            else
            {
                objEst = null;
                return true;
            }
        }
```




Creamos el método Eliminar Usuario

```
public static bool eliminarUsuario(string id)
        {
            ConexionBD objEst = new ConexionBD();
            string sentencia;
            sentencia = "EXECUTE usp_Eliminar '" + id+ "'";

            if (!objEst.EjecutarSentencia(sentencia, false))
            {

                objEst = null;
                return false;

            }
            else
            {
                objEst = null;
                return true;
            }
        }
```




Creamos el método Listar Usuario

```
public static List<Usuario> Listar()
        {
            List<Usuario> oListaUsuario = new List<Usuario>();
            ConexionBD objEst = new ConexionBD();
            string sentencia;
            sentencia = "EXECUTE usp_consultar";

            if (objEst.Consultar(sentencia, false))
            {
                   SqlDataReader dr = objEst.Reader;
                     while (dr.Read())
                        {
                            oListaUsuario.Add(new Usuario()
                            {
                                IdU = Convert.ToInt32(dr["IdU"]),
                                Nombres = dr["Nombres"].ToString(),
                                Telefono = dr["Telefono"].ToString(),
                                Correo = dr["Correo"].ToString(),
                                Ciudad = dr["Ciudad"].ToString(),
                                FechaIngreso = Convert.ToDateTime(dr["FechaIngreso"].ToString())
                            });
                        }

                     return oListaUsuario;
            }
            else
            {
                return oListaUsuario;
            }
        }
```




Creamos el método obtener Usuario

```
public static List<Usuario> Obtener(string id)
        {
            List<Usuario> oListaUsuario = new List<Usuario>();
            ConexionBD objEst = new ConexionBD();
            string sentencia;
            sentencia = "EXECUTE usp_obtener '"+ id +"'";

            if (objEst.Consultar(sentencia, false))
            {

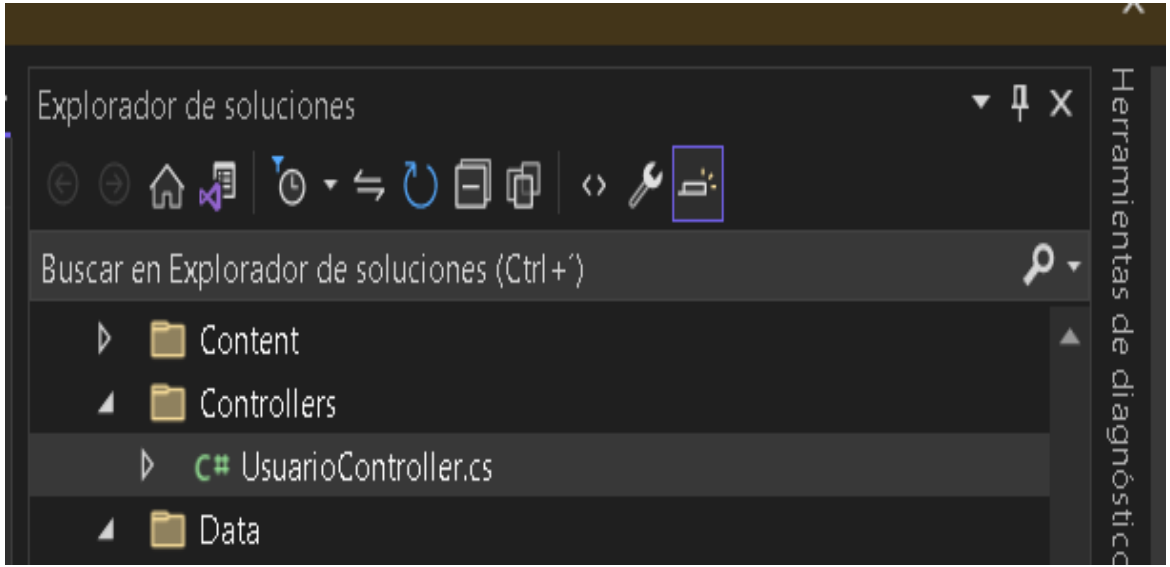
                SqlDataReader dr = objEst.Reader;
                while (dr.Read())
                {
                    oListaUsuario.Add(new Usuario()
                    {
                        IdU = Convert.ToInt32(dr["IdU"]),
                        Nombres = dr["Nombres"].ToString(),
                        Telefono = dr["Telefono"].ToString(),
                        Correo = dr["Correo"].ToString(),
                        Ciudad = dr["Ciudad"].ToString(),
                        FechaIngreso = Convert.ToDateTime(dr["FechaIngreso"].ToString())
                    });
                }

                return oListaUsuario;


            }
            else
            {
                return oListaUsuario;
            }
        }
```

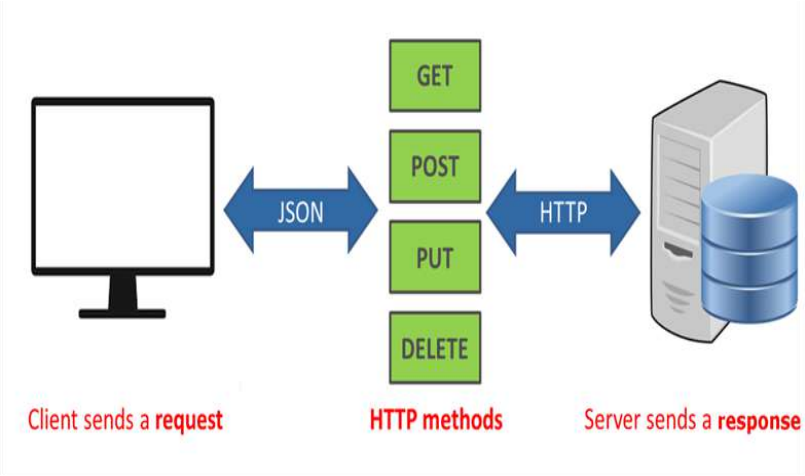
Ahora creamos el controlador:
Dentro de la carpeta Controllers.

1. Click derecho agregar nuevo elemento.
2. C# / web /web Api
3. Escogemos Clase de controlador web Api
4. Asignamos el nombre UsuarioController.




Instructor: Adalberto Cárcamo A.




```
// GET api/<controller>
public List<Usuario> Get()
{
    return UsuarioData.Listar();
}

// GET api/<controller>/5
public List<Usuario> Get(string id)
{
    return UsuarioData.Obtener(id);
}

// POST api/<controller>
public bool Post([FromBody] Usuario oUsuario)
{
    return UsuarioData.registrarUsuario(oUsuario);
}


// PUT api/<controller>/5
public bool Put([FromBody] Usuario oUsuario)
{
    return UsuarioData.actualizarUsuario(oUsuario);
}

// DELETE api/<controller>/5
public bool Delete(string id)
{
    return UsuarioData.eliminarUsuario(id);
}
```

Instructor: Adalberto Cárcamo A.




Para que los métodos de la Api puedan ser leídos desde la ruta que nos asigna la aplicación.

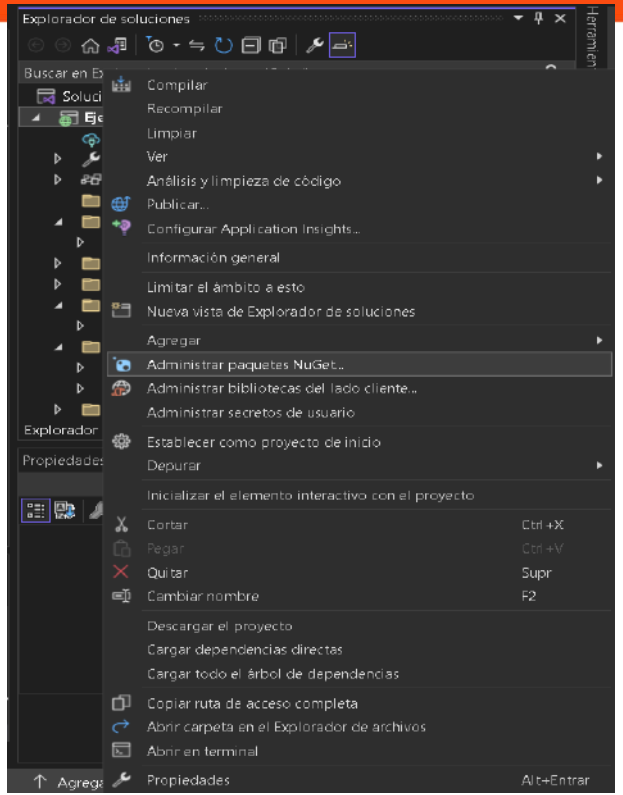
Debemos instalar el siguiente paquete:

**Microsoft.AspNet.Web.WebApi.Cors**

Instructor: Adalberto Cárcamo A.



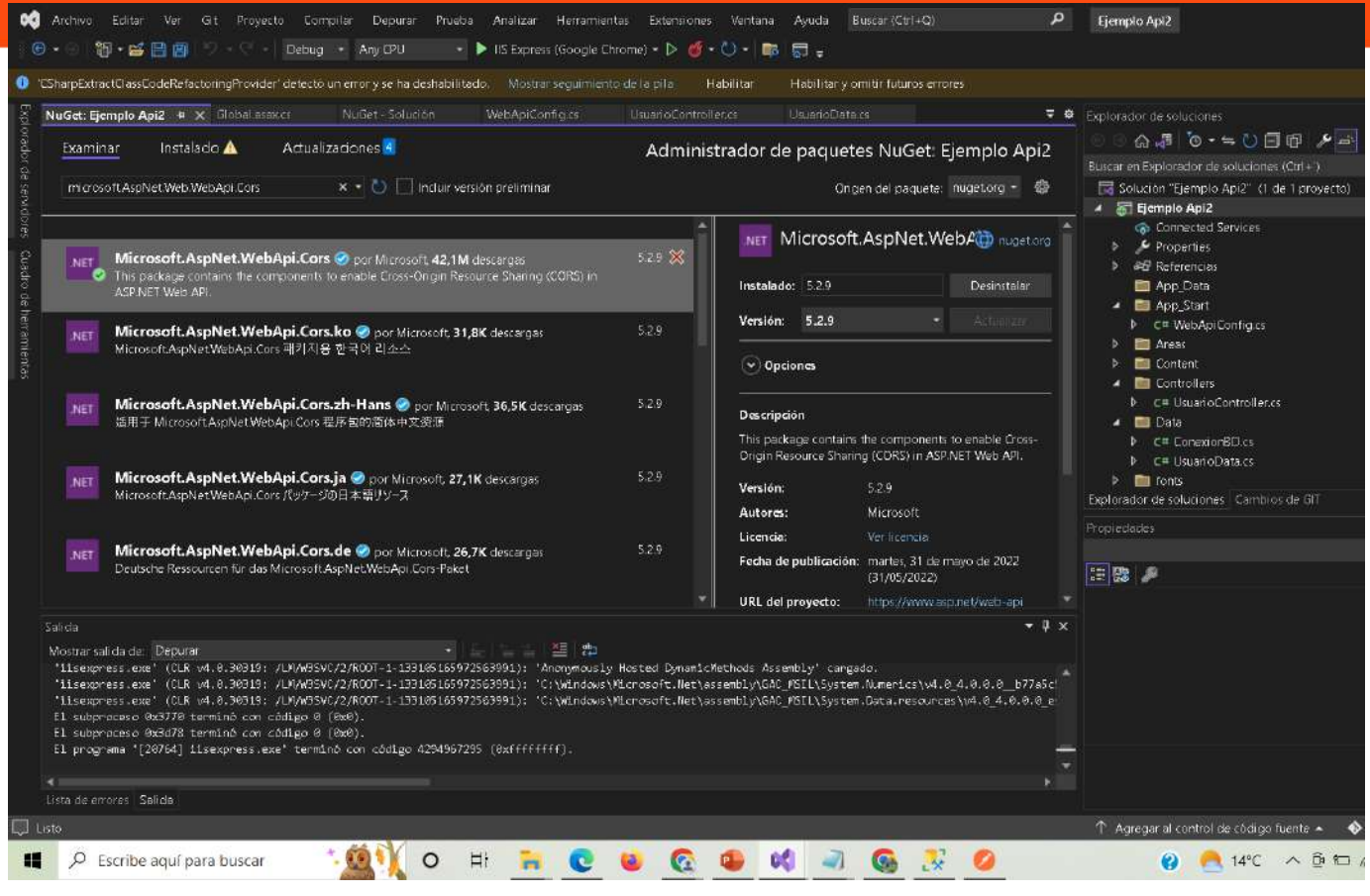

Click derecho en la solución y luego escogemos Administrador paquetes NuGet..




Instructor: Adalberto Cárcamo A.

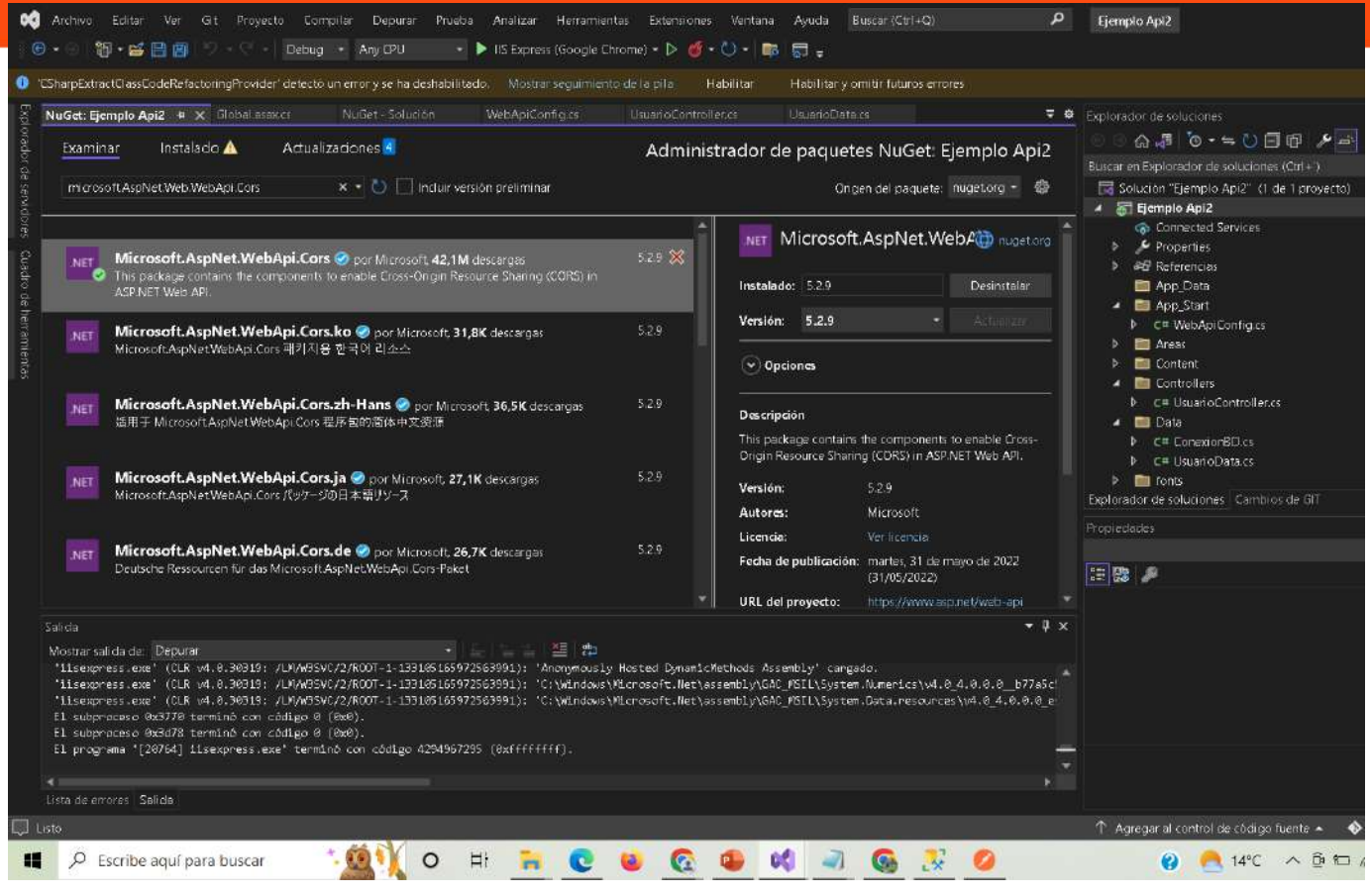



En examinar buscamos el paquete y luego lo instalamos ,habilitando en la solución y el proyecto






En examinar buscamos el paquete y luego lo instalamos ,habilitando en la solución y el proyecto

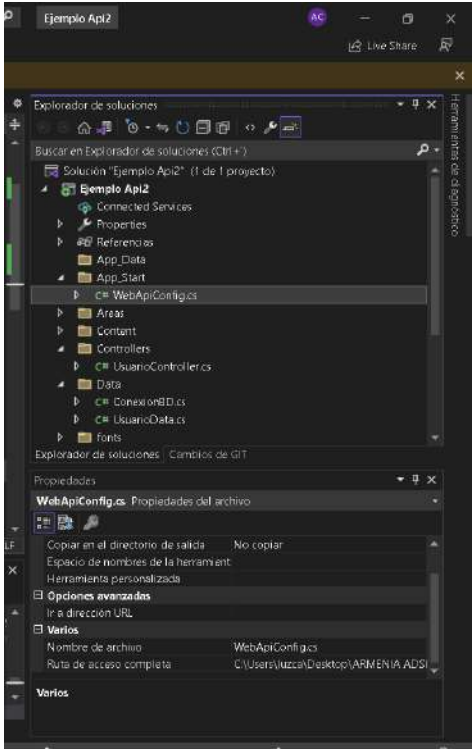





Abrimos el la clase WebConfig de la carpeta App_Start y asignamos el siguiente código que nos permitirá que la Api sea leída desde cualquier aplicación es importante primero usar los paquetes que instalamos :

```
using System.Web.Http;
using System.Web.Http.Cors;
```

```
// Configuración y servicios de Web API
            var cors = new EnableCorsAttribute("*", "*", "*");
            config.EnableCors(cors);
```




Instructor: Adalberto Cárcamo A.




Ejecutamos la Api y nos trae la url en este ejemplo seria el siguiente:

http://localhost:49770/




Instructor: Adalberto Cárcamo A.




Utilizaremos ahora una herramienta que nos permitirá hacer pruebas de las solicitudes de la API.



POSTMAN

Instructor: Adalberto Cárcamo A.
Instructor: Adalberto Cárcamo A.




GET https://jsonplaceholder.typicode

No Environment

https://jsonplaceholder.typicode.com/users

1

GET

2

https://jsonplaceholder.typicode.com/users

3

Send

Params Authorization Headers Body Pre-request Script Tests Cookies Code

| KEY | VALUE | DESCRIPTION |
| --- | --- | --- |
| Key | Value | Description |

Body Cookies (1) Headers (19) Test Results

4

Status: 200 OK Time: 784 ms Size: 6.27 KB

Pretty Raw Preview JSON

5


```
[
    {
        "id": 1,
        "name": "Leanne Graham",
        "username": "Bret",
        "email": "Sincere@april.biz",
        "address": {
            "street": "Kulas Light",
            "suite": "Apt. 556",
            "city": "Gwenborough",
            "zipcode": "92998-3874",
            "geo": {
                "lat": "-37.3159",
                "lng": "81.1496"
            }
        },
        "phone": "1-770-736-8031 x56442",
        "website": "hildegard.org",
        "company": {
            "name": "Romaguera-Crona",
            "catchPhrase": "Multi-layered client-server neural-net",
            "bs": "harness real-time e-markets"
        }
    },
    {
        "id": 2,
        "name": "Ervin Howell",
```


Learn

Instructor: Adalberto Cárcamo A.




# GRACIAS

# Web Api - Rest

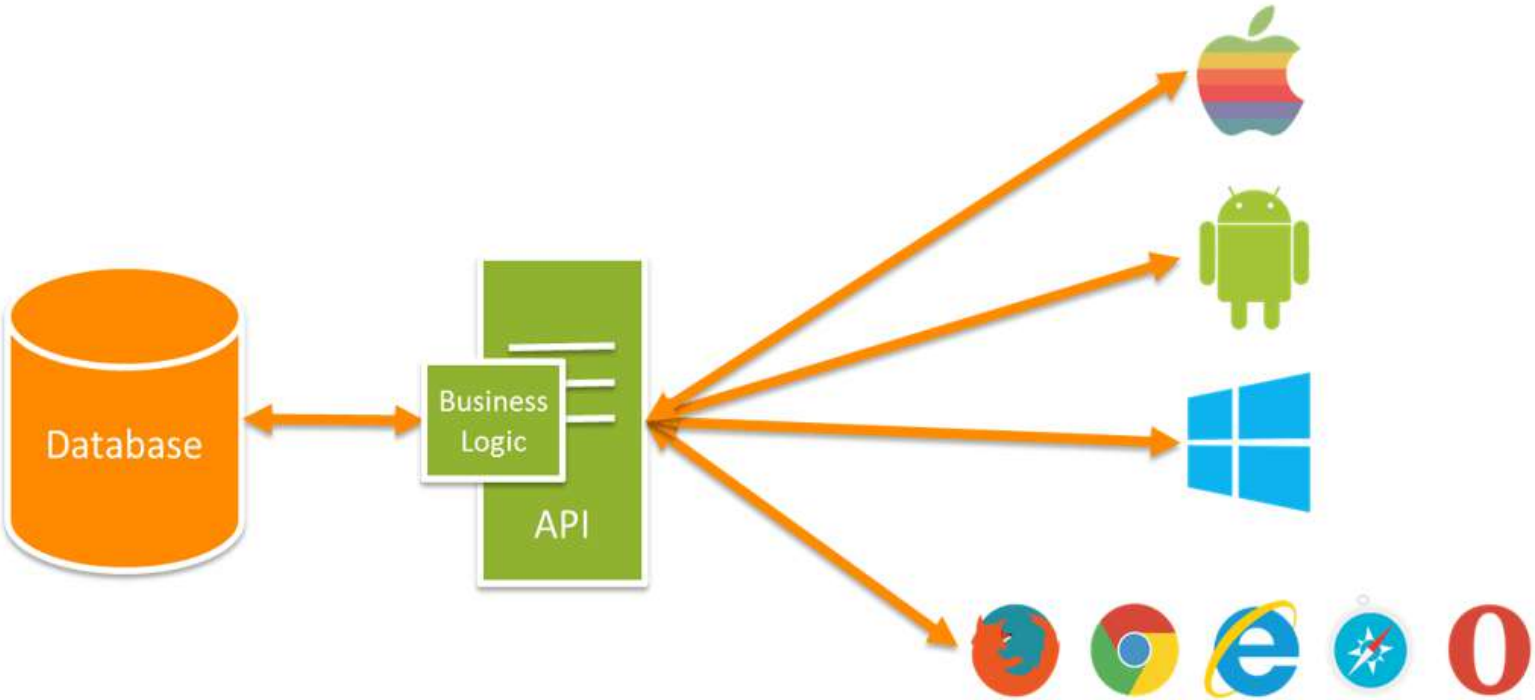








Instructor: Adalberto Cárcamo A.

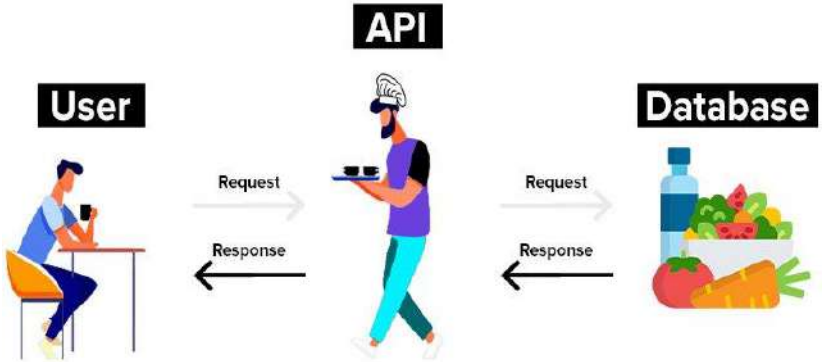

Database
Business Logic
API


API REST
DESARROLLADORES
DE FRONTEND
DESARROLLADORES
IOS/ANDROID
DESARROLLADORES
DE BACKEND




## Qué es una API

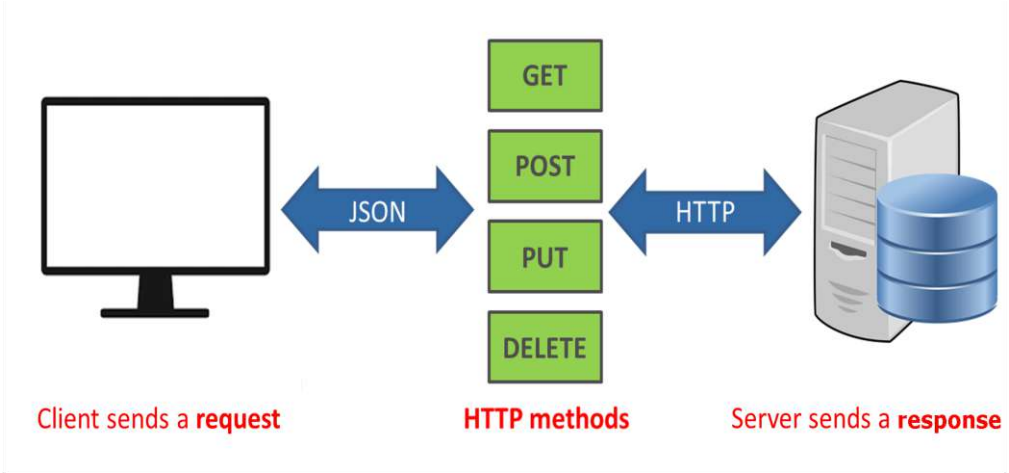
El término API es una abreviatura de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos aplicaciones de software a través de un conjunto de reglas.




Instructor: Adalberto Cárcamo A.




HTTP define métodos que son utilizados para diferentes circunstancias, por lo que trataremos de listar lo más relevantes y más utilizamos en la construcción de servicios REST, los métodos son los siguientes:




**GET**: Es utilizado únicamente para consultar información al servidor, muy parecidos a realizar un SELECT a la base de datos. No soporta el envío del payload

**POST**: Es utilizado para solicitar la creación de un nuevo registro, es decir, algo que no existía previamente, es decir, es equivalente a realizar un INSERT en la base de datos. Soporta el envío del payload.

**PUT**: Se utiliza para actualizar por completo un registro existente, es decir, es parecido a realizar un UPDATE a la base de datos. Soporta el envío del payload.

**PATCH**: Este método es similar al método PUT, pues permite actualizar un registro existente, sin embargo, este se utiliza cuando actualizar solo un fragmento del registro y no en su totalidad, es equivalente a realizar un UPDATE a la base de datos. Soporta el envío del payload

**DELETE**: Este método se utiliza para eliminar un registro existente, es similar a DELETE a la base de datos. No soporta el envío del payload.




# Métodos HTTP

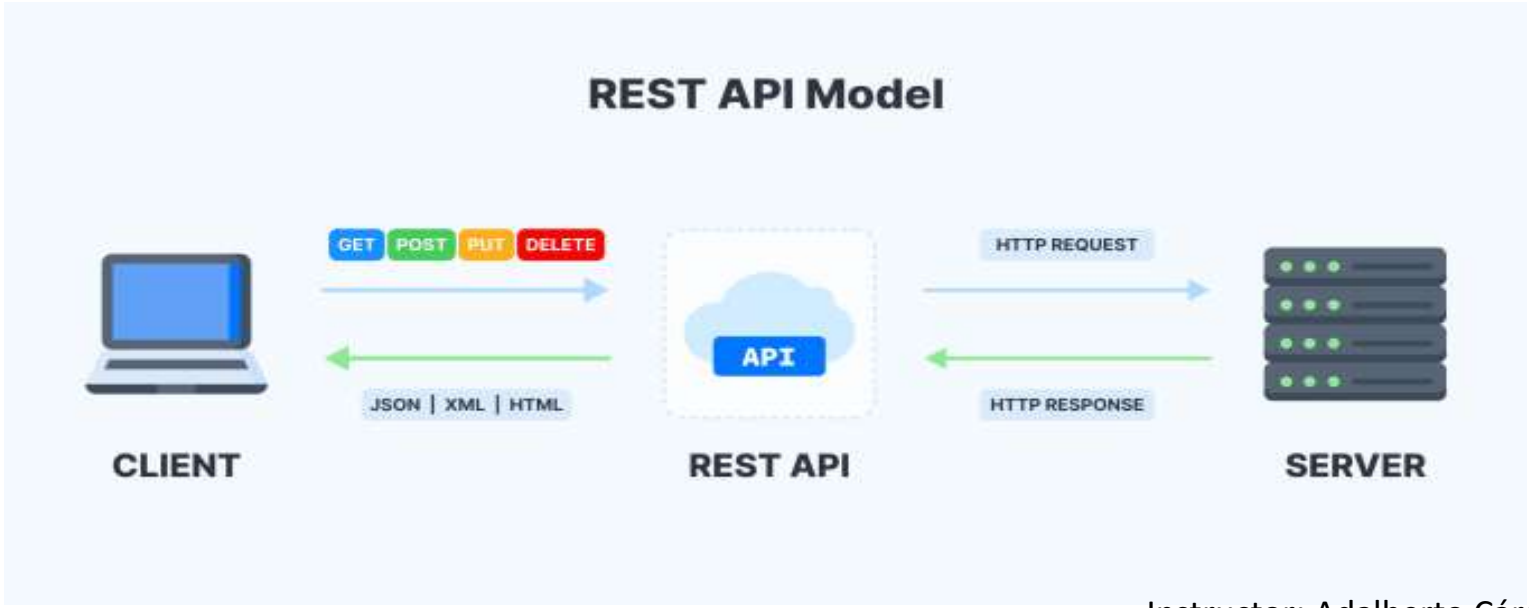
| | | |
|---|---|---|
| GET | → | Leer |
| POST | → | Crear |
| PUT | → | Modificar |
| PATCH | → | Modificar |
| DELETE | → | Eliminar |

Instructor: Adalberto Cárcamo A.




**Realizaremos un ejemplo práctico:**

API Rest  CRUD con el lenguaje de programación  Python Django




Instructor: Adalberto Cárcamo A.




1. Instalamos los paquetes necesarios

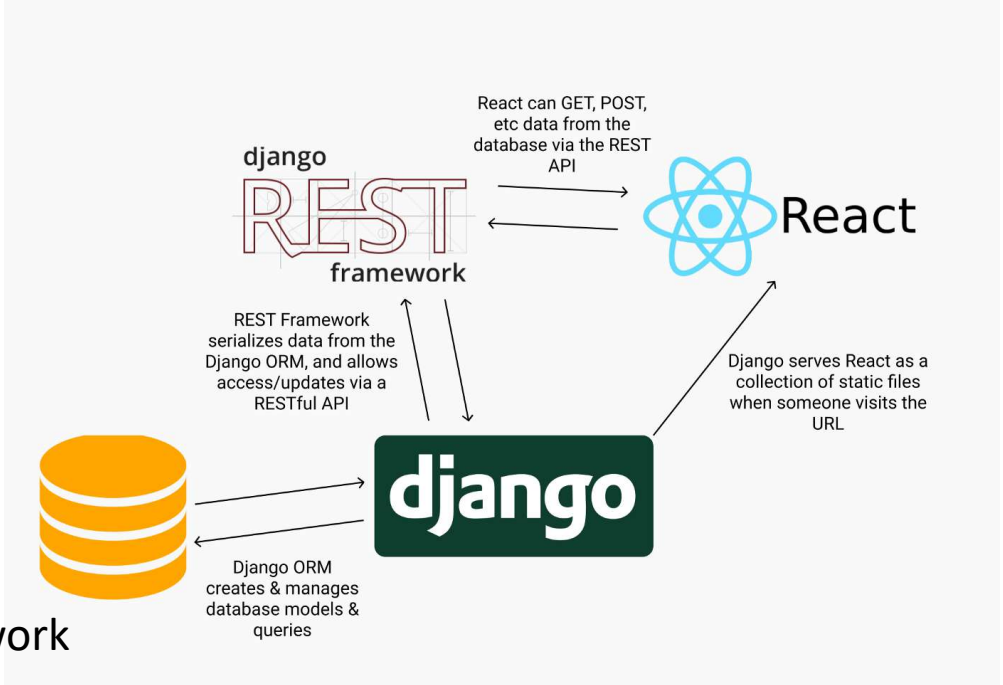
2. Creamos la carpeta del proyecto ejemplo: ApiDjango
3. Dentro de la carpeta creamos el entorno virtual **virtualenv venv**
4. Ahora antes de empezar a usarlo debemos activar el entorno virtual: venv\Scripts\actívate
5. Instalar Django ejecutando el siguiente comando : **pip install django**

Instructor: Adalberto Cárcamo A.










Ahora instalamos el Django Rest Framework

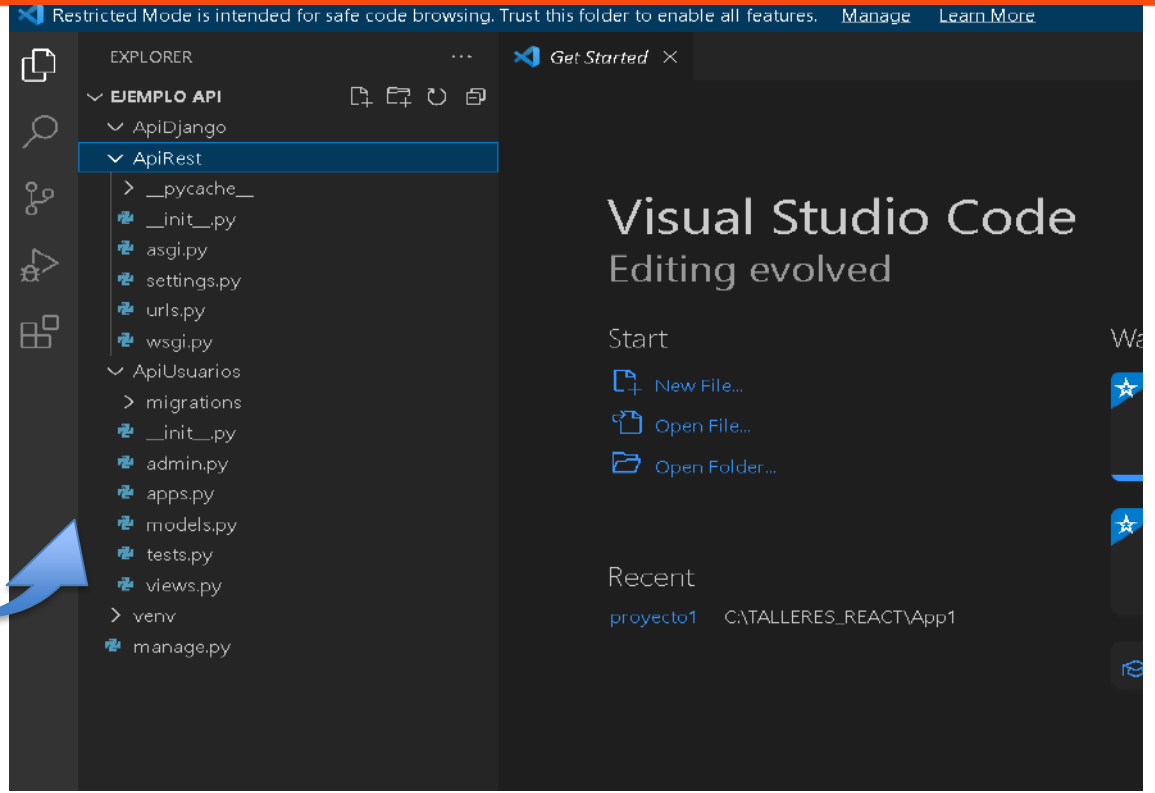
pip install djangorestframework

django REST framework

Creamos ahora el proyecto :
django-admin startproject  ApiRest .

Luego creamos nuestra aplicación:
Python manage.py startapp ApiUsuarios

La estructura de las carpetas nos queda d ela siguiente manera

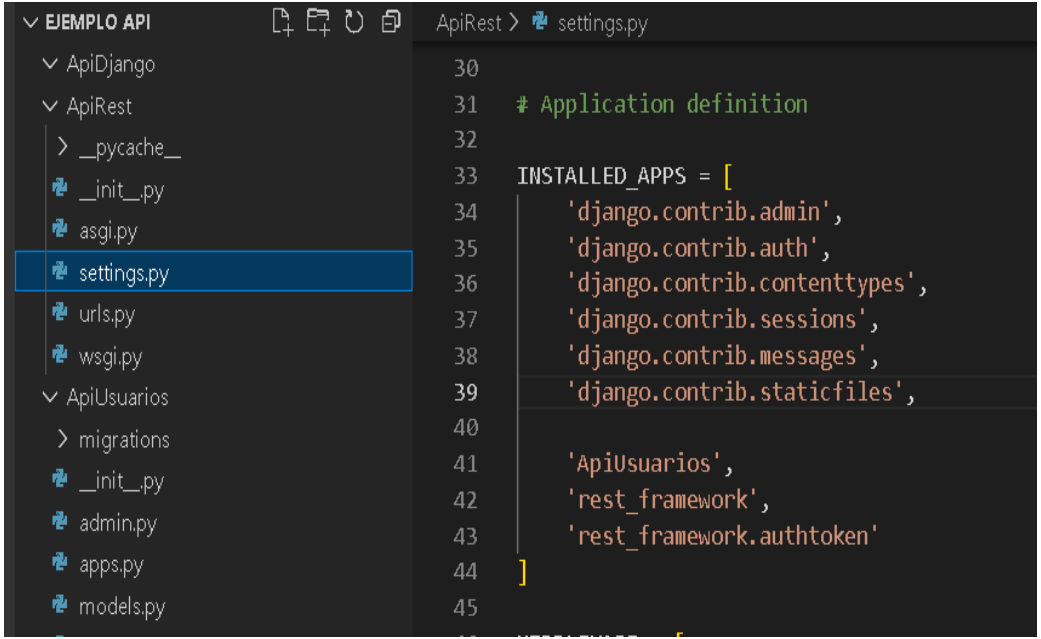


Importamos ahora la aplicación que creamos y el Djangorestframework

Ingresamos al settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    'ApiUsuarios',
    'rest_framework',
    'rest_framework.authtoken'
]
```




Instructor: Adalberto Cárcamo A.




Ahora vamos a crear el modelo: Importamos AbstracBaseUser esta es la clase base de usuarios y PermissionsMixin

```
from django.db import models
from django.contrib.auth.models import AbstracBaseUser
from django.contrib.auth.models import PermissionsMixin

"""permite manipular el modelo creado """
from django.contrib.auth.models import BaseUserManager


Para manipula los objetos de la clase Usuario
```

Creamos ahora la clase heredando de las clases importadas

```
class Usuario(AbstracBaseUser,PermissionsMixin):
```

django REST framework

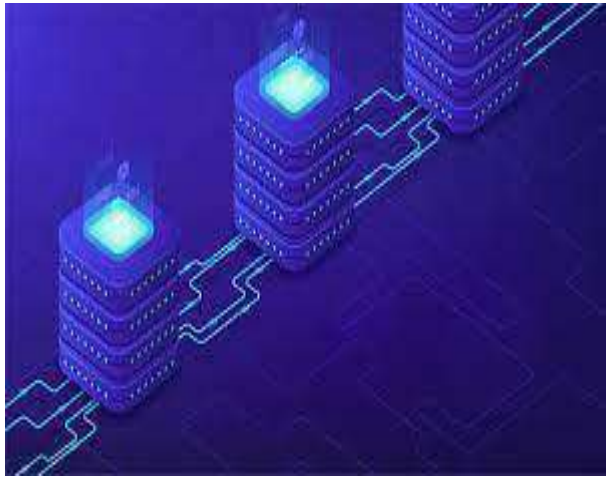
Creamos el modelo de la base de datos

```python
class Usuario(AbstracBaseUser,PermissionsMixin):
    email=models.EmailField(max_length=100,unique=True)
    nombre=models.CharField(max_length=100)
    activo=models.BooleanField(default=True)
    esdelequipo=models.BooleanField(default=False)

    """Objeto creado para realizar procesos con el modelo"""
    objects=UserProfileManager()
    """usar el email como validacion """
    USERNAME_FIELD='email'
    """nombre obligatorio o requerido"""
    REQUIRED_FIELDS=['nombre']

    def get_nombre(self):
      """obtener el nombre """
      return self.nombre

    def __str__(self):
      """retornar cadena representtiva del usuario"""
      return self.email
```



Instructor: Adalberto Cárcamo A.

django REST framework

```
class UsuarioManager(BaseUserManager):
    """para perfiles de usuario"""
    """creamos ahora las funciones para la crecion de usuarios"""
    def create_user(self,email,nombre,password=None):
        if not email:
            raise ValueError('Usuario debe tener un Email')
        """Normaliza el email ingresado convirtiendola en
minuscula"""
        email=self.normalize_email(email)
        user =self.model(email=email,nombre=nombre)
        user.set_password(password)
        user.save(using=self._db)

        return user

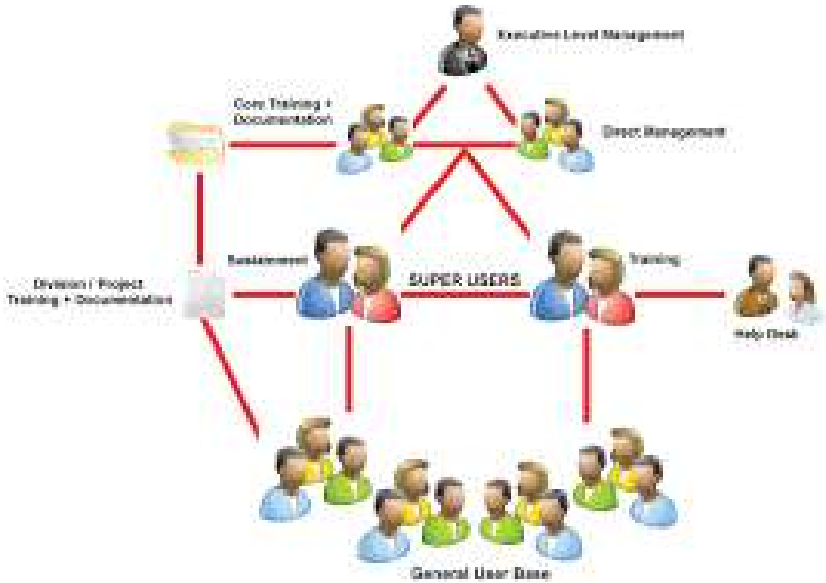
    """ creamos ahora el super Usuario """
    """este si debe tener pasword"""
    def SuperUsuario(self,email,nombre,password):

        user=self.create_user(email, nombre,password)

        user.is_superuser =True
        user.is_staff=True
        user.save(using=self._db)

        return user
```

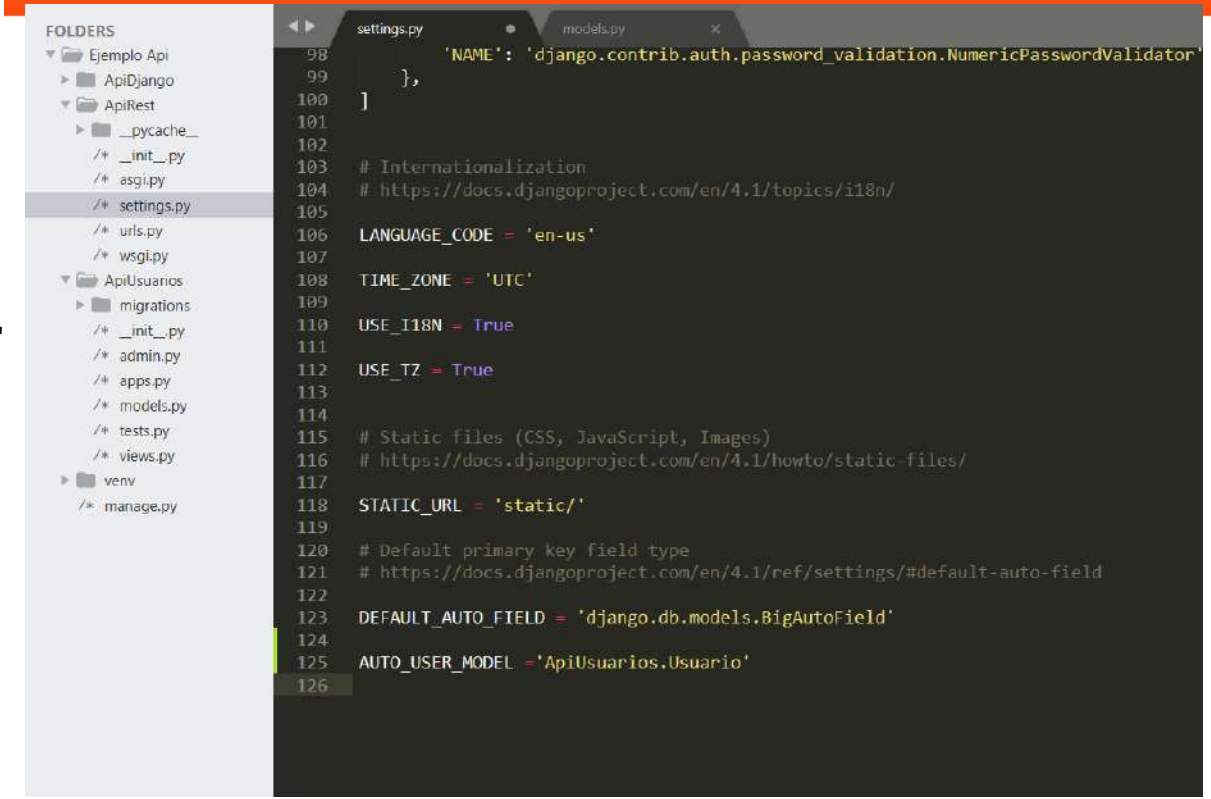
Ahora manipulamos el modelo creando las Funciones para el usuario y el super usuario




Instructor: Adalberto Cárcamo A.

Luego en el settings.py
Digitamos lo siguiente para que reconozca el modelo creado.

AUTO_USER_MODEL ='ApiUsuarios.Usuario'



Archivo de Migraciones Django la usa para configurar la base de datos junto con nuestros modelos
En el CMD digitamos :

```
python manage.py makemigrations ApiUsuarios
```

```
python manage.py migrate
```

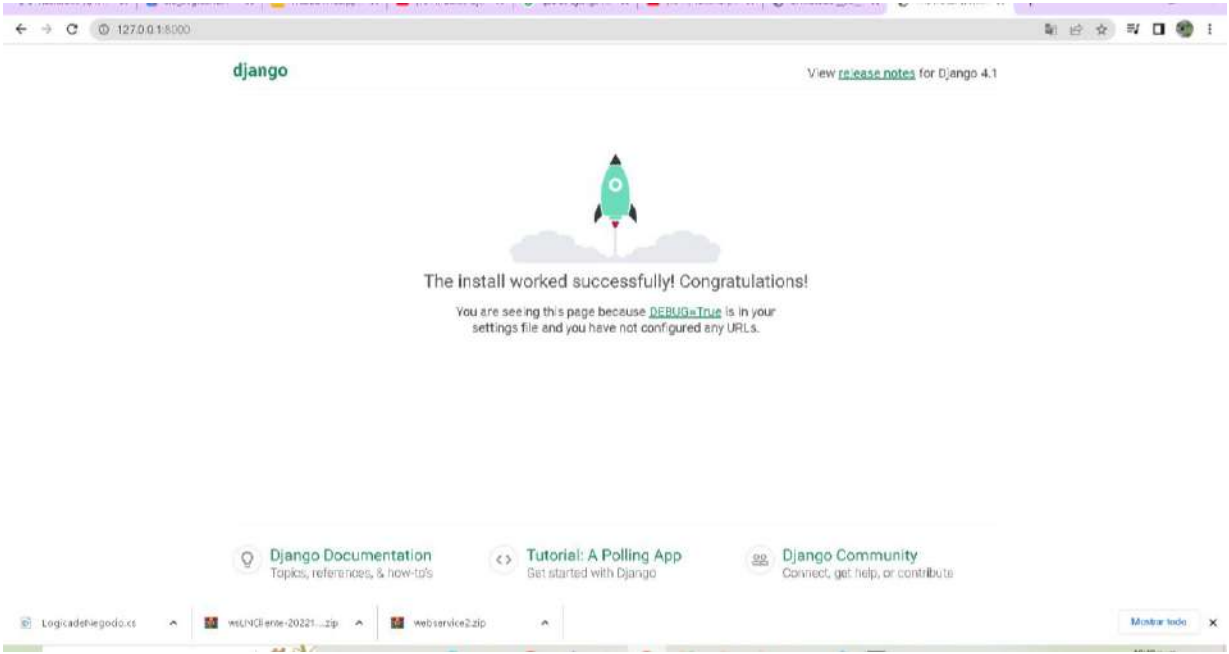
En el archivo Admin.py de proyecto digitamos:
.

```
from django.contrib import admin
from ApiUsuarios import models
admin.site.register(models.Usuario)
```

Con esto permite abrir nuestro administrador de django con el modelo que realizamos.
Luego enla terminal digitamos nuestro primer usuario:

```
python manage.py createsuperuser
```

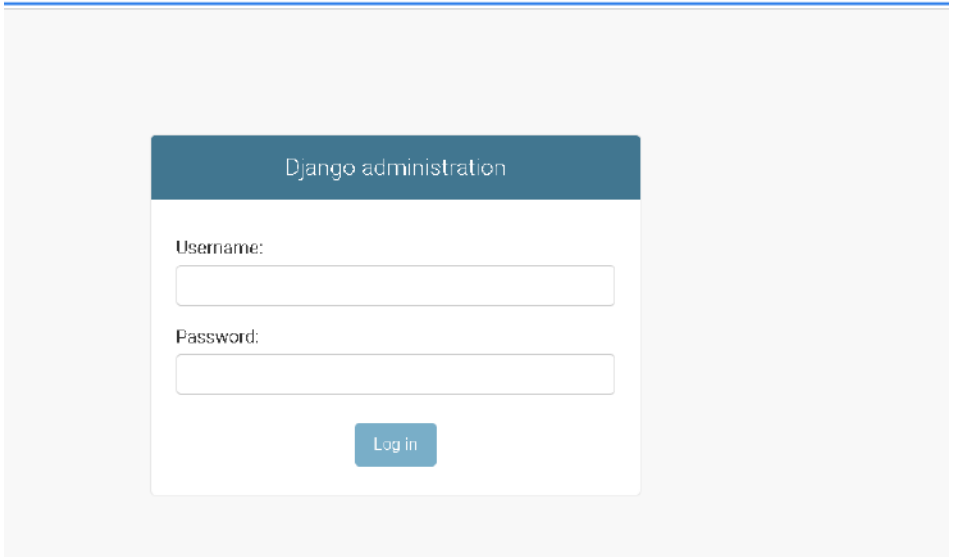
Ejecutamos el proyecto :
python manage.py runserver



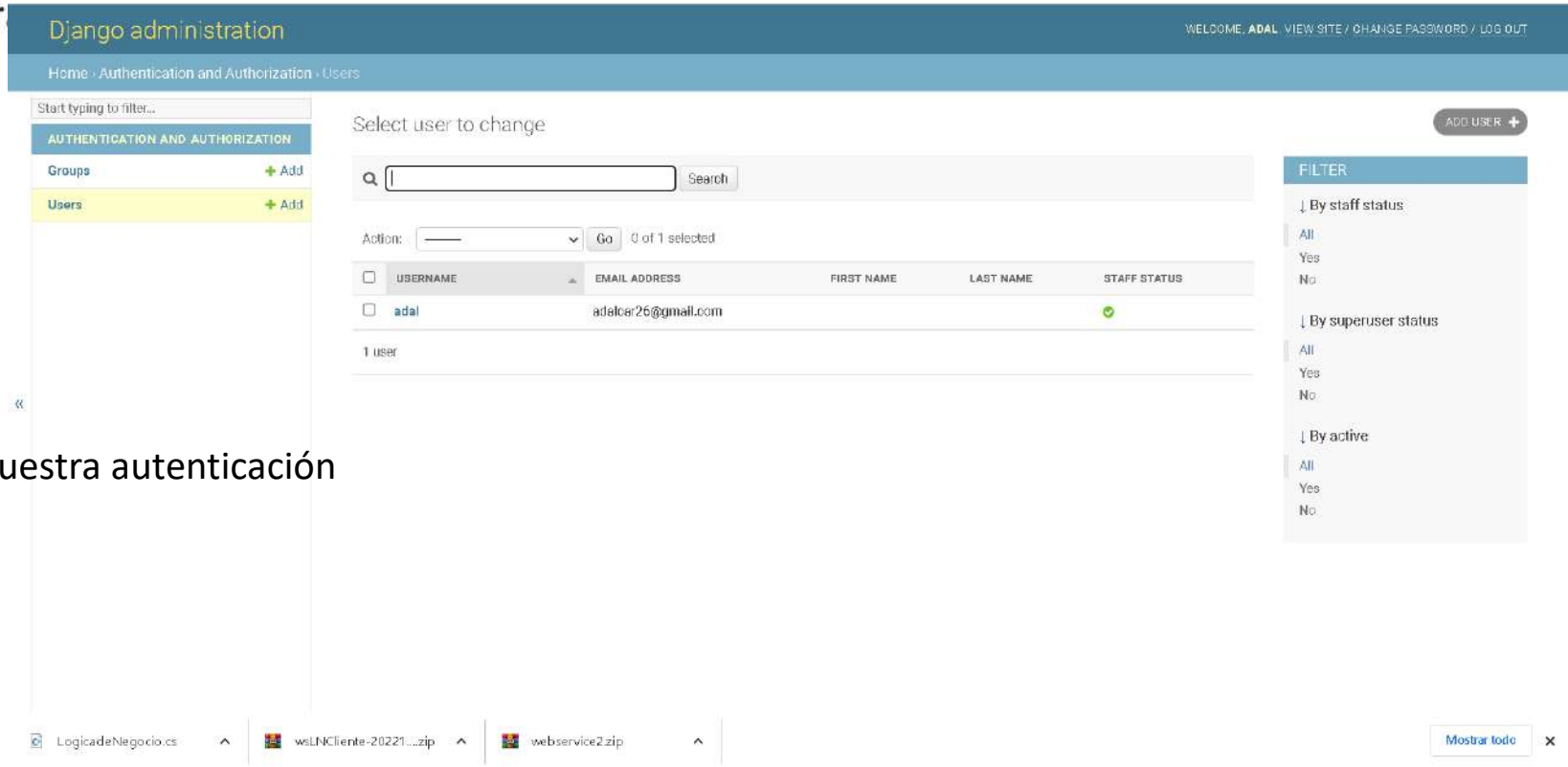

Instructor: Adalberto Cárcamo A.

En la URL ingresamos al administrador de Diango:

http://127.0.0.1:8000/admin



Digitamos nuestra autenticación






# GRACIAS

# Web Api - Rest

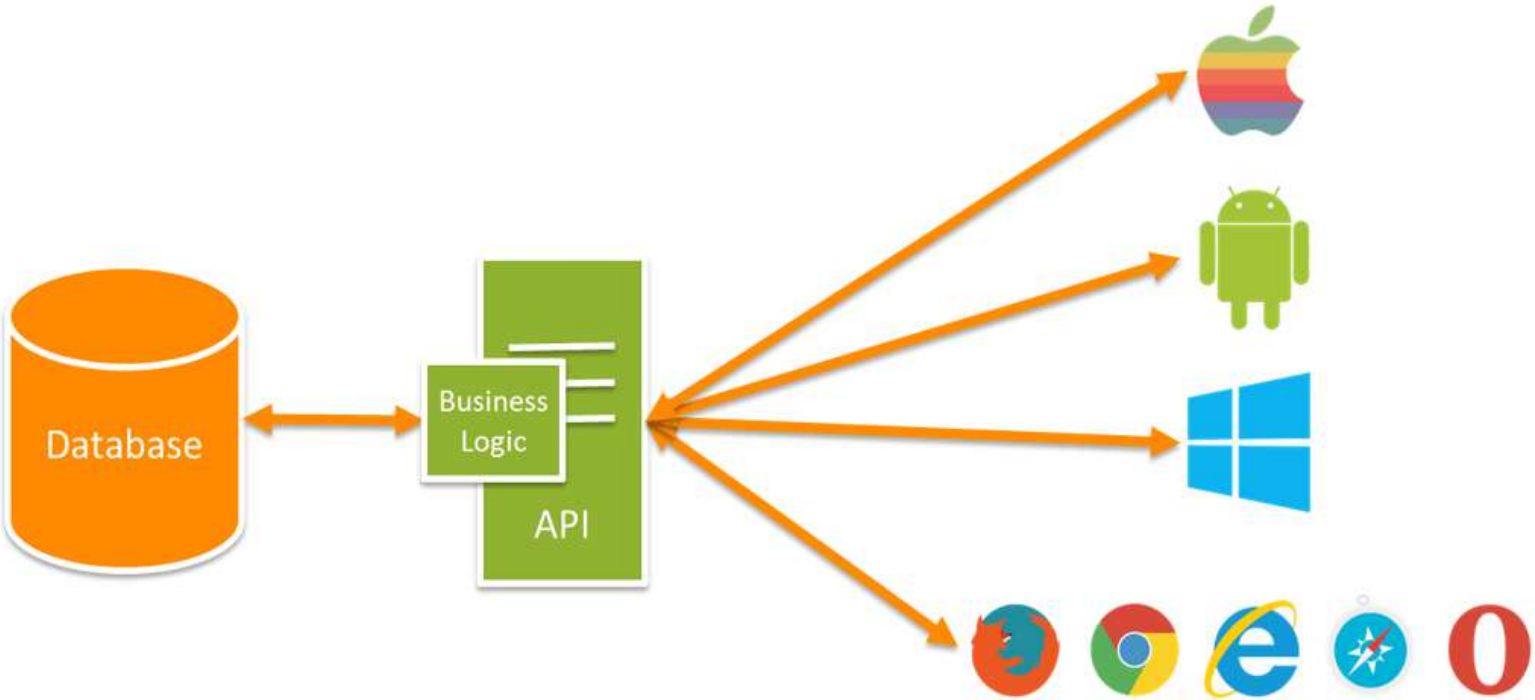








Instructor: Adalberto Cárcamo A.

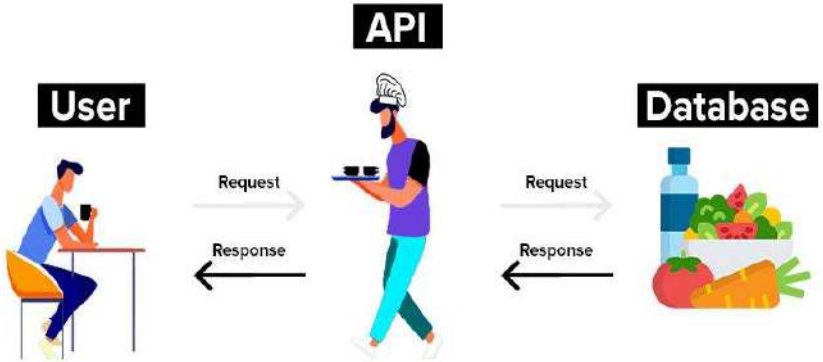

Database
Business Logic
API


API REST
DESARROLLADORES
DE FRONTEND
DESARROLLADORES
IOS/ANDROID
DESARROLLADORES
DE BACKEND




## Qué es una API

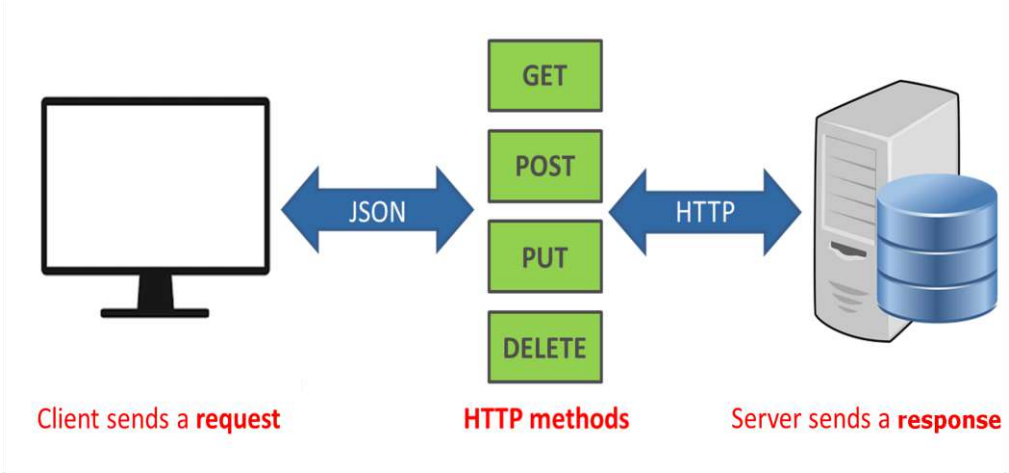
El término API es una abreviatura de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos aplicaciones de software a través de un conjunto de reglas.




Instructor: Adalberto Cárcamo A.




HTTP define métodos que son utilizados para diferentes circunstancias, por lo que trataremos de listar lo más relevantes y más utilizamos en la construcción de servicios REST, los métodos son los siguientes:




**GET**: Es utilizado únicamente para consultar información al servidor, muy parecidos a realizar un SELECT a la base de datos. No soporta el envío del payload

**POST**: Es utilizado para solicitar la creación de un nuevo registro, es decir, algo que no existía previamente, es decir, es equivalente a realizar un INSERT en la base de datos. Soporta el envío del payload.

**PUT**: Se utiliza para actualizar por completo un registro existente, es decir, es parecido a realizar un UPDATE a la base de datos. Soporta el envío del payload.

**PATCH**: Este método es similar al método PUT, pues permite actualizar un registro existente, sin embargo, este se utiliza cuando actualizar solo un fragmento del registro y no en su totalidad, es equivalente a realizar un UPDATE a la base de datos. Soporta el envío del payload

**DELETE**: Este método se utiliza para eliminar un registro existente, es similar a DELETE a la base de datos. No soporta el envío del payload.




# Métodos HTTP

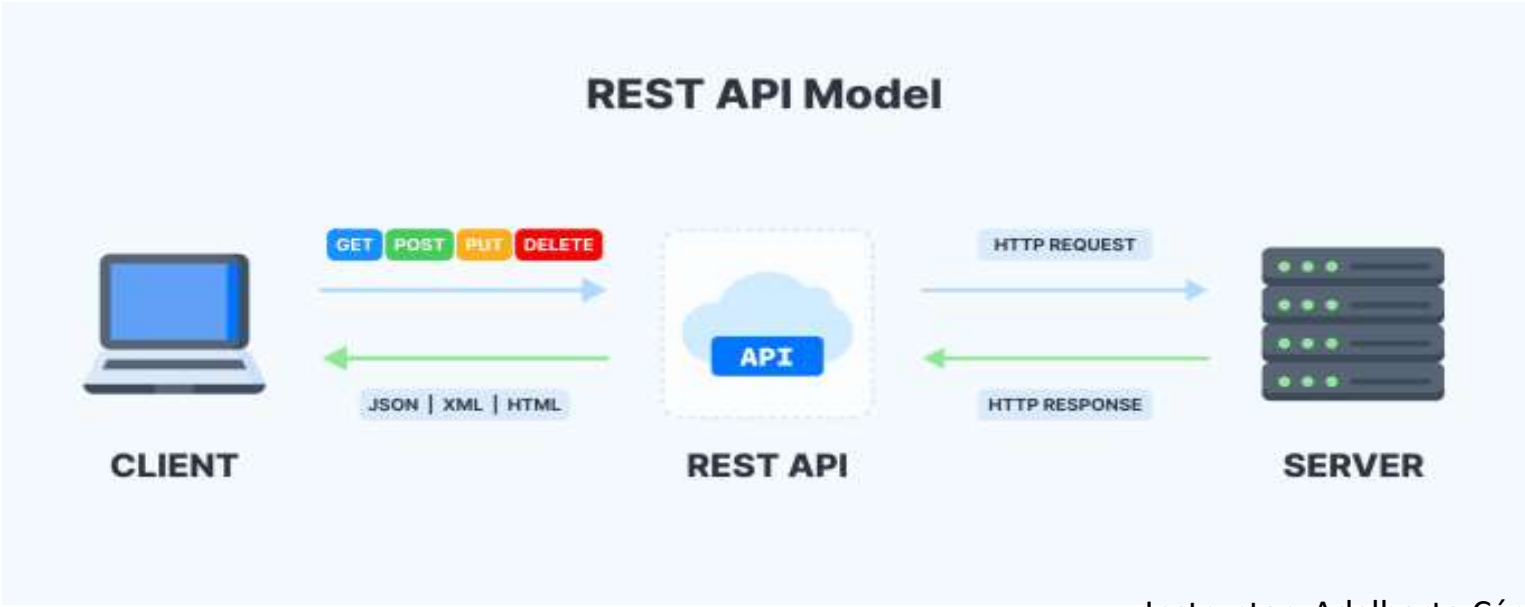
| | | |
|---|---|---|
| GET | → | Leer |
| POST | → | Crear |
| PUT | → | Modificar |
| PATCH | → | Modificar |
| DELETE | → | Eliminar |

Instructor: Adalberto Cárcamo A.




**Realizaremos un ejemplo práctico:**

API Rest  CRUD con el lenguaje de programación  Python Django




Instructor: Adalberto Cárcamo A.




1. Instalamos los paquetes necesarios

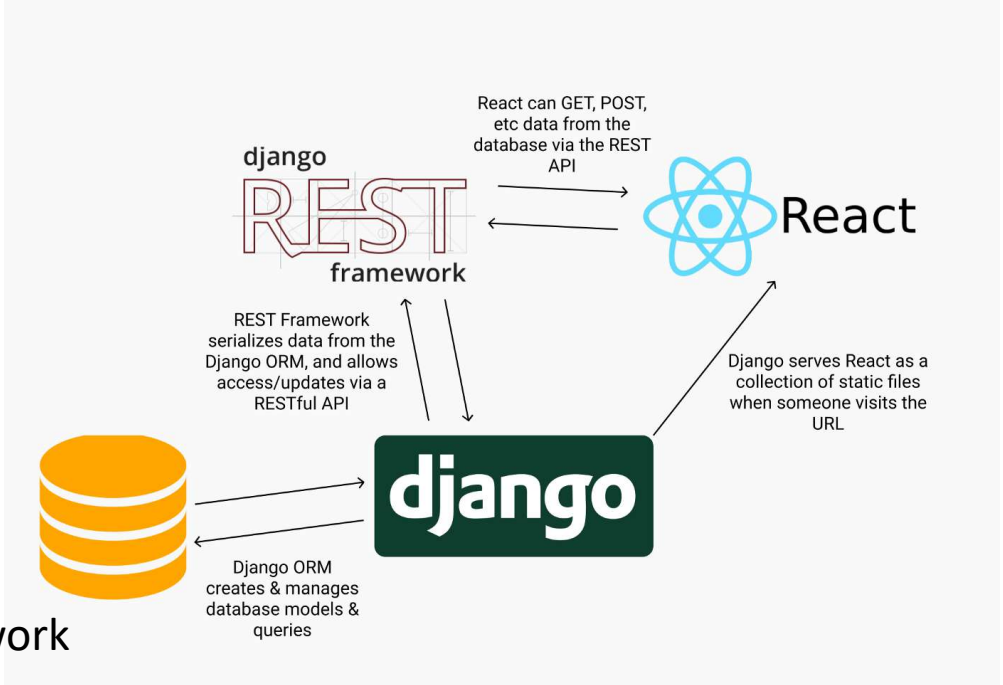
2. Creamos la carpeta del proyecto ejemplo: ApiDjango
3. Instalamos el VirtulaEnv **pip install virtualenv**
4. Dentro de la carpeta creamos el entorno virtual **virtualenv venv**
5. Ahora antes de empezar a usarlo debemos activar el entorno virtual: **venv\Scripts\activate**
6. Instalar Django ejecutando el siguiente comando : **pip install django**

Instructor: Adalberto Cárcamo A.










Ahora instalamos el Django Rest Framework

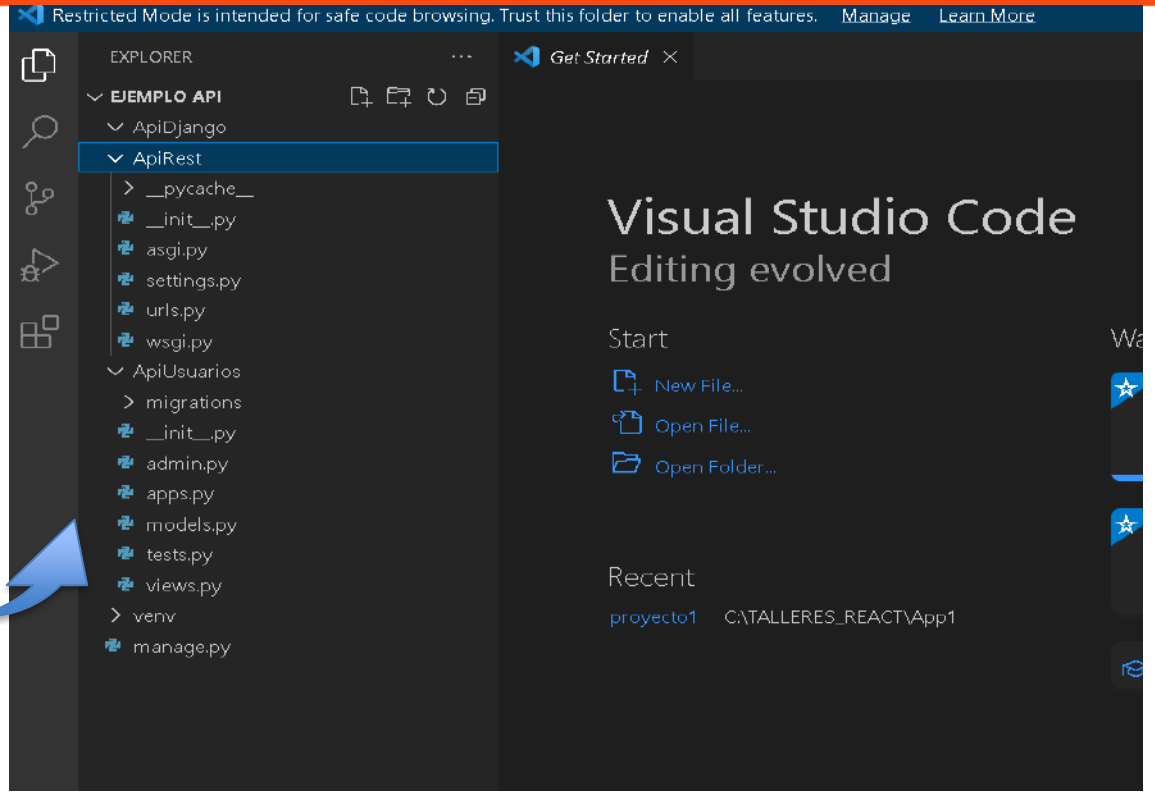
pip install  djangorestframework

Instructor: Adalberto Cárcamo A.

Creamos ahora el proyecto :
django-admin startproject  ApiRest .

Luego creamos nuestra aplicación:
Python manage.py startapp ApiUsuarios

La estructura de las carpetas nos queda
d ela siguiente manera



Importamos ahora la aplicación que creamos y el Djangorestframework

Ingresamos al settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    'appApi',
    'rest_framework',
]
```



```
ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'appApi',
    'rest_framework',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddl
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMi
    'django.contrib.messages.middleware.MessageMiddl
    'django.middleware.clickjacking.XFrameOptionsMid
]
```

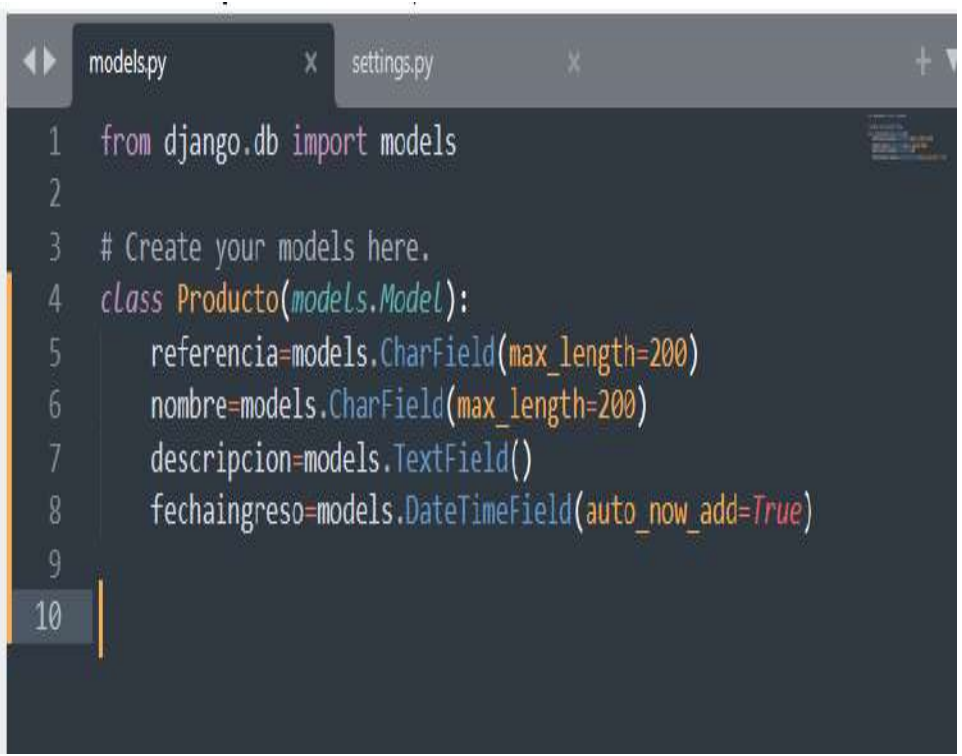
Instructor: Adalberto Cárcamo A.

Ahora vamos a crear el modelo:

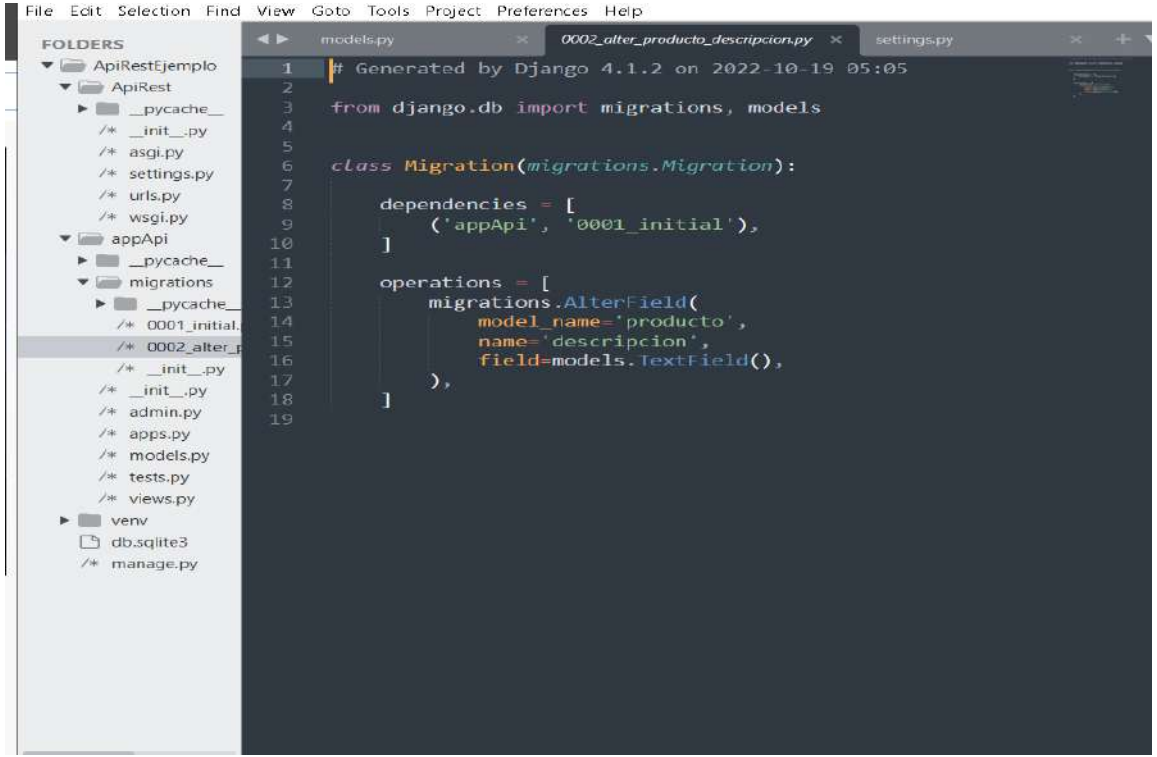
```
from django.db import models

# Create your models here.

class Producto(models.Model):
        referencia=models.CharField(max_length=200)
        nombre=models.CharField(max_length=200)
        descripcion=models.TextField()
        fechaingreso=models.DateTimeField(auto_now_add=True)
```



Nos genera la siguiente migración en la carpeta **migrations**



Archivo de Migraciones Django la usa para configurar la base de datos junto con nuestros modelos
En el CMD  digitamos :

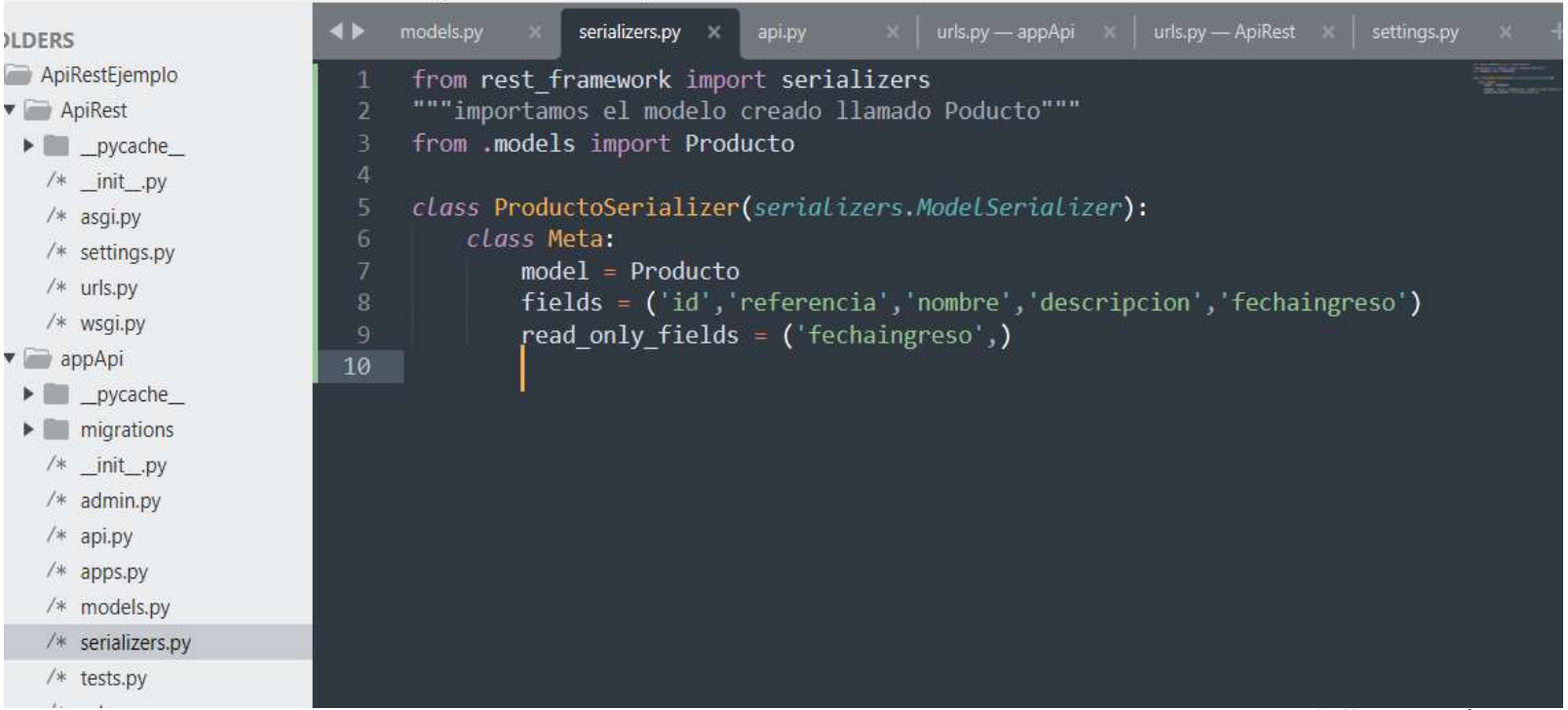
```
python manage.py makemigrations
```

```
python manage.py migrate
```

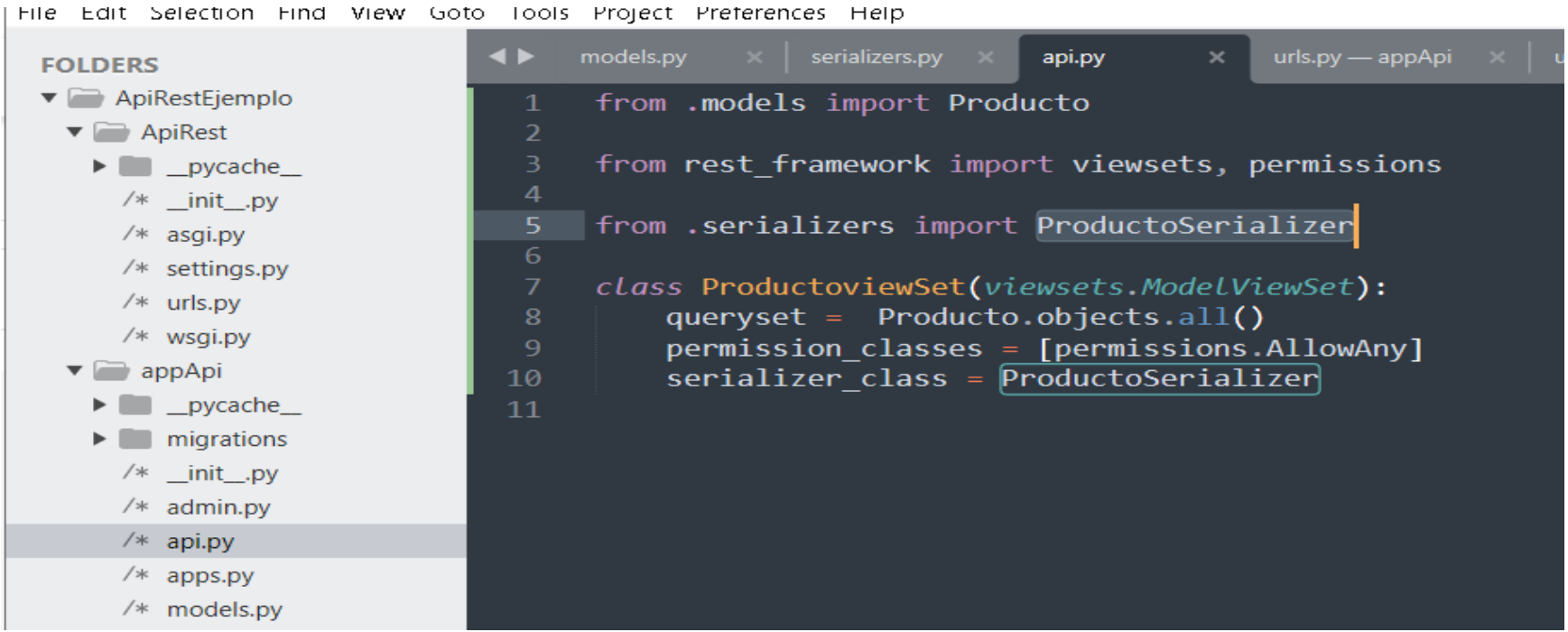


django REST framework

Creamos un archivo llamado serializers.py en la aplicación

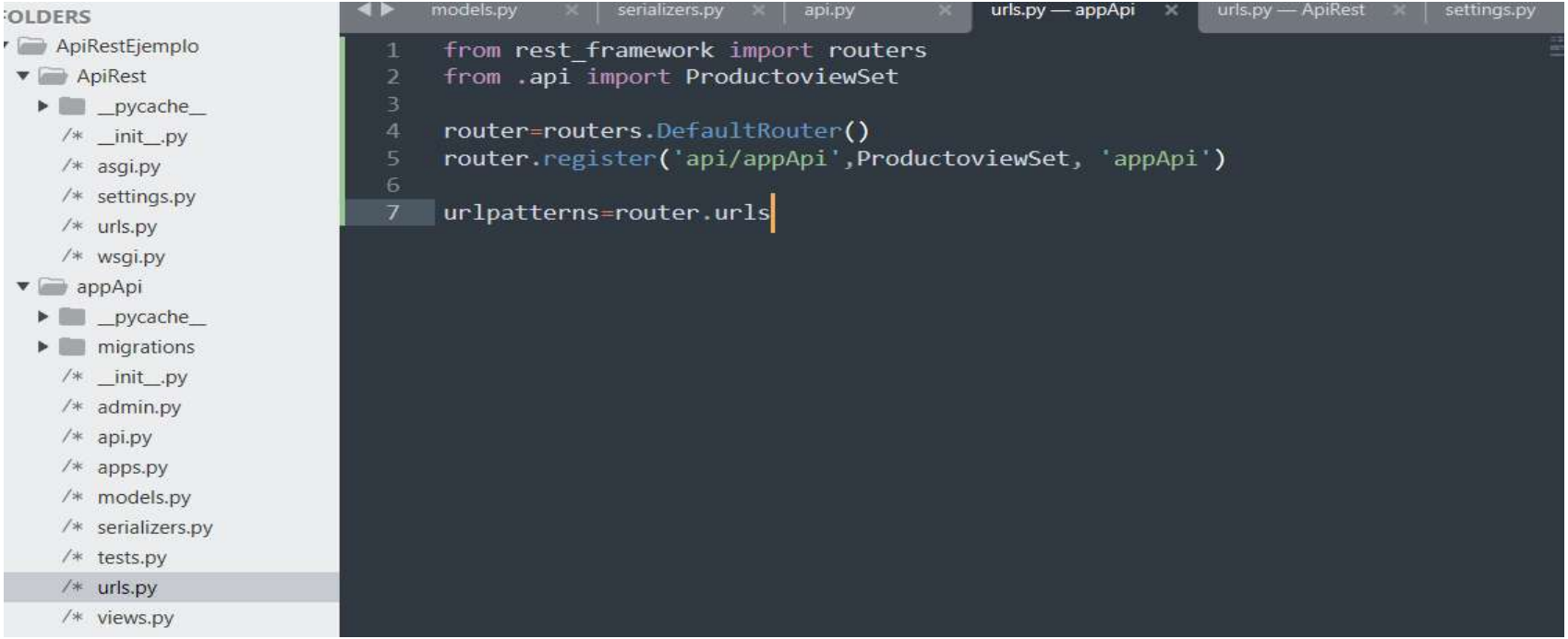


# Creamos la api.py en la aplicación



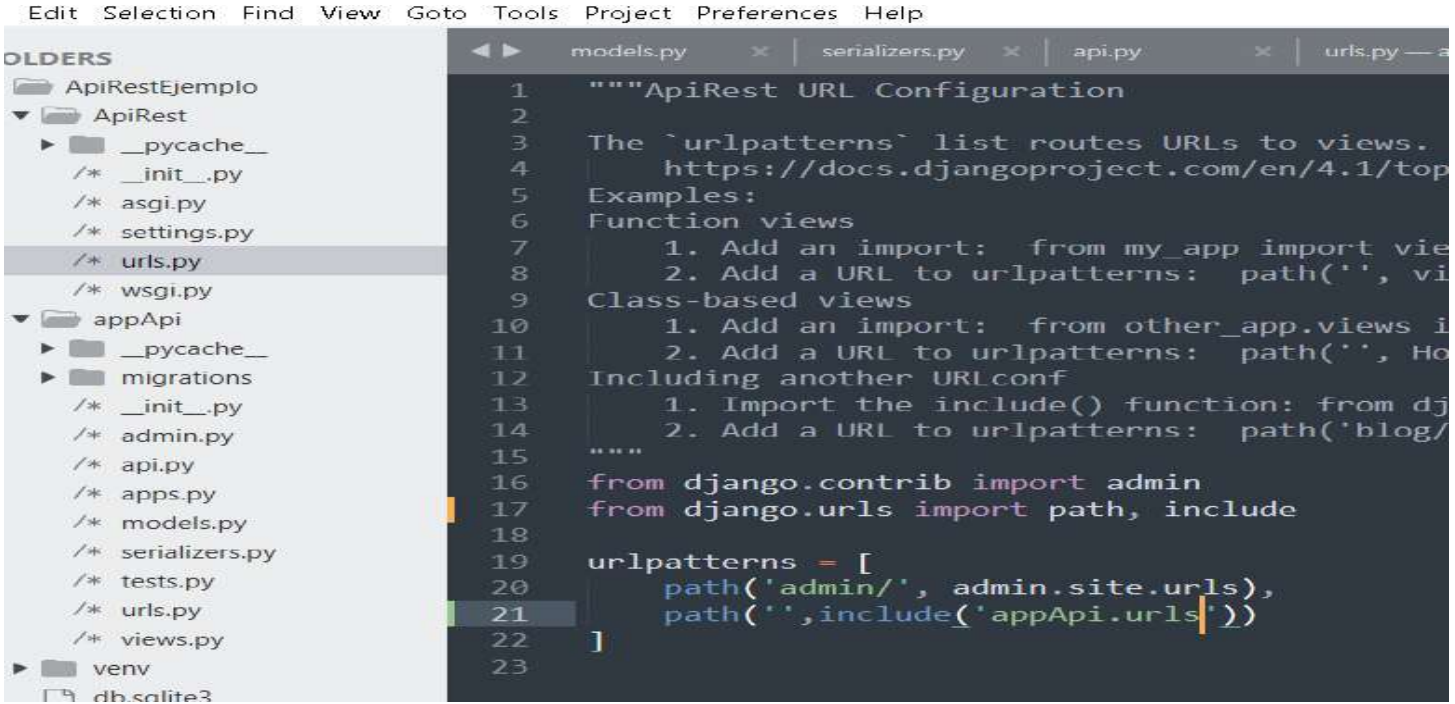
django REST framework

## Creamos el archivo urls.py en la aplicación



django REST framework

## En la urls.py del proyecto




Instructor: Adalberto Cárcamo A.




Django REST framework

Api Root

# Api Root

OPTIONS GET

The default basic root view for DefaultRouter

```
GET /
```

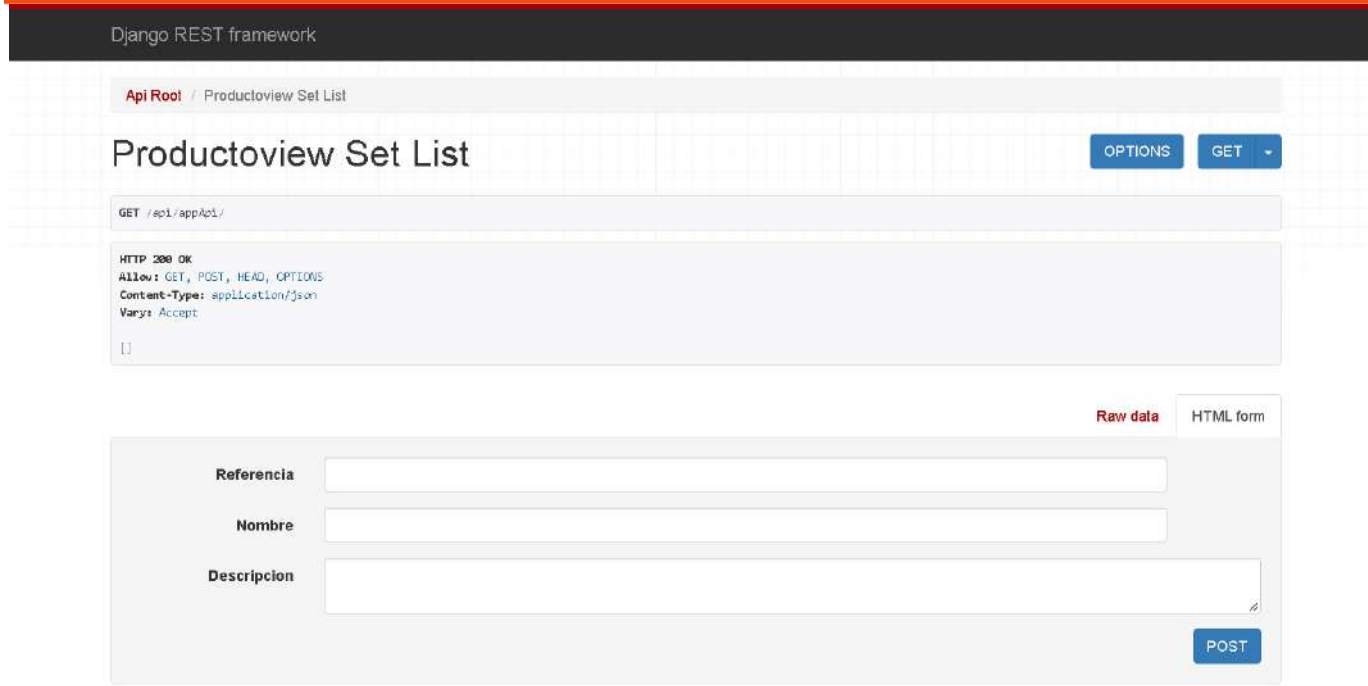
```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "api/appApi": "http://127.0.0.1:8000/api/appApi/"
}
```

Ejecutamos el proyecto y nos muestra la ruta generada

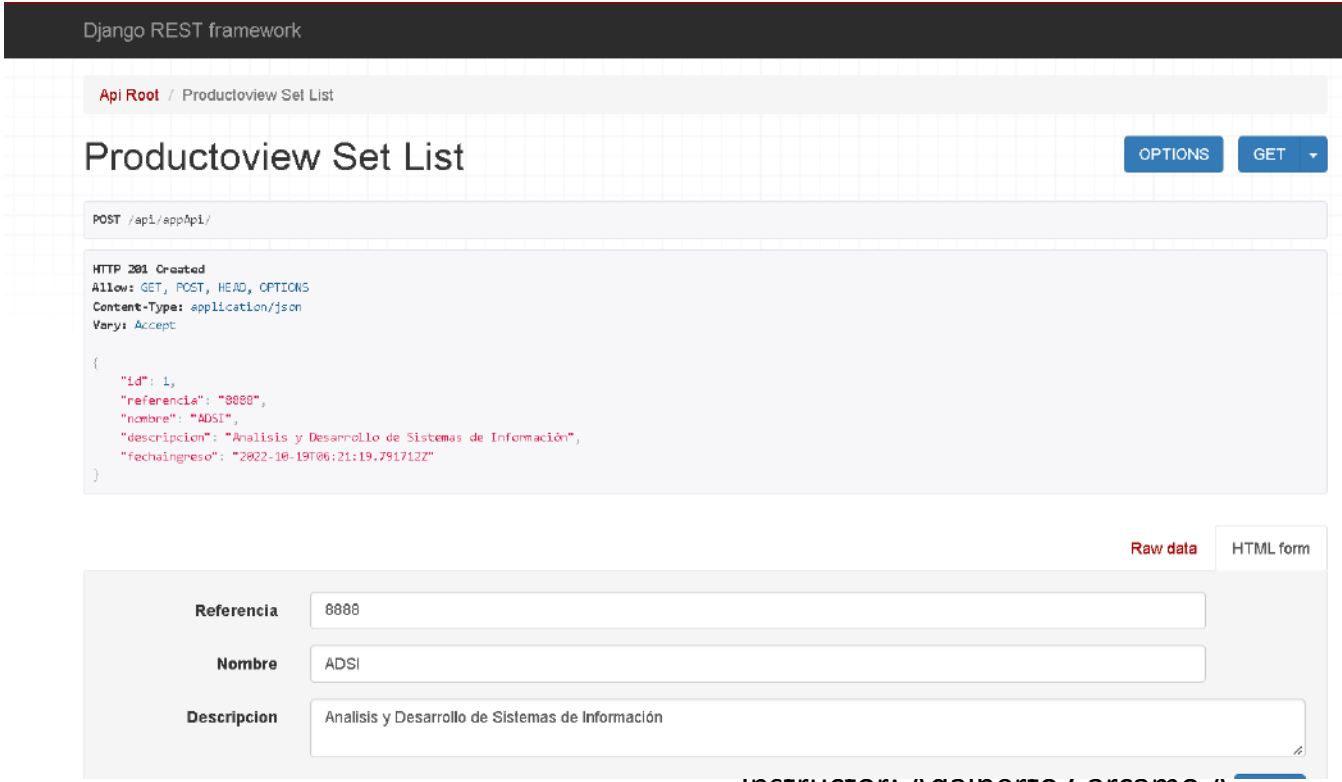
Asignamos la ruta de nuestra api

http://127.0.0.1:8000/api/appApi/




Instructor: Adalberto Cárcamo A.

Verificamos ingresando Información usando PUT



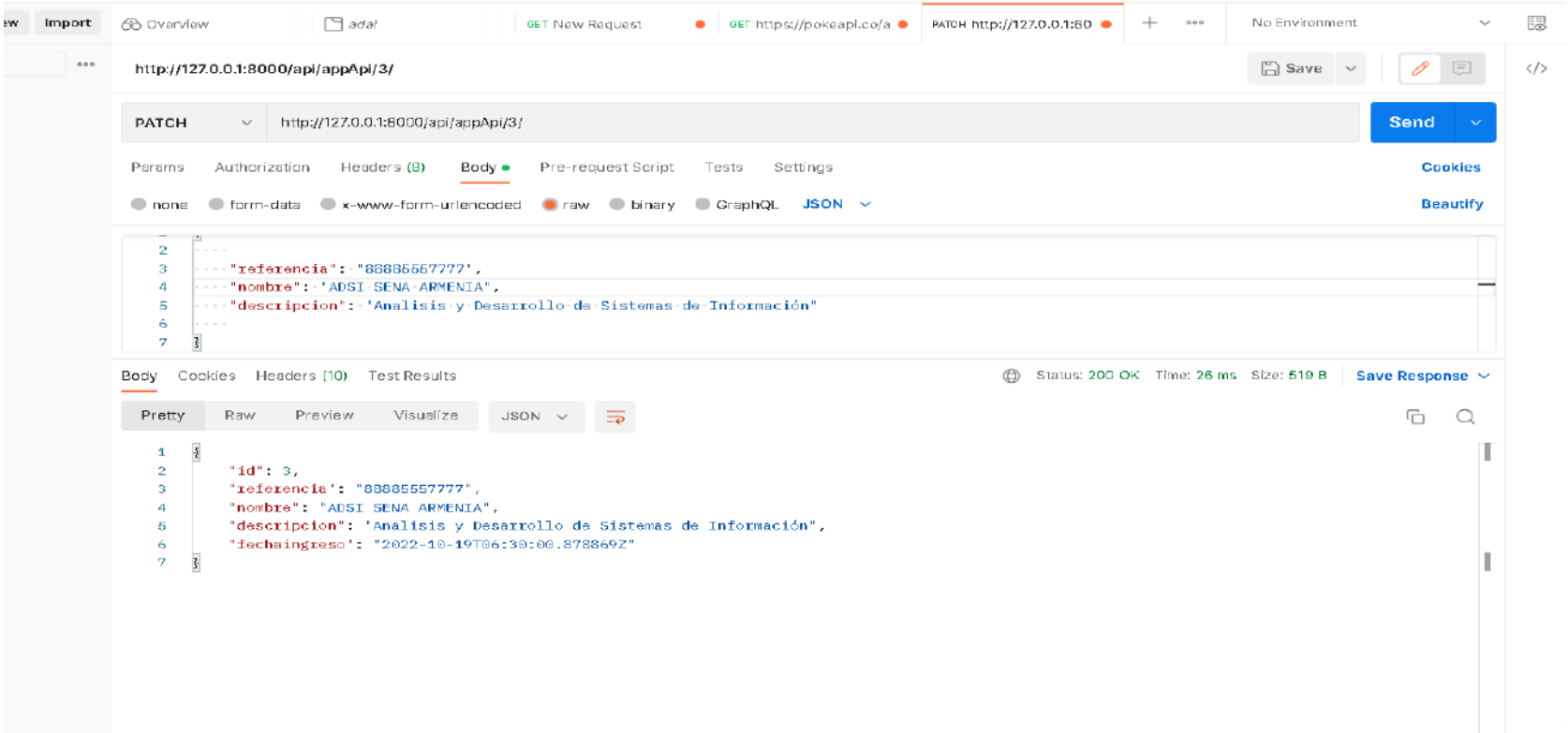
Utilizando el Postman realizaremos las pruebas



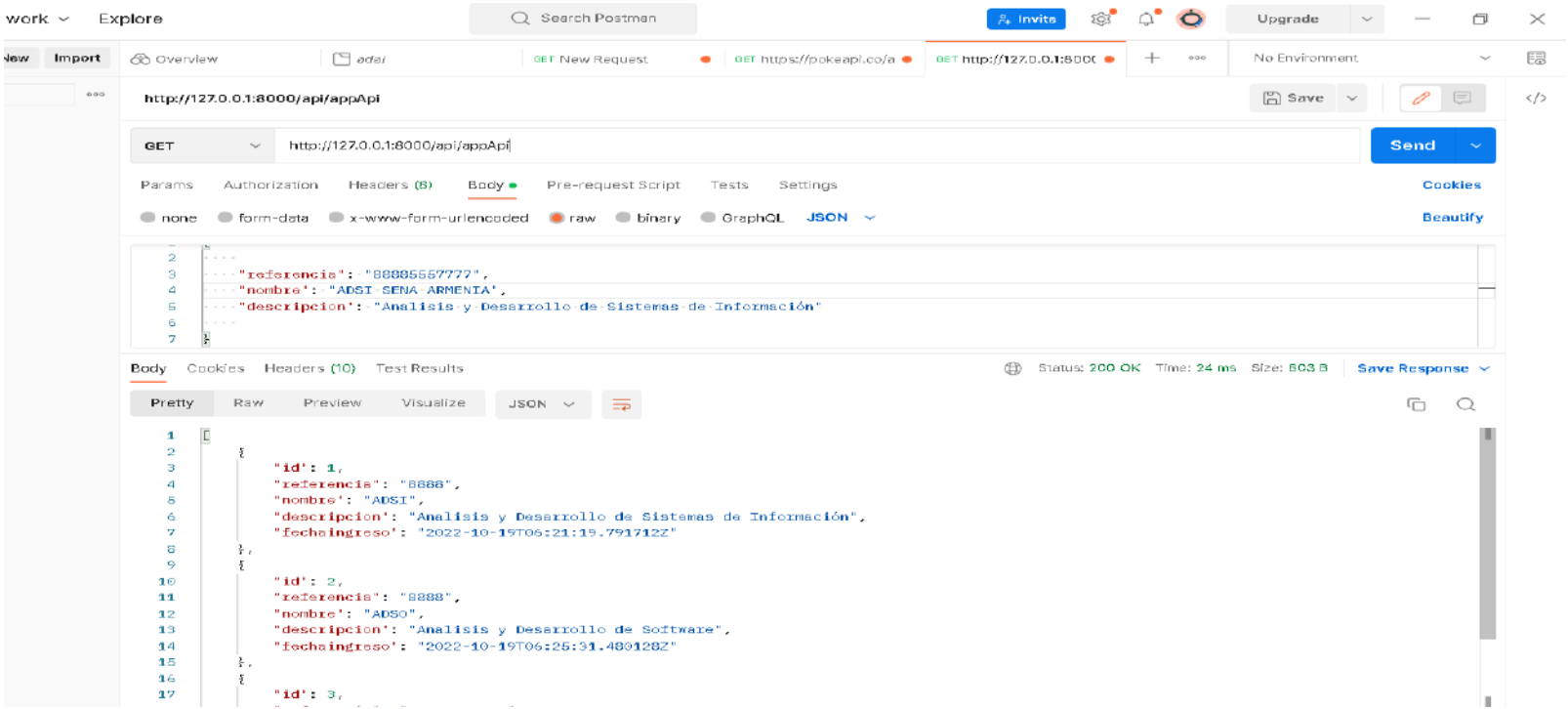
POSTMAN

Instructor: Adalberto Cárcamo A.

# django REST framework



# django REST framework




Instructor: Adalberto Cárcamo A.




# GRACIAS



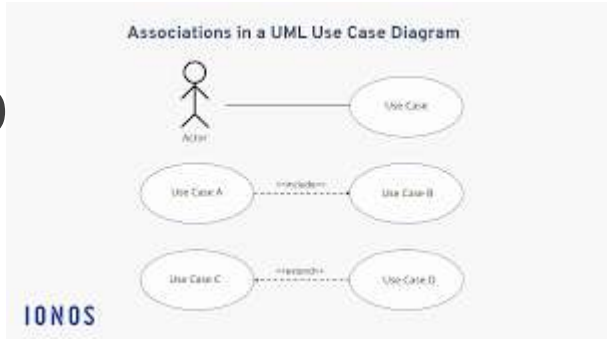

# UML

## Lenguaje Unificado de Modelado

## Casos de Uso

GC-F-004v.04

# Diagrama Casos de Uso



# CASOS DE USO

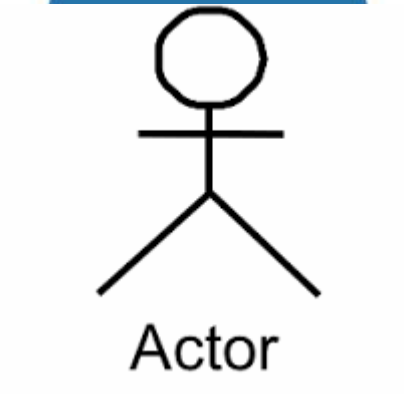
El diagrama de casos de uso es una forma de diagrama de comportamiento en lenguaje de modelado unificado (UML, del inglés *Unified Modelling Language*), con la que se representan procesos empresariales, así como sistemas y procesos de programación orientada a objetos. Por lo tanto, UML no es un lenguaje de programación, sino un lenguaje de modelado, es decir, un método estandarizado para representar sistemas planificados o ya existentes. En este diagrama, todos los objetos involucrados se estructuran y se relacionan entre sí.



# Diagrama de casos de uso

En el diagrama de casos de uso, las **funciones del sistema** en cuestión se representan desde el **punto de vista del usuario** (llamado “actor” en UML). Este actor no tiene que ser necesariamente un usuario humano, sino que el rol también puede atribuirse a un sistema externo que accede a otro sistema. De este modo, el diagrama de casos de uso muestra la **relación entre un actor y sus requisitos o expectativas del sistema**, sin representar las acciones que tienen lugar o ponerlas en un orden lógico..





# Diagrama casos de Uso

En la práctica, esta estructura es adecuada para representar claramente las funciones y/o objetivos más importantes de un sistema. Por esta razón, a la hora de **desarrollar un software o planificar nuevos procesos empresariales**, crear un diagrama de casos de uso suele ser uno de los primeros pasos, ya que permite visualizar clara y fácilmente qué casos de uso deben tenerse en cuenta durante el desarrollo para que los actores (y, en un sentido más amplio, también los operadores o clientes) logren su objetivo, en principio independientemente de la viabilidad técnica.

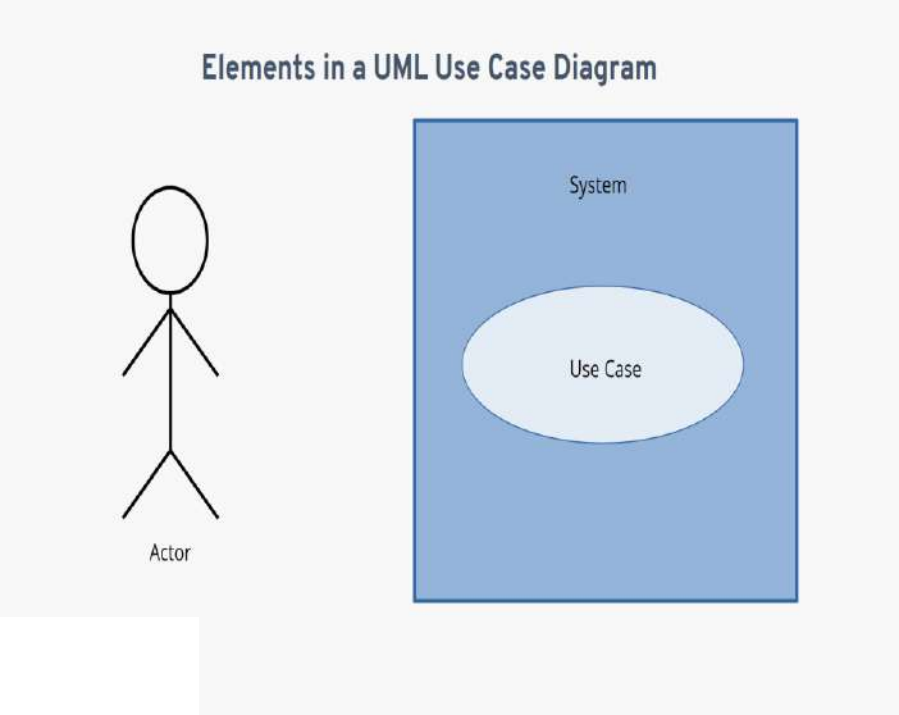




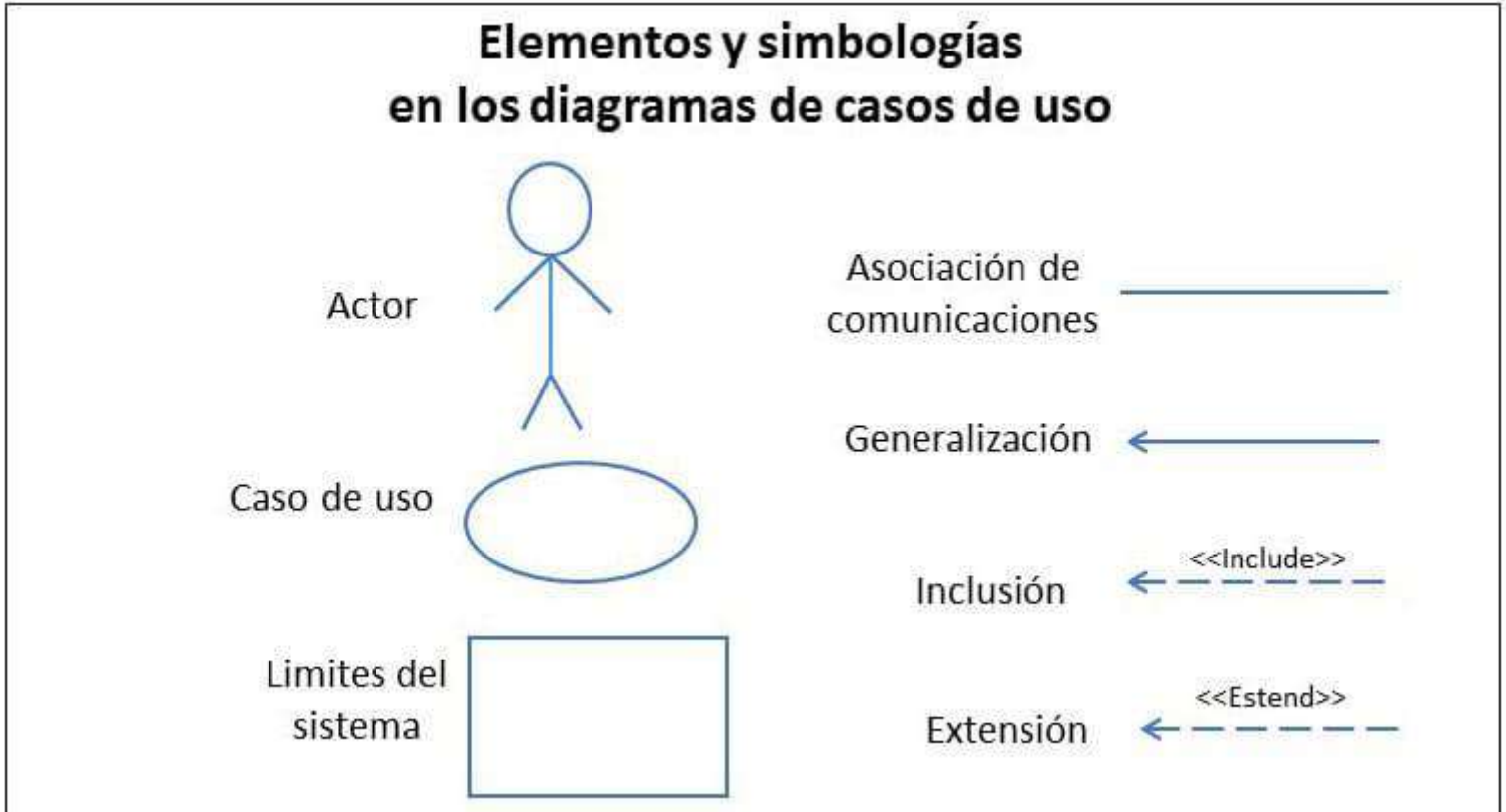
# Elementos y estructura del diagrama de casos de uso

Para garantizar que el diagrama de casos de uso se comprensible para todo el mundo de un vistazo, se utiliza elementos estandarizados para elaborarlo. En prim lugar, hay tres elementos principales:

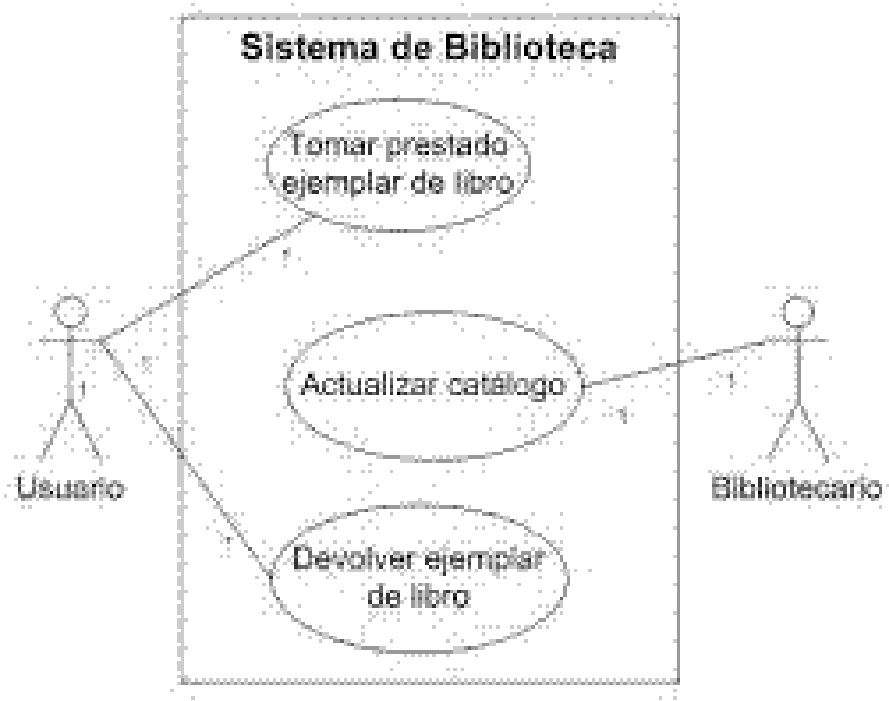
- **Actor:** tanto si es una persona, como un sistema, se representa con el dibujo de una figura humana esquemática.
- **Sistema:** el sistema al que se refiere el caso de uso tiene forma de rectángulo.
- **Caso de uso:** se muestra como una elipse que suele incluir un texto describiendo brevemente el proceso.



# Elementos del diagrama de casos de Uso



# Ejemplo


Sistema de Biblioteca
Tomar prestado ejemplar de libro
Actualizar catálogo
Devolver ejemplar de libro
Usuario
Bibliotecario




# GRACIAS

**Clase conexión:**

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Data;
using System.Linq;
using System.Web;

namespace Ejemplo_Api2.Data
{

    public class ConexionBD
    {
        #region "Atributos"
        private string strError;
        private bool blnBDAbierta;
        private string strCadenaCnx;
        private SqlConnection objCnnBD;
        private SqlCommand objCmdBD;
        private SqlDataReader objReader;
        private SqlDataAdapter dapGenerico; //Intermediario para llenar el dataset
        private DataSet dts;
        private string strVrUnico;
        private SqlParameter objParametro;
        #endregion
        #region "Constructor"
        public ConexionBD()
        {
            objCnnBD = new SqlConnection();
            objCmdBD = new SqlCommand();
            dapGenerico = new SqlDataAdapter();
            strVrUnico = "";
            objParametro = new SqlParameter();
            strError = "";
        }
        #endregion
        #region "Propiedades"
        public SqlDataReader Reader
        {
            get { return objReader; }
        }
        public DataSet DataSet_Retornado
        {
            get { return dts; }
        }
        public string Error
        {
            set { strError = value; }
            get { return strError; }
        }
        public string ValorUnico
        {
            get { return strVrUnico; }
        }
        #endregion
        #region "Metodos Privados"

        private bool AbrirConexion()
        {

            try
            {
                strCadenaCnx = "Data Source=DESKTOP-UGSFCI7\\SQLEXPRESS;Initial
Catalog=DB_API_DATOS;Integrated Security=True";
                objCnnBD.ConnectionString = strCadenaCnx;
                objCnnBD.Open();
                blnBDAbierta = true;
                return true;
```

```
        }
        catch (Exception ex)
        {
            blnBDAbierta = false;
            strError = "Error al abrir la conexion -" + ex.Message;
            return false;
        }

    }
    #endregion
    #region "Metodos Publicos"
    public bool Consultar(string SentenciaSQL, bool blnCon_Parametros)
    {
        if (SentenciaSQL == "")
        {
            strError = "Error en instrucción SQL";
            return false;
        }
        if (blnBDAbierta == false)
        {
            if (AbrirConexion() == false)
            {
                return false;
            }
        }
        objCmdBD.Connection = objCnnBD;
        if (blnCon_Parametros)
            objCmdBD.CommandType = CommandType.StoredProcedure;
        else
            objCmdBD.CommandType = CommandType.Text;
        objCmdBD.CommandText = SentenciaSQL;
        try
        {
            objReader = objCmdBD.ExecuteReader();
            return true;
        }
        catch (Exception ex)
        {
            strError = "Falla en ejecutar comando -" + ex.Message;
            return false;
        }
    }
    public bool EjecutarSentencia(string SentenciaSQL, bool blnCon_Parametros)
    {
        if (SentenciaSQL == "")
        {
            strError = "No se ha definido la sentencia a ejecutar ";
            return false;
        }
        if (blnBDAbierta == false)
        {
            if (AbrirConexion() == false)
            {
                return false;
            }
        }
        objCmdBD.Connection = objCnnBD;
        if (blnCon_Parametros)
            objCmdBD.CommandType = CommandType.StoredProcedure;
        else
            objCmdBD.CommandType = CommandType.Text;
        objCmdBD.CommandText = SentenciaSQL;
        try
        {
            objCmdBD.ExecuteNonQuery();
            return true;
        }
        catch (Exception ex)
        {
```

```
                strError = "Error al ejecutar la instrucción -" + ex.Message;
                return false;
            }
        }
        public bool ConsultarValorUnico(string SentenciaSQL, bool blnCon_Parametros)
        {
            if (SentenciaSQL == "")
            {
                strError = "No se ha definido la sentencia a ejecutar ";
                return false;
            }
            if (blnBDAbierta == false)
            {
                if (AbrirConexion() == false)
                {
                    return false;
                }
            }
            objCmdBD.Connection = objCnnBD;
            if (blnCon_Parametros)
                objCmdBD.CommandType = CommandType.StoredProcedure;
            else
                objCmdBD.CommandType = CommandType.Text;
            objCmdBD.CommandText = SentenciaSQL;
            try
            {
                strVrUnico = Convert.ToString(objCmdBD.ExecuteScalar());
                return true;
            }
            catch (Exception ex)
            {
                strError = "Error al ejecutar instrucción -" + ex.Message;
                return false;
            }
        }
        public void CerrarConexion()
        {
            try
            {
                objCmdBD = null; //Destruye objeto Command
            }
            catch (Exception ex)
            {
                strError = "Falla en cerrar Command -" + ex.Message;
            }
            try
            {
                objCnnBD.Close(); //Cierra la conexión
                objCnnBD = null; //Destruye en la emmoria
            }
            catch (Exception ex)
            {
                strError = "Falla en cerrar conexión -" + ex.Message;
            }
        }
        public bool LlenarDataSet(string NombreTabla, string SentenciaSQL, bool
blnCon_Parametros)
        {
            // con este booleano se sabe si hay conexion abierta o no
            if (blnBDAbierta == false)
            {
                if (AbrirConexion() == false)
                {
                    return false;
                }
            }
            objCmdBD.Connection = objCnnBD;
            if (blnCon_Parametros)
                objCmdBD.CommandType = CommandType.StoredProcedure;
```

```
            else
                objCmdBD.CommandType = CommandType.Text;
            objCmdBD.CommandText = SentenciaSQL;
            try
            {
                dts = new DataSet();
                dapGenerico.SelectCommand = objCmdBD;
                dapGenerico.Fill(dts, NombreTabla);
                return true;
            }
            catch (Exception ex)
            {
                strError = ex.Message;
                return false;
            }
        }
        public bool AgregarParametro(ParameterDirection Direccion, string Nombre_En_SP,
        SqlDbType TipoDato, Int16 Tamaño, object Valor)
        {
            try
            {
                objParametro.Direction = Direccion;
                objParametro.ParameterName = Nombre_En_SP;
                objParametro.SqlDbType = TipoDato;
                objParametro.Size = Tamaño;
                objParametro.Value = Valor;
                objCmdBD.Parameters.Add(objParametro);
                objParametro = new SqlParameter();
                return (true);
            }
            catch (Exception ex)
            {
                strError = ex.Message;
                return (false);
            }
        }
        #endregion
    }
}
```

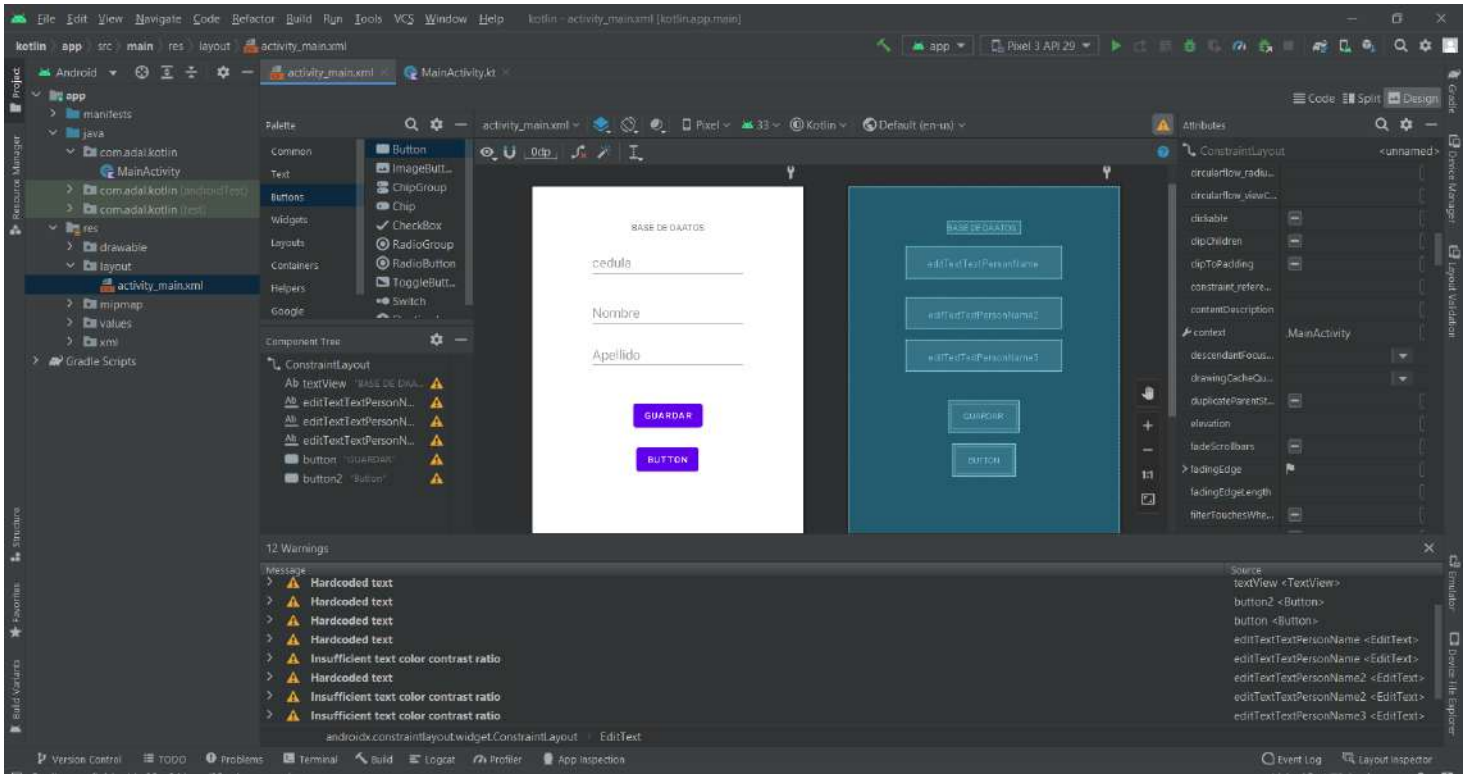
GC-F-004v.04

# Guardar datos en Firebase Realtime

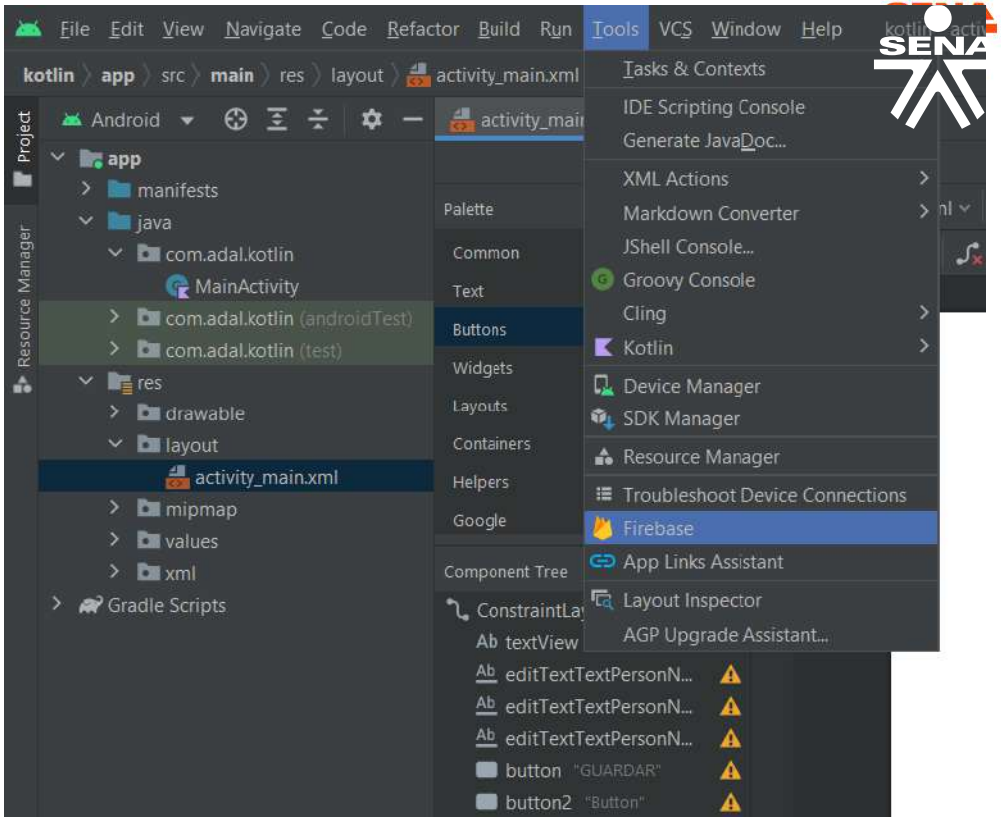





Realizar el siguiente formulario:

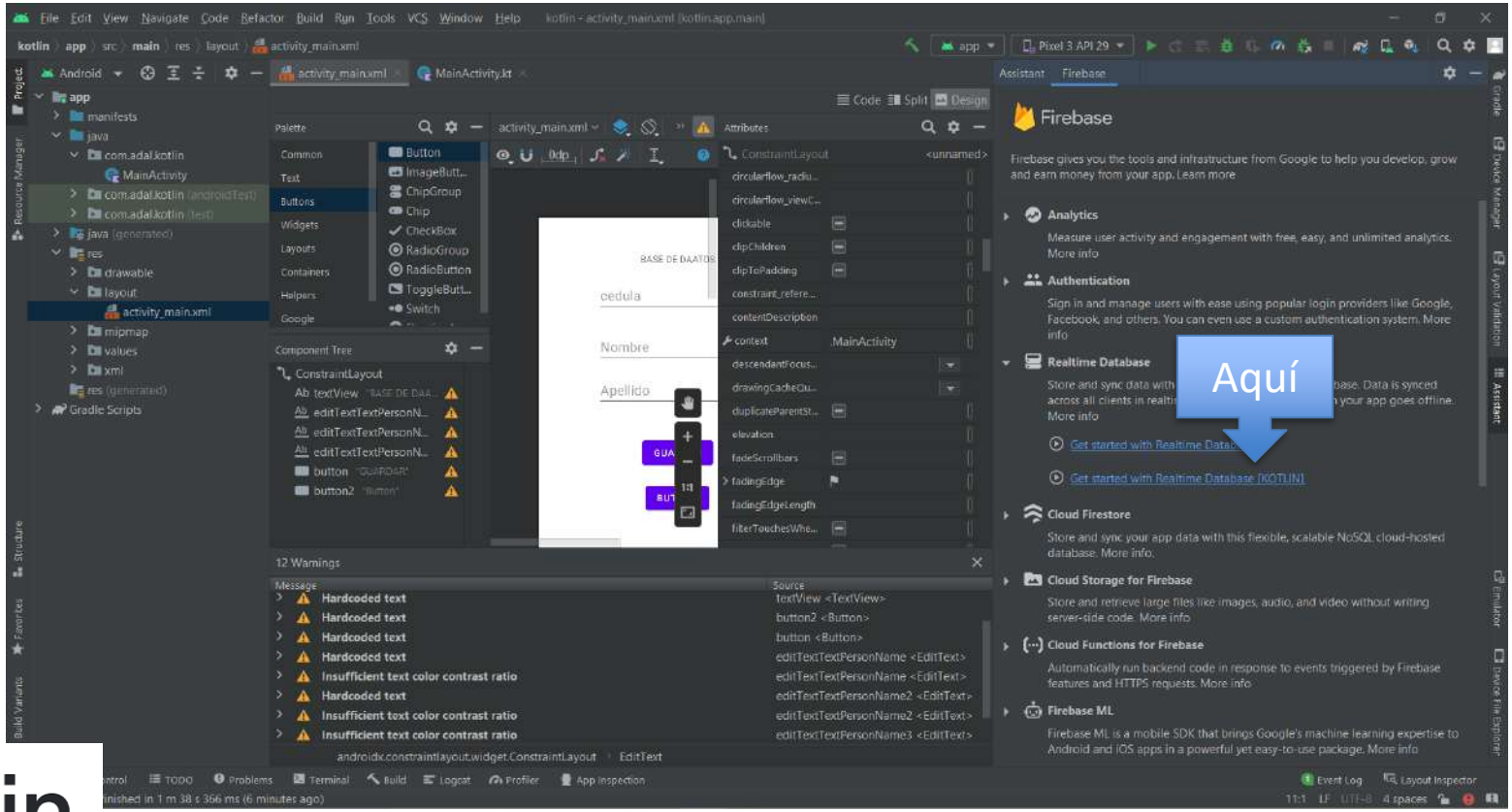


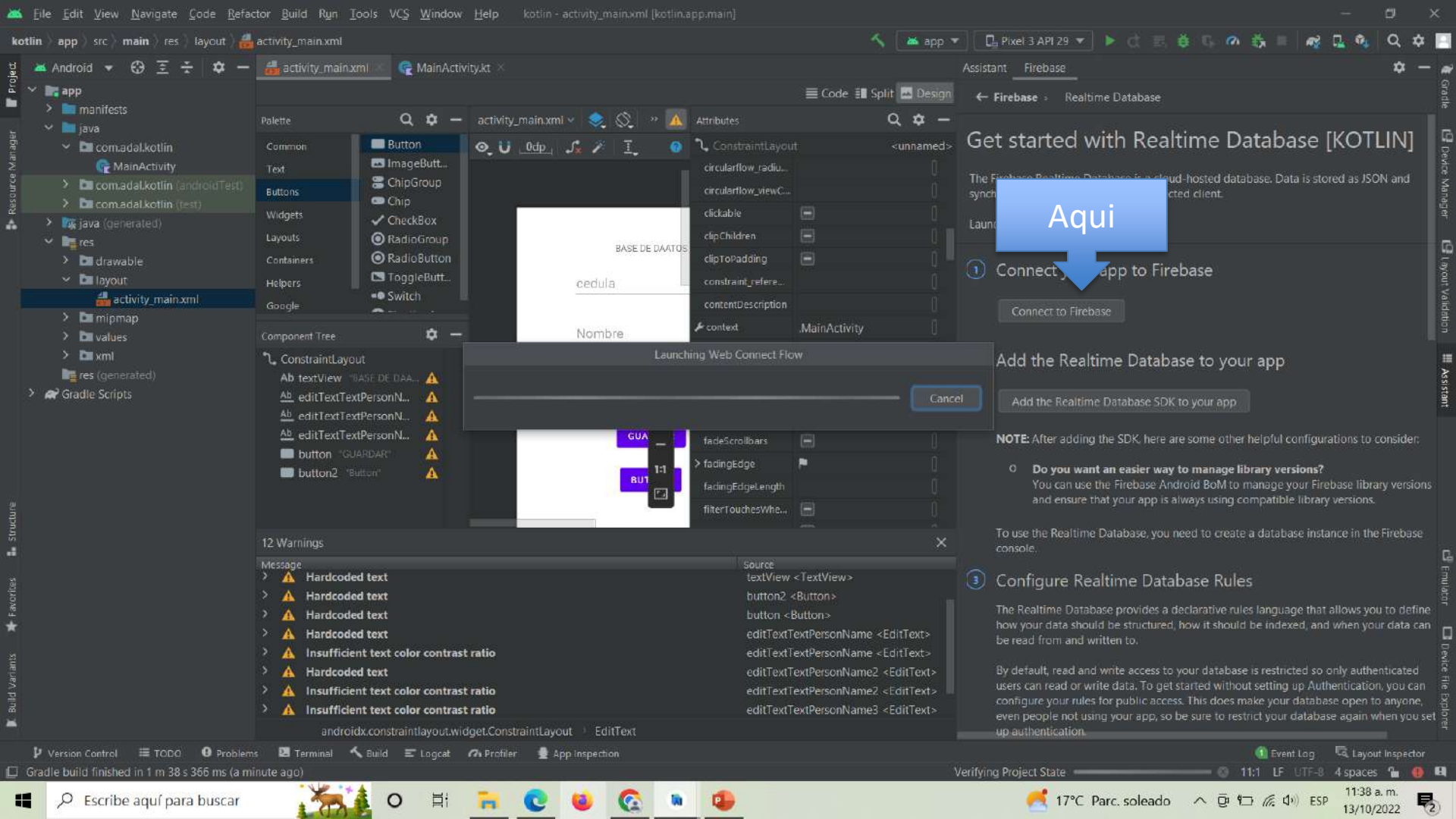
1. Vamos al menú tools, Firebase



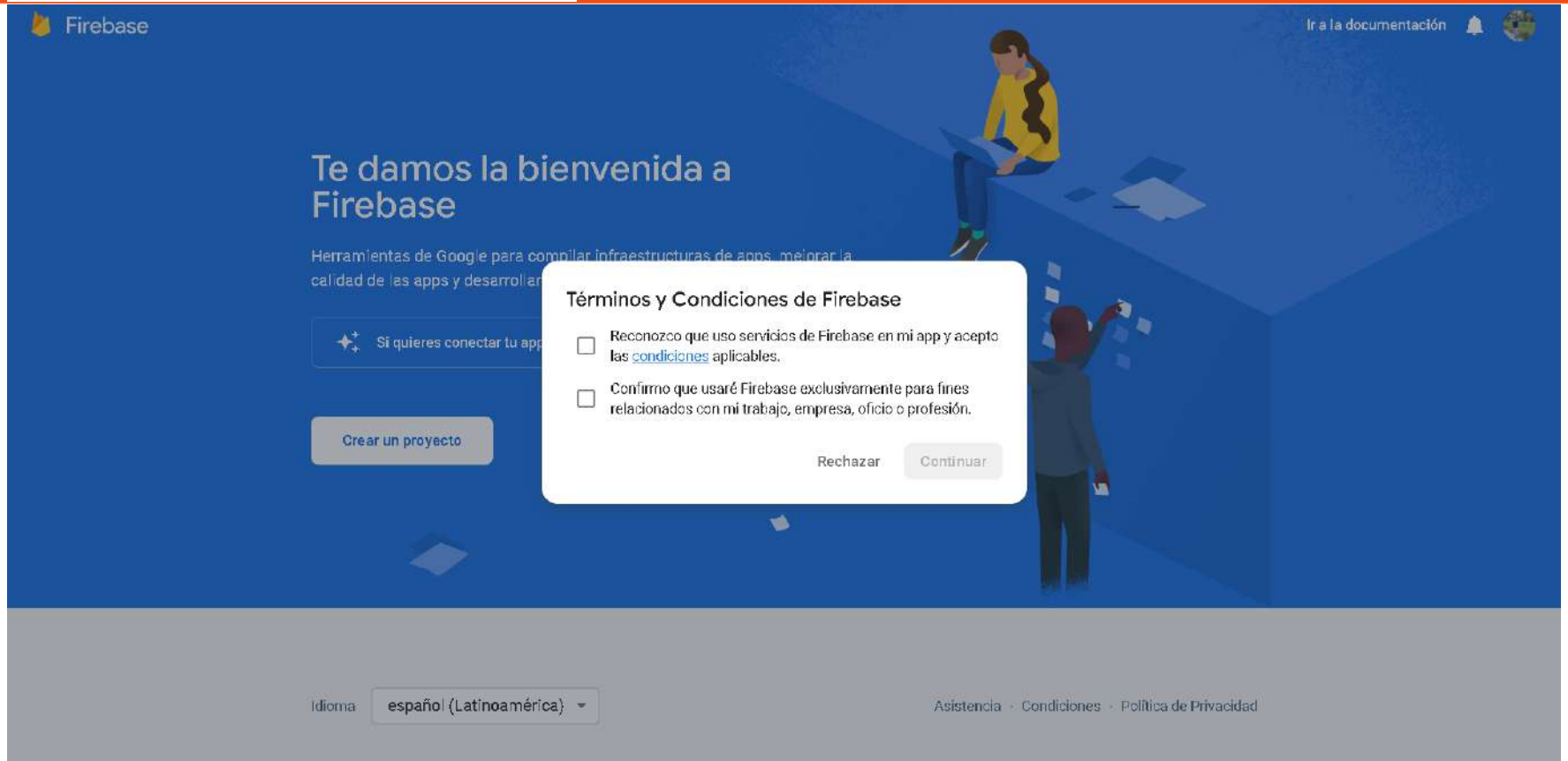

Kotlin

# 2. Realtime Database/ Get started with Realtime Database Kotlin




Aqui
Get started with Realtime Database [KOTLIN]
Connect to Firebase
Add the Realtime Database to your app
Add the Realtime Database SDK to your app
Launching Web Connect Flow
Cancel
Configure Realtime Database Rules

# Firebase


Firebase
Ir a la documentación
Te damos la bienvenida a Firebase
Herramientas de Google para compilar infraestructuras de apps, mejorar la calidad de las apps y desarrollar
Si quieres conectar tu app
Crear un proyecto
Términos y Condiciones de Firebase
Reconozco que uso servicios de Firebase en mi app y acepto las condiciones aplicables.
Confirmo que usaré Firebase exclusivamente para fines relacionados con mi trabajo, empresa, oficio o profesión.
Rechazar
Continuar
Idioma
español (Latinoamérica)
Asistencia · Condiciones · Política de Privacidad

# Firebase


Crear un proyecto(paso 1 de 3)
Comencemos con el nombre de tu proyecto
Nombre del proyecto
kotlin
kotlin-98738
Seleccionar recurso superior
Continuar

# Firebase

Crear un proyecto(paso 2 de 3)

Google Analytics es una solución de analítica ilimitada y gratuita que permite usar la segmentación, los informes y otras funciones en Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing y Cloud Functions.

Google Analytics habilita las siguientes funciones:

- Pruebas A/B
- Segmentación de usuarios y orientación a ellos en los productos de Firebase
- Usuarios que no experimentan fallas
- Activadores de Cloud Functions basados en eventos
- Informes ilimitados y gratuitos

Habilitar Google Analytics para este proyecto
Recomendado

Anterior

Continuar



# Firebase

Crear un proyecto(paso 3 de 3)

## Configurar Google Analytics

Ubicación de Analytics

Estados Unidos

Configuración de uso compartido de datos y Condiciones de Google Analytics

Usar la configuración predeterminada para compartir datos de Google Analytics
Más información

- ✕ Compartir tus datos de Analytics con Google para mejorar nuestros productos y servicios
- ✓ Compartir tus datos de Analytics con Google para habilitar las comparativas
- ✓ Compartir tus datos de estadísticas con Google para habilitar la asistencia técnica
- ✓ Compartir tus datos de Analytics con los especialistas en Cuentas de Google

Acepto las Condiciones de Google Analytics

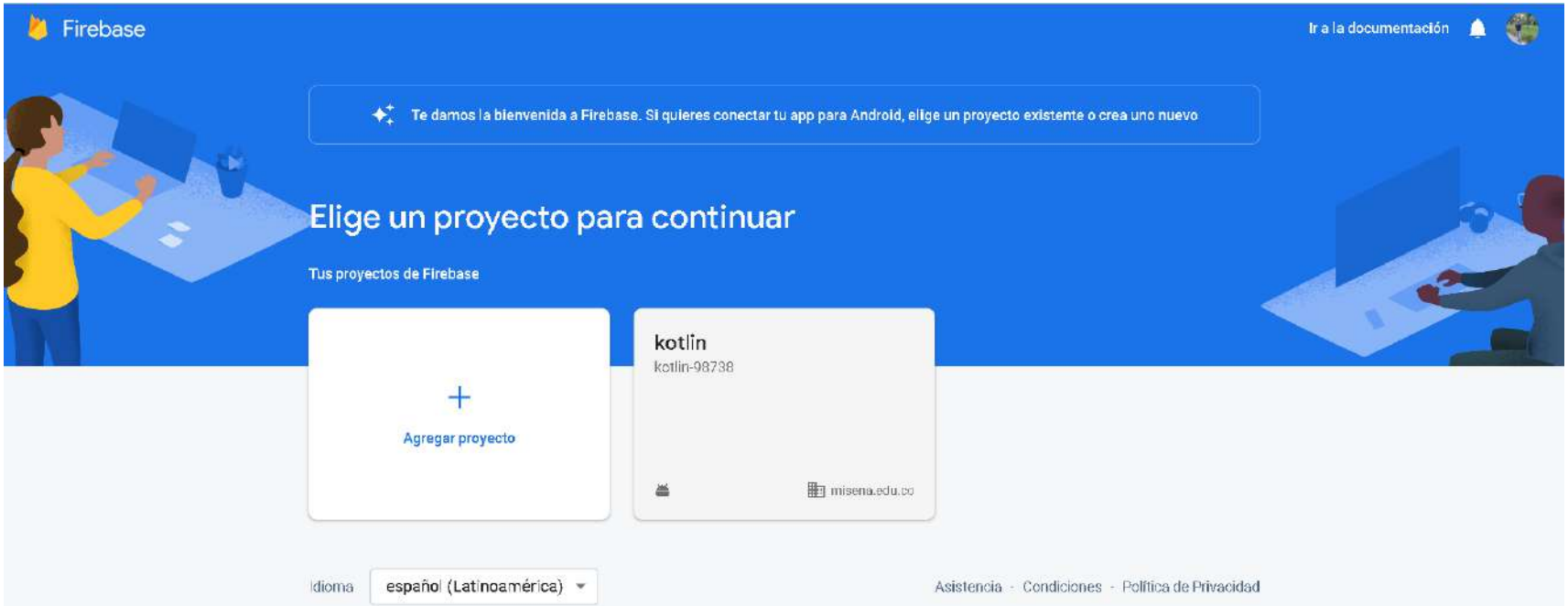
Cuando se cree el proyecto, también se creará una nueva propiedad de Google Analytics y se vinculará a tu proyecto de Firebase. Este vínculo permitirá el flujo de datos entre los productos. Los datos que se exportan de tu propiedad de Google Analytics a Firebase están sujetos a las Condiciones del Servicio de Firebase, mientras que los datos de Firebase que se importan a Google Analytics están sujetos a las Condiciones del Servicio de Google Analytics. Obtén más información.

Anterior

Crear proyecto

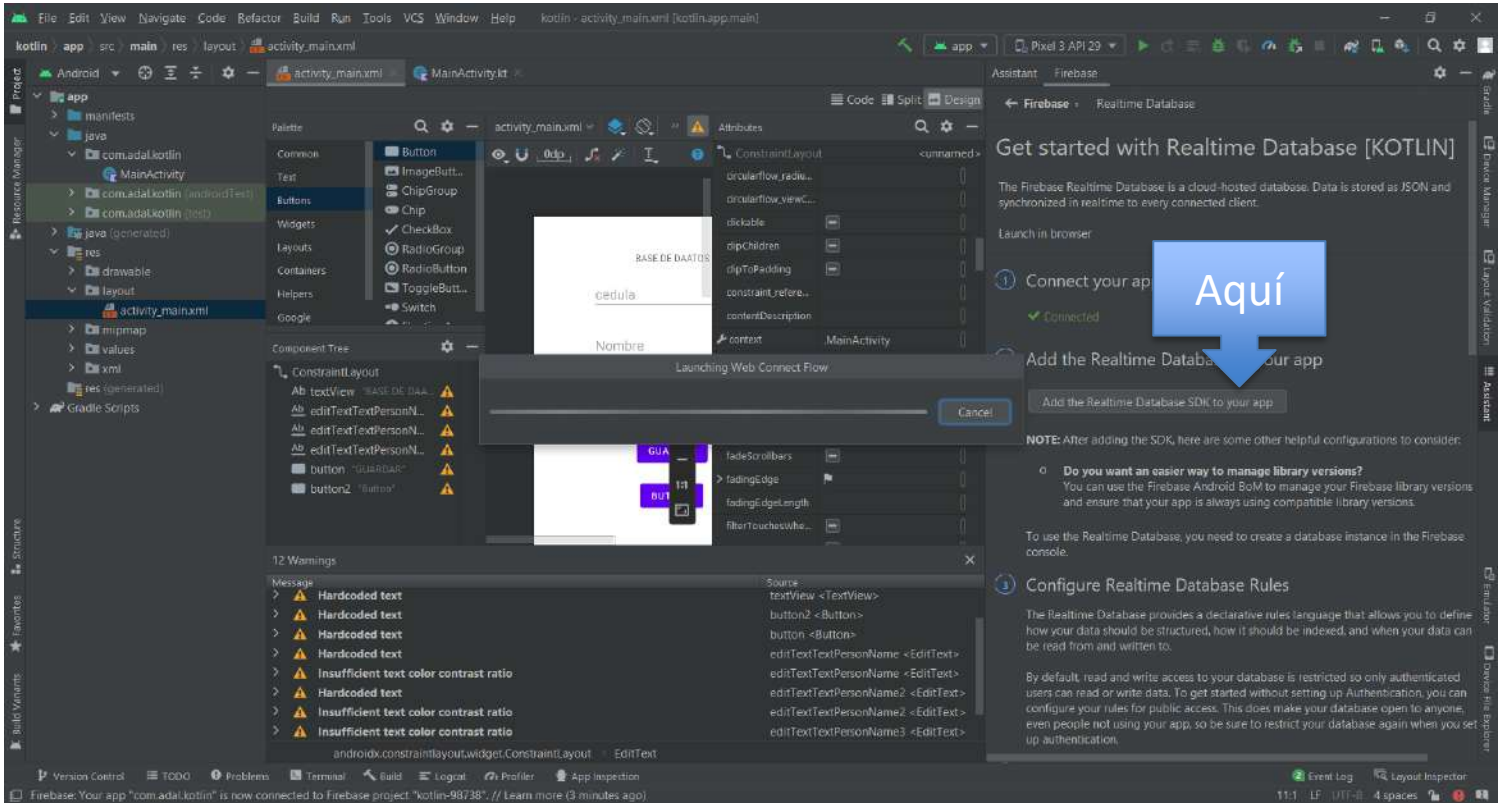


# Firebase

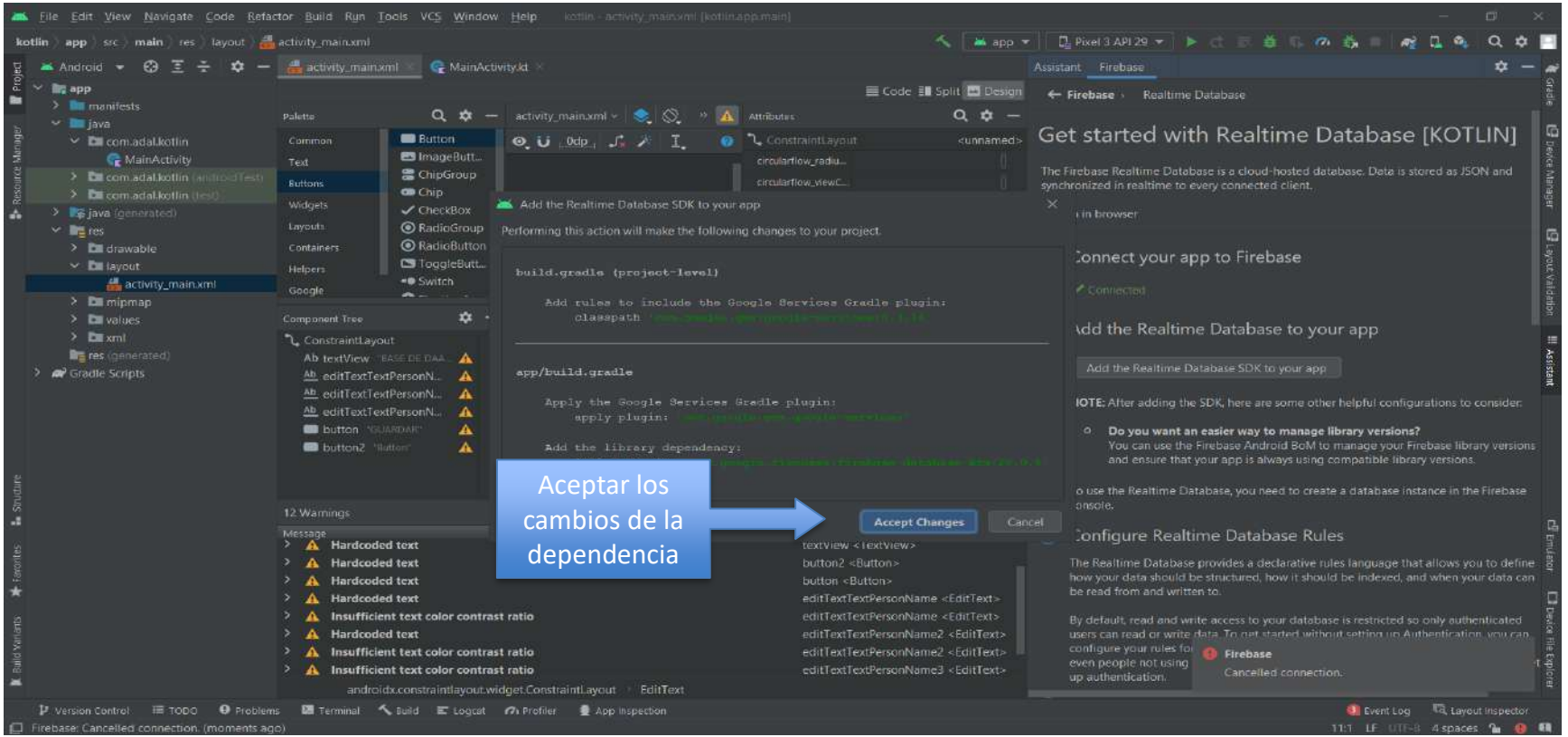



Escojo el proyecto creado en este caso Kotlin

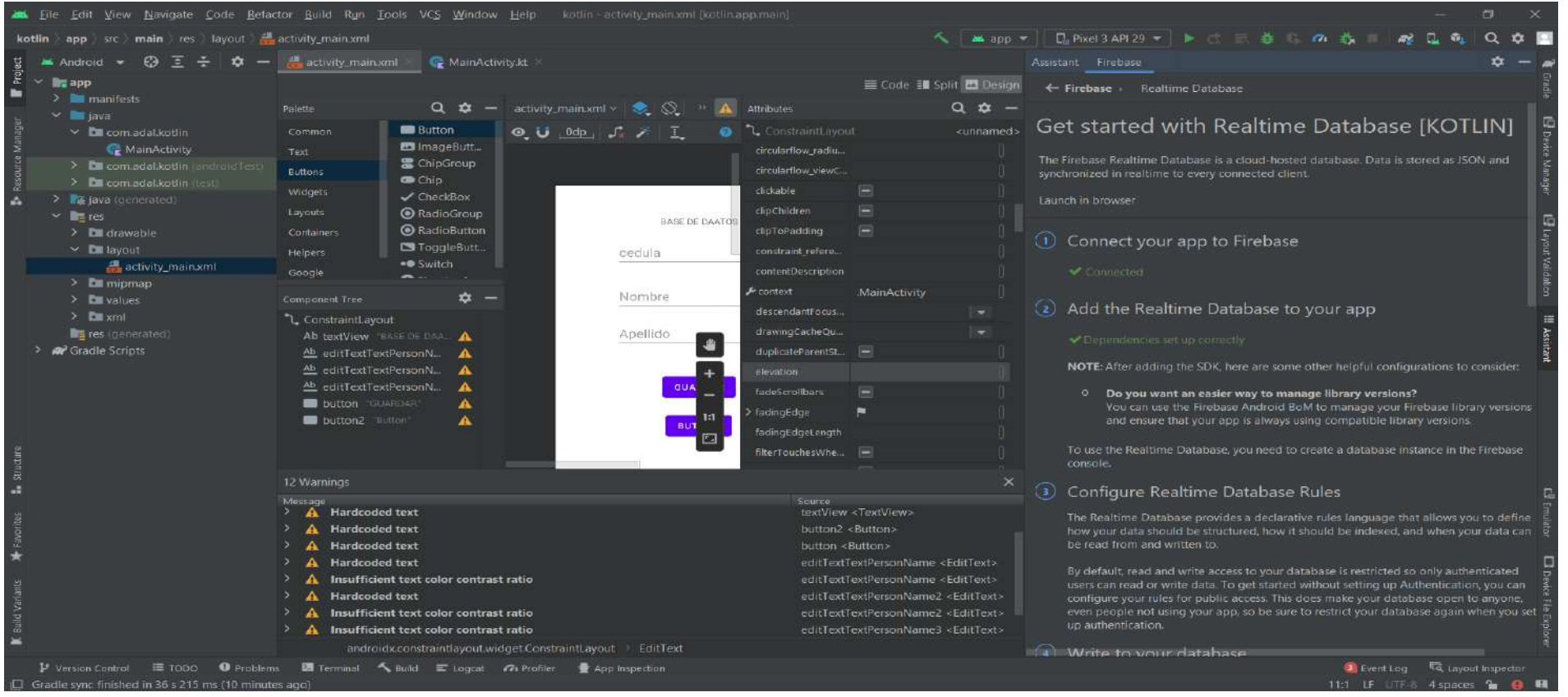
# Firebase


Aquí
Get started with Realtime Database [KOTLIN]
Launching Web Connect Flow
Cancel
Add the Realtime Database SDK to your app
Configure Realtime Database Rules

# Firebase


Aceptar los cambios de la dependencia
Add the Realtime Database SDK to your app
Performing this action will make the following changes to your project.
build.gradle (project-level)
app/build.gradle
Accept Changes
Cancel
Get started with Realtime Database [KOTLIN]
Connect your app to Firebase
Add the Realtime Database to your app
Add the Realtime Database SDK to your app
Configure Realtime Database Rules
Firebase
Cancelled connection.

# Firebase


Get started with Realtime Database [KOTLIN]
The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client.
Launch in browser
1 Connect your app to Firebase
Connected
2 Add the Realtime Database to your app
Dependencies set up correctly
NOTE: After adding the SDK, here are some other helpful configurations to consider:
Do you want an easier way to manage library versions?
You can use the Firebase Android BoM to manage your Firebase library versions and ensure that your app is always using compatible library versions.
To use the Realtime Database, you need to create a database instance in the Firebase console.
3 Configure Realtime Database Rules
The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to.
By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up Authentication, you can configure your rules for public access. This does make your database open to anyone, even people not using your app, so be sure to restrict your database again when you set up authentication.
4 Write to your database



Firebase

Creamos una data clase llamada Usuario y creamos el siguiente constructor

```
package com.adal.kotlin

data class Usuario(val cedula: String?=null,val nombre:String?=null,val apellido:String? =null){

}
```

Firebase

Digitamos el siguiente código en el MainActivity

```
package com.adal.kotlin

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button
import android.widget.TextView
import android.widget.Toast
import com.google.firebase.database.DatabaseReference

class MainActivity : AppCompatActivity() {
  private lateinit var basedatos:DatabaseReference
    lateinit var botonG: Button
    lateinit var id: TextView
    lateinit var nom: TextView
    lateinit var apell: TextView
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        id=findViewById(R.id.txtcedula)
        nom=findViewById(R.id.txtnombre)
        apell=findViewById(R.id.txtapell)
        val b=findViewById<Button>(R.id.btnguardar)
        b.setOnClickListener{

        }

    }

}
```



Firebase

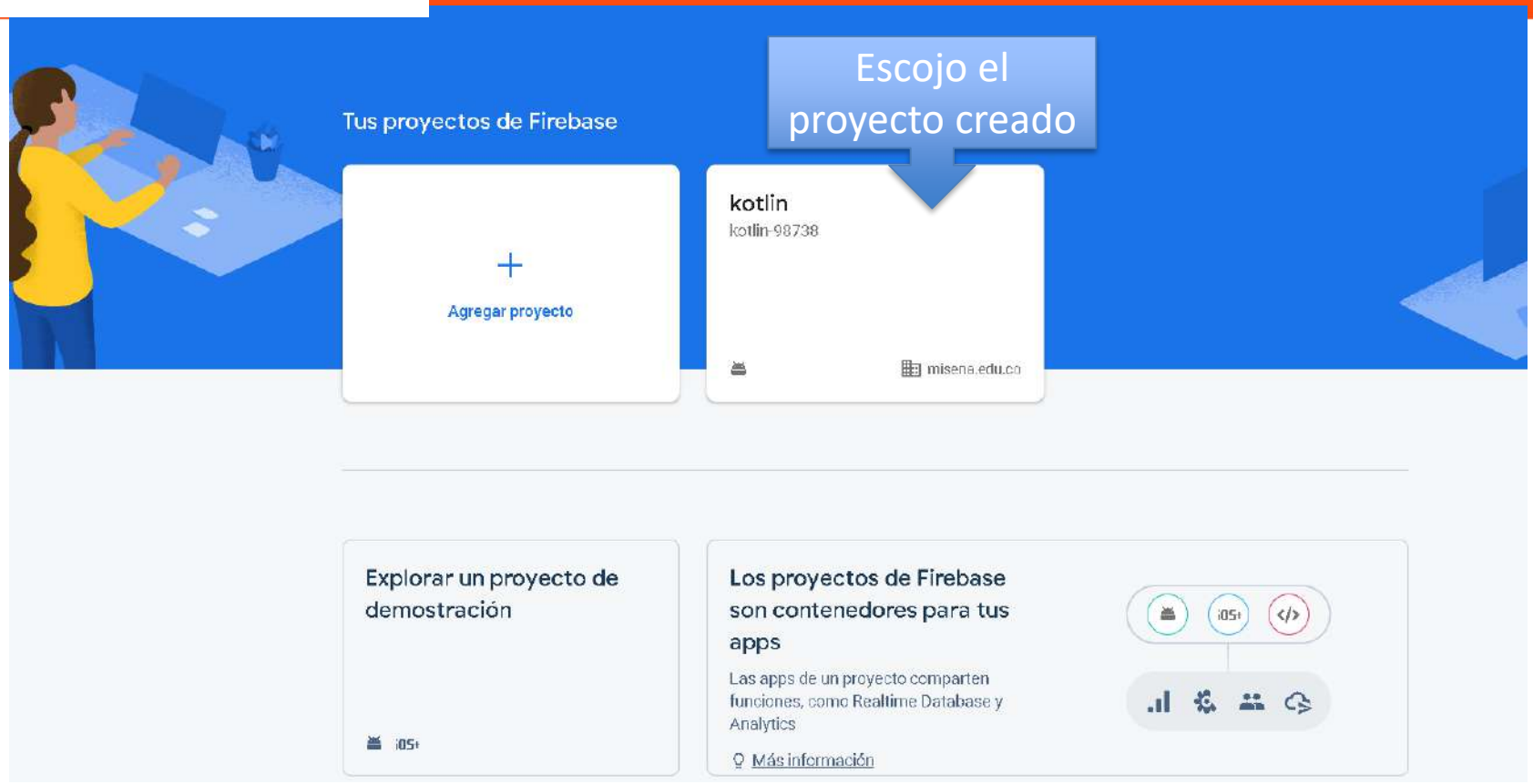
```
        id=findViewById(R.id.txtcedula)
        nom=findViewById(R.id.txtnombre)
        apell=findViewById(R.id.txtapell)
        val b=findViewById<Button>(R.id.btnguardar)
        b.setOnClickListener{
          val cc=id.text.toString()
          val nombre=nom.text.toString()
          val apellido=apell.text.toString()

           basedatos=FirebaseDatabase.getInstance().getReference("Usuario")

         val basedatosusuarios=Usuario(cc,nombre,apellido)
          basedatos.child(cc).setValue(basedatosusuarios).addOnSuccessListener {
             id.text=""
             nom.text=""
             apell.text=""

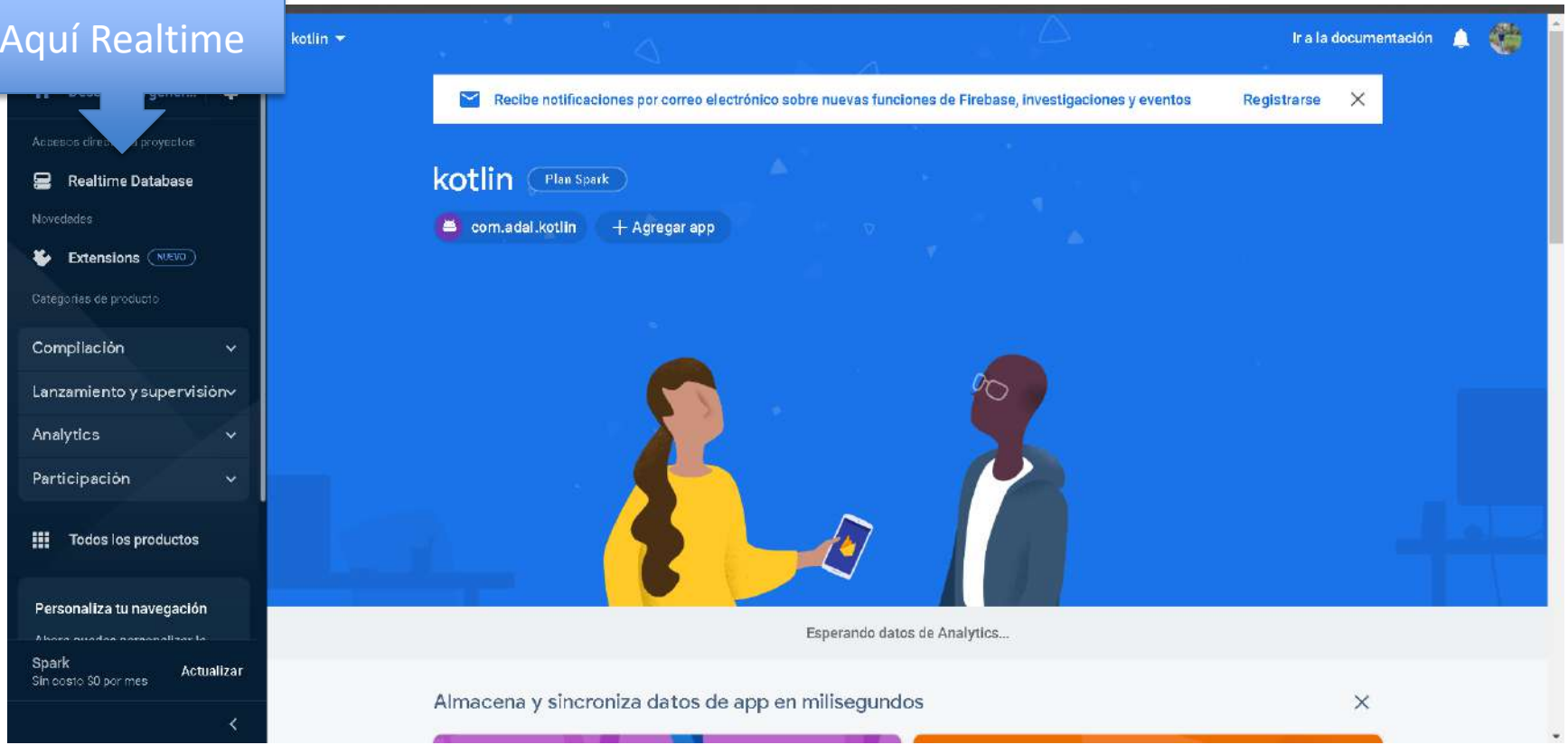
             Toast.makeText(this, "Se guardo Exitosamente", Toast.LENGTH_SHORT).show()
          }.addOnFailureListener{
             Toast.makeText(this, "No se guardo ", Toast.LENGTH_SHORT).show()
          }
        }
    }
}
```

# Firebase

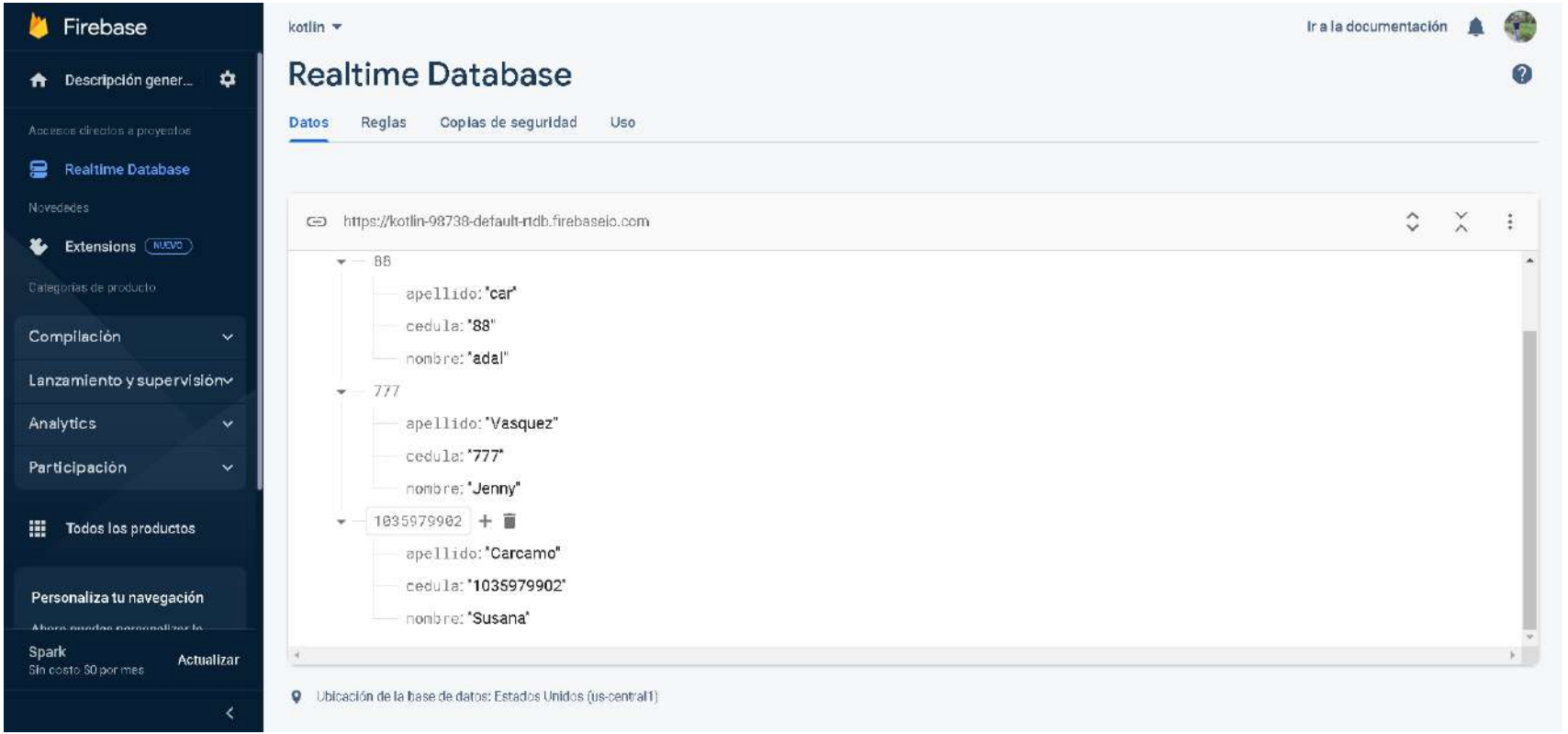

Escojo el proyecto creado
Tus proyectos de Firebase
Agregar proyecto
kotlin
kotlin-98738
misena.edu.co
Explorar un proyecto de demostración
iOS+
Los proyectos de Firebase son contenedores para tus apps
Las apps de un proyecto comparten funciones, como Realtime Database y Analytics
Más información

# Firebase

Aquí Realtime



# Firebase


Firebase
Descripción gener...
Accesos directos a proyectos
Realtime Database
Novedades
Extensions
NUEVO
Categorías de producto
Compilación
Lanzamiento y supervisión
Analytics
Participación
Todos los productos
Personaliza tu navegación
Spark
Sin costo $0 por mes
Actualizar
kotlin
Ir a la documentación
Realtime Database
Datos
Reglas
Copias de seguridad
Uso
https://kotlin-98738-default-rtdb.firebaseio.com
88
apellido: "car"
cedula: "88"
nombre: "adal"
777
apellido: "Vasquez"
cedula: "777"
nombre: "Jenny"
1035979902
apellido: "Carcamo"
cedula: "1035979902"
nombre: "Susana"
Ubicación de la base de datos: Estados Unidos (us-central1)




# GRACIAS

# Mapas Google

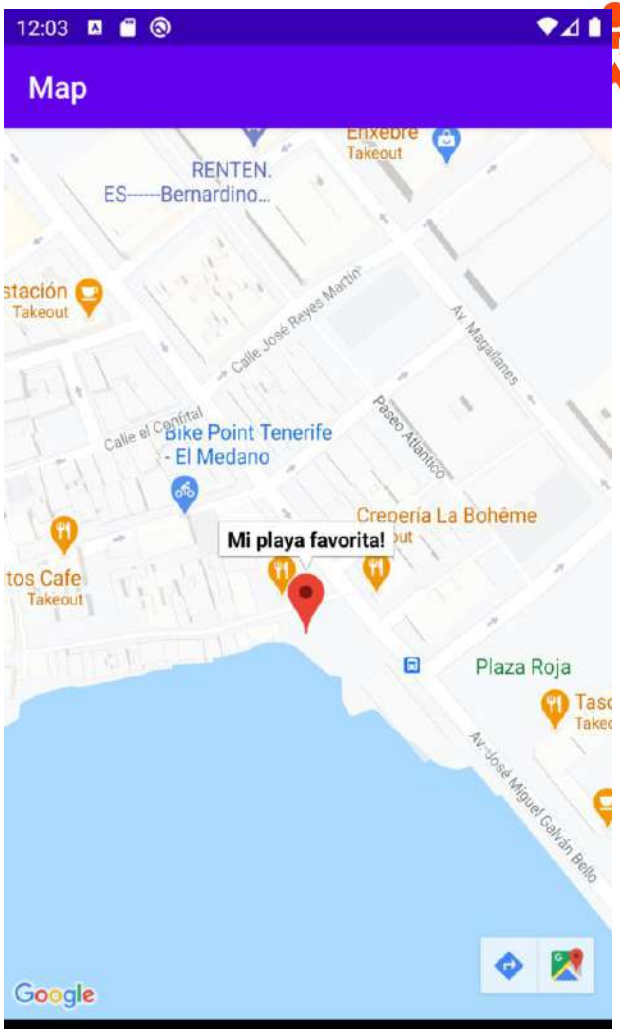


# Maps para Android

***Actividades***

•Habilitarás y usarás el SDK de Maps para Android a fin de agregar Google Maps a una app para Android.

•Agregarás, personalizarás y agruparás marcadores.

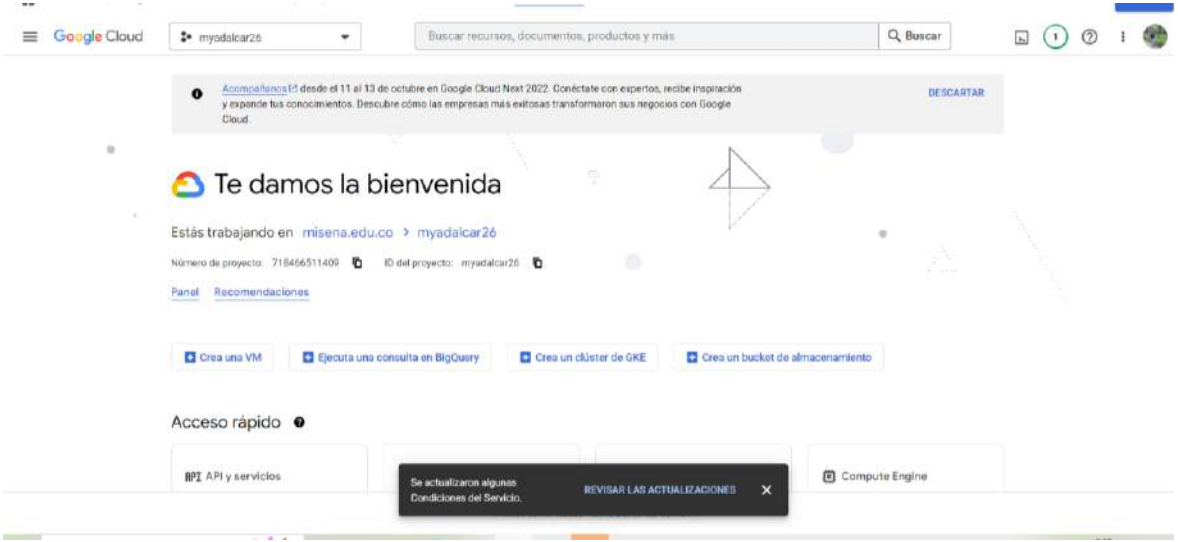


Google Cloud Platform  google.com

Se requiere una Api.Key

Para acceder a la api de Google map realizamos los siguientes pasos

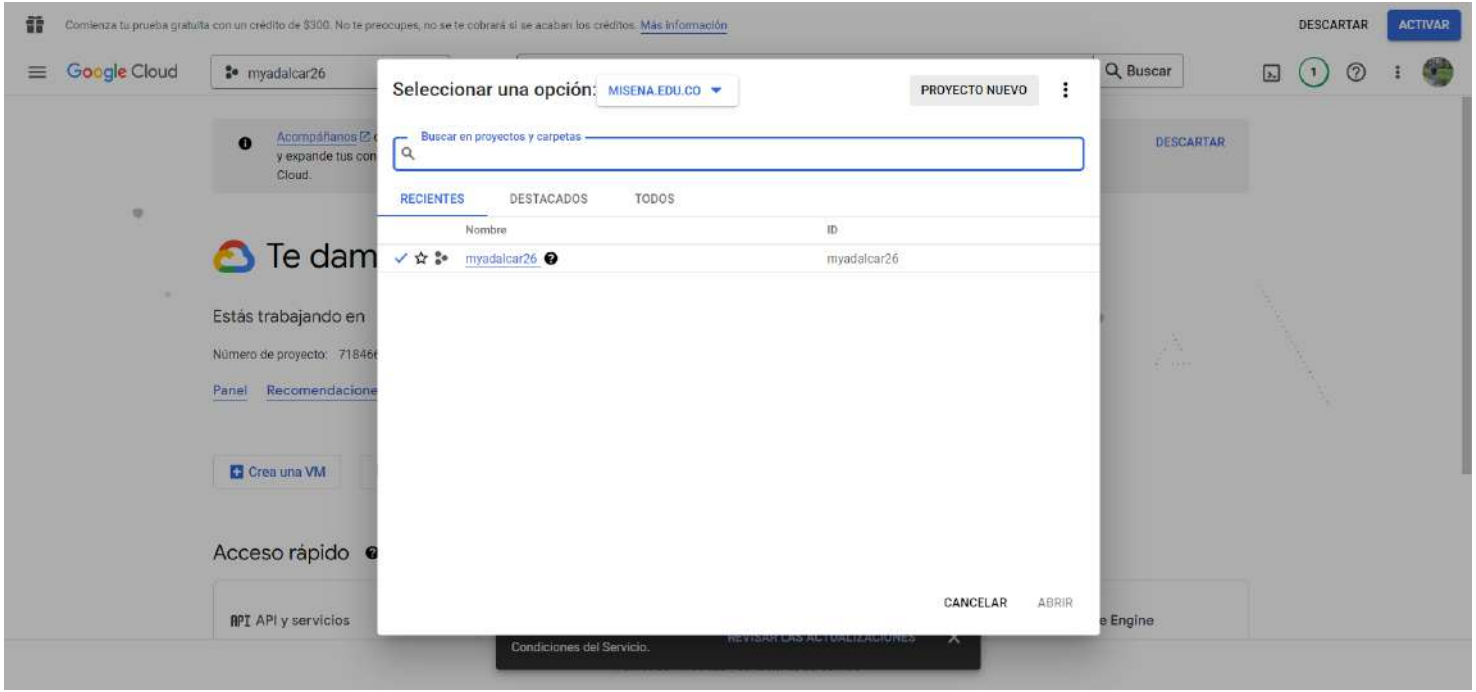
1. Ingresamos al link Console.cloud.google.com




Kotlin

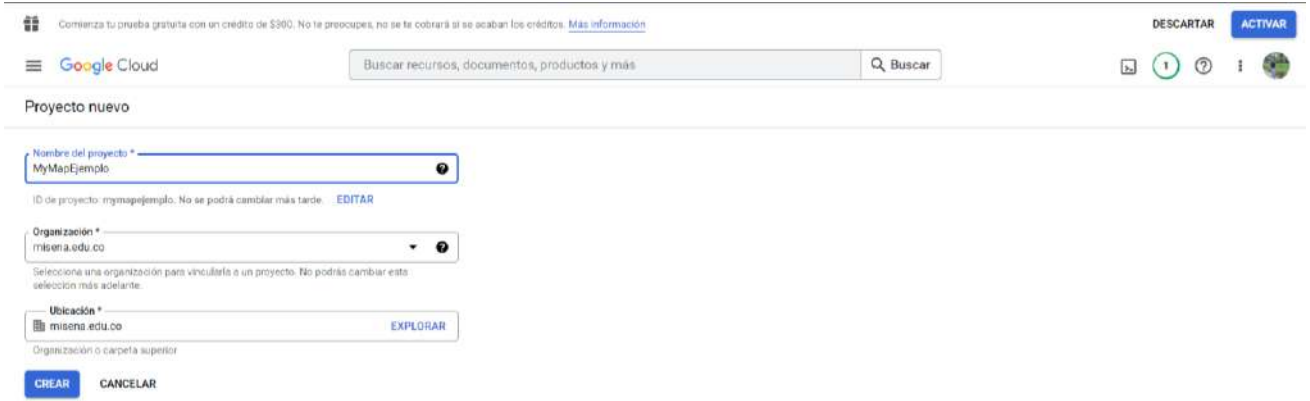
# Maps para Android

Creamos un nuevo proyecto:



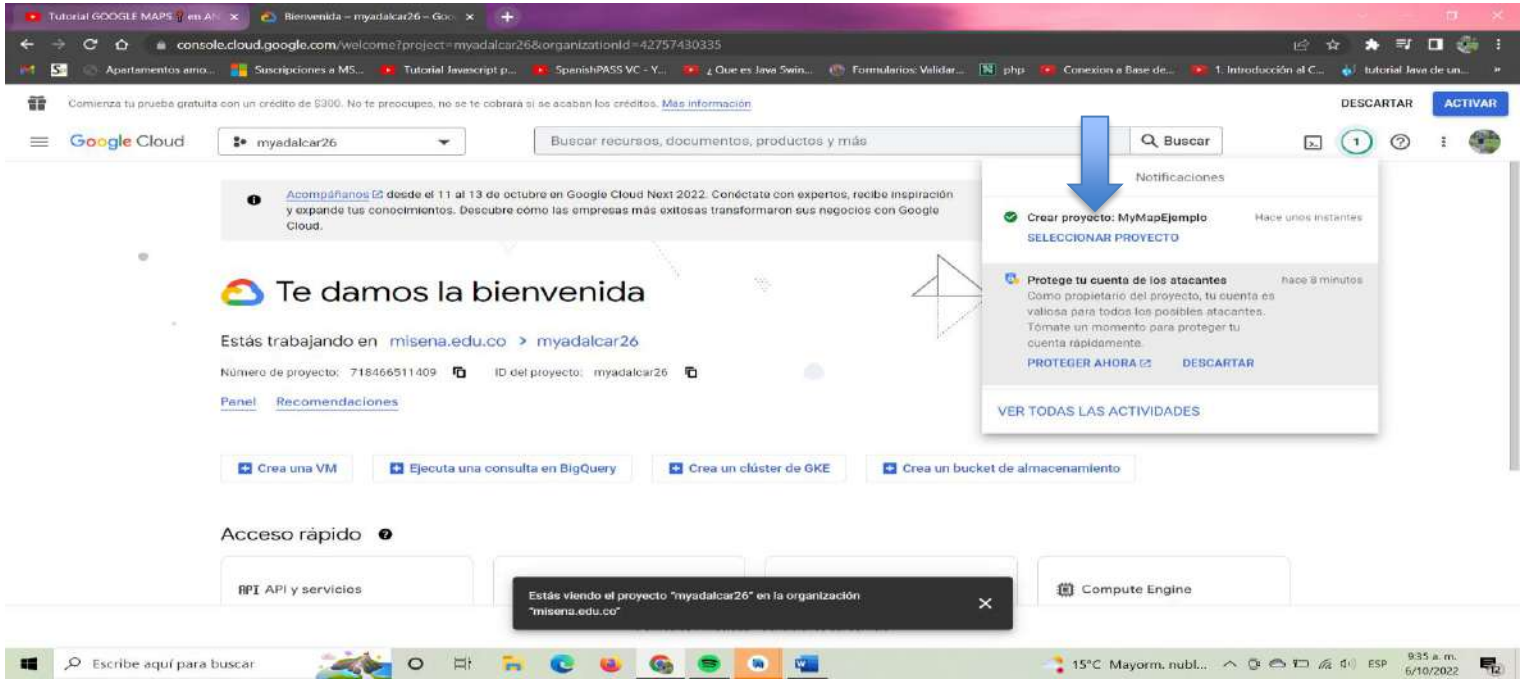
# Maps para Android

Creamos un nuevo proyecto:



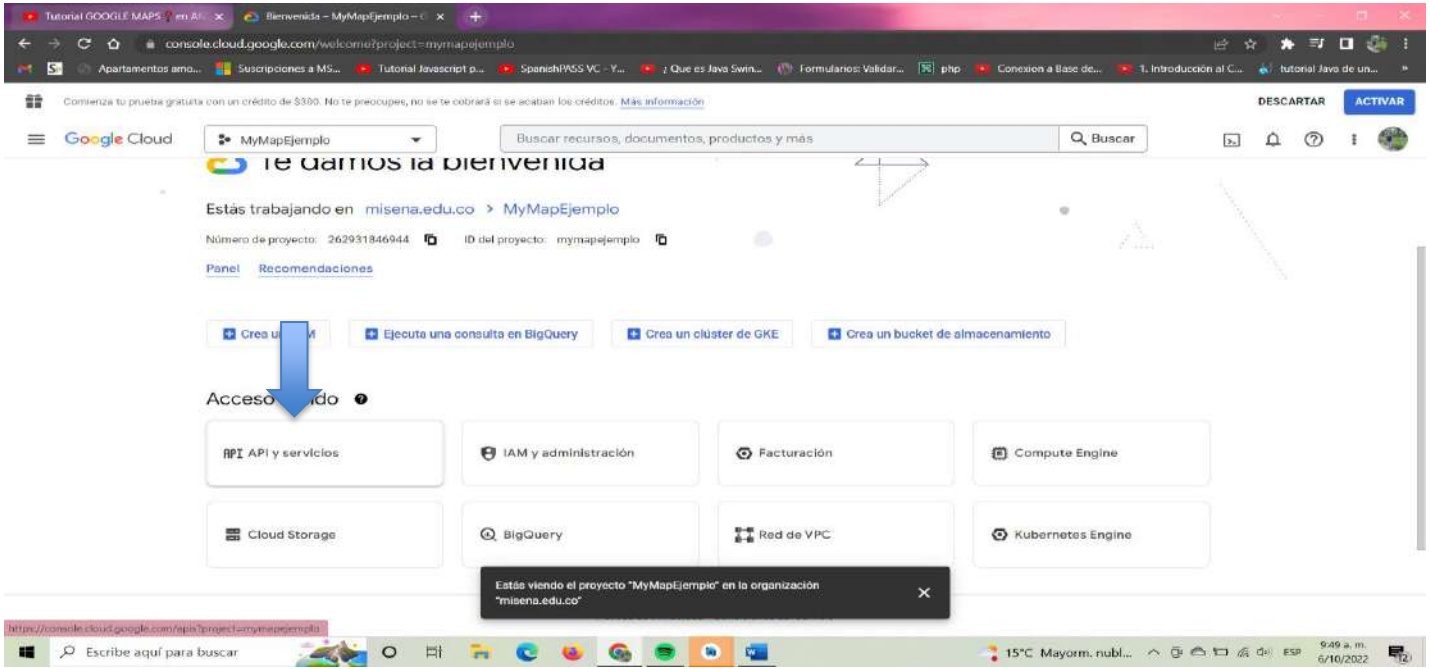
# Maps para Android

Creamos el  proyecto y lo seleccionamos



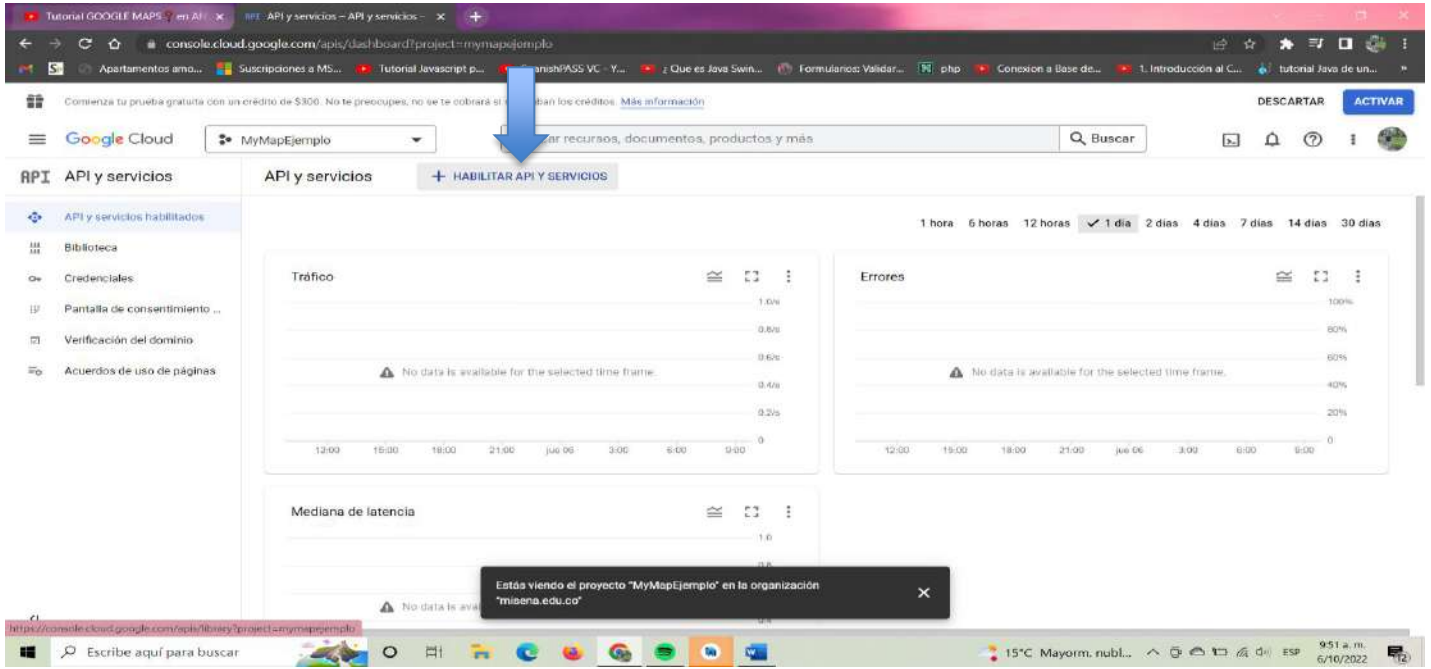
# Maps para Android

Seleccionamos Api y Servicios



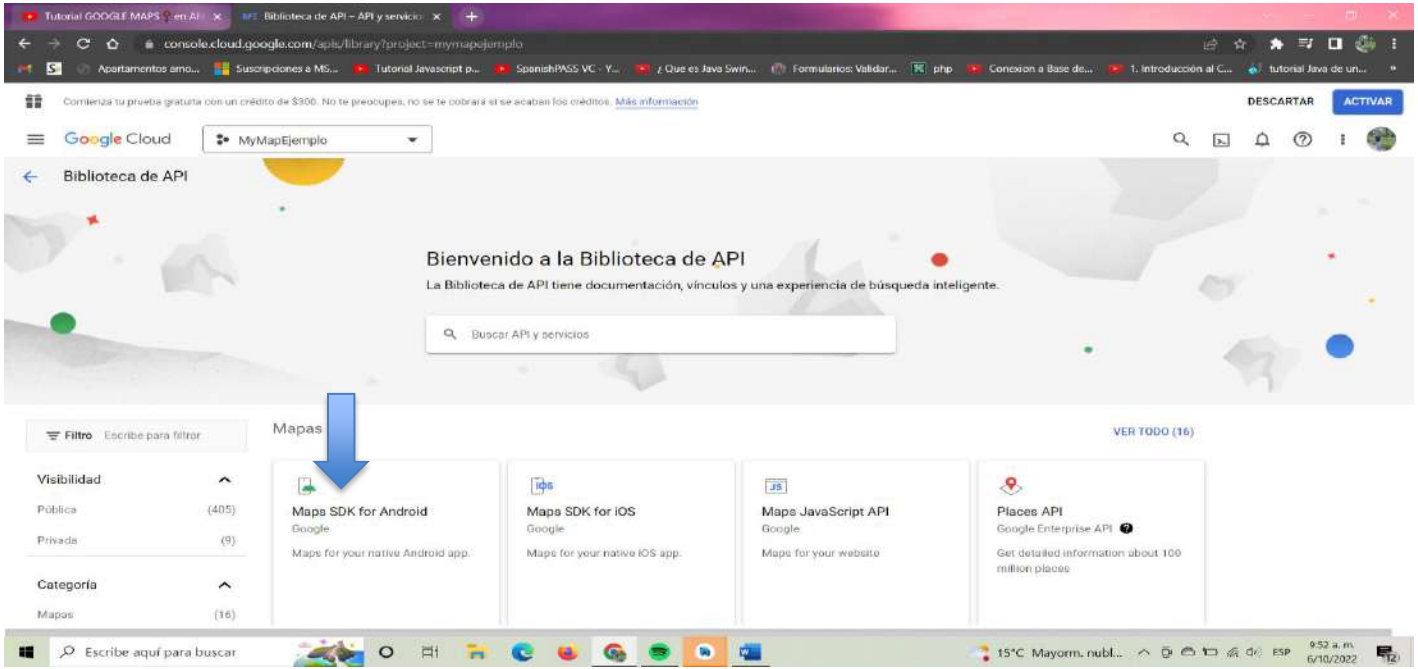
# Maps para Android

Habilitamos los Servicios



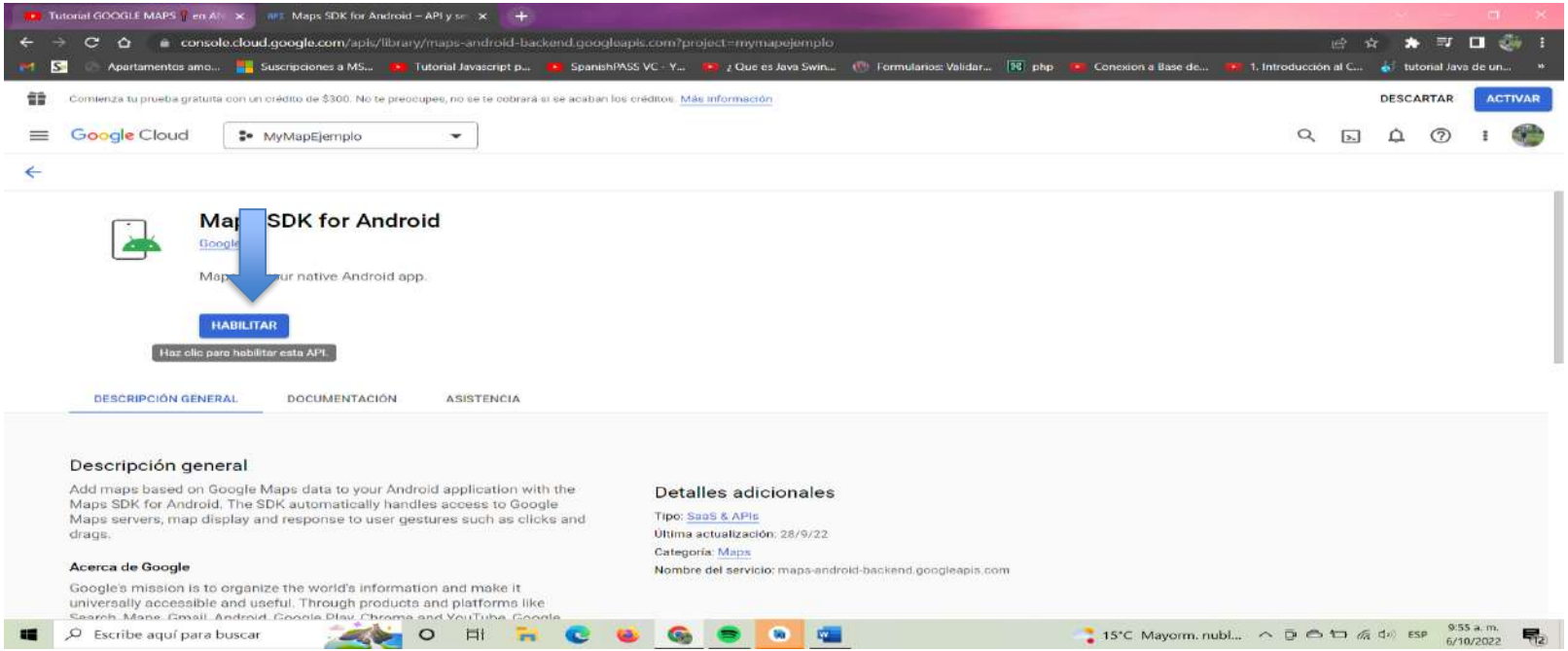
# Maps para Android

Escogemos Maps SDK for Android



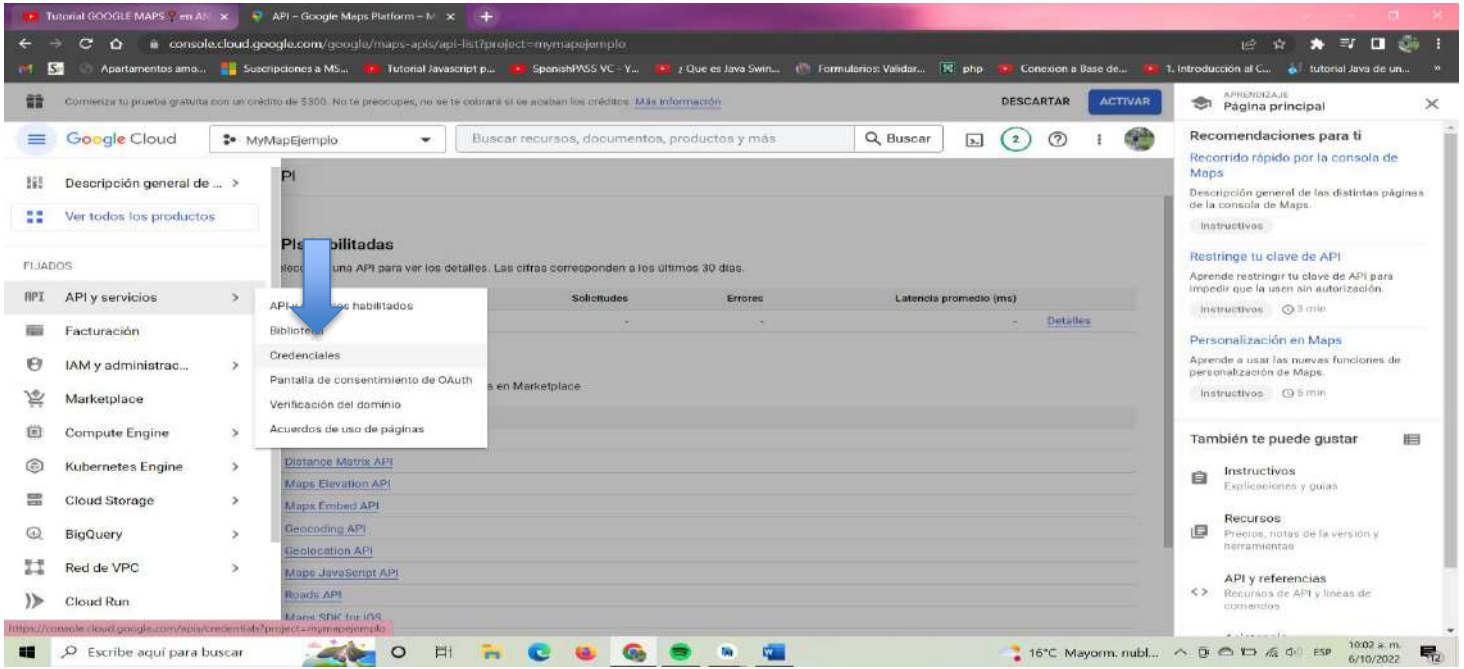
# Maps para Android

Habilitamos el  Maps SDK for Android



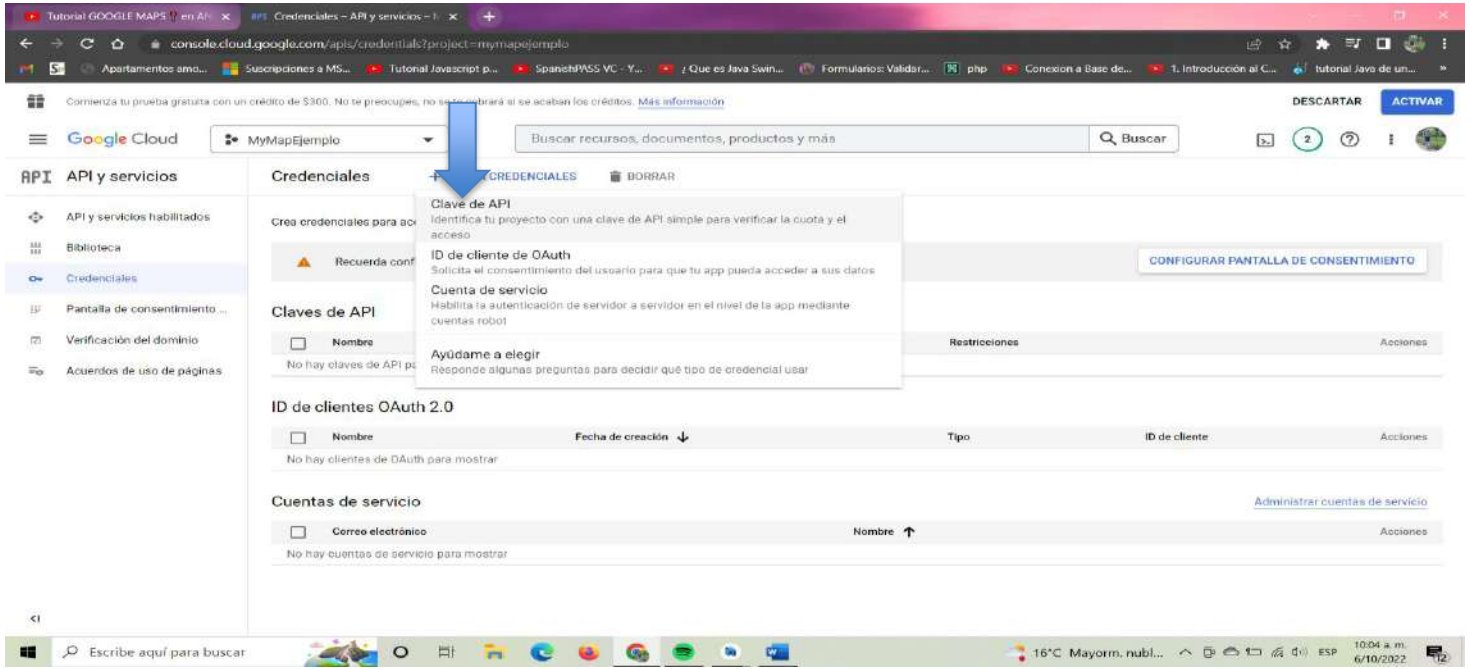
# Maps para Android

Api y servicios ,luego creamos credenciales



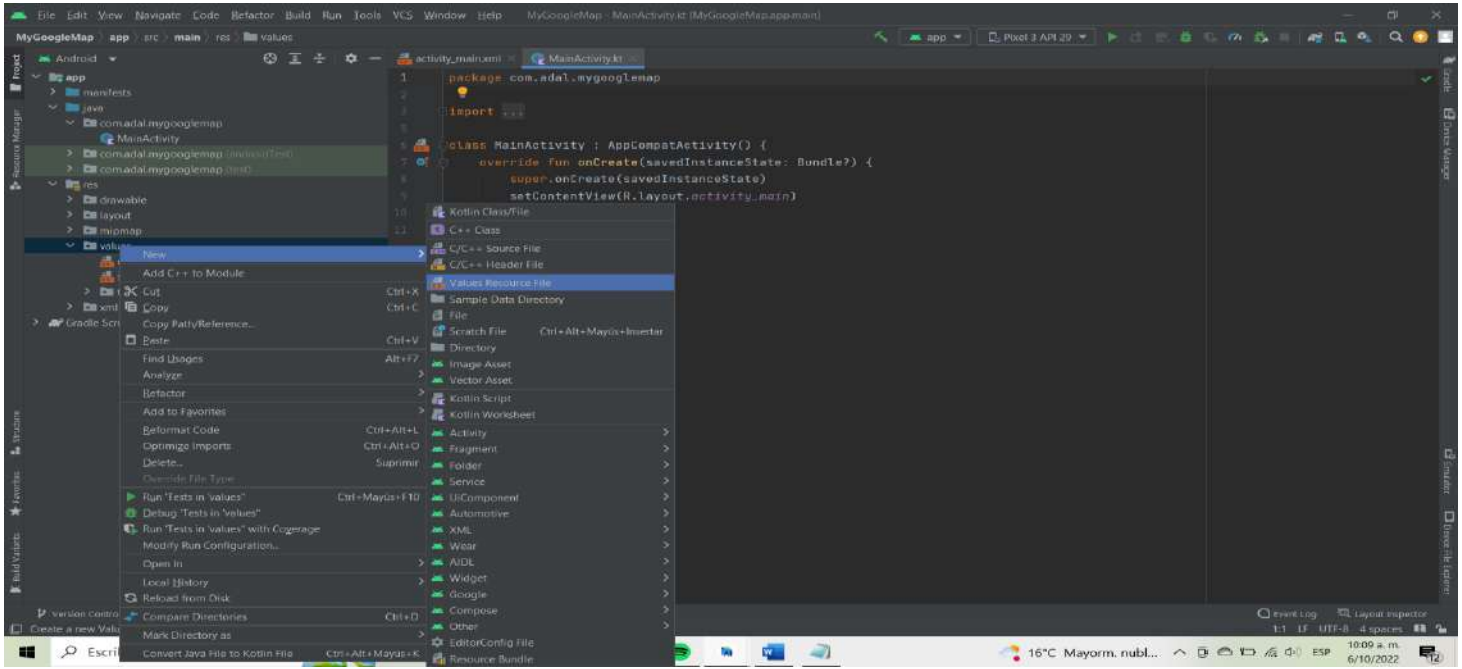
# Maps para Android

Creamos la Clave

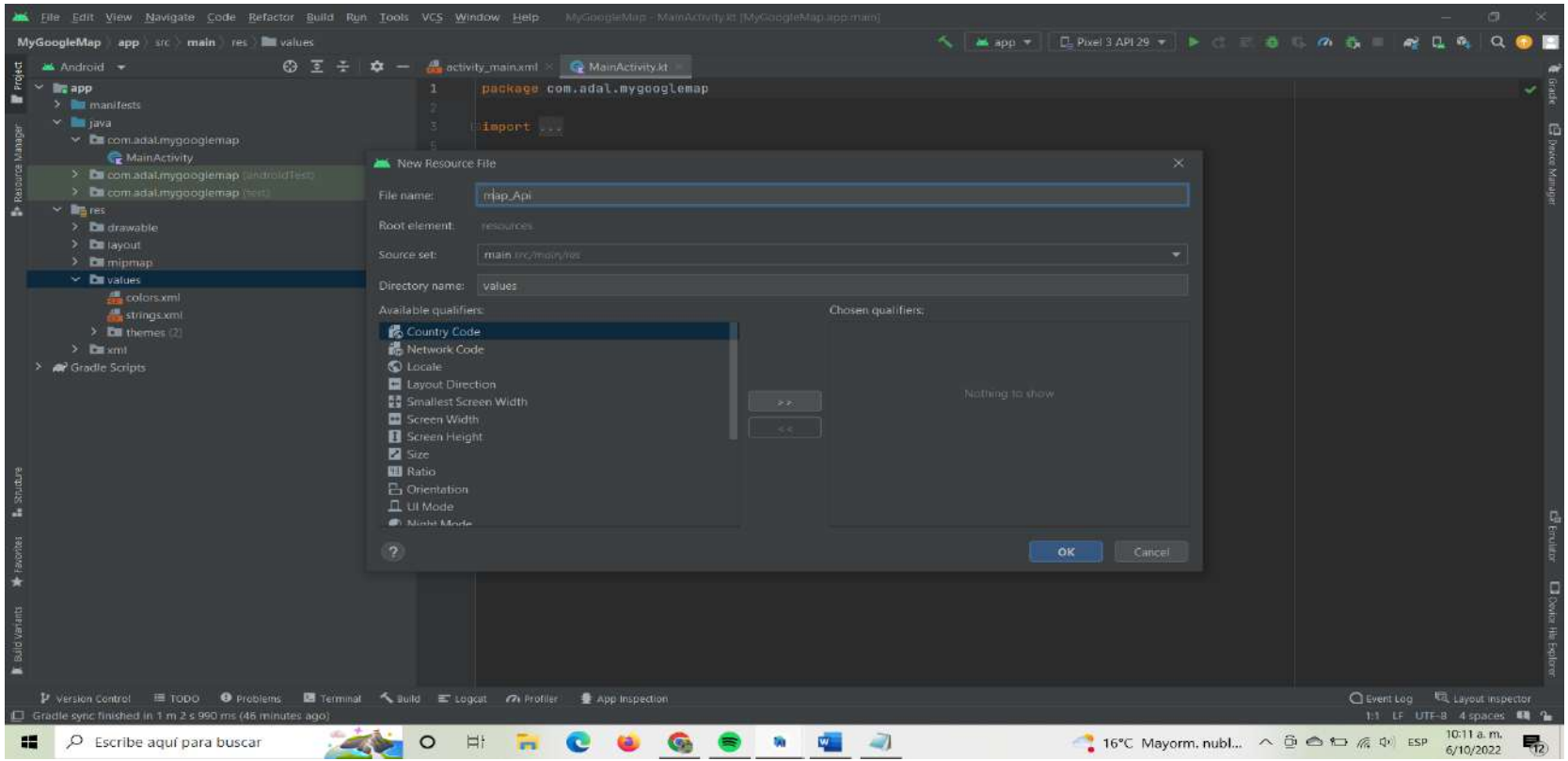


# Maps para Android

Creamos en Android Un nuevo proyecto en blanco,  creamos un nuevo archivo en la siguiente dirección

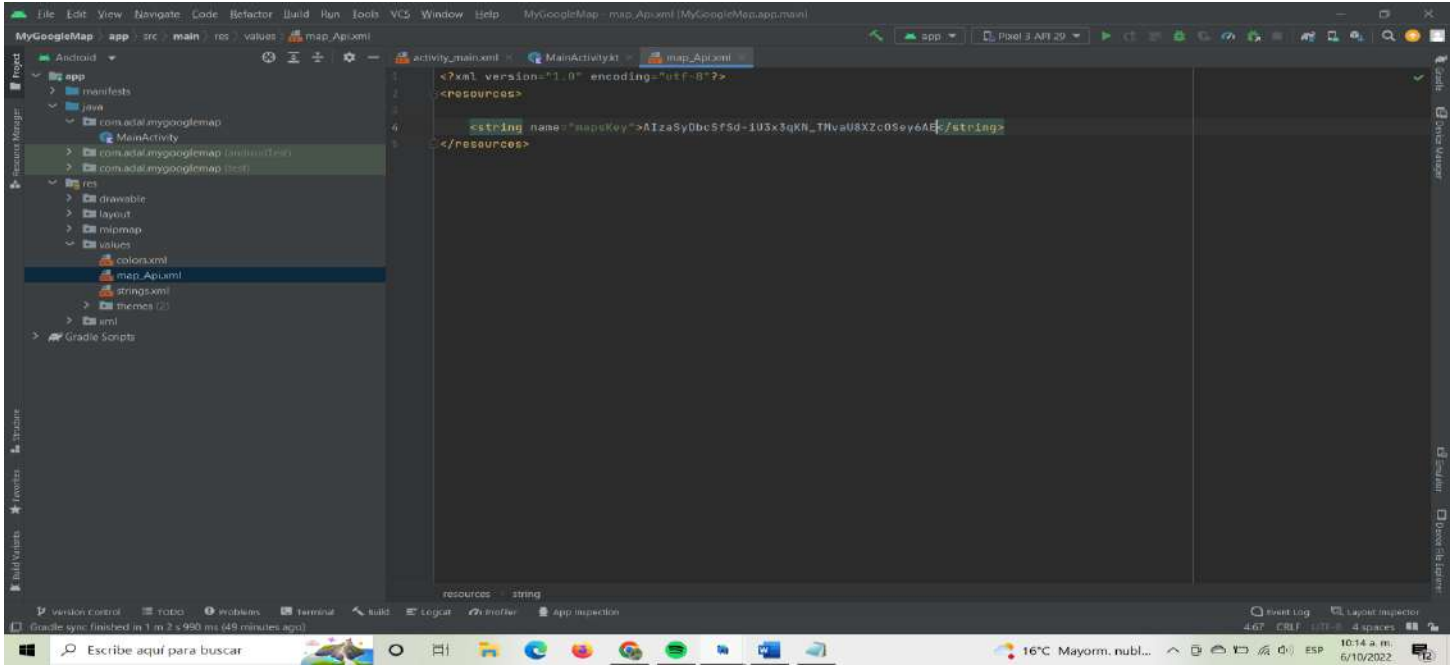


# Maps para Android



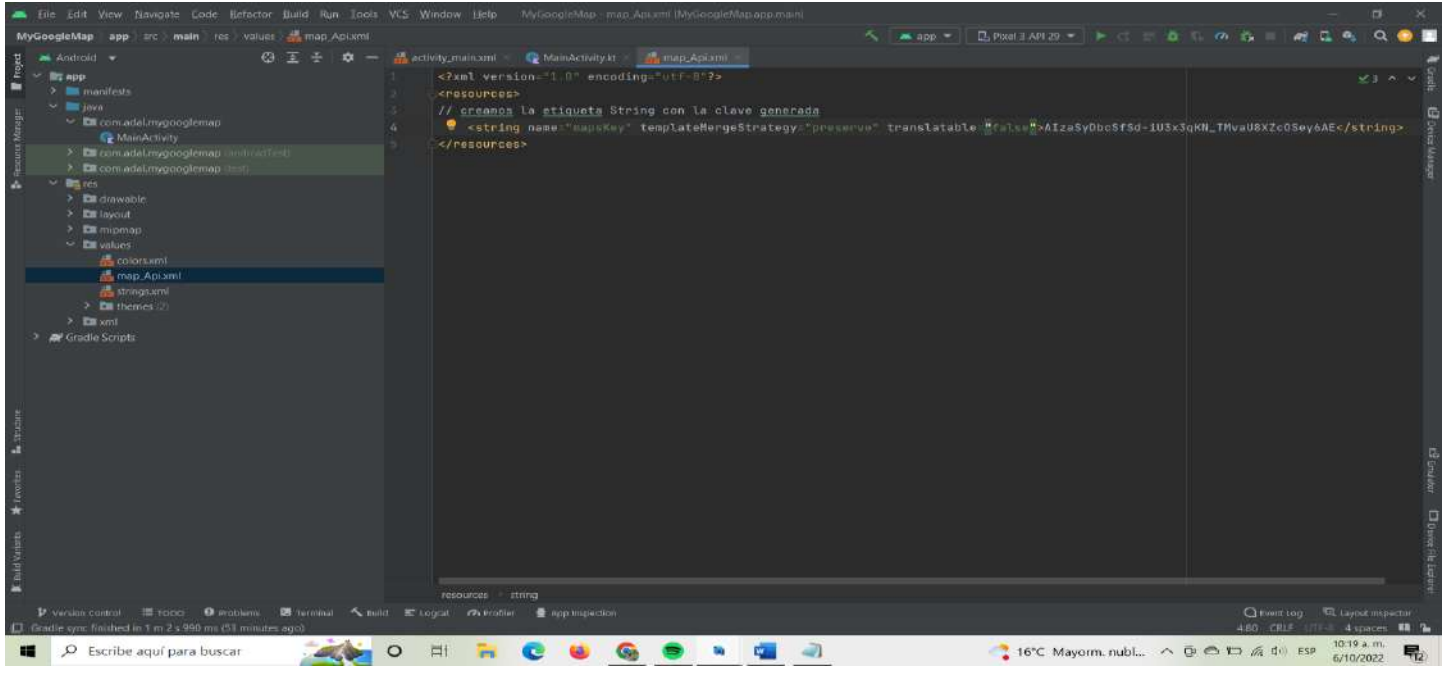
# Maps para Android

Creamos el siguiente código asignando la clave generada



# Maps para Android

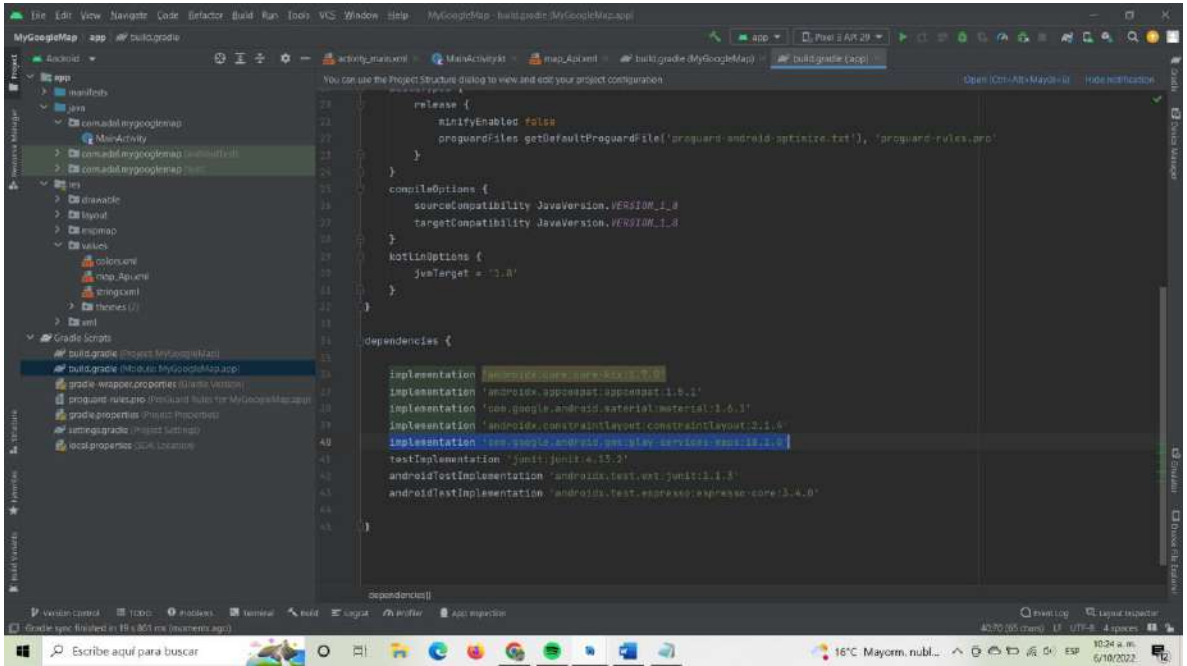
Para que la clave no sea cambiada por el modo de idiomas asignamos las siguientes propiedades



# Maps para Android

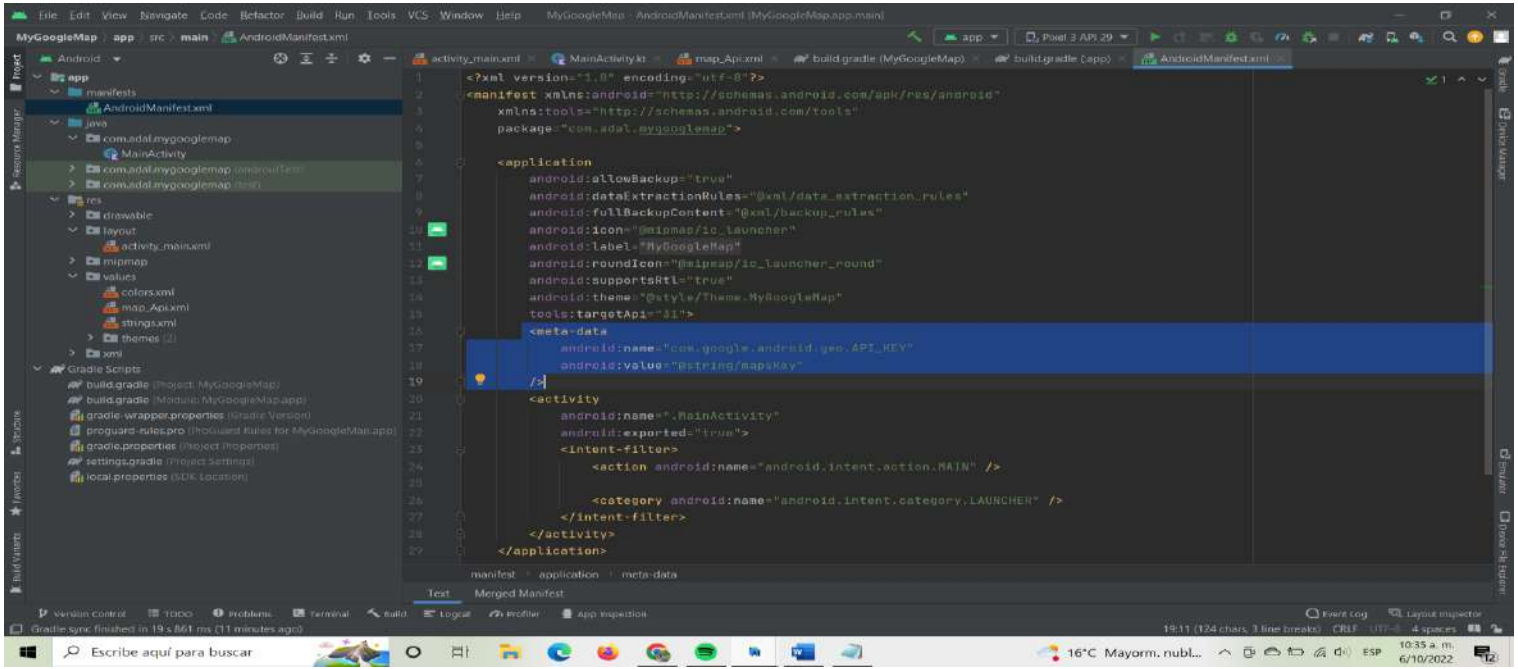
Añadir la dependencia de Google maps

```
implementation 'com.google.android.gms:play-services-maps:18.1.0'
```



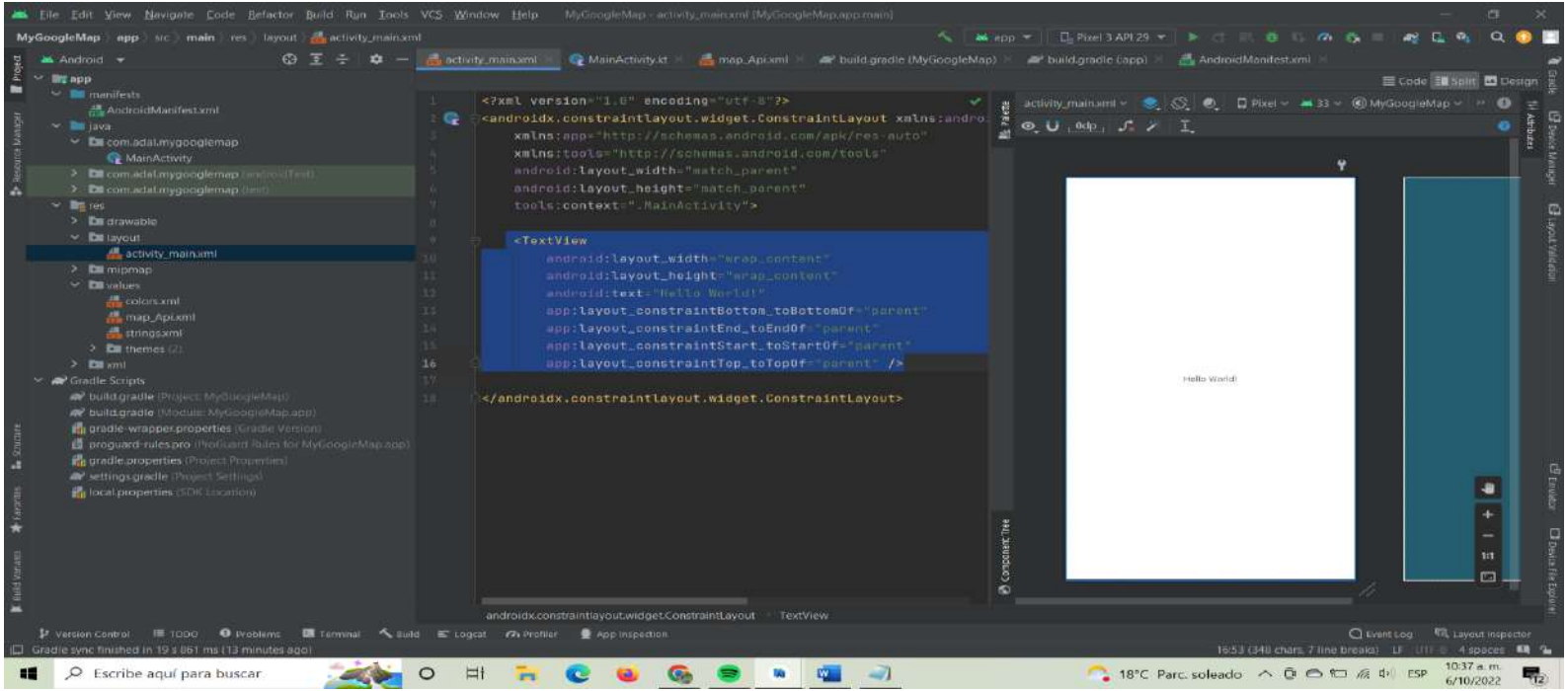
# Maps para Android

Luego vamos al Manifest



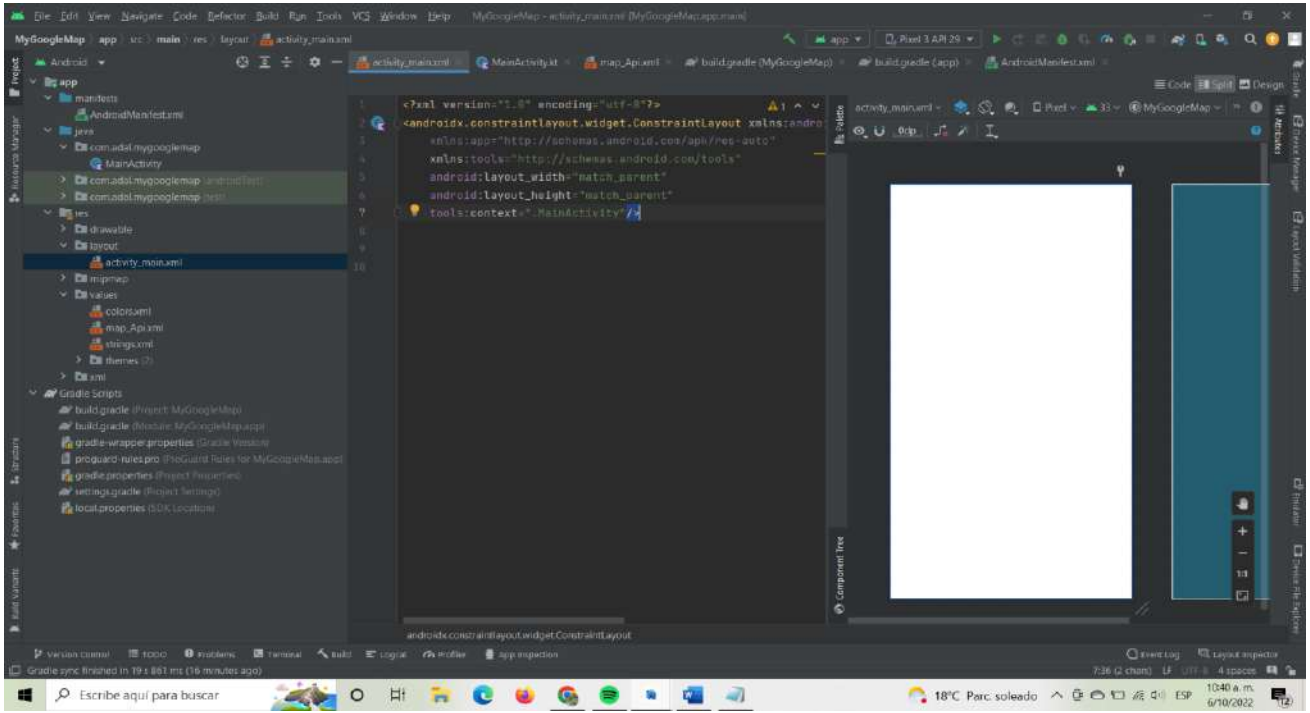
# Maps para Android

En el código del activity Main borramos  el objeto text que trae por defecto.



# Maps para Android

Cerramos la etiqueta final



# Maps para Android

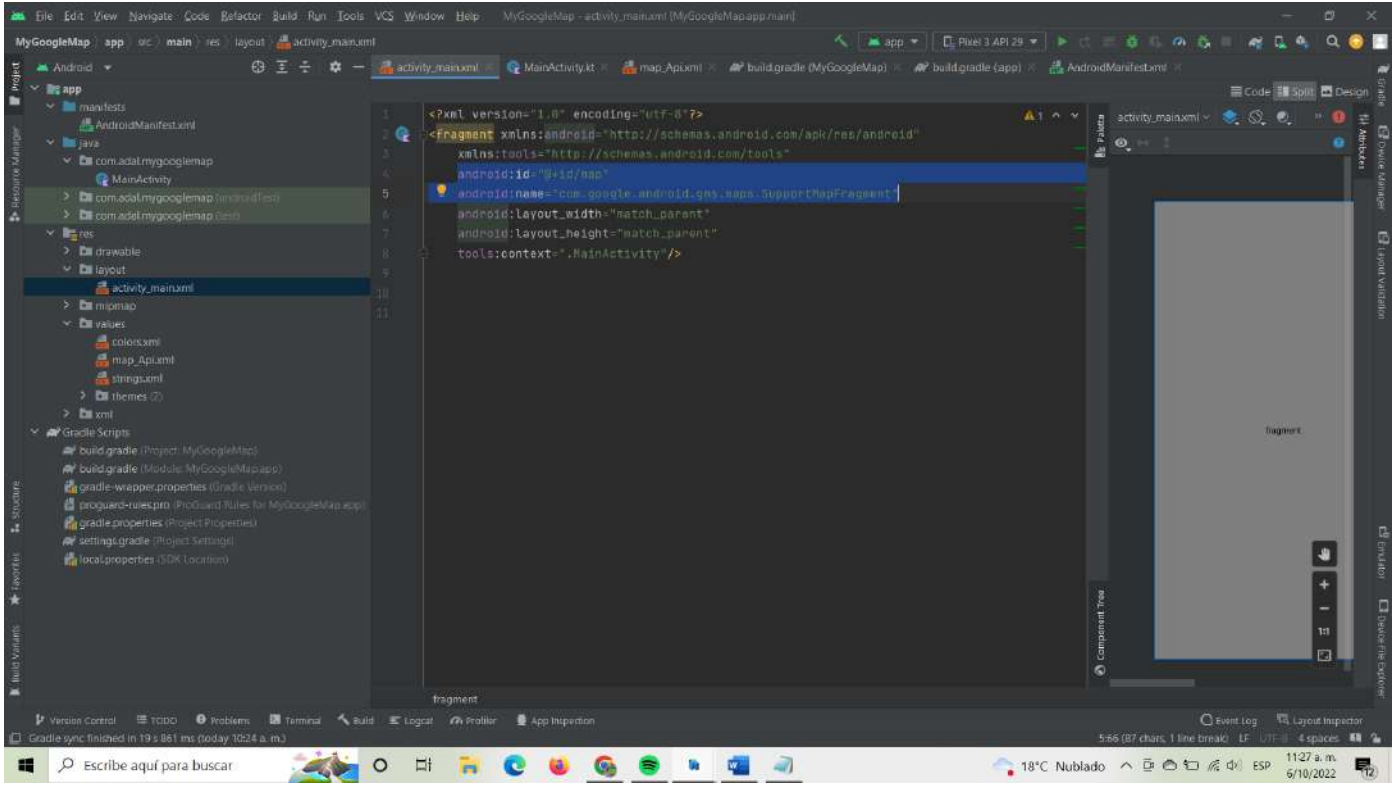
Nos quedaría d ela siguiente manera cambiando por un fragment

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"/>
```

Asignamos la siguiente línea

```
android:id="@+id/map"
android:name="com.google.android.gms.maps.SupportMapFragment"
```

# Maps para Android



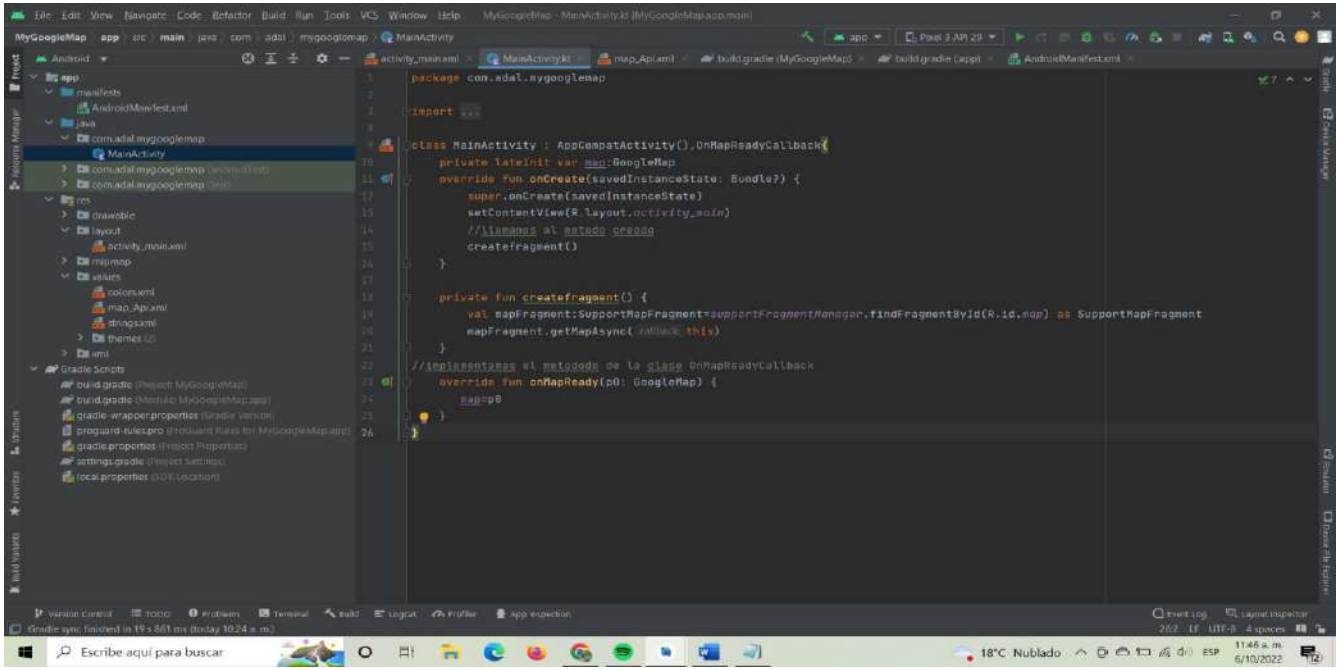
# Maps para Android

En el código del MainActivity

```
class MainActivity : AppCompatActivity(),OnMapReadyCallback{
    private lateinit var map:GoogleMap
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        //llamamos al metodo creado
        createfragment()
    }

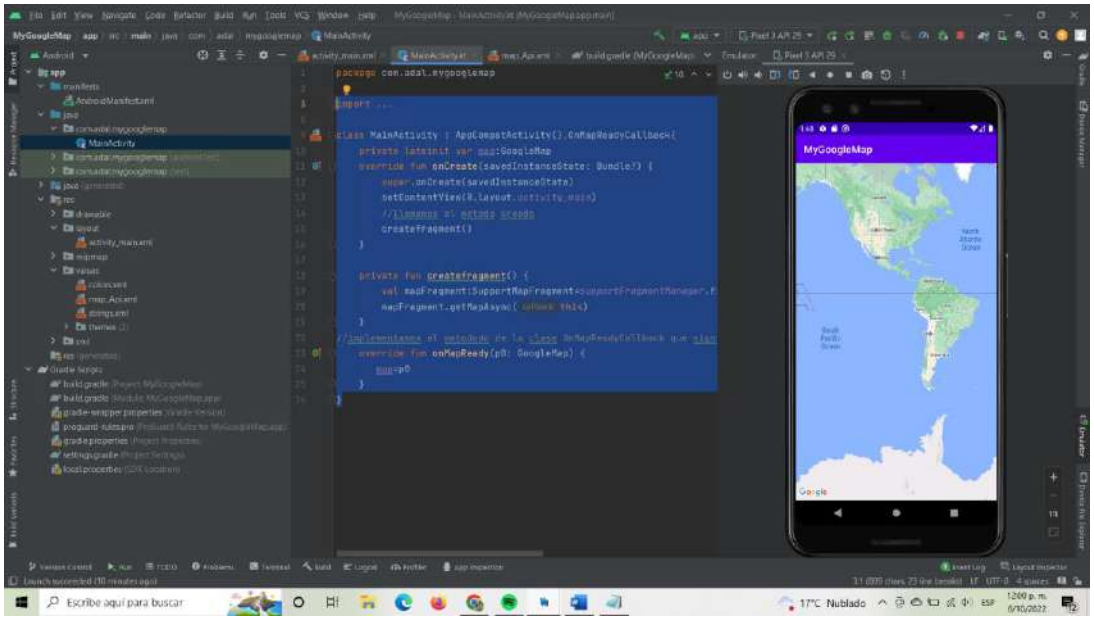
    private fun createfragment() {
        val mapFragment:SupportMapFragment=supportFragmentManager.findFragmentById(R.id.map)
as SupportMapFragment
        mapFragment.getMapAsync(this)
    }
//implementamos el metododo de la clase OnMapReadyCallback
    override fun onMapReady(p0: GoogleMap) {
      map=p0
    }
}
```

# Maps para Android



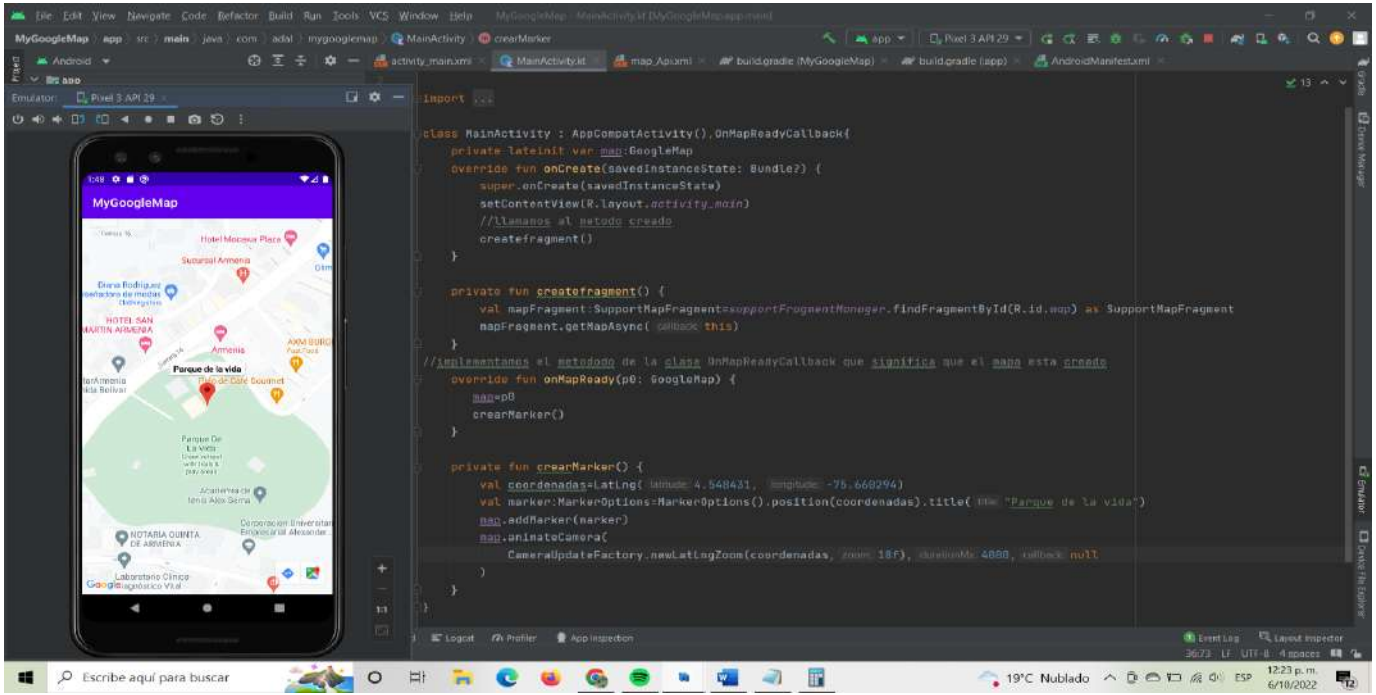
# Maps para Android

Ejecutamos la aplicación



# Maps para Android

Crear marca en el mapa






# GRACIAS

GC-F-004v.06

El empleo es de todos

Mintrabajo

# Análisis y Desarrollo de Sistemas de Información

SENA

# Conceptos Integración Continua, DevOps, Versión final manual de configuración del sistema, Versión final Documento Informe General del sistema



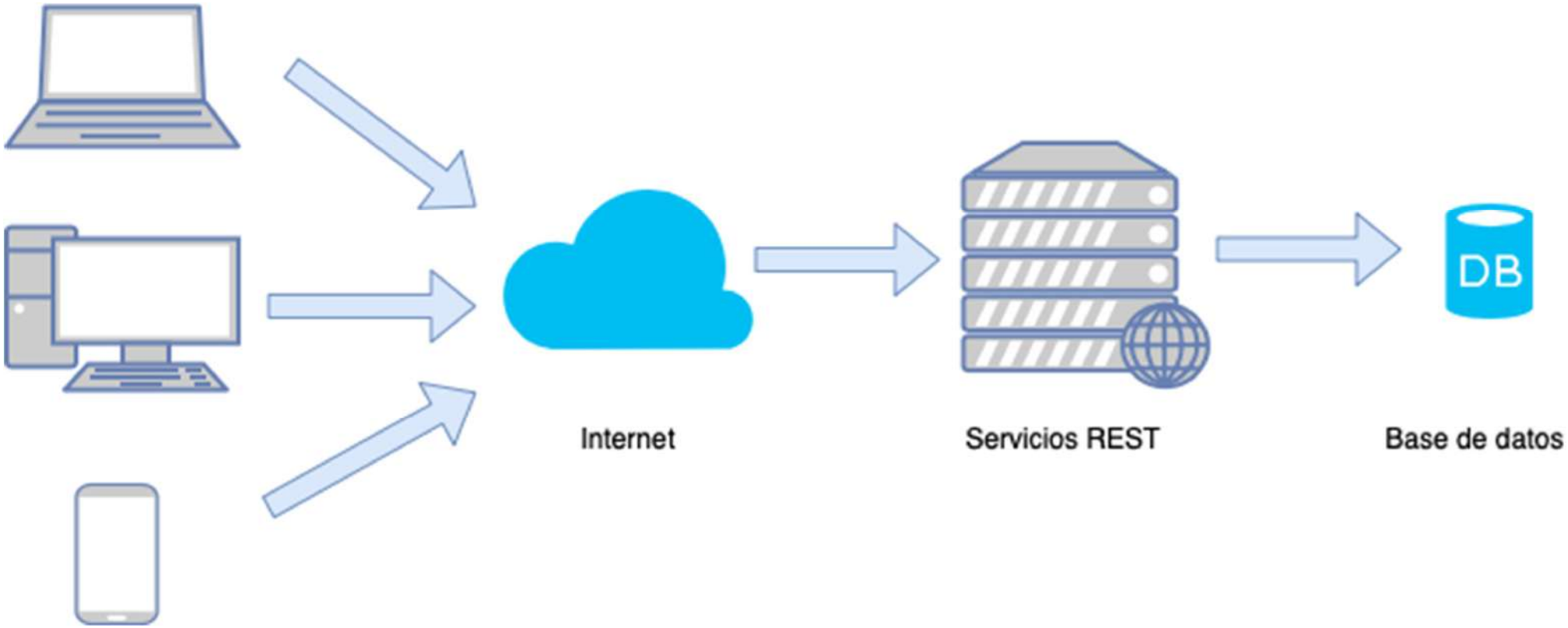
Una aplicación web es un conjunto de archivos que se almacenan en un servidor y que pueden ser entregados a petición. La comunicación suele efectuarse a través de internet.

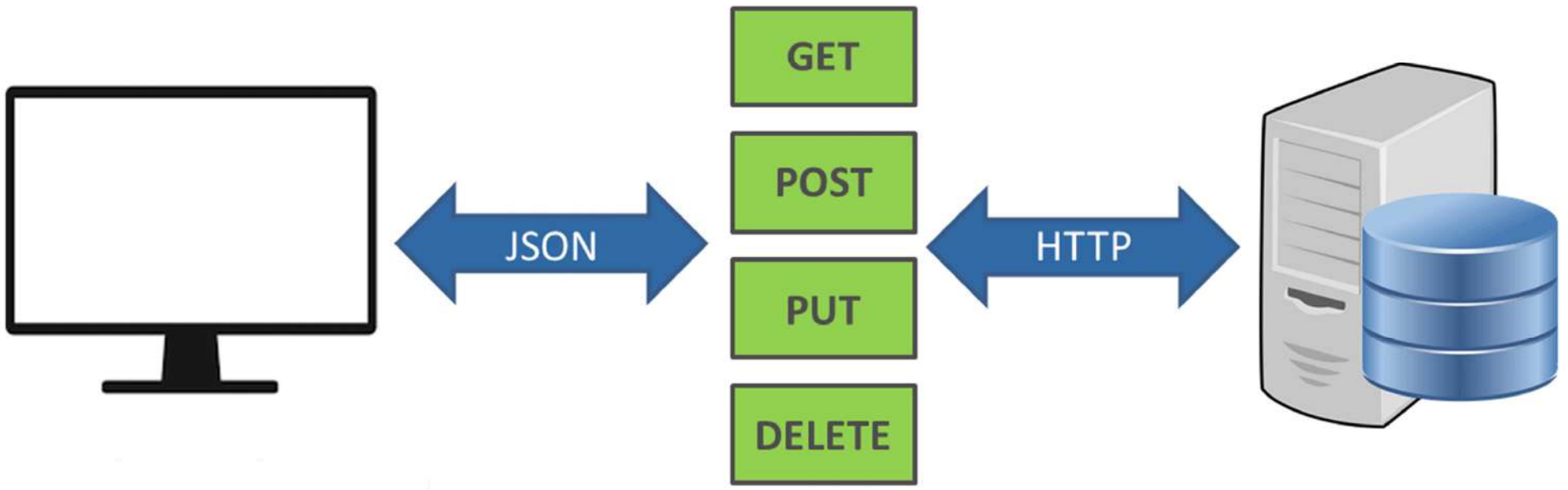
Un cliente permite la recuperación y el despliegue de dichos contenidos. Esto funciona muy bien debido al alto nivel de estandarización que se logro, se basa en el protocolo http.

Transparencia de la ubicación ¿A alguno le importa en que lugar esta el servidor al que acceden?

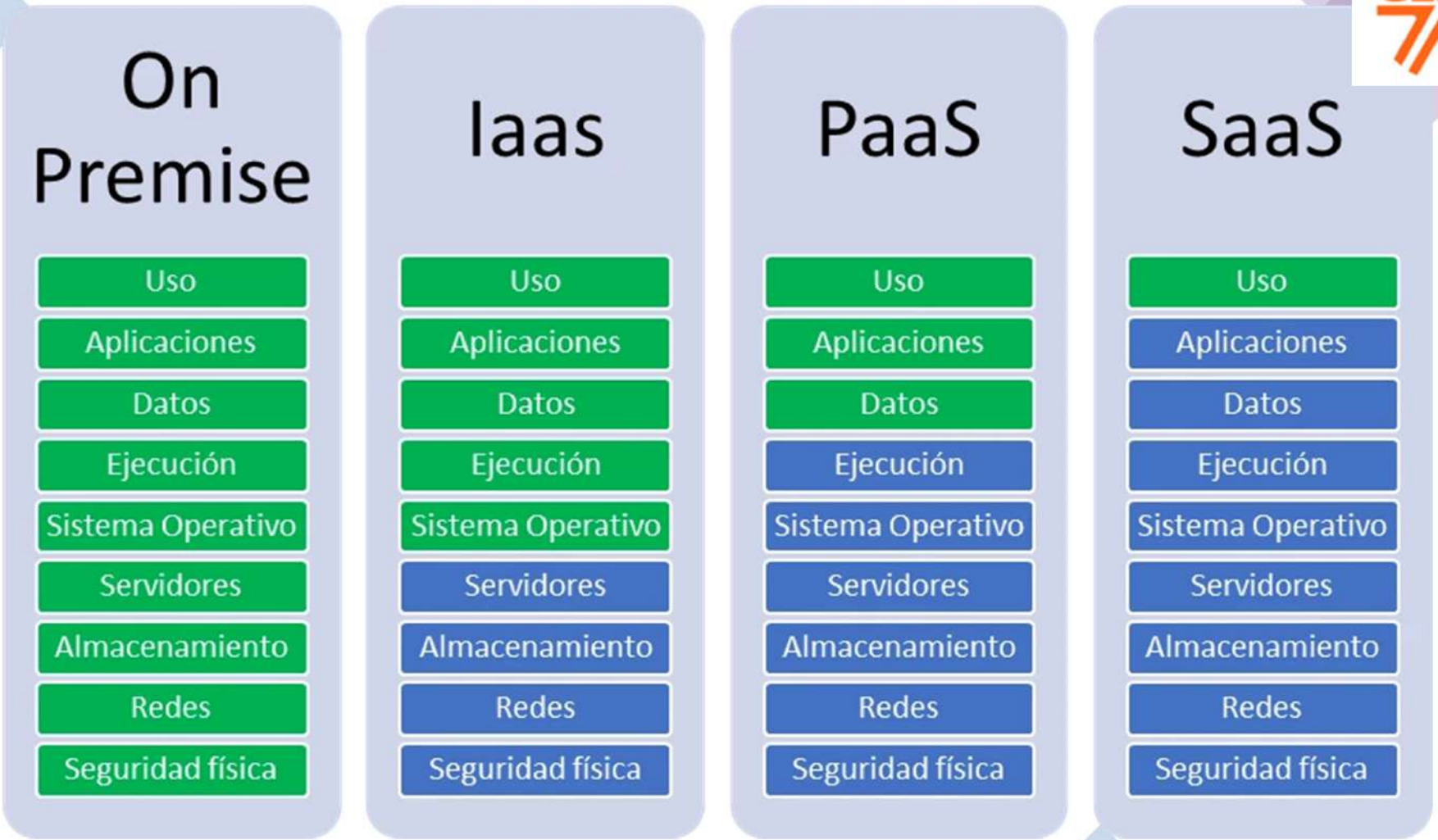
Transparencia de plataforma ¿A alguien le interesa el sistema operativo del servidor que les responde




DB
Internet
Servicios REST
Base de datos


GET
POST
PUT
DELETE
JSON
HTTP
Client sends a **request**
**HTTP methods**
Server sends a **response**

| | WITH CLOUD | WITH ON-PREMISE |
|---|---|---|
| DATA SECURITY | Minimal risk of data breach | Company data is managed internally for increased security |
| COSTS | No increased costs for additional equipment or infrastructure | Large initial cash outlay for equipment (and every few years for new equipment or maintenance) |
| MAINTENANCE | No need for dedicated IT staff | Dedicated IT staff for equipment maintenance |
| AGILITY | Third party doing updates and product enhancements on a regular basis | Feature "lock-in"; usually limited to solution features at time of purchase |
| MOBILITY | Broad accessibility for internet-enabled devices | Access generally limited to devices on-network |


On Premise
Uso
Aplicaciones
Datos
Ejecución
Sistema Operativo
Servidores
Almacenamiento
Redes
Seguridad física
Iaas
Uso
Aplicaciones
Datos
Ejecución
Sistema Operativo
Servidores
Almacenamiento
Redes
Seguridad física
PaaS
Uso
Aplicaciones
Datos
Ejecución
Sistema Operativo
Servidores
Almacenamiento
Redes
Seguridad física
SaaS
Uso
Aplicaciones
Datos
Ejecución
Sistema Operativo
Servidores
Almacenamiento
Redes
Seguridad física

# Modelos de negocios

- IaaS :: Hardware(CPU, Memoria, Disco y RED).
- PaaS :: Hardware(CPU, Memoria, Disco y RED) + S.O + Aplicaciones básicas.
- SaaS :: Hardware(CPU, Memoria, Disco y RED) + S.O + Aplicaciones básicas + Aplicaciones a utilizar.
- LaaS :: Servicio de almacenamiento y análisis centralizados de logs (bitácoras de auditoria) aplicaciones.
- FaaS :: Funciones como servicios :: Pago por la ejecución de una rutina (Microservicio)

# Servicios en la nube

Almacenamiento en la nube :: Es una de las aplicaciones mas comunes de los servicios basados en la nube. Surge para resolver inconveniente de portabilidad, perdida de archivos (al activar de forma automática un sistema de gestión de versiones) y consumo de recursos en el equipo del usuario.

Actualmente se puede obtener el servicio de almacenamiento en la nube bajo dos de los modelos de negocios estudiados.

- PaaS :: AWS(Simple Storage Service) S3, S3 glacier, Google Cloud Platform, Microsoft Azure MASS, La forma natural de acceder a dichos servicios es a través de APIs, CLI.

- SaaS :: Provee una interfaz grafica para la gestión del almacenamiento. Google Drive, OneDrive, DropBox, etc..

# Bases de datos

Una base de dato representa una forma de almacenamiento persistente de grandes volúmenes de información; dicho almacenamiento recibe una organización que facilita la gestión de contenido CRUD.



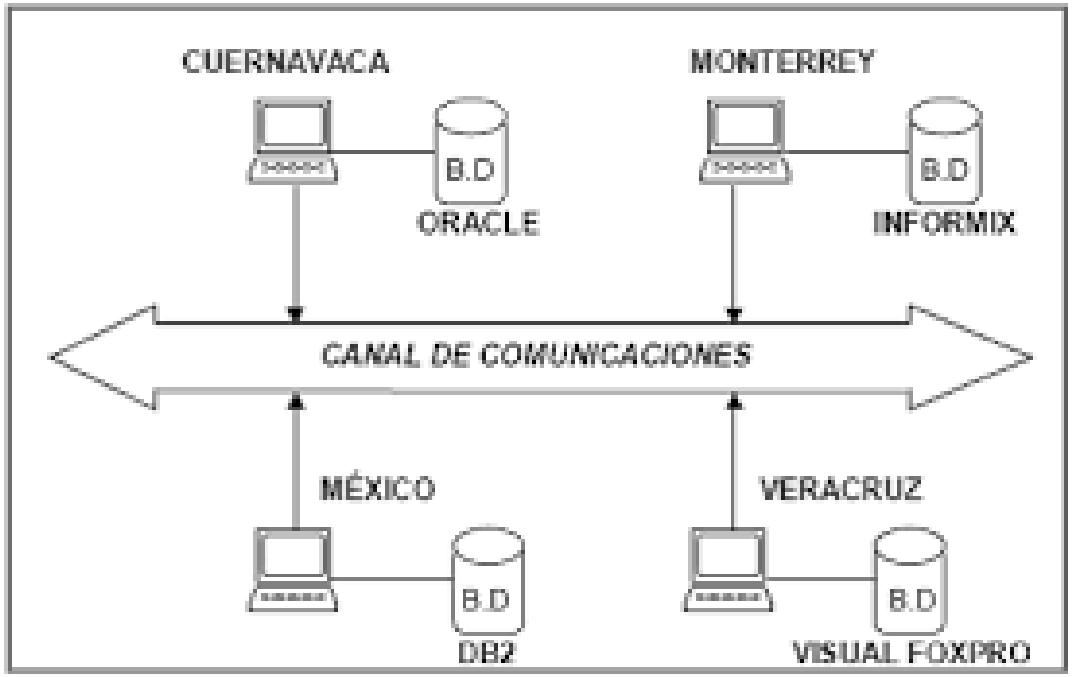
# Bases de datos centralizadas

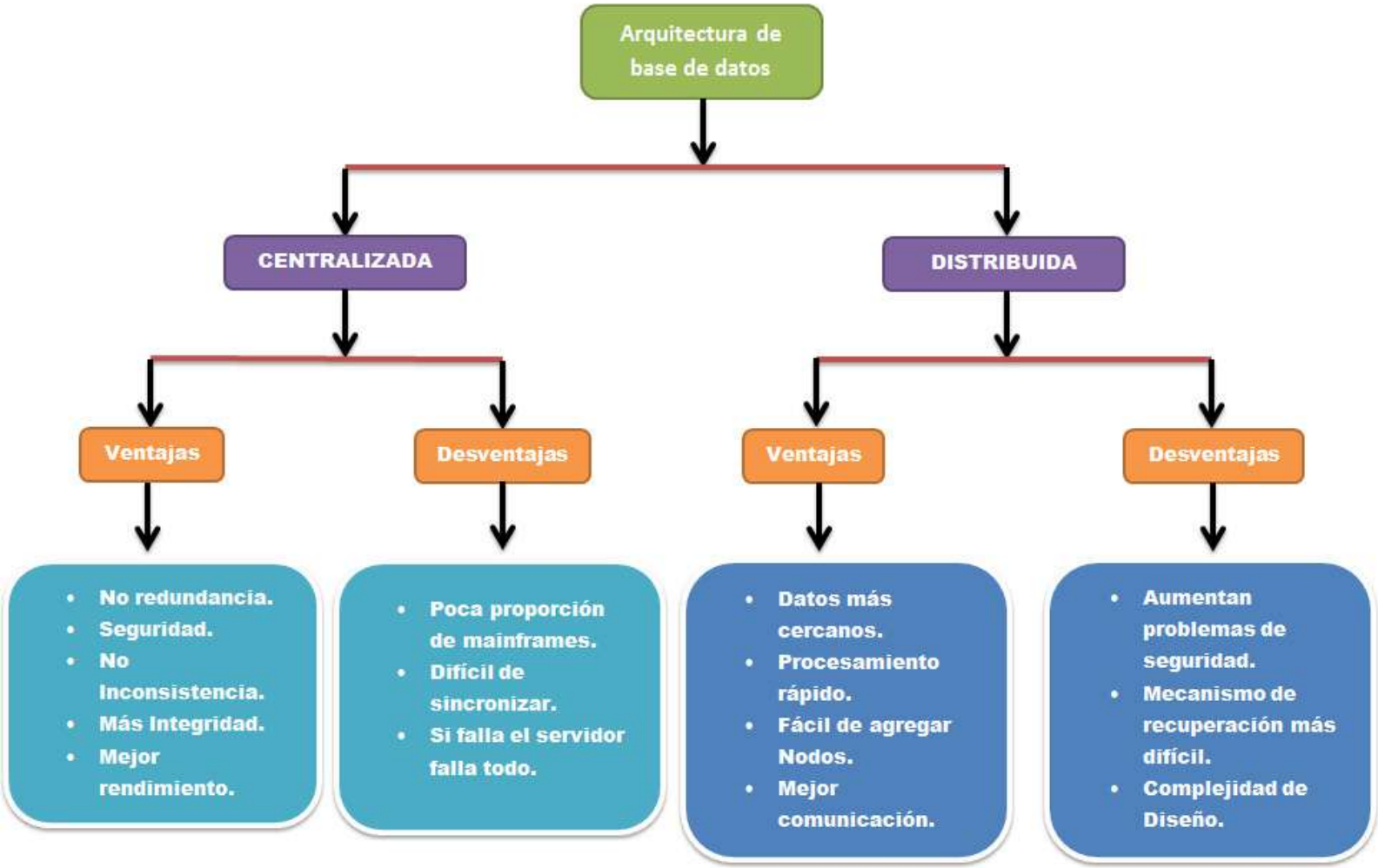
1. Existe un solo archivo de base  de datos el cual puede encontrarse en cualquiera de las ubicaciones físicas.
2. Facilidad en la gestión de la base de datos.
3. Facilidad en la recuperación de fallas.
4. Mejora en cuanto a la integridad de los datos.



# Bases de datos distribuidas

1. Permite la continuidad del negocio.
2. Pueden ejecutar operaciones en paralelos mejorando tiempo de respuesta y carga sobre los servidores.
3. Se puede mejorar el sistema introduciendo replicas.

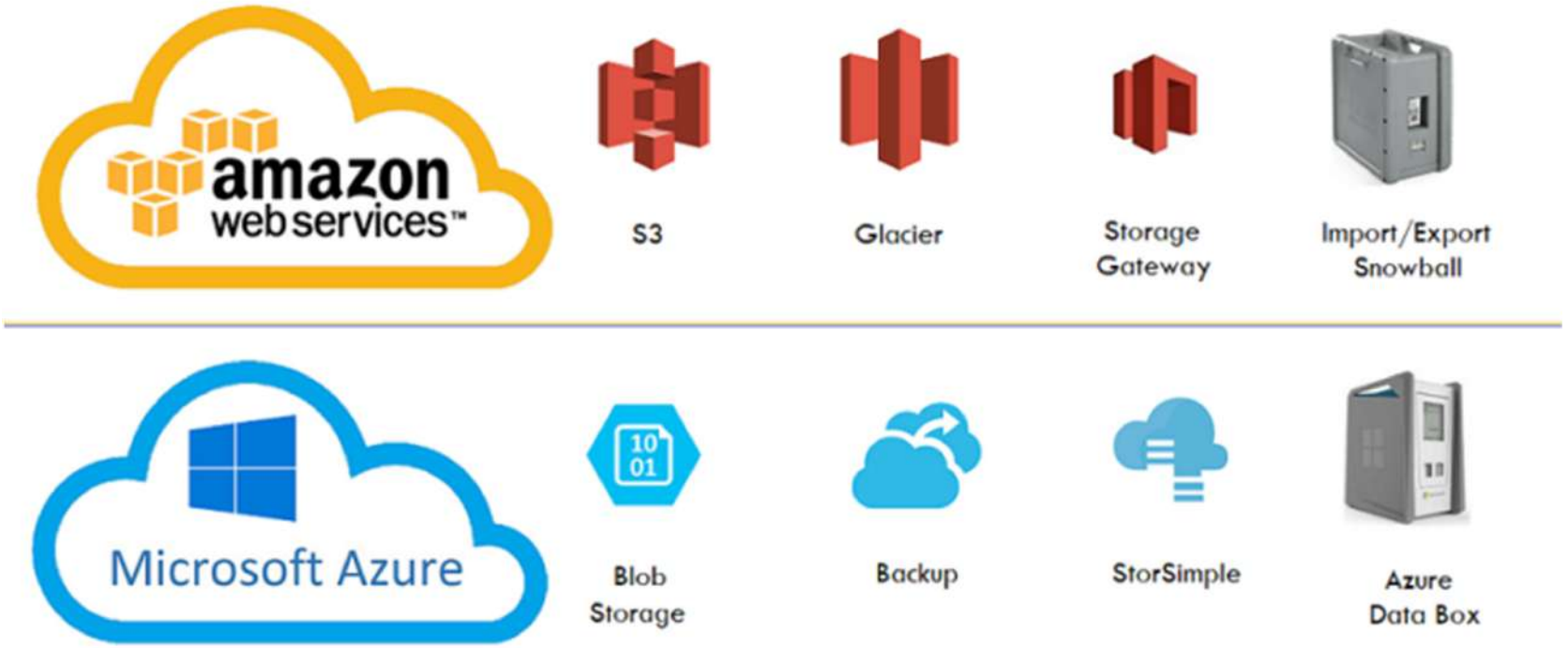



Arquitectura de base de datos
CENTRALIZADA
DISTRIBUIDA
Ventajas
Desventajas
Ventajas
Desventajas
No redundancia.
Seguridad.
No Inconsistencia.
Más Integridad.
Mejor rendimiento.
Poca proporción de mainframes.
Difícil de sincronizar.
Si falla el servidor falla todo.
Datos más cercanos.
Procesamiento rápido.
Fácil de agregar Nodos.
Mejor comunicación.
Aumentan problemas de seguridad.
Mecanismo de recuperación más difícil.
Complejidad de Diseño.

AWS
- RDS (SQL)
- Aurora (SQL)
- DynamoDB (NoSQL)

MongoDB Atlas
Firebase Firestore
ORACLE
IBM

Microsoft
- Azure SQL Database (SQL)
- Azure Cosmos DB (NoSQL)



# PaaS SQL DATABASES - DATABASE ENGINES AND RELEASE DATES

| SERVICE NAME | aws<br>AMAZON RDS (Relational Database Service) | Azure<br>AZURE SQL DATABASE MANAGED INSTANCE<br>AZURE SQL DATABASE FOR MySQL/PostgreSQL/MariaDB | GCP CLOUD SQL |
|---|---|---|---|
| SQL SERVER | MAY, 2012 | OCT, 2018 | FEB, 2020 |
| ORACLE | JUNE, 2011 | ✗ | ✗ |
| MySQL | OCT, 2009 | MARCH, 2018 | AUG, 2018 |
| PostgreSQL | NOV, 2013 | MARCH, 2018<br>HYPERSCALE (CITUS): MAY, 2019 | APR, 2018 |
| MariaDB | OCT, 2015 | DEC, 2018 | ✗ |




https://youtu.be/07mAdMTwRHs

# Referencias bibliograficas

https://aws.amazon.com/es/devops/continuous-integration/

https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/integracion-continua/

# GRACIAS

Línea de atención al ciudadano: 018000 910270
Línea de atención al empresario: 018000 910682

@SENAcomunica

www.sena.edu.co

GC-F-004v.06

El empleo es de todos

Mintrabajo

# Asistencia en Organización de Archivos

SENA

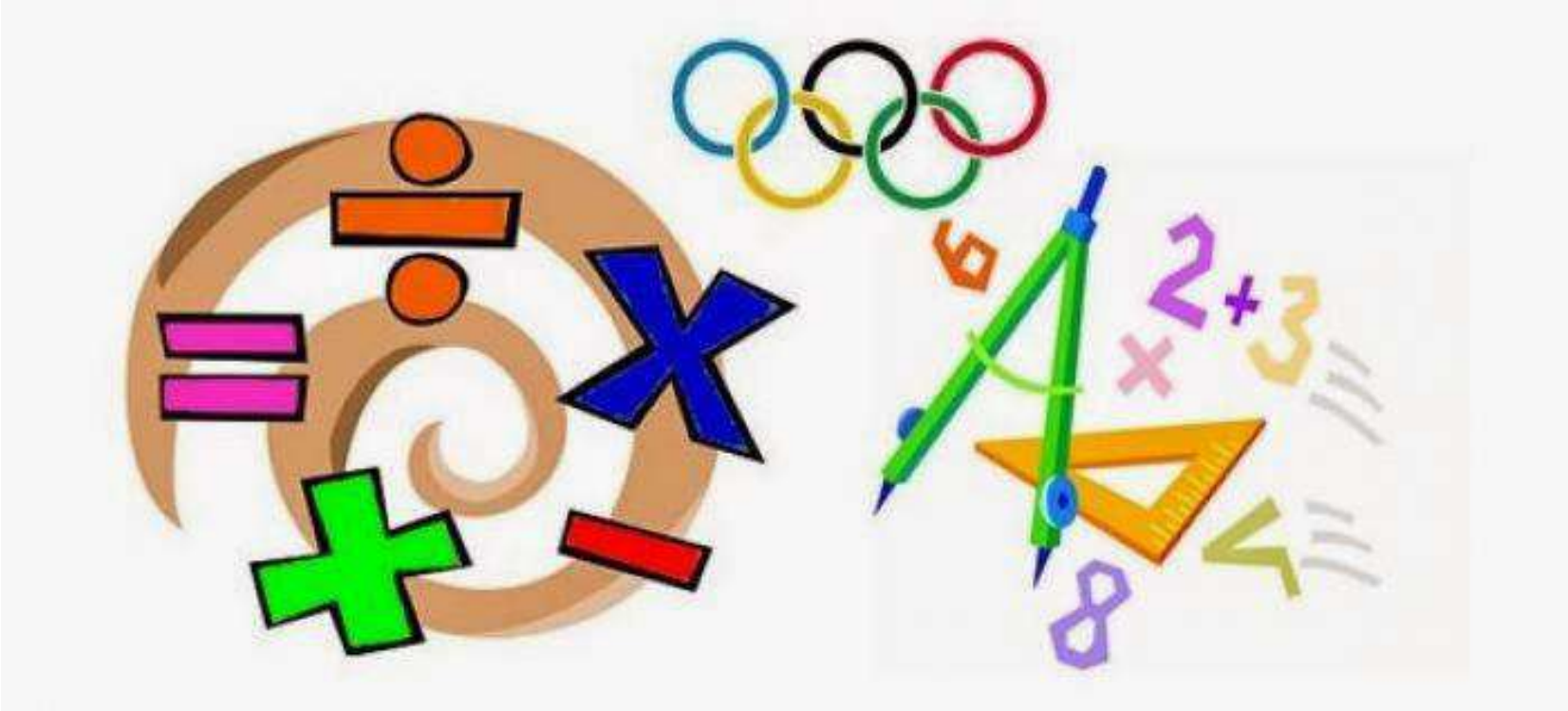
# II TRIMESTRE 2022

Gestor del Proyecto: Andres Hernando Zambrano Cardona
Instructor Transversal: Alvaro Antonio Rodríguez Fernández

# Orden del día

1. Presentación
2. Explicación metodología de aprendizaje
3. Evidencias de aprendizaje
4. Seguimiento evidencia de aprendizaje
5. Sección de dudas e inquietudes

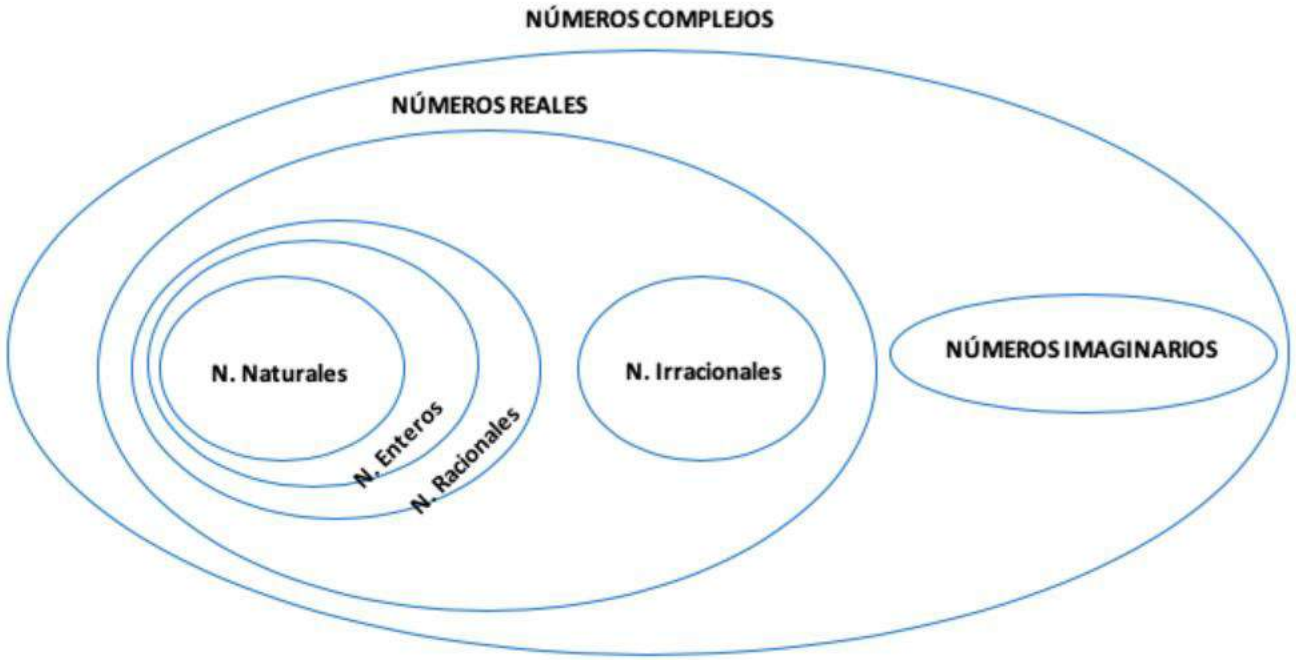
# 1. CONJUNTOS NUMÉRICOS



# 1. CONJUNTOS NUMÉRICOS

Los conjuntos numéricos son, en otras palabras, los tipos de números que las personas tenemos a nuestra disposición para realizar operaciones, tanto cotidianas como a un nivel más sofisticado (por parte de ingenieros o científicos, por ejemplo).

Estos conjuntos son creación de la mente humana, y forman parte de una abstracción. Es decir, no existen materialmente hablando.



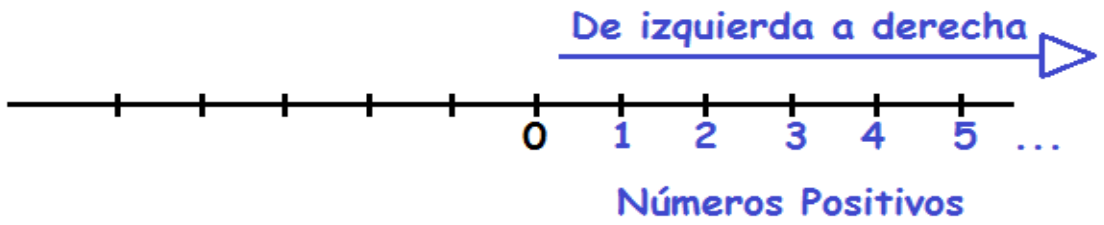
# NUMEROS NATURALES

Los números naturales son aquellos que toman intervalos discretos de una unidad, y empiezan con el número 1, extendiéndose hasta el infinito. Una forma de distinguir estos números es como aquellos que sirven para contar.

En términos formales, el conjunto de números naturales se expresa con la letra N y de la siguiente forma:

se caracteriza por:

- Tener un numero infinito de elemento
- Cada elemento tiene un sucesor y todos excepto el 1, tiene un antecesor
- El sucesor de un número natural, se obtiene sumando uno (+1); el antecesor se obtiene restando uno (-1)



# NÚMEROS ENTEROS

Surge de la necesidad de  incluir números negativos para dar solución a problemas de matemáticas como la sustracción pues cuando el sustraendo es mayor que el minuendo, esta sustracción no tiene solución en los conjuntos naturales, se denotan con la letra (z)

Este conjunto numérico nos permite realizar las operaciones básicas como:

- Para la suma y la resta se utiliza la resta numérica
- Para la multiplicación y división se hace uso de la ley de signos

**+ por + = +**  **- por - = +**

**+ por - = -**  **- por + = -**



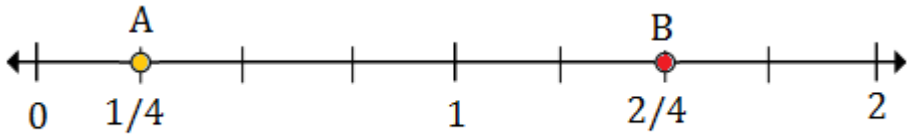
# NÚMEROS RACIONALES

El conjunto de los números racionales se creó debido a las limitaciones de cálculo que se presentaban en el conjunto de los números naturales y números enteros.

- Los números racionales son aquellos que nos permiten dividir una unidad en partes iguales y tomar de ella lo necesario, denotan con la letra (Q).

Como el resultado no es un número entero, Para solucionar esta dificultad, se creó este conjunto, el cual está formado por todos los números de la forma a/b. Esta fracción en la cual el numerador es a, es un número entero y el denominador b, es un número entero distinto de cero.

$$Q = \{. . ., -\tfrac{3}{4}, -\tfrac{1}{2}, -\tfrac{1}{4}, 0, \tfrac{1}{4}, \tfrac{1}{2}, \tfrac{3}{4}, . . .\}$$



Los números racionales incluyen no solo aquellos enteros, sino también los que pueden expresarse como el cociente de dos números enteros, de manera que pueden tener una parte decimal.

El conjunto de números racionales puede expresarse de la siguiente forma:
Números Racionales

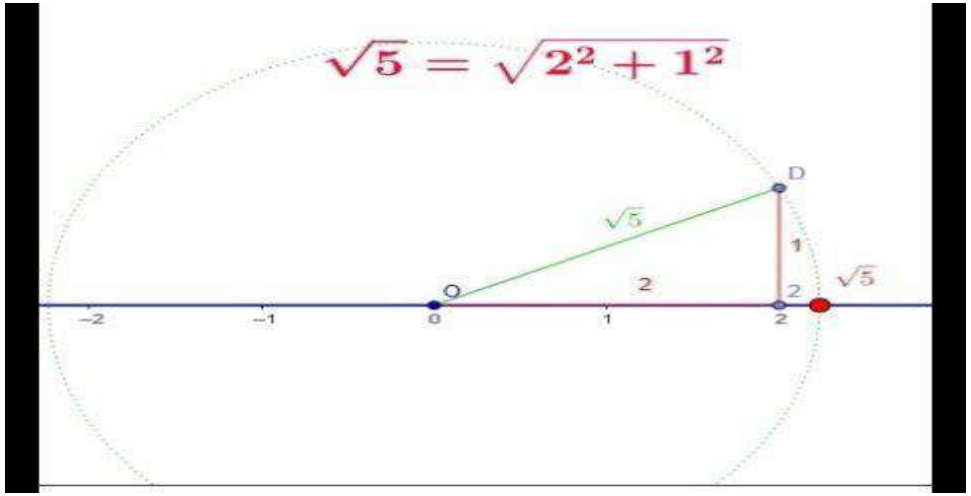
$$\mathbb{Q} = \frac{\mathbb{Z}}{\mathbb{Z}}$$

Conviene señalar que la parte decimal de u ro racional puede repetirse indefinidamente, caso en el cual se le denomina periódico. Así pues, puede tratarse de un periódico puro, cuando la parte decimal contiene uno o más números que se repiten al infinito, o un periódico mixto, cuando después de la coma decimal hay algún número, o algunos números, que no se repiten, mientras que el resto sí se prolonga al infinito

# Números irracionales

Denotado con la letra (Q*) con un asterisco, es un conjunto de números decimales infinitos no periódicos.
Este conjunto surgió de la necesidad de reunir a ciertos números que no pertenecen a los conjuntos anteriores.

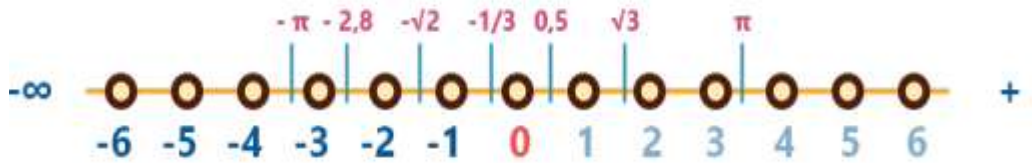
- Entre ellos se pueden citar a las raíces inexactas, el número Pi, etc.
- A él pertenecen todos los números decimales infinitos puros, es decir aquellos números que no pueden transformarse en una Fracción.



| Número | En fracción | Es… |
|---|---|---|
| 1,25 | 5/4 | Racional |
| 0,33333333333333… | 1/3 | Racional |
| 2 | 8/4 | Racional |
| 1,4142135623… (√2) | ¡No se puede! | Irracional |
| 0,97 | 97/100 | Racional |
| 3,1415926535… (π) | ¡No se puede! | Irracional |
| 1,090909090909090909… | 12/11 | Racional |

# Números reales

Denotados por la letra (R). Surgen de la necesidad de reunir los racionales y los irracionales en un solo conjunto. Se denotan por R.
R = { . . . , - 10, -1, - ¾, - ½, - ¼, 0, ¼ , √2, 5 , . . . }




**NUMEROS COMPLEJOS E IMAGINARIOS**
son utilizados frecuentemente en los campos de la física como en la mecánica cuántica, electrónica, telecomunicaciones, ingenierías, entre otras.

## Ejercicios propuestos

1. Determinar el menor conjunto numérico al cual pertenecen los siguientes números y ubicarlos en la recta real:

| | | |
|---|---|---|
| $a.\ -12$ | $d.\ \frac{16}{3}$ | $g.\ -4,161616...$ |
| $b.\ 3$ | $e.\ \sqrt[3]{8}$ | $h.\ \frac{11}{2}$ |
| $c.\ 1,25$ | $f.\ \sqrt{7}$ | $i.\ -\pi.$ |

2. Hallar la representación decimal de los siguientes números racionales:

| | | |
|---|---|---|
| $a.\ \frac{2}{5}$ | $d.\ \frac{2}{9}$ | $g.\ \frac{7}{3}$ |
| $b.\ -\frac{1}{5}$ | $e.\ \frac{3}{8}$ | $h.\ -\frac{17}{6}$ |
| $c.\ \frac{5}{2}$ | $f.\ -\frac{8}{5}$ | $i.\ \frac{4}{2}$ |

## Respuestas

1. *a.* $\mathbb{Z}$ *d.* $\mathbb{Q}$ *g.* $\mathbb{Q}$

   *b.* $\mathbb{N}$ *e.* $\mathbb{N}$ *h.* $\mathbb{Q}$

   *c.* $\mathbb{Q}$ *f.* $\mathbb{I}$ *i.* $\mathbb{I}$.

2. *a.* $0,4$ *d.* $0,\overline{2}$ *g.* $2,\overline{3}$

   *b.* $-0,2$ *e.* $0,375$ *h.* $-2,8\overline{3}$

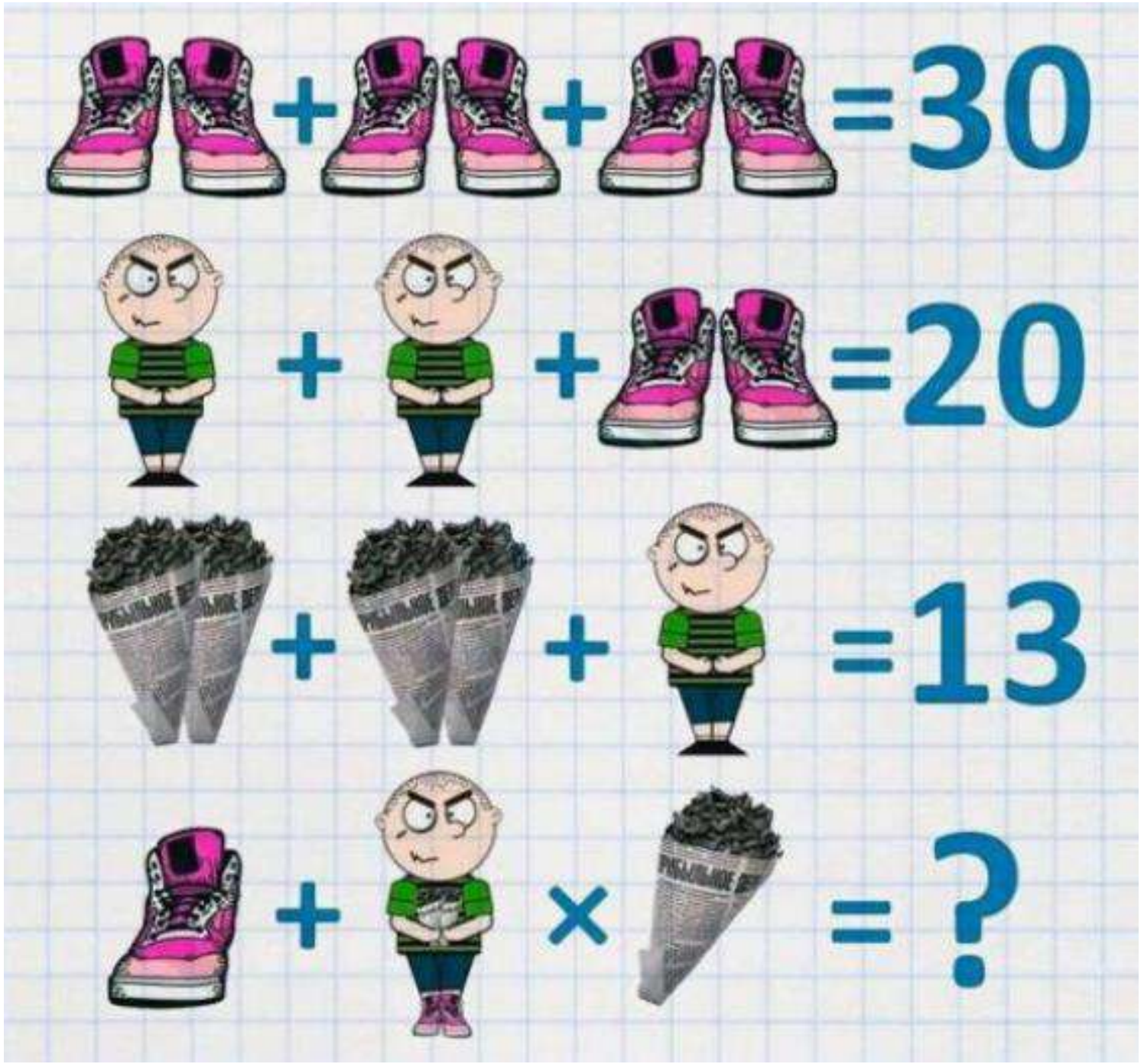
   *c.* $2,5$ *f.* $-1,6$ *i.* $2,0$.

# OPERACIONES ARITMETICAS

La aritmética es la rama de las matemáticas que relaciona los números y las diferentes operaciones
que se pueden realizar con los números (Suma, resta, multiplicación y división) así mismo las propiedades implícitas en dichas operaciones.

- para las operaciones combinadas en el conjunto de los números reales es necesario conocer la **jerarquía** que es una regla que nos determina cuáles operaciones se deben realizar primero en un determinado ejercicio que contenga varias operaciones .

1) Las operaciones que se encuentren agrupadas en paréntesis, llaves y corchetes.
2) Luego deben resolverse las potencias y raíces.
3) Posteriormente las divisiones y multiplicaciones en orden (De izquierda a derecha).
4) Por último debemos resolver las sumas y restas.


+ + =30
+ + =20
+ + =13
+ × =?

# Operaciones en números enteros

**Suma y resta de números enteros:**
Cuando los números enteros tienen el mismo signo. Se suman los valores y se deja el signo que tengan, si son positivos signo positivo y si son negativos signo negativo. Si no se pone nada delante del número se entiende que es +.

**Ejemplo:** (+6) + (+2) = +8
**Es los mismo que:** 6 + 2 = 8

- (-6) + ( -2) = - 8
- -6 - 2 = 8

**Cuando los números enteros tienen distinto signo**:
Se restan sus valores absolutos y se pone el signo del sumando de mayor valor absoluto. (Se restan y se deja el signo del más grande en valor absoluto).
**Ejemplo:**

- ( + 30 ) + ( - 20 ) = 30 – 20 = + 10
- ( - 8 ) + ( + 3 ) = - 8 + 3 = - 5

# Multiplicación y división de números enteros

Para estas operaciones es importante enunciar las reglas de los signos como se muestra a continuación:

| **Multiplicar** | **Dividir** |
|---|---|
| + por + = + | + entre + = + |
| - por - = + | - entre - = + |
| + por - = - | + entre - = - |
| - por + = - | - entre + = - |

**Para multiplicar dos números enteros:** Se multiplican sus valores absolutos y se aplica la regla de los signos. Cuando van dos signos seguidos hay que separarlos utilizando paréntesis.

**Ejemplo :**

- ( + 6 ) * ( + 4 ) = + 24
- ( - 4 ) * ( - 2 ) = + 8
- ( -2 ) * ( 3 ) = - 6

**Para dividir dos números enteros: Se** divide el dividendo entre el divisor y se aplica la regla de los signos. Una división es exacta cuando el resto es 0.

**ejemplo:**

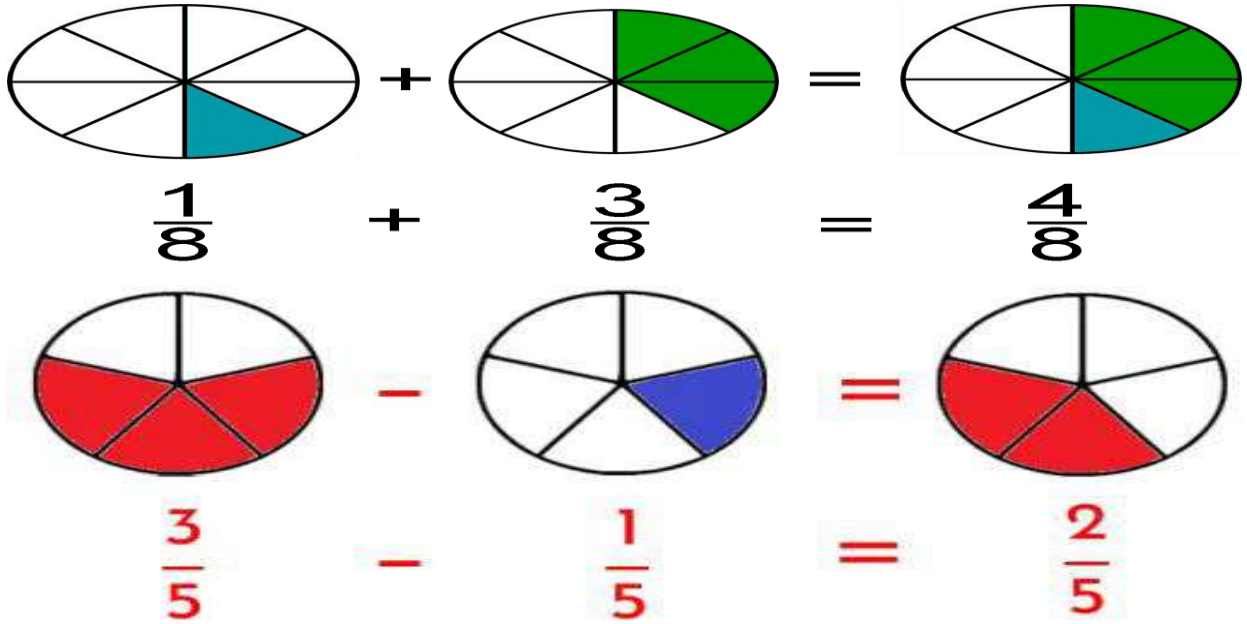
- ( - 15 ) ÷ ( 3 ) = - 5
- 6 ÷ 2 = + 3

# Operaciones en los números racionales

Recordemos que los números racionales también pueden ser nombrados como fracciones.
Y que una fracción es dividir cualquier cantidad en cantidades iguales y tomar algunas de esas partes.

**Operaciones de suma y resta de fracciones homogéneas:** Son aquellas que tienen igual denominador, Para sumar o restar números racionales homogéneos se coloca el mismo denominador y se suman o restan los numeradores.

**Ejemplo:**



# OPERACIONES CON FRACCIONES HETEROGÉNEAS.

**Para adicionar o sustraer números racionales heterogéneos:** se multiplican los denominadores entre sí y ese será el denominador, Se multiplican el numerador de la primera fracción por el denominador de la segunda, Se coloca el signo más. ( +,- ) Se multiplica el numerador de la segunda fracción por el denominador de la primera, Por último, sumamos los dos números que nos quedan en el en la parte de arriba.

**Ejemplo:**

$$\frac{5}{6} + \frac{3}{10} = \frac{(5 \times 10) + (6 \times 3)}{6 \times 10}$$
$$= \frac{68}{60}$$
$$= \frac{17}{15}$$

# Multiplicación De Números Racionales (Fracciones)

El producto de dos números racionales es otro número racional y se realiza multiplicando los numeradores entre sí y denominadores entre sí.

$$\frac{8}{3} x \frac{4}{3} = \frac{8x4}{3x3} = \frac{32}{9}$$

# División de fracciones

Para dividir dos números racionales se multiplica el numerador del primer número racional con el denominador del segundo número racional, y el numerador del segundo número con el denominador del primero. (Se multiplica en cruz)

$$\frac{4}{3} \cdot \frac{5}{4} = \frac{20}{12}$$

# Potencias y raíces

**La potenciación** es la multiplicación de un número (n) veces, el mismo número entero o racional, en el la base es el número racional que multiplicaremos y n (Denominado exponente) es el número de veces que multiplicamos la base.

**Propiedades de la potenciación:**

| *Propiedad* | *Exposición* | *Ejemplo* |
|---|---|---|
| **Potencia de base cero.** | Cualquier potencia de base 0 es igual a 0. $0^m = 0$ | $0^4 = 0$ |
| **Potencia de exponente cero.** | Cualquier potencia de exponente 0 es igual a 1. $X^0 = 1$ | $5^0 = 1$ |
| **Potencia de exponente 1.** | Cualquier potencia de exponente 1 es el mismo número. $X^1 = X$ | $7^1 = 7$ |
| **Producto de potencias de la misma base.** | El producto de potencias de la misma base es la misma base y el exponente la suma de los exponentes. $X^m + X^n = X^{m+n}$ | $2^4 * 2^6 = 2^{4+6} = 2^{10}$ |
| **Cociente de potencias de la misma base.** | El producto de potencias de la misma base es la misma base y el exponente de la suma de los exponentes. $X^m / X^n = X^{m-n}$ | $3^5 / 3^2 = 3^{5-2} = 3^3$ |
| **Potencia de una potencia.** | La potencia de una potencia es la misma base y el exponente el producto de los exponentes. $(X^m)^n = X^{m*n}$ | $(6^2)^4 = 6^{2*4} = 6^8$ |
| **Producto de potencias con el mismo exponente.** | El producto de potencias con el mismo exponente es una potencia con el mismo exponente y como base el producto de las bases. $X^m * Y^m = (X * Y)^m$ | $2^3 * 4^3 = (2 * 4)^3$ |

| *Propiedad* | *Exposición* | *Ejemplo* |
| --- | --- | --- |
| **Cociente de potencias con el mismo exponente.** | El cociente de potencias es una potencia con el mismo exponente y con base el cociente de las bases. $X^m / Y^m = (X / Y)^m$ | $6^3 / 2^3 = (6 / 2)^3 = 3^3$ |
| **Potencia de exponente negativo.** | La potencia de exponente negativo es igual al inverso de la potencia, con exponente positivo. $X^{-m} = 1 / X^m$ | $4^{-2} / 2 = 1 / 4^2$ |
| **Potencia de exponente racional.** | La potencia de exponente racional es igual a una raíz, donde el denominador es el índice de la raíz y el numerador el exponente del radicando. $X^{m/n} = \sqrt[n]{X^m}$ | $7^{4/3} = \sqrt[3]{7^4}$ |

# RADICACION

es conocida como la operación contraria a la potenciación en donde se conoce el índice de la raíz y el radicando de este modo se puede encontrar un número que multiplicado el número de veces que enuncie el índice nos de como resultado el radicando.

**Propiedades de la potencia**

| *Operación* | *Ejemplo* |
|---|---|
| Producto: $\sqrt[n]{a} * \sqrt[n]{b} = \sqrt[n]{a * b}$ | $\sqrt[3]{4} * \sqrt[3]{6} = \sqrt[3]{24}$ |
| Cociente: $\sqrt[n]{a} : \sqrt[n]{b} = \sqrt[n]{a / b}$ | $\sqrt[3]{4} : \sqrt[3]{6} = \sqrt[3]{4 / 6}$ |
| Potencia: $(\sqrt[n]{a})^m = \sqrt[n]{a^m}$ | $(\sqrt[5]{4})^6 = \sqrt[5]{4^6}$ |
| Simplificar: $(\sqrt[n]{a^n}) = a$ | $(\sqrt[3]{2^3}) = 2$ |
| Raíz de raíz: $(\sqrt[p]{\sqrt[n]{a}}) = \sqrt[n*p]{a}$ | $(\sqrt[4]{\sqrt[2]{3}}) = \sqrt[8]{3}$ |
| Introducir: $(a \sqrt[n]{a^n}) = \sqrt[n]{a^n * a} = \sqrt[n]{a^{n+1}}$ | $(4 \sqrt[3]{4}) = \sqrt[3]{4^3 * 4} = \sqrt[3]{4^4}$ |

# G R A C I A S

Línea de atención al ciudadano: 018000 910270
Línea de atención al empresario: 018000 910682

@SENAcomunica

www.sena.edu.co

GC-F-004v.06

El empleo es de todos

Mintrabajo

# Gestión Administrativa del Sector Salud

SENA

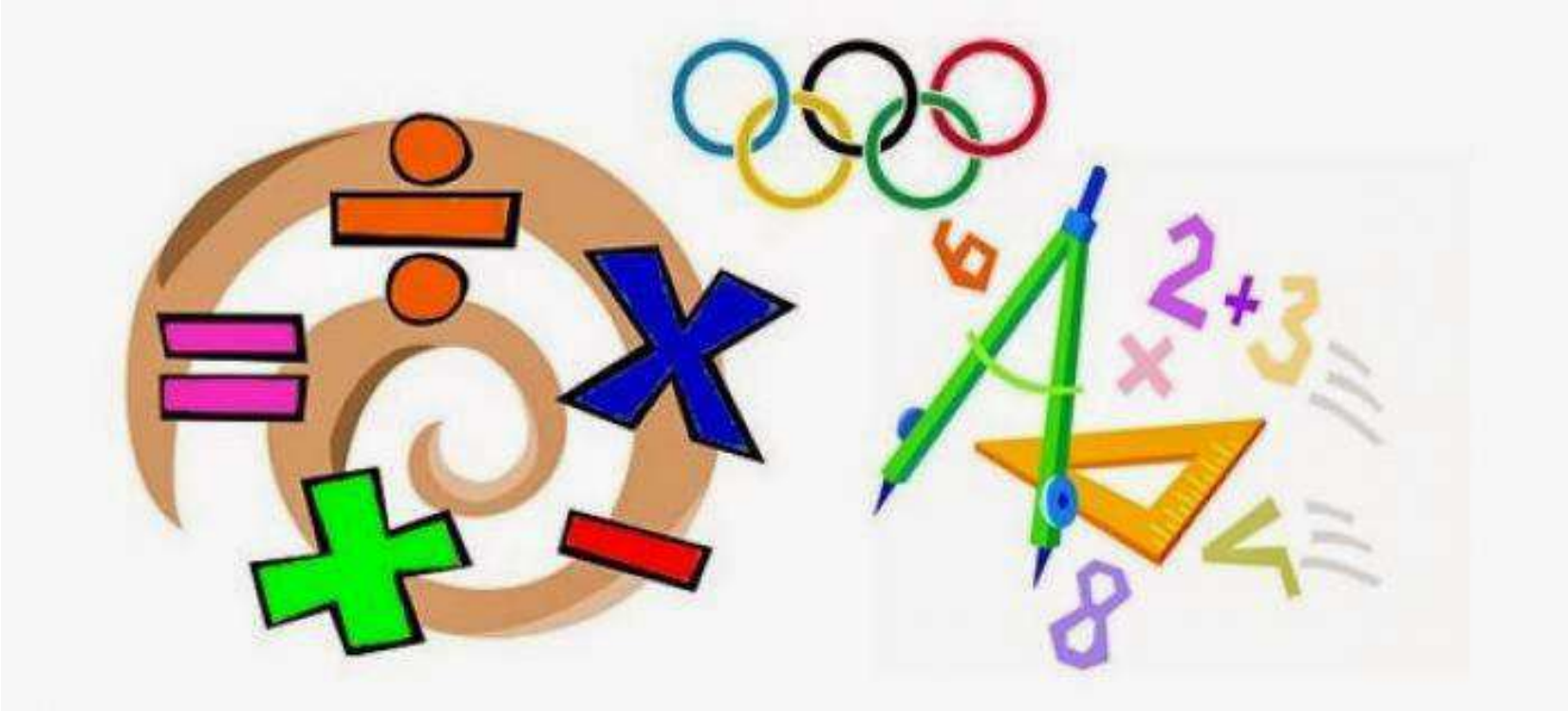
# II TRIMESTRE 2022

Gestor del Proyecto: Lindelia Peralta Paula
Instructor Transversal: Alvaro Antonio Rodríguez Fernández

# Orden del día

1. Presentación
2. Explicación metodología de aprendizaje
3. Evidencias de aprendizaje
4. Seguimiento evidencia de aprendizaje
5. Sección de dudas e inquietudes

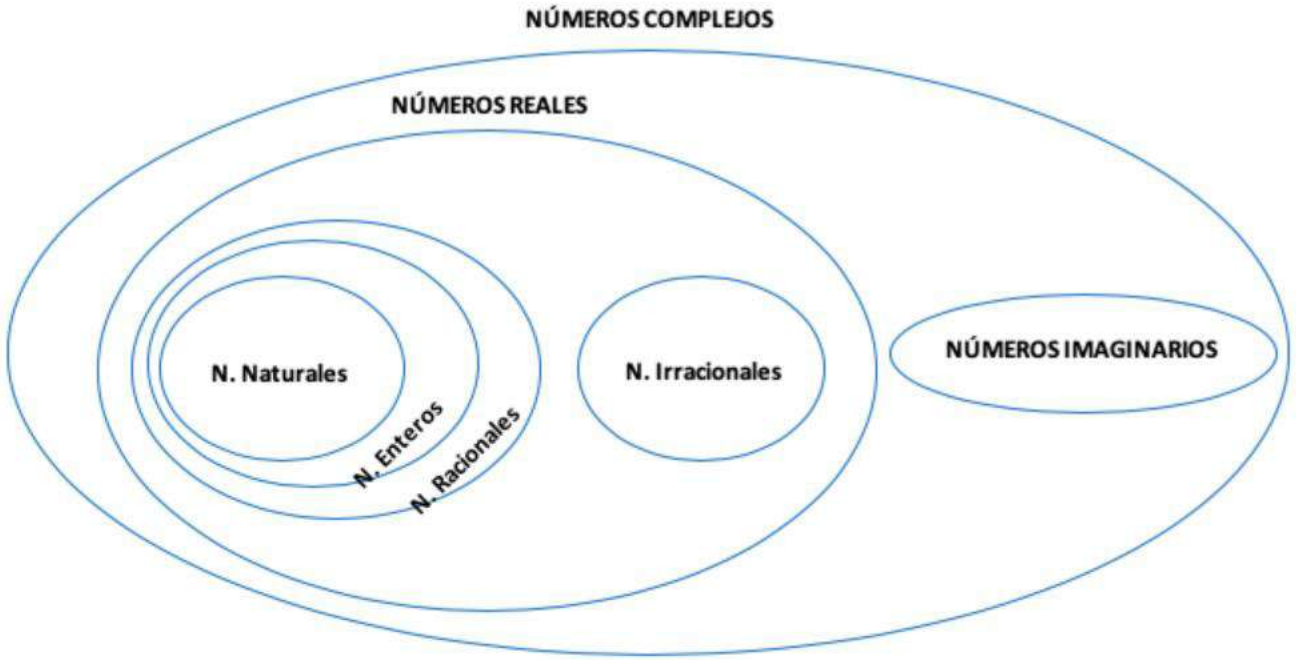
# 1. CONJUNTOS NUMÉRICOS



# 1. CONJUNTOS NUMÉRICOS

Los conjuntos numéricos son, en otras palabras, los tipos de números que las personas tenemos a nuestra disposición para realizar operaciones, tanto cotidianas como a un nivel más sofisticado (por parte de ingenieros o científicos, por ejemplo).

Estos conjuntos son creación de la mente humana, y forman parte de una abstracción. Es decir, no existen materialmente hablando.



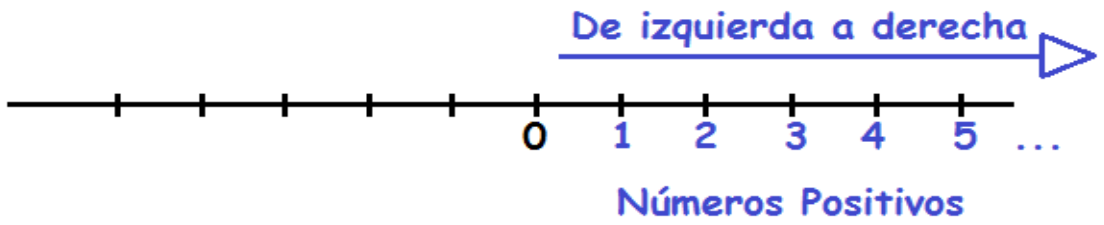
# NUMEROS NATURALES

Los números naturales son aquellos que toman intervalos discretos de una unidad, y empiezan con el número 1, extendiéndose hasta el infinito. Una forma de distinguir estos números es como aquellos que sirven para contar.

En términos formales, el conjunto de números naturales se expresa con la letra N y de la siguiente forma:

se caracteriza por:

- Tener un numero infinito de elemento
- Cada elemento tiene un sucesor y todos excepto el 1, tiene un antecesor
- El sucesor de un número natural, se obtiene sumando uno (+1); el antecesor se obtiene restando uno (-1)



# NÚMEROS ENTEROS

Surge de la necesidad de  incluir números negativos para dar solución a problemas de matemáticas como la sustracción pues cuando el sustraendo es mayor que el minuendo, esta sustracción no tiene solución en los conjuntos naturales, se denotan con la letra (z)

Este conjunto numérico nos permite realizar las operaciones básicas como:

- Para la suma y la resta se utiliza la resta numérica
- Para la multiplicación y división se hace uso de la ley de signos

**+ por + = +**  **- por - = +**

**+ por - = -**  **- por + = -**



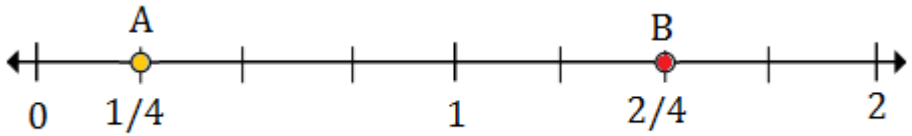
# NÚMEROS RACIONALES

El conjunto de los números racionales se creó debido a las limitaciones de cálculo que se presentaban en el conjunto de los números naturales y números enteros.

- Los números racionales son aquellos que nos permiten dividir una unidad en partes iguales y tomar de ella lo necesario, denotan con la letra (Q).

Como el resultado no es un número entero, Para solucionar esta dificultad, se creó este conjunto, el cual está formado por todos los números de la forma a/b. Esta fracción en la cual el numerador es a, es un número entero y el denominador b, es un número entero distinto de cero.

$$Q = \{. . ., -\tfrac{3}{4}, -\tfrac{1}{2}, -\tfrac{1}{4}, 0, \tfrac{1}{4}, \tfrac{1}{2}, \tfrac{3}{4}, . . .\}$$



Los números racionales incluyen no solo aquellos enteros, sino también los que pueden expresarse como el cociente de dos números enteros, de manera que pueden tener una parte decimal.

El conjunto de números racionales puede expresarse de la siguiente forma:
Números Racionales

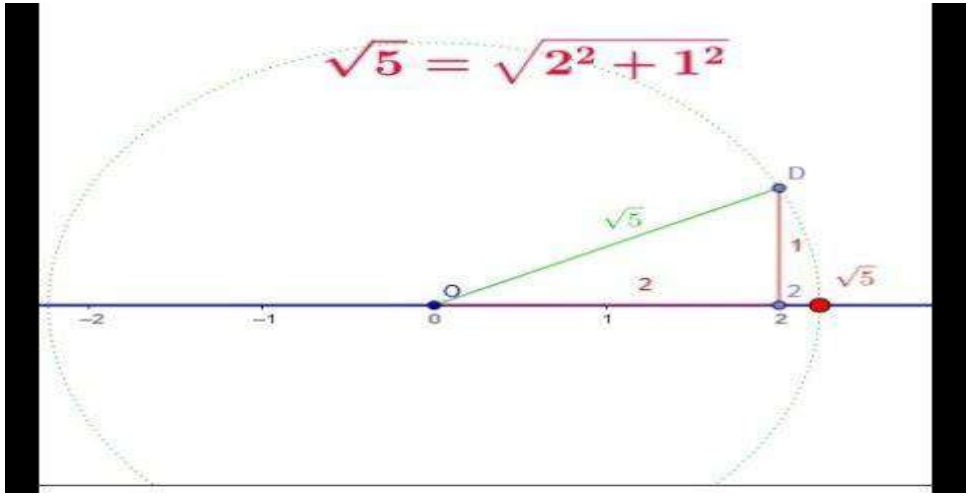
$$\mathbb{Q} = \frac{\mathbb{Z}}{\mathbb{Z}}$$

Conviene señalar que la parte decimal de u ro racional puede repetirse indefinidamente, caso en el cual se le denomina periódico. Así pues, puede tratarse de un periódico puro, cuando la parte decimal contiene uno o más números que se repiten al infinito, o un periódico mixto, cuando después de la coma decimal hay algún número, o algunos números, que no se repiten, mientras que el resto sí se prolonga al infinito

# Números irracionales

Denotado con la letra (Q*) con un asterisco, es un conjunto de números decimales infinitos no periódicos.
Este conjunto surgió de la necesidad de reunir a ciertos números que no pertenecen a los conjuntos anteriores.

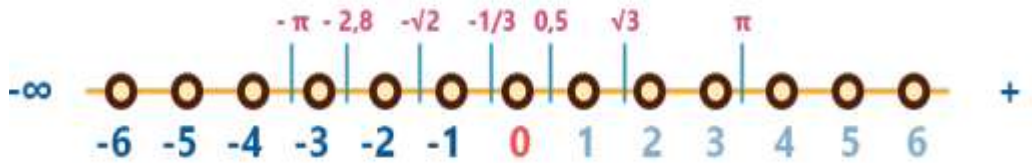
- Entre ellos se pueden citar a las raíces inexactas, el número Pi, etc.
- A él pertenecen todos los números decimales infinitos puros, es decir aquellos números que no pueden transformarse en una Fracción.



| Número | En fracción | Es… |
|---|---|---|
| 1,25 | 5/4 | Racional |
| 0,33333333333333… | 1/3 | Racional |
| 2 | 8/4 | Racional |
| 1,4142135623… (√2) | ¡No se puede! | Irracional |
| 0,97 | 97/100 | Racional |
| 3,1415926535… (π) | ¡No se puede! | Irracional |
| 1,090909090909090909… | 12/11 | Racional |

# Números reales

Denotados por la letra (R). Surgen de la necesidad de reunir los racionales y los irracionales en un solo conjunto. Se denotan por R.
R = { . . . , - 10, -1, - ¾, - ½, - ¼, 0, ¼ , √2, 5 , . . . }




**NUMEROS COMPLEJOS E IMAGINARIOS**
son utilizados frecuentemente en los campos de la física como en la mecánica cuántica, electrónica, telecomunicaciones, ingenierías, entre otras.

## Ejercicios propuestos

1. Determinar el menor conjunto numérico al cual pertenecen los siguientes números y ubicarlos en la recta real:

| | | |
|---|---|---|
| $a.\ -12$ | $d.\ \frac{16}{3}$ | $g.\ -4,161616...$ |
| $b.\ 3$ | $e.\ \sqrt[3]{8}$ | $h.\ \frac{11}{2}$ |
| $c.\ 1,25$ | $f.\ \sqrt{7}$ | $i.\ -\pi.$ |

2. Hallar la representación decimal de los siguientes números racionales:

| | | |
|---|---|---|
| $a.\ \frac{2}{5}$ | $d.\ \frac{2}{9}$ | $g.\ \frac{7}{3}$ |
| $b.\ -\frac{1}{5}$ | $e.\ \frac{3}{8}$ | $h.\ -\frac{17}{6}$ |
| $c.\ \frac{5}{2}$ | $f.\ -\frac{8}{5}$ | $i.\ \frac{4}{2}$ |

## Respuestas

1. *a.* $\mathbb{Z}$ *d.* $\mathbb{Q}$ *g.* $\mathbb{Q}$

   *b.* $\mathbb{N}$ *e.* $\mathbb{N}$ *h.* $\mathbb{Q}$

   *c.* $\mathbb{Q}$ *f.* $\mathbb{I}$ *i.* $\mathbb{I}$.

2. *a.* $0,4$ *d.* $0,\overline{2}$ *g.* $2,\overline{3}$

   *b.* $-0,2$ *e.* $0,375$ *h.* $-2,8\overline{3}$

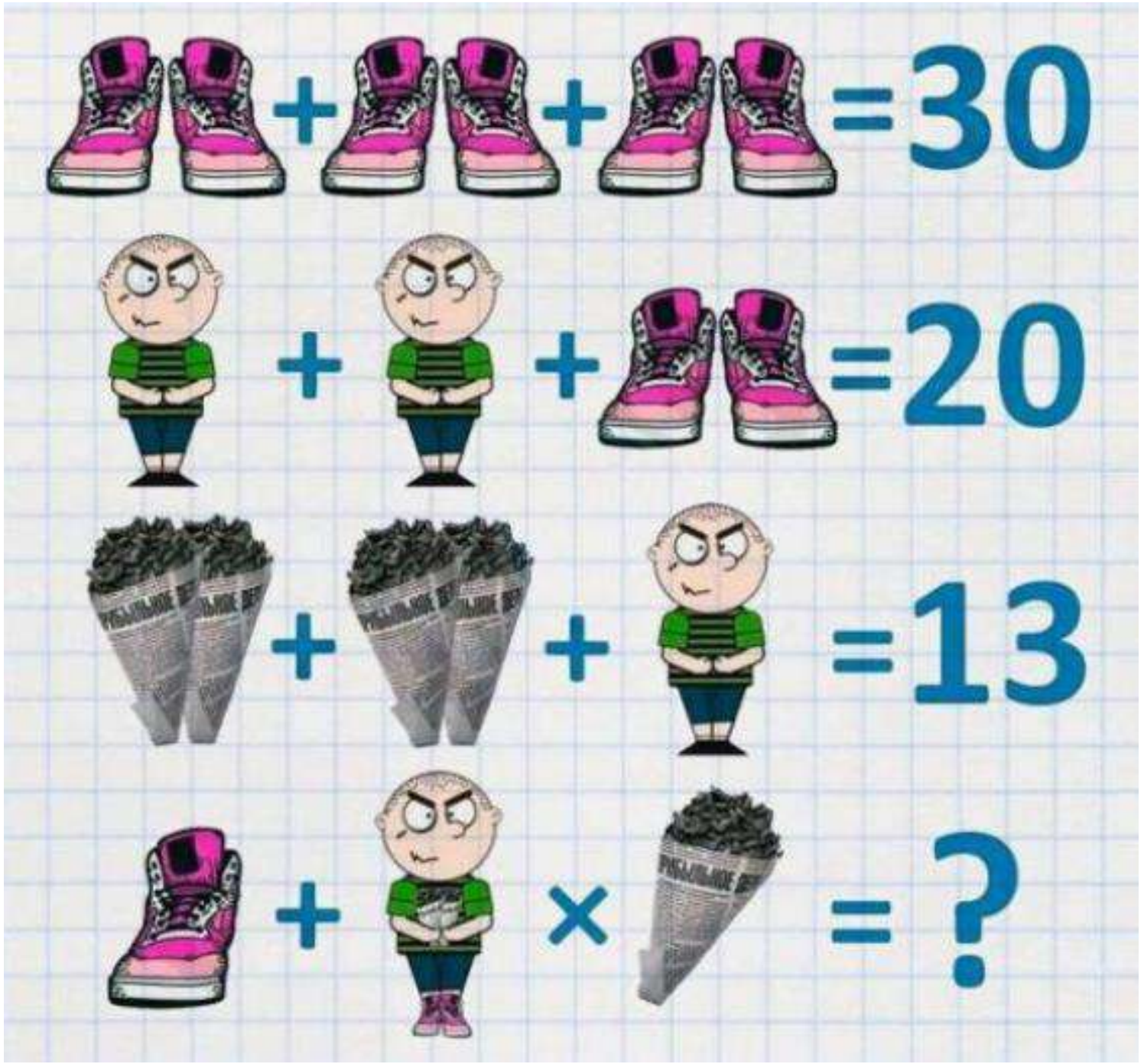
   *c.* $2,5$ *f.* $-1,6$ *i.* $2,0$.

# OPERACIONES ARITMETICAS

La aritmética es la rama de las matemáticas que relaciona los números y las diferentes operaciones
que se pueden realizar con los números (Suma, resta, multiplicación y división) así mismo las propiedades implícitas en dichas operaciones.

- para las operaciones combinadas en el conjunto de los números reales es necesario conocer la **jerarquía** que es una regla que nos determina cuáles operaciones se deben realizar primero en un determinado ejercicio que contenga varias operaciones .

1) Las operaciones que se encuentren agrupadas en paréntesis, llaves y corchetes.
2) Luego deben resolverse las potencias y raíces.
3) Posteriormente las divisiones y multiplicaciones en orden (De izquierda a derecha).
4) Por último debemos resolver las sumas y restas.


+ + =30
+ + =20
+ + =13
+ × =?

# Operaciones en números enteros

**Suma y resta de números enteros:**
Cuando los números enteros tienen el mismo signo. Se suman los valores y se deja el signo que tengan, si son positivos signo positivo y si son negativos signo negativo. Si no se pone nada delante del número se entiende que es +.

**Ejemplo:** (+6) + (+2) = +8
**Es los mismo que:** 6 + 2 = 8

- (-6) + ( -2) = - 8
- -6 - 2 = 8

**Cuando los números enteros tienen distinto signo**:
Se restan sus valores absolutos y se pone el signo del sumando de mayor valor absoluto. (Se restan y se deja el signo del más grande en valor absoluto).
**Ejemplo:**

- ( + 30 ) + ( - 20 ) = 30 – 20 = + 10
- ( - 8 ) + ( + 3 ) = - 8 + 3 = - 5

# Multiplicación y división de números enteros

Para estas operaciones es importante enunciar las reglas de los signos como se muestra a continuación:

| **Multiplicar** | **Dividir** |
|---|---|
| + por + = + | + entre + = + |
| - por - = + | - entre - = + |
| + por - = - | + entre - = - |
| - por + = - | - entre + = - |

**Para multiplicar dos números enteros:** Se multiplican sus valores absolutos y se aplica la regla de los signos. Cuando van dos signos seguidos hay que separarlos utilizando paréntesis.

**Ejemplo :**

- ( + 6 ) * ( + 4 ) = + 24
- ( - 4 ) * ( - 2 ) = + 8
- ( -2 ) * ( 3 ) = - 6

**Para dividir dos números enteros: Se** divide el dividendo entre el divisor y se aplica la regla de los signos. Una división es exacta cuando el resto es 0.

**ejemplo:**

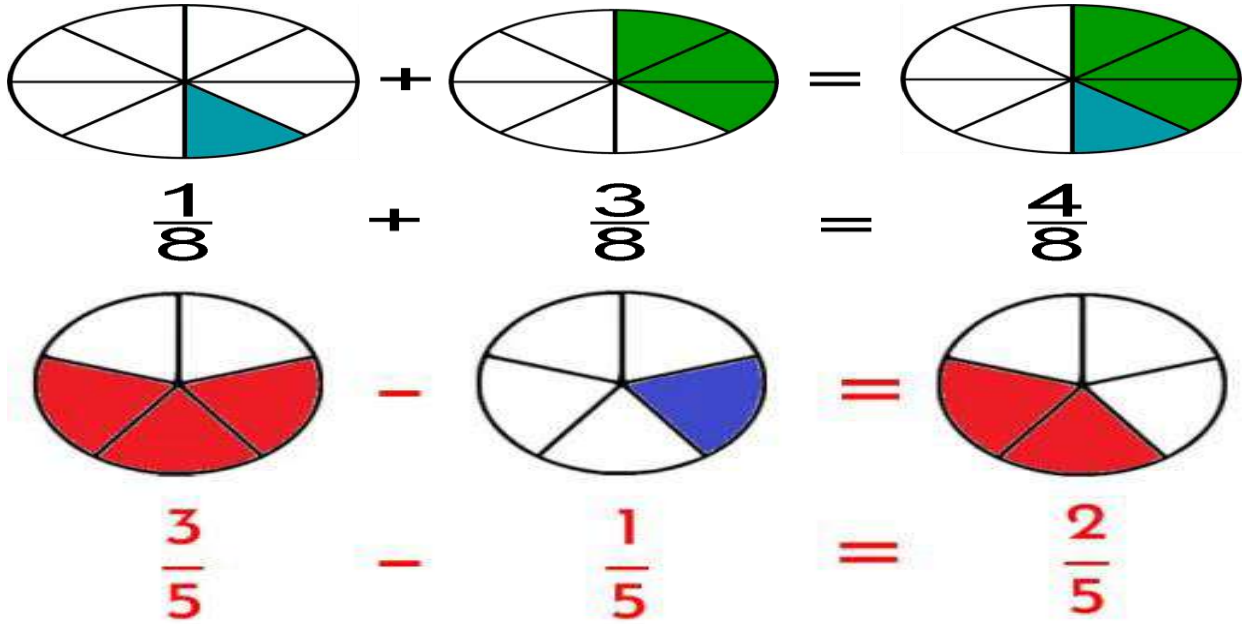
- ( - 15 ) ÷ ( 3 ) = - 5
- 6 ÷ 2 = + 3

# Operaciones en los números racionales

Recordemos que los números racionales también pueden ser nombrados como fracciones.
Y que una fracción es dividir cualquier cantidad en cantidades iguales y tomar algunas de esas partes.

**Operaciones de suma y resta de fracciones homogéneas:** Son aquellas que tienen igual denominador, Para sumar o restar números racionales homogéneos se coloca el mismo denominador y se suman o restan los numeradores.

**Ejemplo:**



# OPERACIONES CON FRACCIONES HETEROGÉNEAS.

**Para adicionar o sustraer números racionales heterogéneos:** se multiplican los denominadores entre sí y ese será el denominador, Se multiplican el numerador de la primera fracción por el denominador de la segunda, Se coloca el signo más. ( +,- ) Se multiplica el numerador de la segunda fracción por el denominador de la primera, Por último, sumamos los dos números que nos quedan en el en la parte de arriba.

**Ejemplo:**

$$\frac{5}{6} + \frac{3}{10} = \frac{(5 \times 10) + (6 \times 3)}{6 \times 10}$$
$$= \frac{68}{60}$$
$$= \frac{17}{15}$$

# Multiplicación De Números Racionales (Fracciones)

El producto de dos números racionales es otro número racional y se realiza multiplicando los numeradores entre sí y denominadores entre sí.

$$\frac{8}{3} x \frac{4}{3} = \frac{8x4}{3x3} = \frac{32}{9}$$

# División de fracciones

Para dividir dos números racionales se multiplica el numerador del primer número racional con el denominador del segundo número racional, y el numerador del segundo número con el denominador del primero. (Se multiplica en cruz)

$$\frac{4}{3} \cdot \frac{5}{4} = \frac{20}{12}$$

# Potencias y raíces

**La potenciación** es la multiplicación de un número (n) veces, el mismo número entero o racional, en el la base es el número racional que multiplicaremos y n (Denominado exponente) es el número de veces que multiplicamos la base.

**Propiedades de la potenciación:**

| *Propiedad* | *Exposición* | *Ejemplo* |
|---|---|---|
| **Potencia de base cero.** | Cualquier potencia de base 0 es igual a 0. $0^m = 0$ | $0^4 = 0$ |
| **Potencia de exponente cero.** | Cualquier potencia de exponente 0 es igual a 1. $X^0 = 1$ | $5^0 = 1$ |
| **Potencia de exponente 1.** | Cualquier potencia de exponente 1 es el mismo número. $X^1 = X$ | $7^1 = 7$ |
| **Producto de potencias de la misma base.** | El producto de potencias de la misma base es la misma base y el exponente la suma de los exponentes. $X^m + X^n = X^{m+n}$ | $2^4 * 2^6 = 2^{4+6} = 2^{10}$ |
| **Cociente de potencias de la misma base.** | El producto de potencias de la misma base es la misma base y el exponente de la suma de los exponentes. $X^m / X^n = X^{m-n}$ | $3^5 / 3^2 = 3^{5-2} = 3^3$ |
| **Potencia de una potencia.** | La potencia de una potencia es la misma base y el exponente el producto de los exponentes. $(X^m)^n = X^{m*n}$ | $(6^2)^4 = 6^{2*4} = 6^8$ |
| **Producto de potencias con el mismo exponente.** | El producto de potencias con el mismo exponente es una potencia con el mismo exponente y como base el producto de las bases. $X^m * Y^m = (X * Y)^m$ | $2^3 * 4^3 = (2 * 4)^3$ |

| *Propiedad* | *Exposición* | *Ejemplo* |
| --- | --- | --- |
| **Cociente de potencias con el mismo exponente.** | El cociente de potencias es una potencia con el mismo exponente y con base el cociente de las bases. $X^m / Y^m = (X / Y)^m$ | $6^3 / 2^3 = (6 / 2)^3 = 3^3$ |
| **Potencia de exponente negativo.** | La potencia de exponente negativo es igual al inverso de la potencia, con exponente positivo. $X^{-m} = 1 / X^m$ | $4^{-2} / 2 = 1 / 4^2$ |
| **Potencia de exponente racional.** | La potencia de exponente racional es igual a una raíz, donde el denominador es el índice de la raíz y el numerador el exponente del radicando. $X^{m/n} = \sqrt[n]{X^m}$ | $7^{4/3} = \sqrt[3]{7^4}$ |

# RADICACION

es conocida como la operación contraria a la potenciación en donde se conoce el índice de la raíz y el radicando de este modo se puede encontrar un número que multiplicado el número de veces que enuncie el índice nos de como resultado el radicando.

**Propiedades de la potencia**

| *Operación* | *Ejemplo* |
|---|---|
| Producto: $\sqrt[n]{a} * \sqrt[n]{b} = \sqrt[n]{a * b}$ | $\sqrt[3]{4} * \sqrt[3]{6} = \sqrt[3]{24}$ |
| Cociente: $\sqrt[n]{a} : \sqrt[n]{b} = \sqrt[n]{a / b}$ | $\sqrt[3]{4} : \sqrt[3]{6} = \sqrt[3]{4 / 6}$ |
| Potencia: $(\sqrt[n]{a})^m = \sqrt[n]{a^m}$ | $(\sqrt[5]{4})^6 = \sqrt[5]{4^6}$ |
| Simplificar: $(\sqrt[n]{a^n}) = a$ | $(\sqrt[3]{2^3}) = 2$ |
| Raíz de raíz: $(\sqrt[p]{\sqrt[n]{a}}) = \sqrt[n*p]{a}$ | $(\sqrt[4]{\sqrt[2]{3}}) = \sqrt[8]{3}$ |
| Introducir: $(a \sqrt[n]{a^n}) = \sqrt[n]{a^n * a} = \sqrt[n]{a^{n+1}}$ | $(4 \sqrt[3]{4}) = \sqrt[3]{4^3 * 4} = \sqrt[3]{4^4}$ |

# G R A C I A S

Línea de atención al ciudadano: 018000 910270
Línea de atención al empresario: 018000 910682

@SENAcomunica

www.sena.edu.co

El empleo es de todos

Mintrabajo

# Análisis y Desarrollo de Sistemas de Información

SENA

# Ciclo de vida del software

También conocido como (SDLC o Systems Development Life Cycle), es el proceso que se sigue para construir y hacer evolucionar un determinado software. El ciclo de vida permite iniciar una serie de fases mediante las cuales se procede a la validación y al desarrollo del software garantizando que se cumplan los requisitos para la aplicación y verificación de los procedimientos de desarrollo; para ello, se utilizan métodos del ciclo del software, que indican distintos pasos a seguir para el desarrollo de un producto.



# A continuación, se indican cuáles son los elementos que integran un ciclo de vida:



## Fases:

Una fase es un conjunto de actividades relacionadas con un objetivo en el desarrollo del proyecto. Se construye agrupando tareas (actividades elementales) que pueden compartir un tramo determinado del tiempo de vida de un proyecto. La agrupación temporal de tareas impone requisitos temporales correspondientes a la asignación de recursos (humanos, financieros o materiales).



## Entregables:

Son los productos intermedios que generan las fases. Pueden ser materiales o inmateriales (documentos, software). Los entregables permiten evaluar la marcha del proyecto mediante comprobaciones de su adecuación o no a los requisitos funcionales y de condiciones de realización previamente establecido.

Las fases del modelo de ciclo del software son: planificación, análisis, diseño, implementación, pruebas y mantenimiento, las cuales se describen en la siguiente figura.

## FASE PLANIFICACIÓN:

En esta primera fase se realiza el planteamiento del problema, se definen alcances y objetivos del software.

Objetivos: estudio de viabilidad, realizar planificación detallada.



## FASE ANÁLISIS (definición de requisitos):

Esta fase busca definir los requisitos que son los que dirigirán el desarrollo del proyecto de software.

Objetivos: conocer los requisitos, asegurar que los requisitos son alcanzables, formalizar acuerdo con el cliente.



## FASE DISEÑO:

En esta fase se estudian posibles opciones de implementación para el software que hay que construir, estructura general del mismo.

Objetivos: Identificar soluciones tecnológicas, asignar recursos materiales, proponer identificar y seleccionar, establecer métodos de validación, ajustar especificaciones.



## FASE PRUEBAS:

Esta fase busca detectar fallos cometidos en las etapas anteriores para corregirlos.

Objetivos: Realizar los ajustes necesarios para corregir posibles errores o inconsistencias.



## FASE MANTENIMIENTO:

En esta fase se realizan tres puntos referenciados: mantenimiento correctivo, mantenimiento adaptativo y mantenimiento perfectivo.

Objetivos: Operación asegurar que el uso del proyecto es el que se pretendía, mantenimiento.



# Paradigmas de los modelos de ciclo de vida del software

- Con la finalidad de proporcionar una metodología común entre el cliente y la empresa de software, se utilizan los modelos de ciclos de vida o paradigmas de desarrollo de software para plasmar las etapas y la documentación necesaria, de manera que cada fase se valide antes de continuar con la siguiente.

- Un modelo de ciclo de vida de software es una vista de las actividades que ocurren durante el desarrollo de software e intenta determinar el orden de las etapas involucradas y los criterios de transición asociados entre estas.

- Existen modelos preestablecidos con los cuales se pude elaborar un proyecto; a continuación, se mencionan los diferentes paradigmas de modelos de ciclo de vida para desarrollar software.



### Paradigma tradicional

Los paradigmas tradicionales se identifican, fundamentalmente, por ser lineales, es decir se trata de completar cada proceso de principio a fin hasta que quede listo para avanzar a la segunda fase del ciclo del software.

Si bien es verdad que las metodologías actuales están basadas en lo que fueron los paradigmas tradicionales, hoy en día se ha evolucionado, sin embargo, los paradigmas tradicionales se mantienen.



### Desventaja:

Pérdida de tiempo si se encuentran errores en una fase avanzada porque al devolverse se debe pasar nuevamente por todas las fases y reestructurar de acuerdo con las modificaciones.



### Paradigma orientado a objetos:

Las etapas de desarrollo de software en el paradigma orientado a objetos, se conforma principalmente por la creación de clases, análisis de requisitos y el diseño. Con este paradigma se pretende que el código fuente sea reutilizable para otros proyectos.



### Paradigma de desarrollo ágil

El objetivo de este paradigma es el desarrollo de proyectos en poco tiempo, se simplifican procesos tediosos, se agilizan las fases del desarrollo y las interacciones se hacen en corto tiempo.

Una de las principales diferencias con los paradigmas anteriores es que el cliente se ve involucrado en el proyecto durante el desarrollo de este, así, el cliente sugiere mejoras, propone ideas y se mantiene al tanto del desarrollo del producto, a diferencia del paradigma tradicional y el orientado objeto donde el cliente únicamente está al principio.

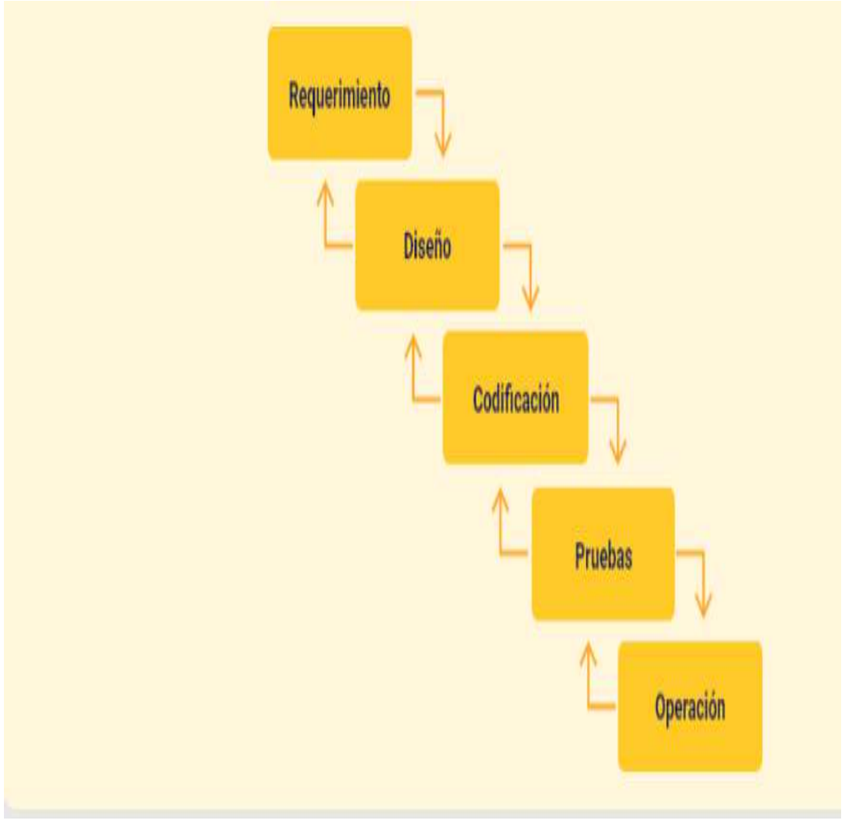



Figura 1. Modelo de cascada del ciclo de vida del desarrollo.

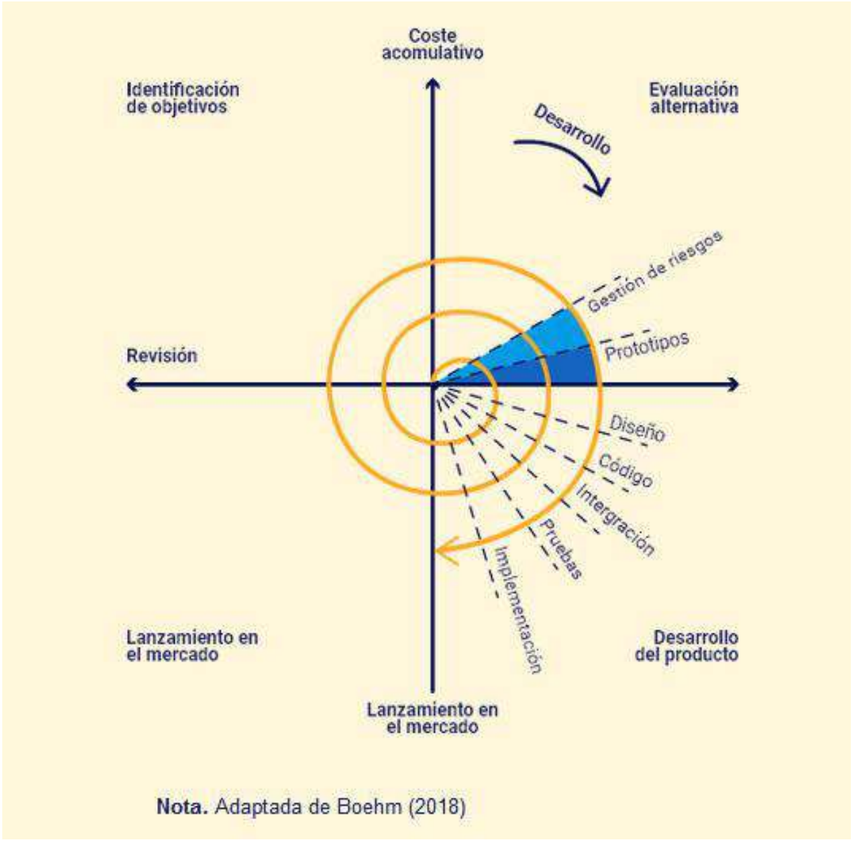



Figura 2. Modelo espiral del ciclo de vida del desarrollo.

- A continuación, se revisan los modelos del paradigma tradicional más utilizados.

## Modelo interactivo o por prototipos.

Este modelo consiste en un procedimiento que permite al equipo de desarrollo diseñar y analizar una aplicación que represente el sistema que será implementado (McCracken y Jackson, 1982).

### Objetivos

**A** Son un medio eficaz para aclarar los requisitos de los usuarios e identificar las características de un sistema que deben cambiarse o añadirse.

**B** Mediante el prototipo se puede verificar la viabilidad del diseño de un sistema.

Las etapas del modelo son:

**1** Colecta y refinamiento de los requerimientos y proyecto rápido.

- Análisis.
- Especificación del prototipo.

**2** Diseño rápido.

**3** Construcción del prototipo.

**4** Evaluación del prototipo por el cliente.

**5** Refinamiento del prototipo.

- Diseño técnico.
- Programación y test.
- Operación y mantenimiento.

**6** Producto de ingeniería.

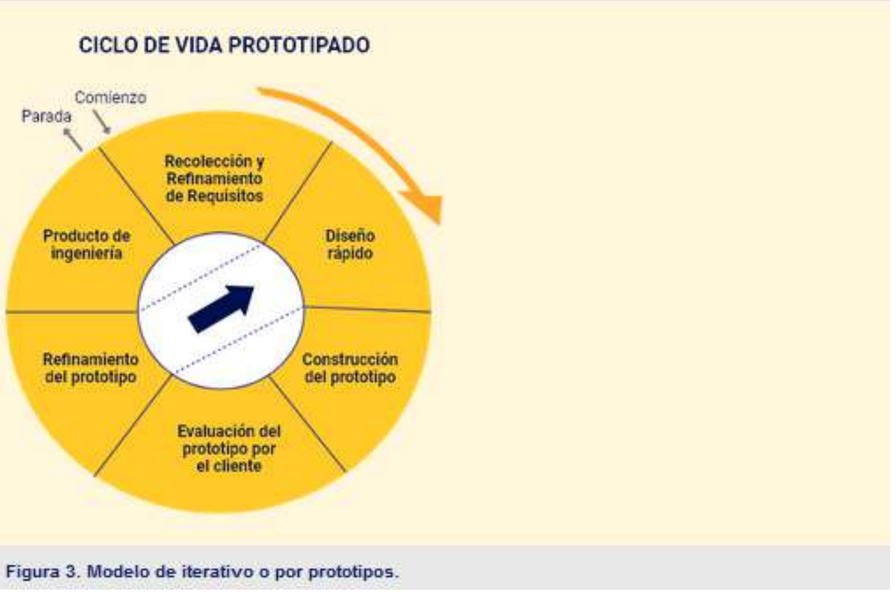



Figura 3. Modelo de iterativo o por prototipos.

# MANIFIESTO AGIL:

Los valores de la metodología Ágil son:

- Individuos e interacciones sobre procesos y herramientas.
- Software funcional sobre documentación completa.
- Colaboración con clientes sobre negociación de contratos.
- Respuesta al cambio sobre el seguimiento de un plan.

# PRINCIPIO AGIL

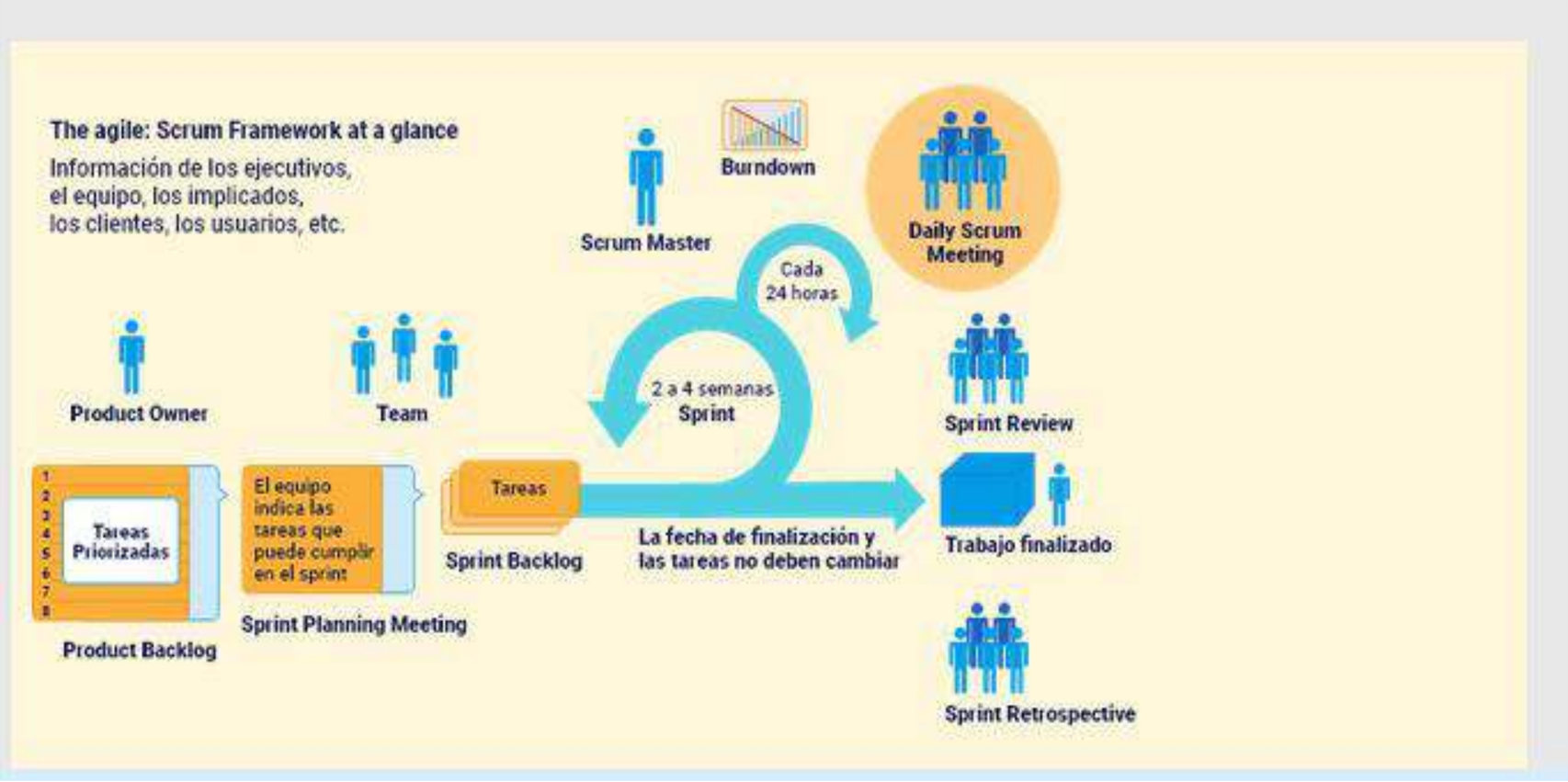
1. Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
3. Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
4. Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
5. Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
6. El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.

7. El software funcionando es la medida principal de progreso.
8. Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
10. La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
12. A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

Modelos del ciclo de vida del paradigma ágil:

## Ventajas

- Gestión regular de las expectativas del usuario: los usuarios participan y proponen soluciones
- Resultados anticipados: no es necesario esperar hasta el final para ver resultados
- Flexibilidad y adaptación: se adapta a cualquier contexto, área o sector.
- Gestión sistemática de riesgos: los problemas son gestionados en el mismo momento de su aparición.




Modelo Scrum

# ANTES DE INICIAR...

## ¿Qué es una metodología?

Es un conjunto de procesos, procedimientos, técnicas y/o herramientas que brindan una guía para el cumplimiento de metas u objetivos...

## ¿Qué es un Framework?

Es un conjunto estandarizado de conceptos, prácticas, herramientas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

# ¿QUÉ ES SCRUM?

Scrum es un marco de trabajo para el desarrollo y el mantenimiento de productos complejos.
Un marco de trabajo dentro del cual las personas pueden afrontar complejos problemas adaptativos, a la vez que entregan productos del máximo valor posible de forma productiva y creativa.
Con SCRUM abordaremos un proyecto y dividiremos su desarrollo para realizar entregas funcionales.
Scrum es:

- Ligero
- Fácil de entender
- Extremadamente difícil de llegar a dominar

## Artefactos ::

- Product Backlog
- Sprint Backlog
- Diagrama de incremento




## Ceremonias::

- Daily Standup.
- Sprint Retrospective.
- Global Retrospective



# TERMINOLOGIA AGIL

- **Scrum Master** :: Coordinar el trabajo, permear la filosofía SCRUM al equipo y mantener un clima socioemocional atractivo (Mantener al equipo motivado).

- **Product Owner** :: Es el encargado de añadir elementos de trabajo al backlog.

- **Scrum Team** :: Son los encargados de agregar (o modificar) funcionalidades al proyecto, cumplen las tareas especificadas en backlogs.

- **Stake Holders** :: Evaluar el incremento de funcionalidad al final de cada iteración (Sprint).

# TERMINOLOGIA AGIL

- **Backlog** :: Base de datos que almacena todo los requerimientos en un formato de historias de usuarios, ponderadas y priorizadas.

- **Sprint** :: Corresponde a un periodo de tiempo, durante el cual se desarrollan aquellas actividades del producto backlog que fueron seleccionadas (sprint backlog).

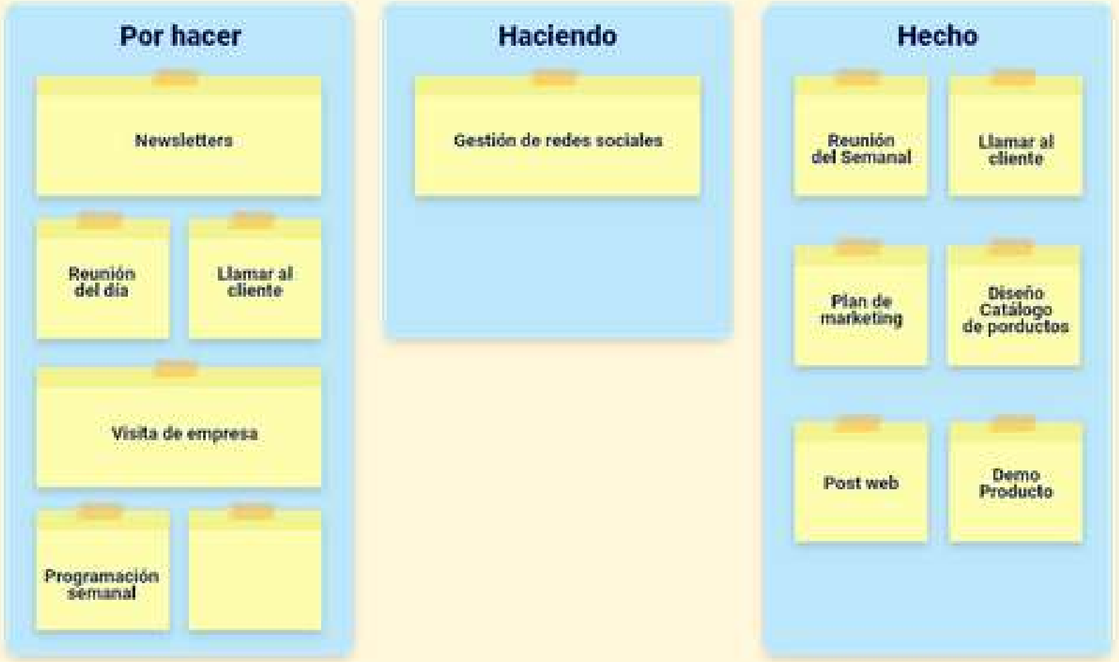
- **Restrocpetivas** :: Reuniones del equipo, con la finalidad de evaluar aspectos acontecidos durante el ciclo; se constituyen en la principal herramienta de gestión de conocimiento y aprendizaje de equipo.

- **Incremento de funcionalidad** :: Resultados finales de cada sprint y su aporte a la adición de funcionalidades (de forma incremental) al producto final.

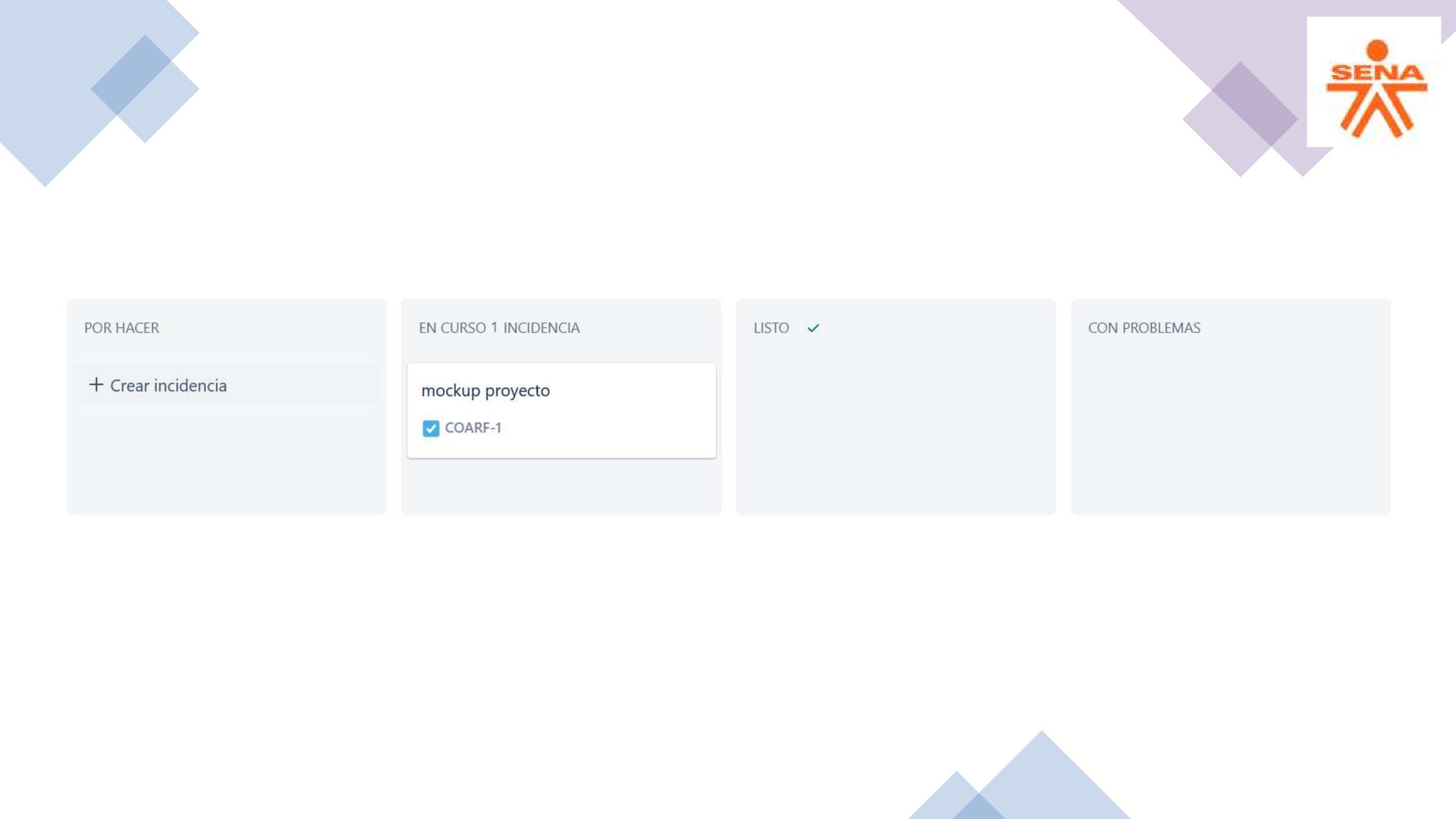
Mediante la metodología japonesa Kanban se:

1. Define el flujo de trabajo.
2. Establecen las fases del ciclo de producción.
3. Stop Starting, start finishing.
4. Tiene un control.

## Kanban.




**Nota**. Adaptada de Anderson.


POR HACER
+ Crear incidencia
EN CURSO 1 INCIDENCIA
mockup proyecto
COARF-1
LISTO
CON PROBLEMAS

# Ejemplo: Historia de Usuario

| |
|---|
| **Como** Neil Armstrong<br>**Quiero** Un traje espacial<br>**Para** dar un paseo por la superficie lunar |
| **PRIORIDAD** 10 |
| **TAMAÑO** 15 |

| **CRITERIO DE ACEPTACIÓN** |
|---|
| **Dado** un astronauta que se encuentra en un viaje a la luna<br>**cuando** aluniza y desciende de la nave espacial<br>**entonces** no se congela y puede respirar. |

# Referencias bibliograficas

**https://agilemanifesto.org/iso/es/principles.html**

https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-Latin-South-American.pdf

# G R A C I A S

Línea de atención al ciudadano: 018000 910270
Línea de atención al empresario: 018000 910682

@SENAcomunica

www.sena.edu.co

# JIRA Software:

Con Jira, los equipos pueden planificar, asignar, supervisar y gestionar el trabajo, así como elaborar informes al respecto. Crea hábitos entre los equipos en todos los ámbitos, desde el desarrollo de software ágil, pruebas y la atención al cliente. :

**Planificación:**
Cree historias de usuario e incidencias, planifica sprints y distribuya tareas entre el equipo de software de hasta 10 integrantes.

**Supervisar:**
Priorice y analice el trabajo de un equipo en su contexto y con una completa visibilidad.

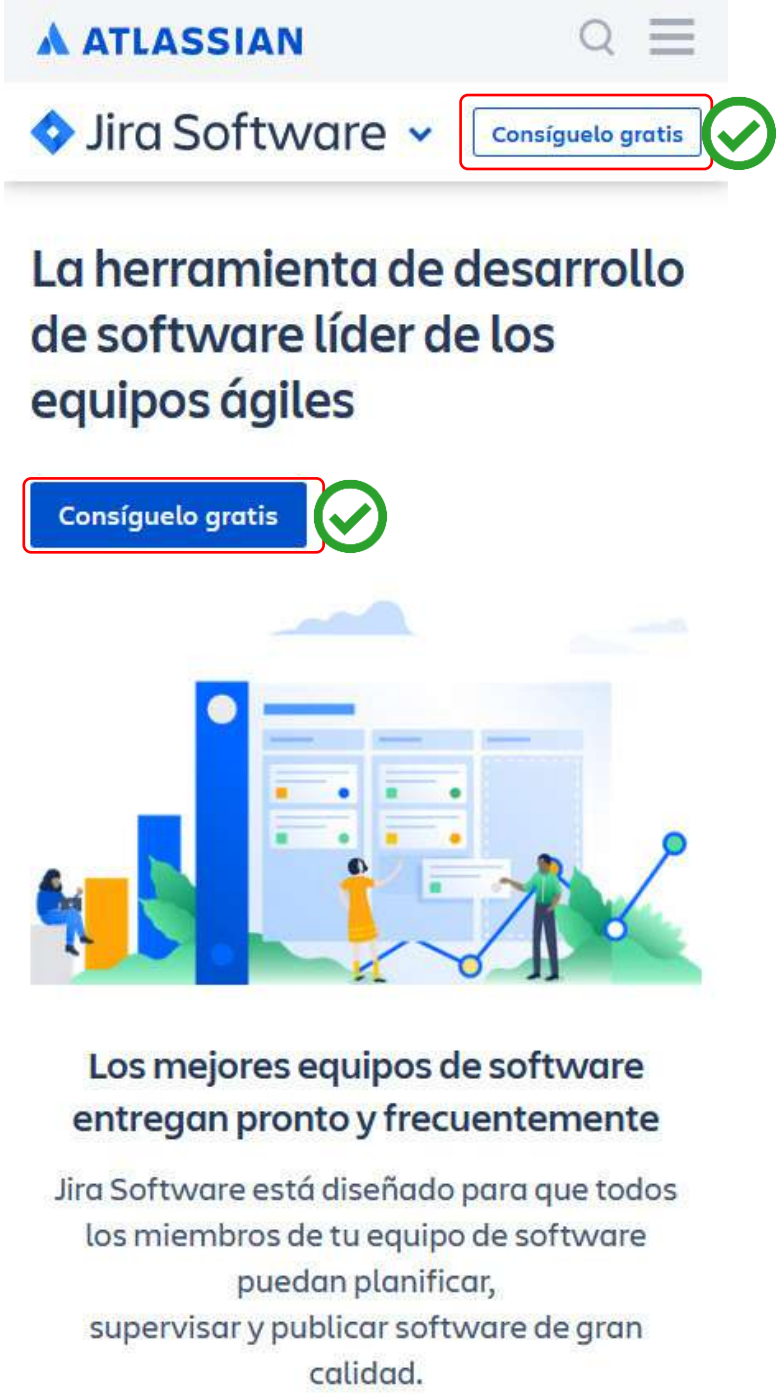
**Informes:**
Mejore el rendimiento del equipo con diferentes gráficos en tiempo real que el equipo puede generar y emplear para el mejoramiento continuo.

# Registro:

Proceso de registro en la plataforma de **Jira Software, preferible usar correo @misena.edu.co**

Ingresar  a la dirección:

https://www.atlassian.com/es/software/jira



# Registro:


ATLASSIAN
Nuestros productos de la nube juntos incluso funcionan mejor.
Consigue otro gratuito
CADA PRODUCTO DEL PLAN FREE:
Admite hasta 10 usuarios o 3 agentes
Incluye 2 GB de almacenamiento
Ofrece soporte de la Comunidad
Siempre es gratuito, no se necesita tarjeta de crédito
¿Tienes un equipo más grande?
Comparar planes
SELECCIONASTE
Jira Software
Seguimiento de proyectos e incidencias
SELECCIONA UN SEGUNDO PRODUCTO
Confluence
Colaboración en documentos
Seleccionar
Agrega contexto a tus proyectos con una única fuente de información para los requisitos del producto, las notas de la versión y la documentación.
Jira Service Management
ITSM de alta velocidad
Seleccionar
Autoriza a los equipos de desarrollo y operaciones a colaborar a gran velocidad, de modo que puedan responder a los cambios empresariales y ofrecer experiencias de servicio rápidamente.
Siguiente
NO SE NECESITA TARJETA DE CRÉDITO

# Registro:


Jira Software
Cloud Free
Empieza ya
Gratis para hasta 10 usuarios
Continuar con Google
o
Registrarse con el correo electrónico
ATLASSIAN
Más de 65 000 equipos de todo el mundo ya confían en esta solución
Square
Spotify
cisco
airbnb
Escala prácticas ágiles
Consolida los flujos de trabajo
Amplía la visibilidad
Planificación, supervisión y publicación

Aquí es donde se crea su espacio en la plataforma



# Registro:

## Te damos la bienvenida a Jira Software,

Si respondes a estas rápidas preguntas, nos ayudarás a personalizar tu experiencia. Mantendremos tu información segura, de acuerdo con lo establecido en nuestra política de privacidad.

¿En qué tipo de equipo trabajas?

Operaciones | Legal | Ventas

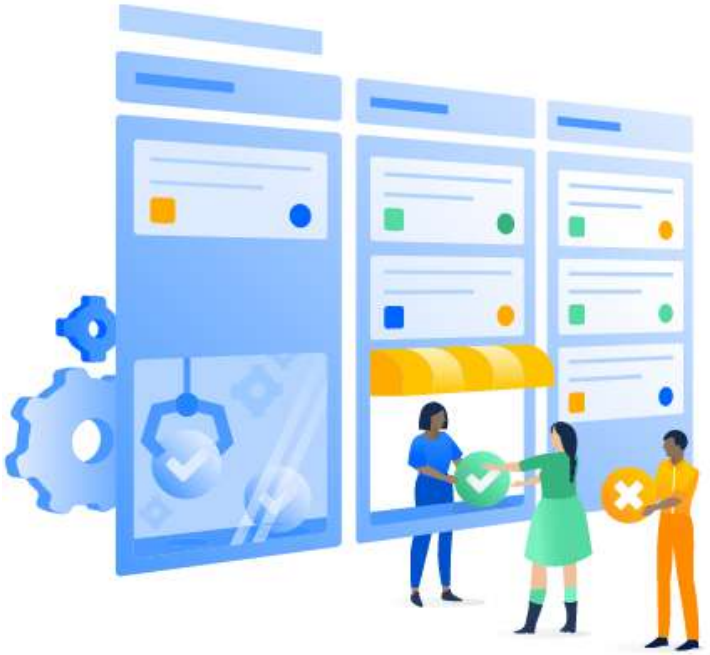
Servicio de atención al cliente | Soporte de TI

Desarrollo de software | Recursos humanos

Marketing | Finanzas

Otro

Saltar pregunta



## Invitar a los compañeros de equipo

¡Reúne a tu equipo para el viaje!

Añadir dirección de correo electrónico

@example.com

Añadir dirección de correo electrónico

@example.com

Añadir dirección de correo electrónico

@example.com

Dejar que mis compañeros de equipo inviten a otras personas a nuestro sitio

Puedes modificar esta configuración en cualquier momento.

Continuar

# Registro:



## Recomienda un proyecto

Responde unas cuantas preguntas para que podamos recomendarte un tipo de proyecto que se adapte lo mejor posible a ti y a tu equipo.

## Ayúdanos a configurar tu Jira

Soy nuevo en Jira.

Mi equipo es nuevo en metodologías ágiles.

Invertimos nuestro tiempo trabajando en las funcionalidades .

Tenemos un programa flexible para finalizar nuestro trabajo.

Omitir Siguiente

# Registro:

## Plantillas de proyectos

Desarrollo de software

PRODUCTOS

Jira Software

MÁS

Importar tu trabajo

## Desarrollo de software

Planifica, monitoriza y publica software de gran calidad. Ponte manos a la obra en seguida con plantillas que se adaptan a la forma de trabajar de tu equipo. Además, contarás con integraciones para los equipos de DevOps que quieran conectar el trabajo en toda su cadena de herramientas.



### Kanban RECOMENDADO

Visualiza tu proyecto y hazlo avanzar utilizando incidencias en un tablero lleno de posibilidades.



### Scrum

Realiza un sprint hacia los objetivos de tu proyecto con un tablero, un backlog y una hoja de ruta.



### Seguimiento de errores

Administra una lista de errores y tareas de desarrollo.

# Registro:


1 Plantilla del proyecto
Scrum
Realiza un sprint hacia los objetivos de tu proyecto con un tablero, un backlog y una hoja de ruta.
r plantilla
2 Elegir un tipo de proyecto
Si decides cambiar de tipo de proyecto más adelante, tendrás que crear un proyecto nuevo.
Gestionados por el equipo
Gestionados por la empresa
Configurado y mantenido por tu equipo.
Para equipos que quieren controlar sus propios procesos y prácticas de trabajo en un espacio independiente. Combina funciones de metodología ágil para respaldar a tu equipo a medida que creces en tamaño y complejidad.
Configuración simplificada
Ponte en marcha rápidamente con una configuración simplificada.
Cualquiera de tu equipo puede configurarlo y mantenerlo
Selecciona un proyecto gestionado por el equipo
Configuración y mantenimiento a cargo de tus administradores de Jira
Para equipos que quieren trabajar con otros equipos en muchos proyectos de forma estándar. Fomenta y promueve las prácticas recomendadas y los procesos de la organización mediante una configuración común.
Configuración de los partners
Sácale partido a un control total con la configuración, personalización y flexibilidad de los partners.
Configuración y mantenimiento a cargo de tus administradores de Jira
Selecciona un proyecto gestionado por la empresa

# Registro:

## Crear proyecto

Nombre *

Estudios

Clave *

Plantilla

**Scrum**

Realiza un sprint hacia los objetivos de tu proyecto con un tablero, un backlog y una hoja de ruta.

Cambiar plantilla

Vincula el trabajo en todas tus herramientas

Sincroniza el trabajo desde otras herramientas para que tu equipo pueda acceder fácilmente al código, las implementaciones, la wiki y mucho más desde tu proyecto de Jira.

Siguiente

rrar



**Conecta tu proyecto**

Sincroniza el trabajo desde tus herramientas conectadas para que puedas acceder a todo desde tu proyecto de Jira. Siempre puedes añadir, eliminar o modificar estas conexiones más tarde. Más información

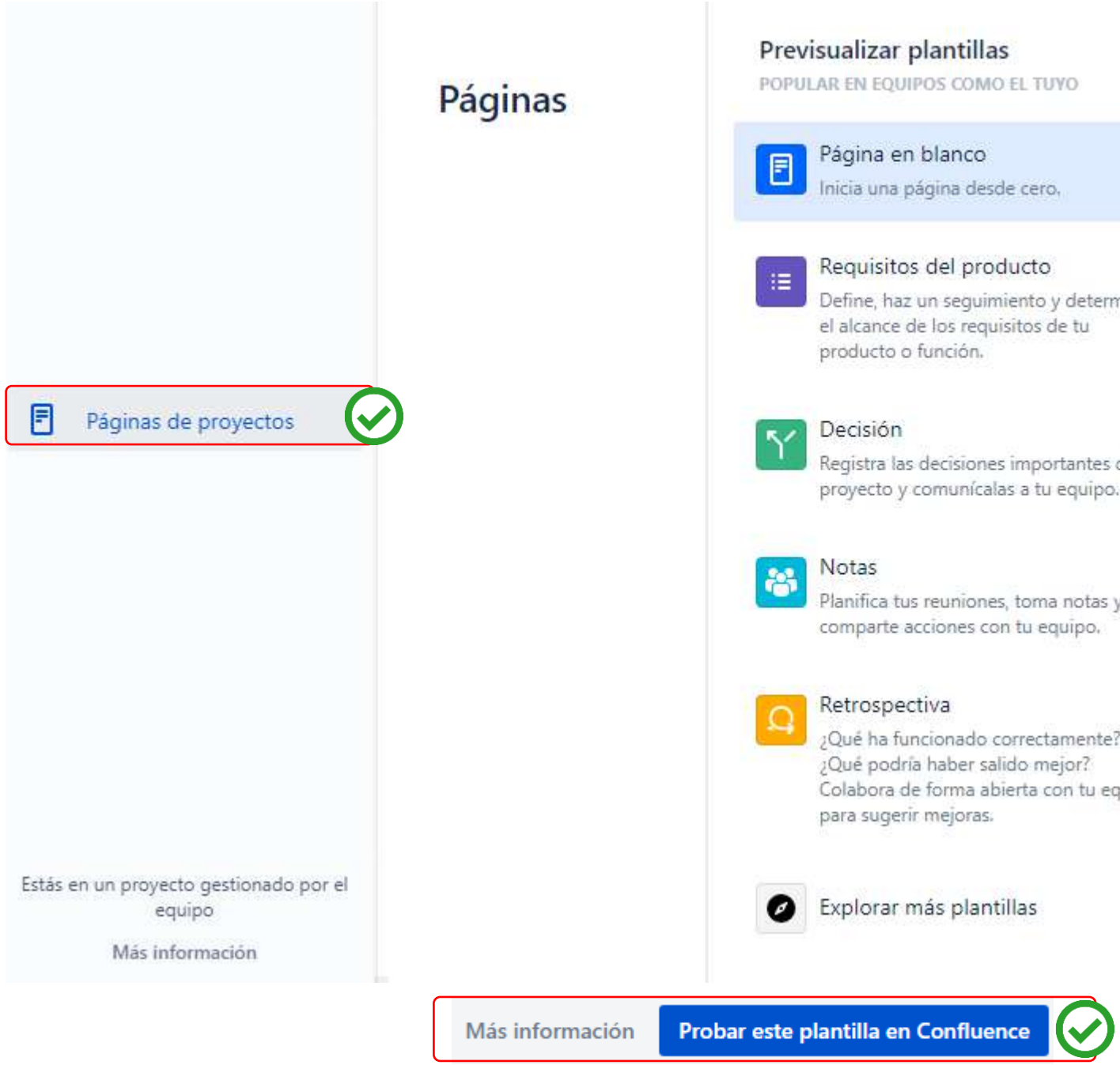
**Todavía no has conectado ninguna herramienta**

Para sincronizar el trabajo con este proyecto, primero deberás conectar las herramientas a tu sitio de Jira. Siempre puedes hacerlo más tarde.

Buscar herramientas para establecer la conexión

Ir al proyecto

# Registro:


Páginas
Páginas de proyectos
Estás en un proyecto gestionado por el equipo
Más información
Previsualizar plantillas
POPULAR EN EQUIPOS COMO EL TUYO
Página en blanco
Inicia una página desde cero.
Requisitos del producto
Define, haz un seguimiento y determina el alcance de los requisitos de tu producto o función.
Decisión
Registra las decisiones importantes del proyecto y comunícalas a tu equipo.
Notas
Planifica tus reuniones, toma notas y comparte acciones con tu equipo.
Retrospectiva
¿Qué ha funcionado correctamente? ¿Qué podría haber salido mejor? Colabora de forma abierta con tu equipo para sugerir mejoras.
Explorar más plantillas
Más información
Probar este plantilla en Confluence

# ¿Qué es un diagrama de Gantt?

El diagrama de Gantt es una herramienta de gestión de proyectos en la que se recoge la planificación de un proyecto.

Normalmente tiene dos secciones: en la parte izquierda se incluye una lista de tareas y, en la derecha, un cronograma con barras que representan el trabajo. Los diagramas de Gantt también pueden incluir las fechas de inicio y de finalización de las tareas, los hitos, las dependencias entre tareas y las personas asignadas. Para cumplir con las demandas del desarrollo de software moderno, las herramientas de **hoja de ruta** como se llaman en Jira Software incluyen funciones como una estructura de tareas plegable y paneles de gestión de recursos.

Estas herramientas de hoja de ruta ayudan a los equipos a mantener una estrategia coherente en los proyectos a pesar de la naturaleza iterativa de los procesos de desarrollo de software.

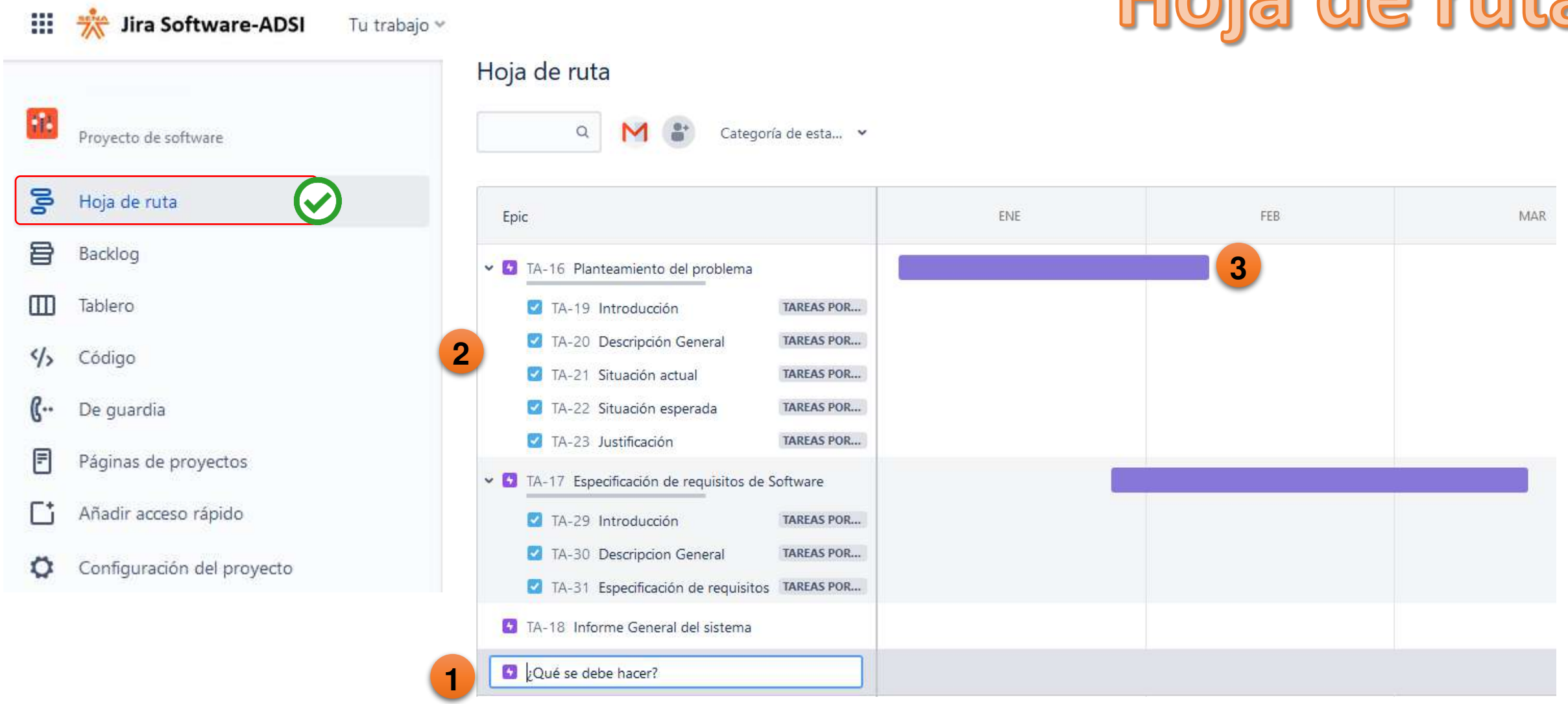
**¿Para qué se utiliza un diagrama de Gantt?**

Los gestores de proyectos utilizan los diagramas de Gantt con tres fines principales:

**Crear y gestionar un proyecto completo**

Los diagramas de Gantt sirven para visualizar los componentes básicos de un proyecto y para organizarlo en tareas más pequeñas y gestionables. Las pequeñas tareas resultantes se programan en la línea de tiempo del diagrama de Gantt, junto con las dependencias entre las tareas, las personas asignadas y los hitos.

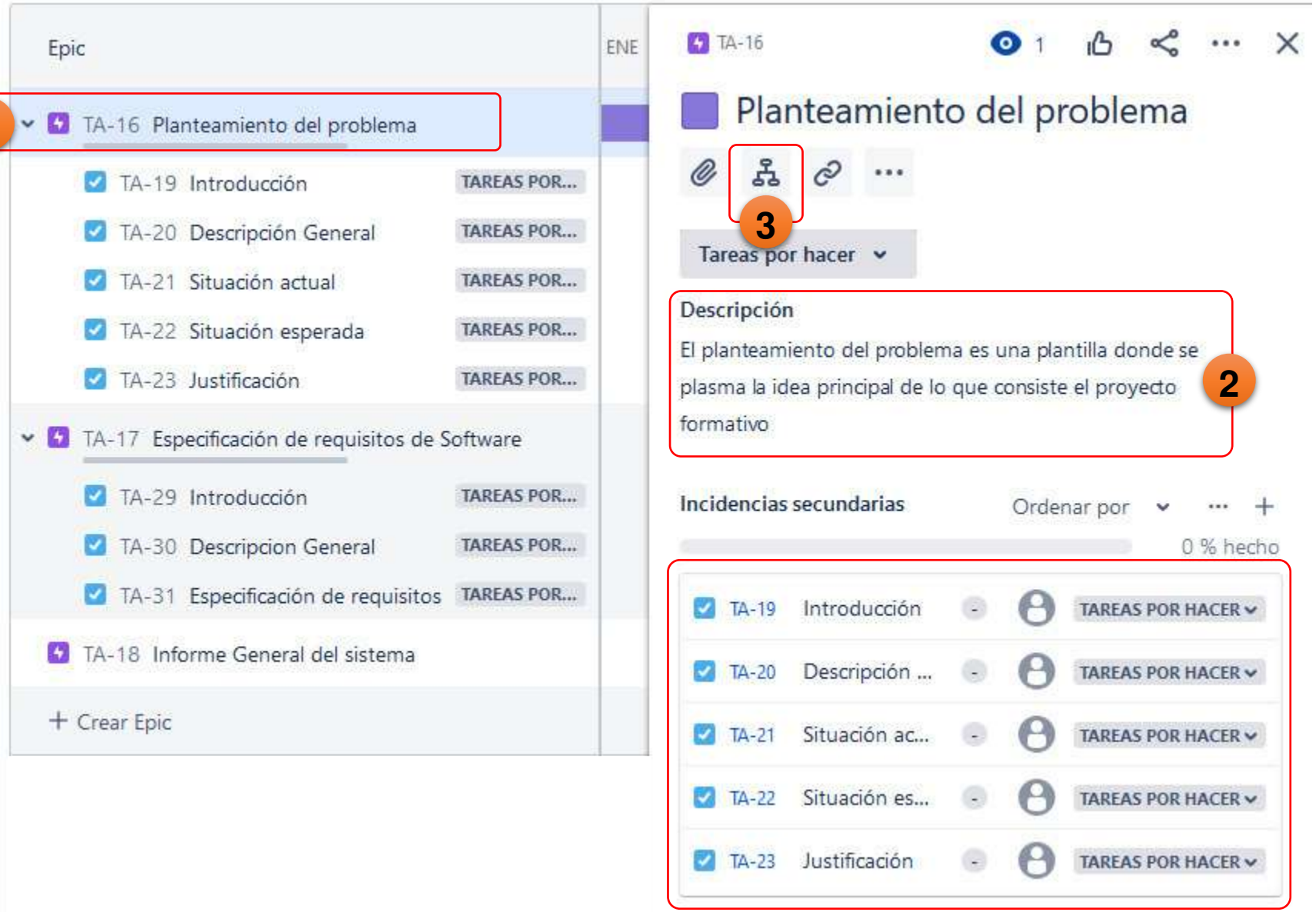
# Hoja de ruta




1. 1: Cree las tareas mas grandes, en Jira se llaman Épicas
2. 2: A las épicas se le asignan tareas
3. 3: se puede definir el tiempo asignado a cada épica, haciendo clic sostenido

# Detalles de la épica

Aplica también para historias de usuario y tareas




1. Clic sobre la épica, a la derecha de la pantalla se activa una sección para detalles como:
2. Descripción: cuadro de texto con el fin de detallar en que consiste la épica/tarea/historia de usuario.
3. Para asignar subtareas a una épica/historia/tarea

Las tareas de la épica/historia/tarea se agrupan

# Detalles de las tareas.

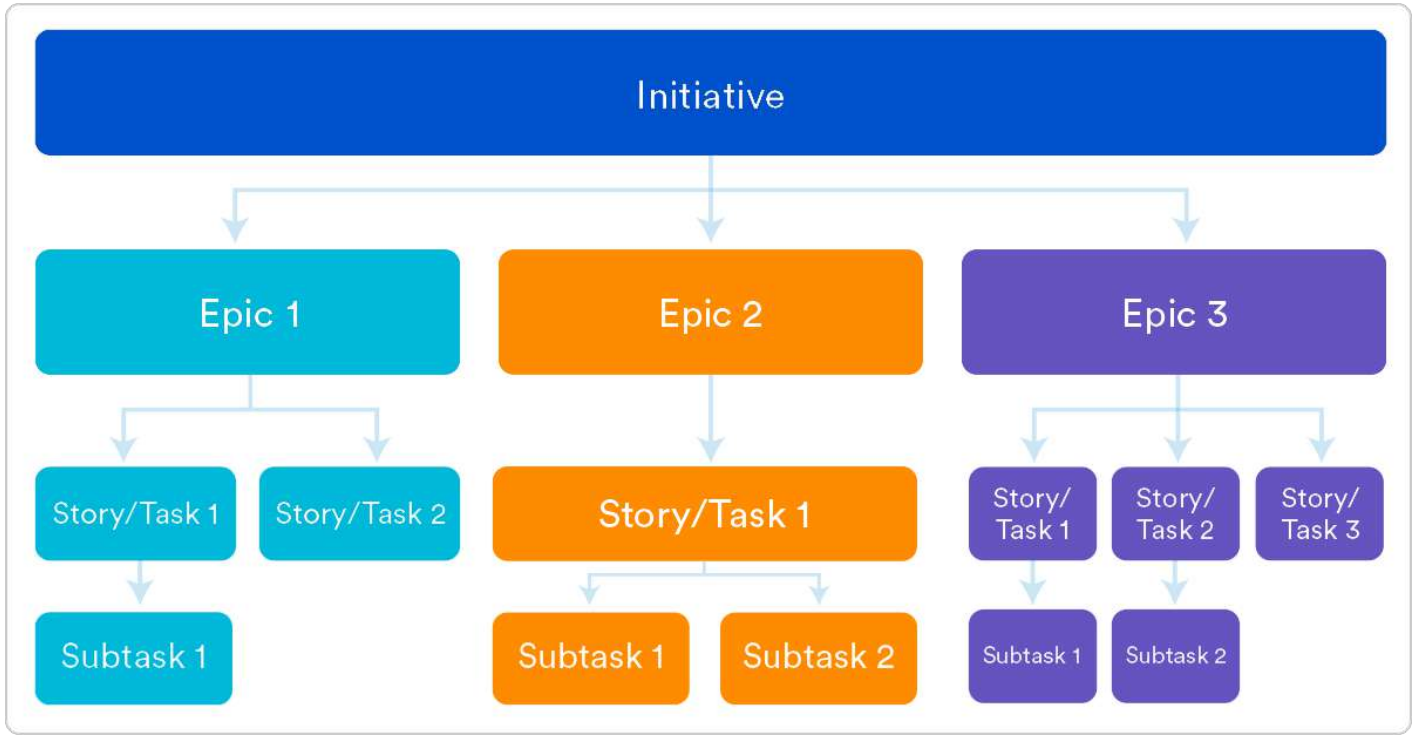



1. Sub tarea de una tarea: en el ejemplo observamos que la tarea TA-24 es una subtarea de la tarea TA-19.
2. Realice una descripción de la tarea, recuerde que esta tarea será asignada a alguien, y esa persona deberá saber en que consiste para poder realizarla.
3. Se selecciona a un integrante del equipo a quien s ele asignara la terea, o en la reunión diaria un integrante del equipo puede tomar la tarea a realizar.
4. Puntos de historia o estimado de dificultad de la tarea.
5. El informador es la persona que creo la tarea o incidente

# Épicas e historias

**Ejemplos de una historia ágil:**

- Los usuarios de iPhone necesitan acceder a una vista vertical de la alimentación en vivo cuando utilizan la aplicación móvil.
- Los usuarios de equipos de escritorio necesitan un botón de visualización en pantalla completa en la esquina inferior derecha del reproductor de vídeo.
- Los usuarios de Android deben estar vinculados al Apple Store.

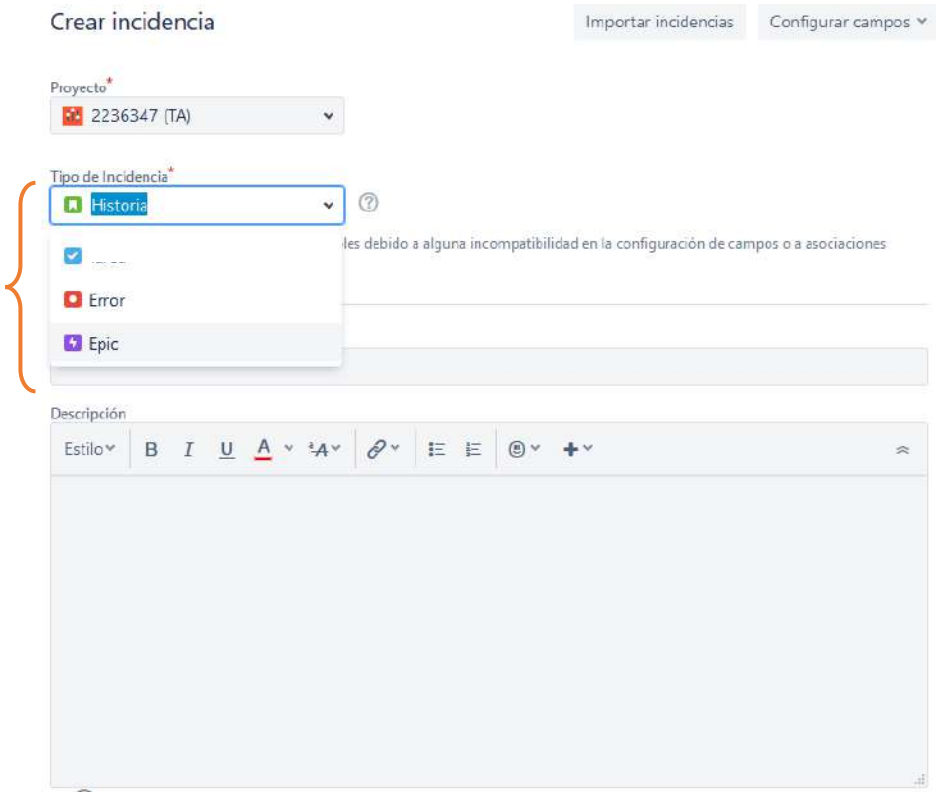



Las anteriores historias están relacionadas entre sí y podrían considerarse tareas independientes que conducen a la realización de un trabajo de mayor volumen (un epic o épica). En este caso, la épica podría ser **"mejorar el servicio de lanzamiento del primer trimestre"**.

# Resumiendo




En la interfaz de jira en la parte superior todo el tiempo hay un botón azul que indica: Crear, al dar clic allí podemos crear todos los elementos de un proyecto: Épicas, Historias de usuario, tareas y errores



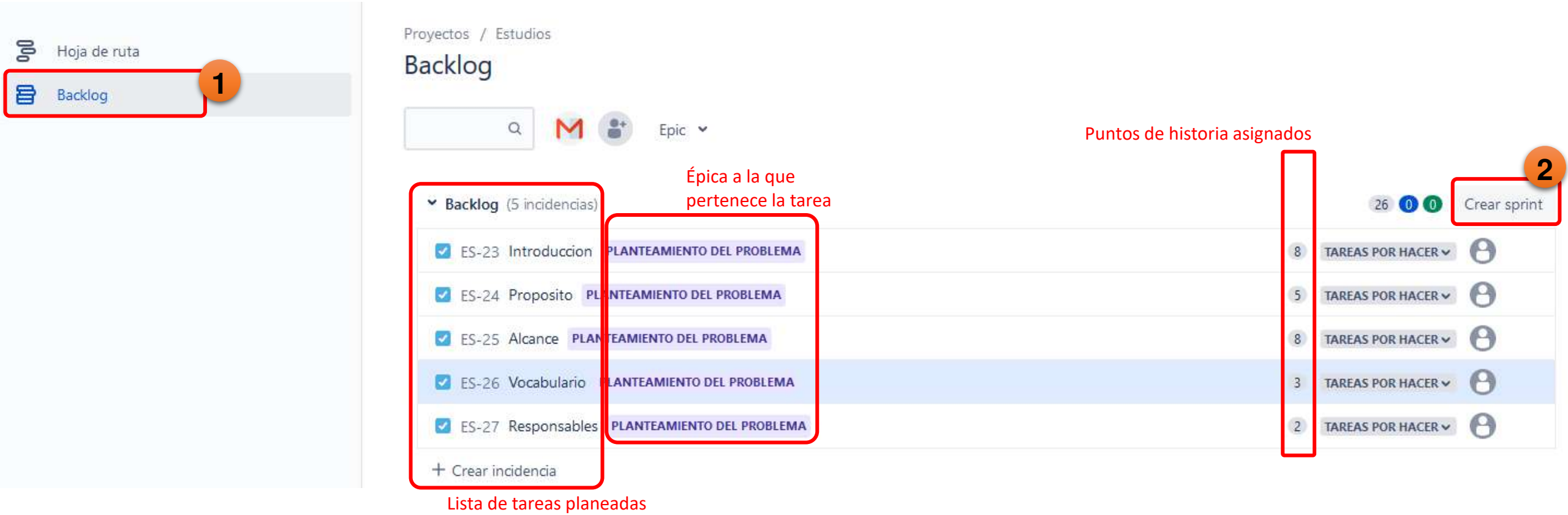
# ¿Qué son los sprints?

Un sprint es un período breve de tiempo fijo en el que un equipo de scrum trabaja para completar una cantidad de trabajo establecida. Los sprints se encuentran en el corazón de las metodologías scrum y ágil, y hacer bien los sprints ayudará al equipo ágil a lanzar mejor software con menos dificultades.

Muchos asocian los sprints de la metodología scrum con el desarrollo de software ágil, hasta tal punto que suelen pensar que las metodologías scrum y ágil son lo mismo, pero no es así. La metodología ágil constituye una serie de principios y la metodología scrum es un marco de trabajo para conseguir resultados.

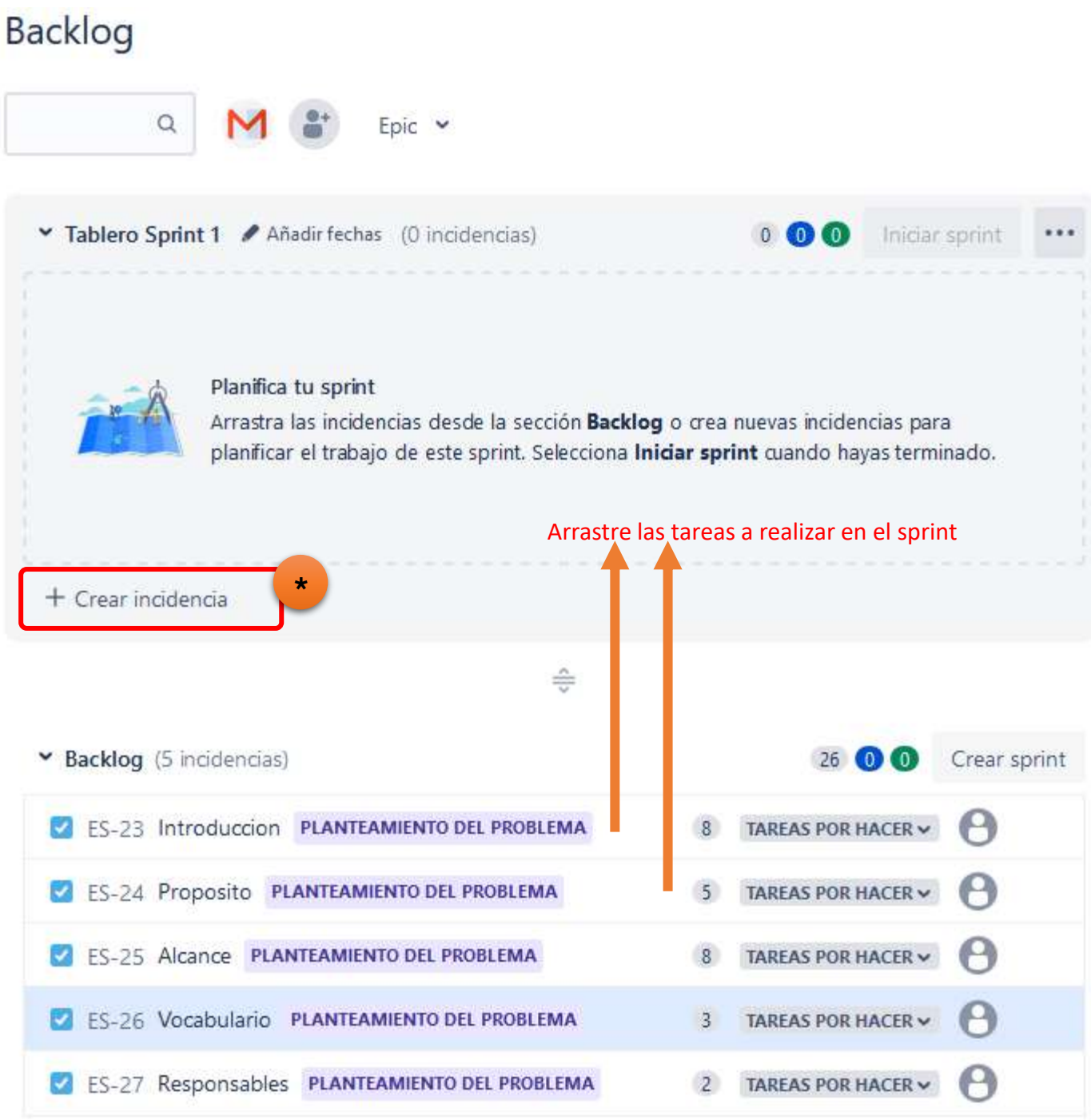
# Crear un Sprint

Cuando ya se tienen las tareas a realizar durante el tiempo que dura el sprint y realizada la planeación del sprint se procede a crearlo desde las opciones de la izquierda en la opción Backlog, luego Crear sprint

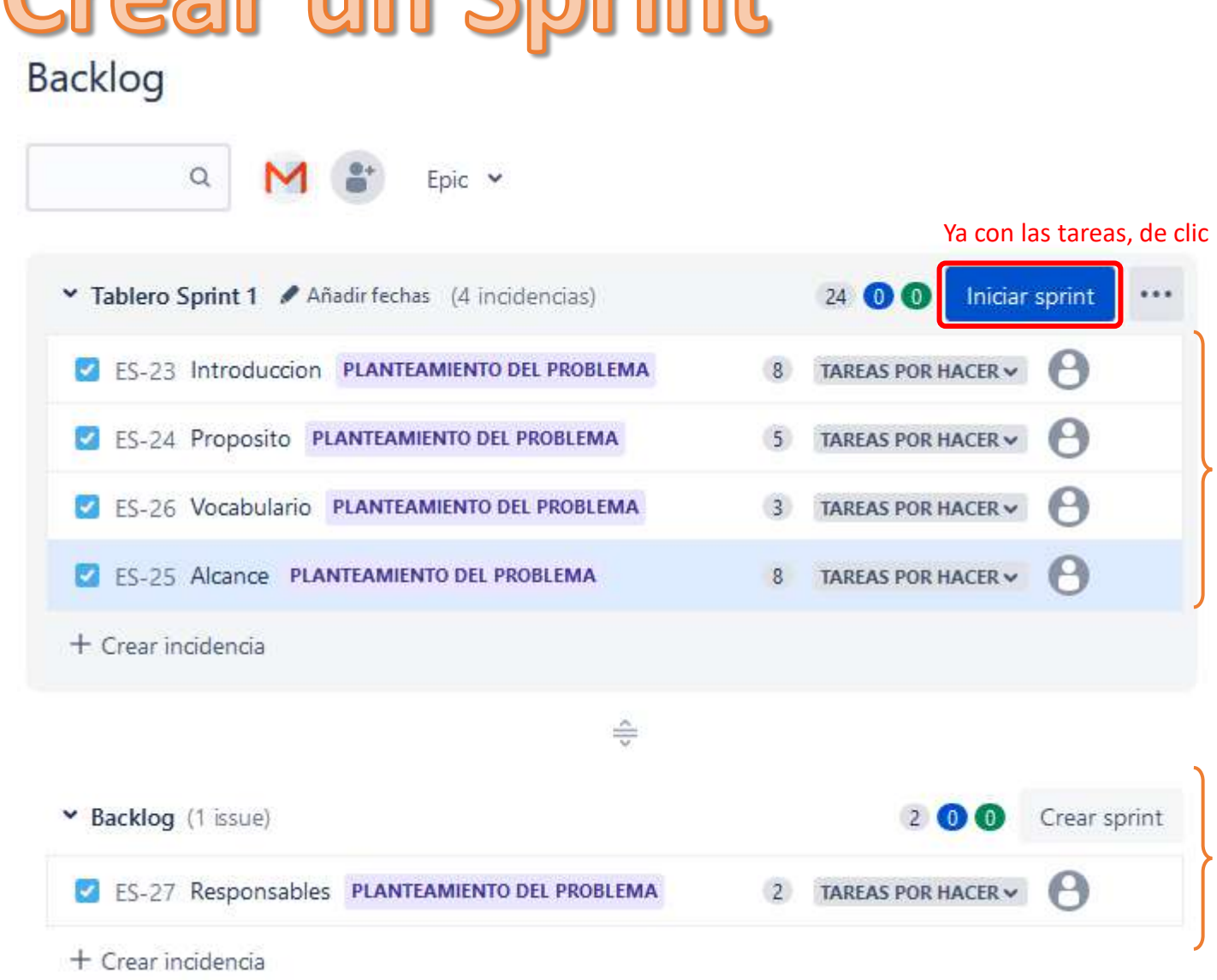


# Crear un Sprint

La pantalla cambiara y lo que se debe hacer es arrastrar las tareas a realizar durante el sprint ala zona o parte superior, también hay una opción para crear directamente una incidencia (tarea, épica, historia de usuario o error), en el sprint



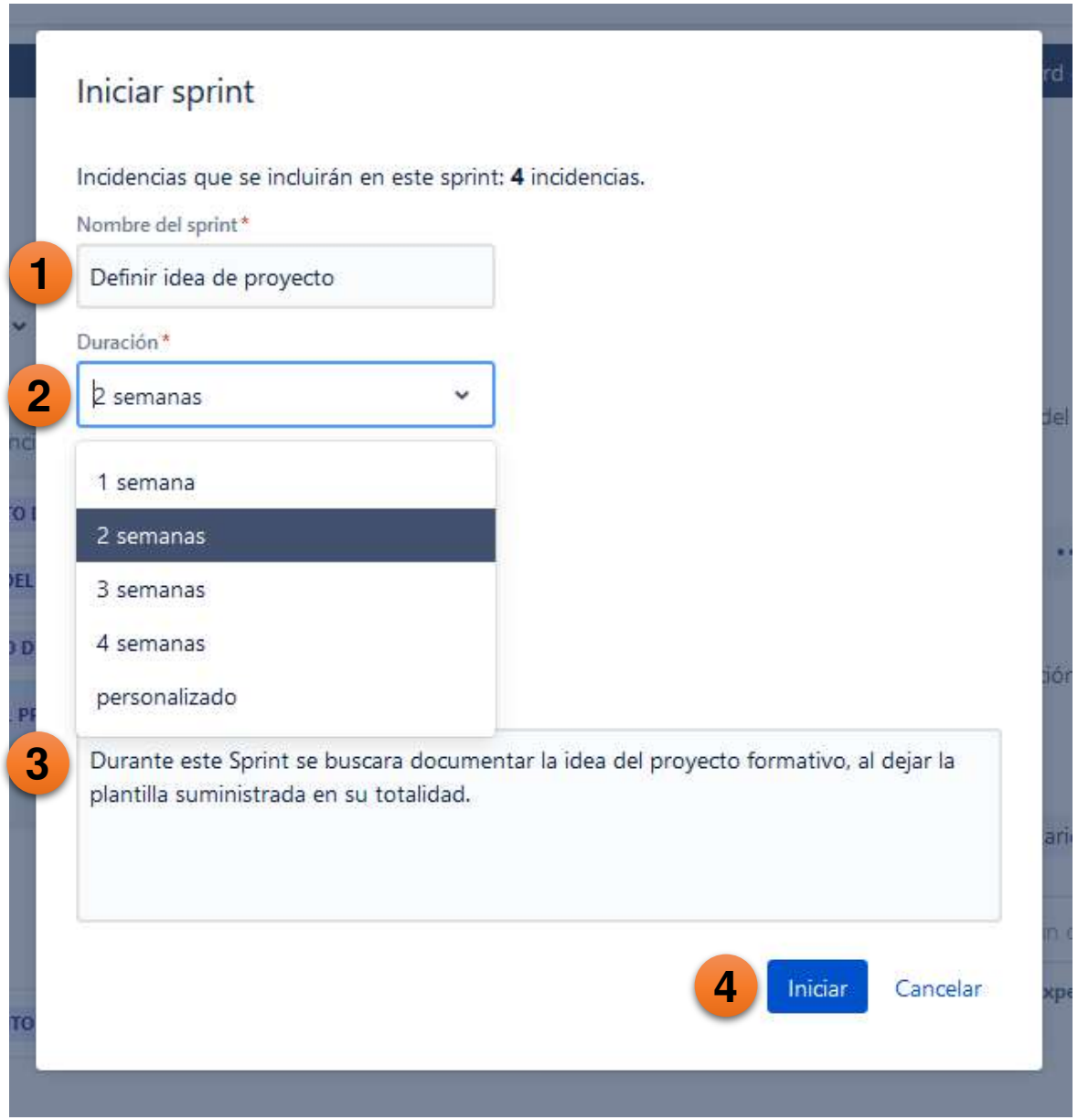
# Crear un Sprint




Tareas a realizar durante el sprint planeado, es ideal que esta queden en el orden de prioridad de arriba abajo, siendo las que están arriba las que primero se deban hacer

Lista de tareas que quedan planeadas para otro sprint, tareas que por su prioridad pueden esperar

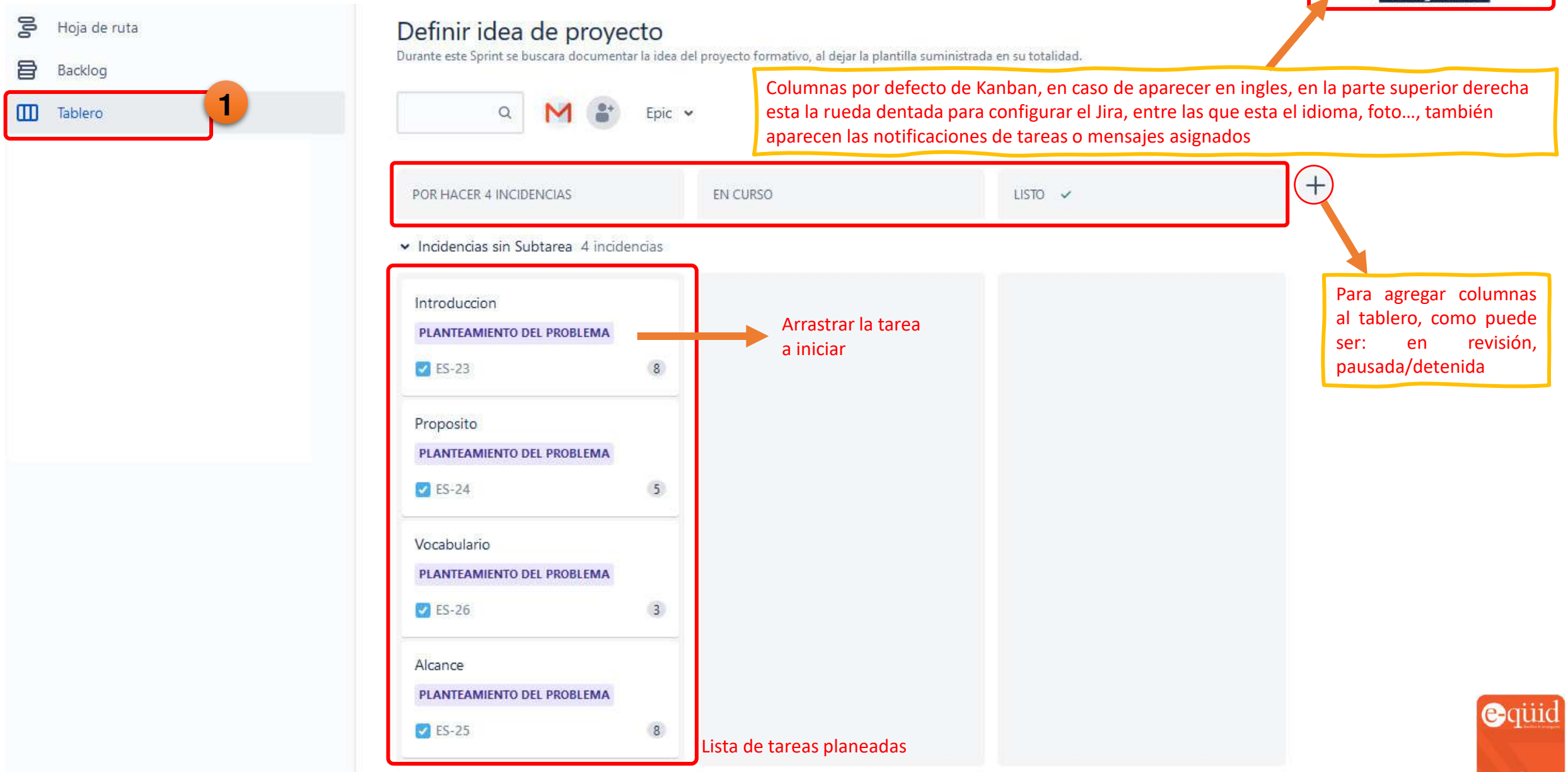
# Crear un Sprint




Al dar clic en iniciar sprint aparecerá una ventana emergente solicitando los siguientes datos:

1. Nombre del Sprint: Se puede personalizar y que este nombre tenga sentido
2. Duración del Sprint: Lista para seleccionar según lo planeado, ya sea 1,2,4 o semanas o personalizar la fecha de inicio y terminación especificando las horas
3. Objetivo del Sprint: Porque es importante realizar el sprint, es una descripción de lo que se quiere alcanzar.
4. Clic para crear el mismo y la plataforma a los segundos pasa al tablero Kanban

# El tablero Kanban



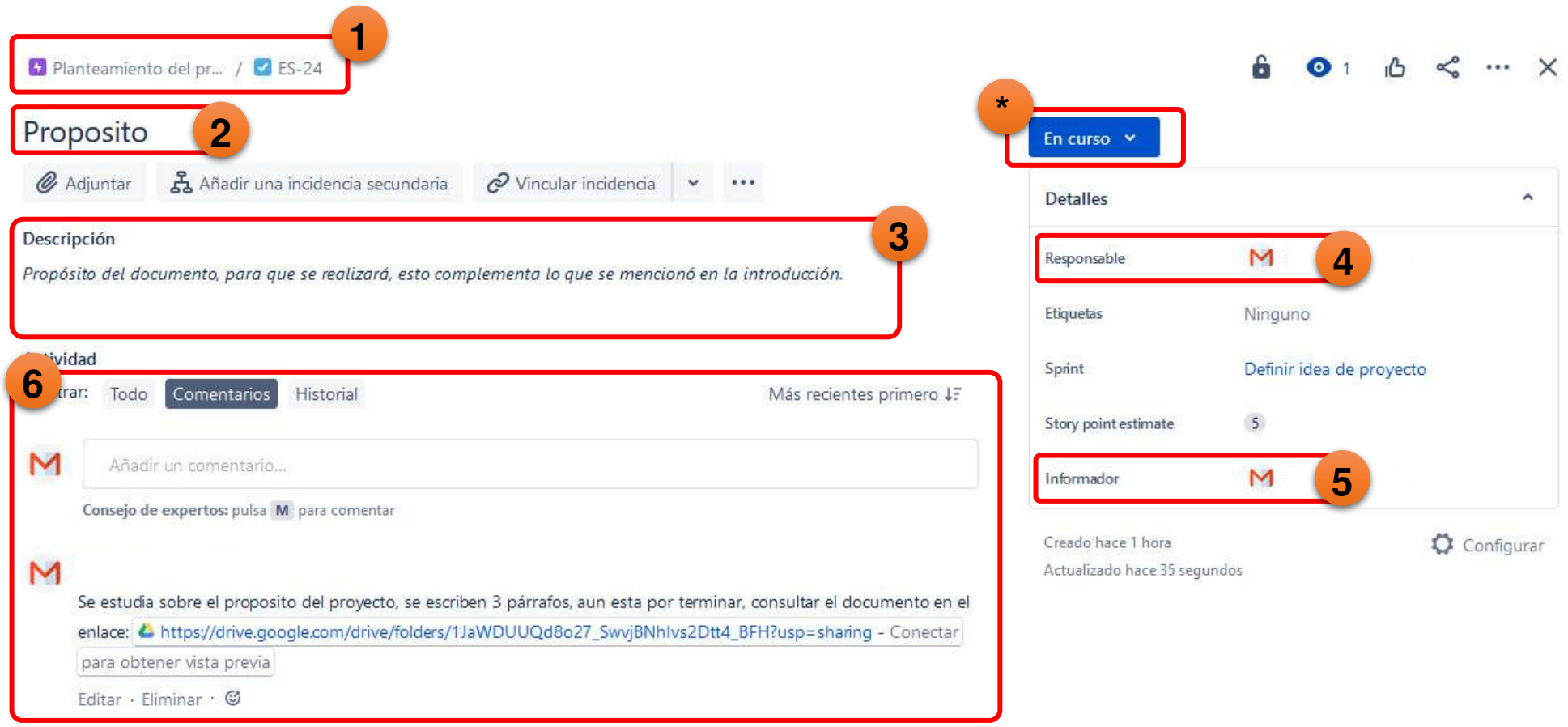
# El tablero Kanban




En el tablero podemos identificar el nombre del Sprint y su objetivo, además de:

1. Personas que pertenecen al equipo
2. Tarea, que al dar clic sobre la misma aparece una ventana emergente que se explica en la siguiente pagina
3. Persona que tiene asignada la tarea o responsable de realizarla, una tarea solo puede ser asignada a un integrante del equipo

# Documentar una tarea




En la ventana emergente aparecen los detalles de la tarea:

1. Código asignado y épica a la que pertenece
2. Nombre de la tarea
3. Descripción de la misma
4. Persona a la que le asignaron la tarea
5. Persona que creo la tarea.

6. Al dar clic en comentarios podemos escribir el avance realizado en la misma, se recomienda usar enlaces a documentos, también al finalizar la tarea documentar la misma

# Terminando el Sprint



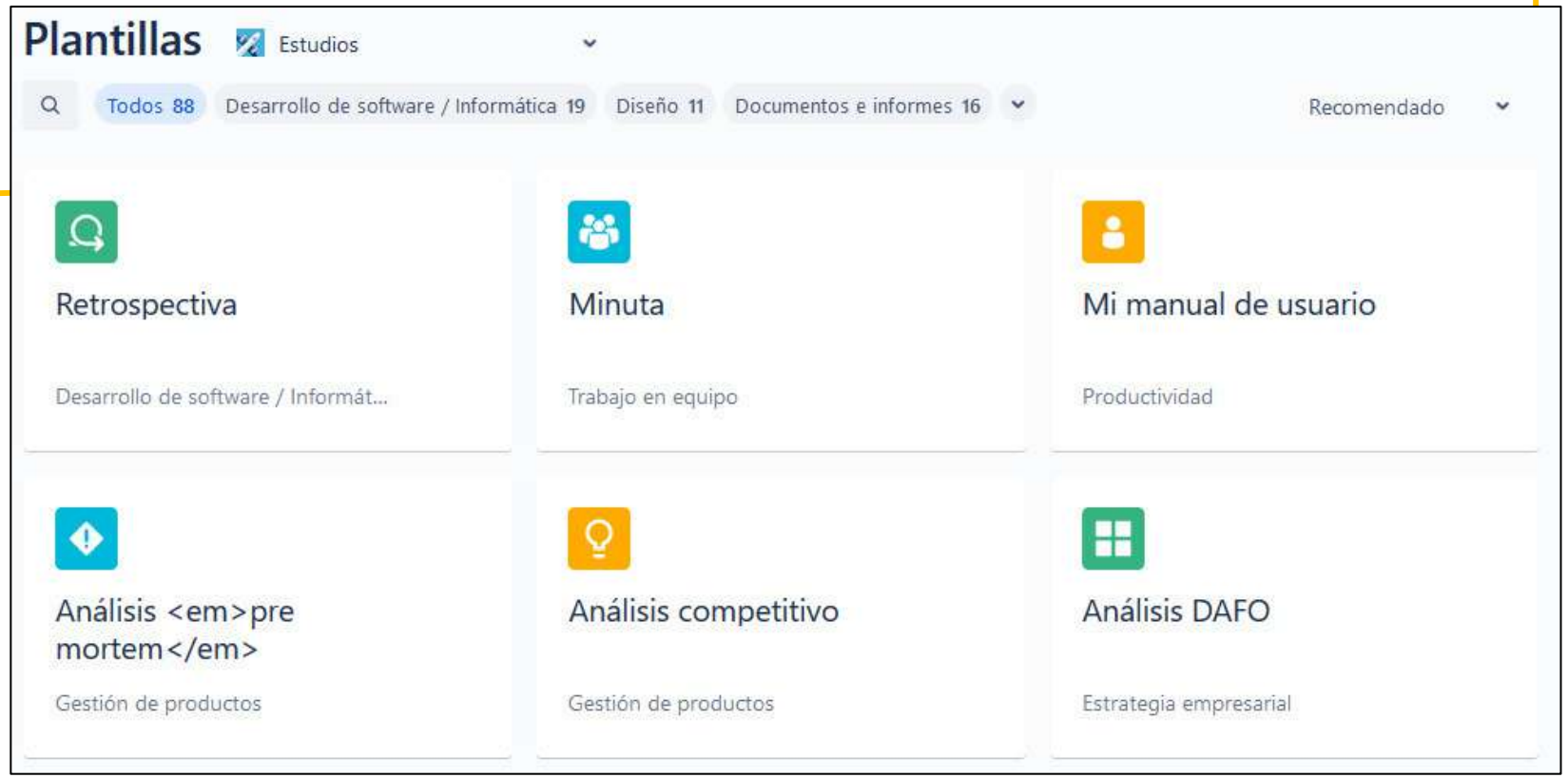

En el mismo tablero de Jira tipo Kanban, aparecen detalles del Sprint como los días que faltan para terminar o que han trascurrido (atrasados), al dar clic en la opción Terminar, aparecerá una ventana informando cuantas tareas se han terminado y cuales no, las tareas que no se han terminado se pueden pasar a otro sprint YA CREADO, o en su defecto egresan al backlog en el estado que estaban (por hacer / en curso).

Se supone que mientas el equipo esta trabajando en el tablero, el Product Owner sigue alimentando el Backlog para el siguiente sprint.

# Documentando…

Este proceso se hace todo el tiempo y por cada integrante del equipo en la parte de “Comentarios”, el Product Owner cuando crea una tarea y la describe: aun así Jira posee plantillas para realizar el proceso como:

- Requisitos
- Actas o notas de reuniones
- Documento de retrospectiva
- Y muchas más…



# Documentando...

Hoja de ruta

Backlog

Tablero

Páginas de proyectos

Opción para acceder a las plantillas

## Páginas de proyectos

Actas

Acta del 20

Manual de usuario

Retrospectives

Documentos ya realizados en la plataforma, estos quedan disponibles y en algunos casos hasta agrupados

Vista previa de la plantilla

### Requisitos del producto

Fill in project details in the table below. Type /date to quickly add the target release date, /jira to link to Jira epics or issues, and @mention functional owners to keep everyone on the same page.

| | |
|---|---|
| Target release | |
| Epic | |
| Document status | DRAFT |
| Document owner | |
| Designer | |
| Tech lead | |
| Technical writers | |
| QA | |

**Objective**

Provide context on this project and explain how it fits into your organization's strategic goals

**Success metrics**

List project goals and the metrics you'll use to judge its success

| Goal | Metric |
|---|---|
| e.g., Simplify the user experience | e.g., Customer satisfaction score increases |
| | |

**Assumptions**

List any assumptions you have about your users, technical constraints, or business goals (e.g., Most users will access this feature from a tablet)

•

**Milestones**

Use the roadmap planner (/roadmap) to help your team stay on track. To edit workstreams or dates, select the placeholder below and tap the pencil icon

### Crear una página

PLANTILLAS POPULARES

Página en blanco
Inicia una página desde cero.

Requisitos del producto
Define, haz un seguimiento y determina el alcance de los requisitos de tu producto o función.

Crear esta página en Confluence

Decisión
Registra las decisiones importantes del proyecto y comunícalas a tu equipo.

Notas
Planifica tus reuniones, toma notas y comparte acciones con tu equipo.

Retrospectiva
¿Qué ha funcionado correctamente?
¿Qué podría haber salido mejor?
Colabora de forma abierta con tu equipo para sugerir mejoras.

Más plantillas

Al seleccionar alguna el navegador se abrirá una nueva pestaña debido a que estos documentos quedan almacenados en otra plataforma llamada Confluence

GC-F-004v.06

El empleo es de todos

Mintrabajo

# Análisis y Desarrollo de Sistemas de Información

SENA

# Conceptos Integración Continua, DevOps, Versión final manual de configuración del sistema, Versión final Documento Informe General del sistema





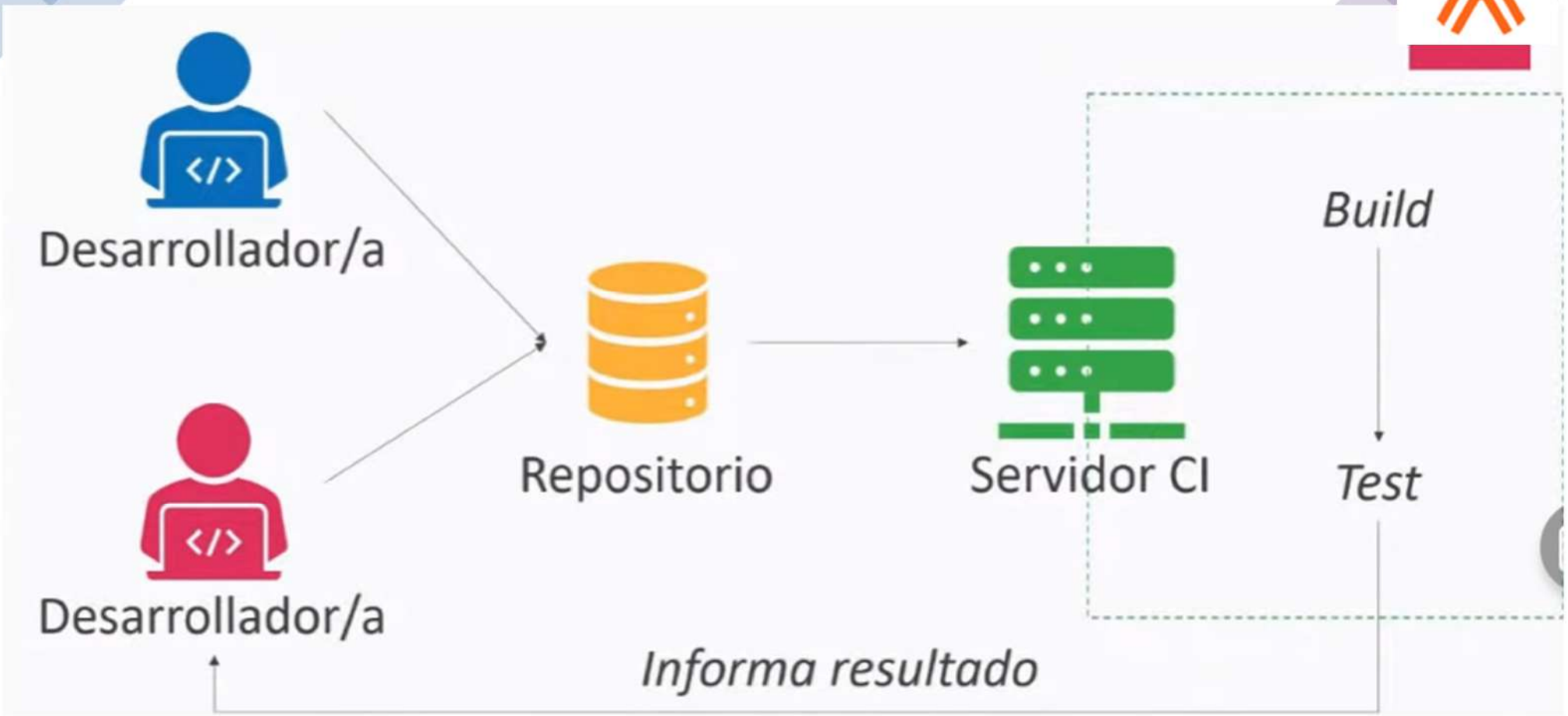

Esta foto de Autor desconocido está bajo licencia CC BY

## INTEGRACIÓN CONTINUA

- La integración continua es una práctica de desarrollo de software mediante la cual los desarrolladores combinan los cambios en el código en un repositorio central de forma periódica, tras lo cual se ejecutan versiones y pruebas automáticas. La integración continua se refiere en su mayoría a la fase de creación o integración del proceso de publicación de software y conlleva un componente de automatización (p. ej., CI o servicio de versiones) y un componente cultural (p. ej., aprender a integrar con frecuencia). Los objetivos clave de la integración continua consisten en encontrar y arreglar errores con mayor rapidez, mejorar la calidad del software y reducir el tiempo que se tarda en validar y publicar nuevas actualizaciones de software.

*Integración continua (CI) es una práctica de desarrollo de software, donde los miembros de un equipo integran su trabajo frecuentemente.*

*Martin Fowler*


Desarrollador/a
Desarrollador/a
Repositorio
Servidor CI
Build
Test
Informa resultado

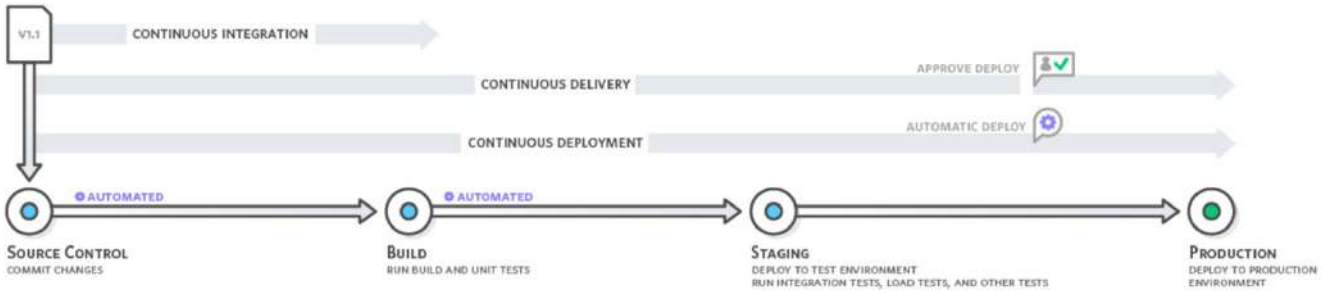
## Integración continua

### ¿Por qué es necesaria la integración continua?

Anteriormente, era común que los desarrolladores de un equipo trabajasen aislados durante un largo periodo de tiempo y solo intentasen combinar los cambios en la versión maestra una vez que habían completado el trabajo. Como consecuencia, la combinación de los cambios en el código resultaba difícil y ardua, además de dar lugar a la acumulación de errores durante mucho tiempo que no se corregían. Estos factores hacían que resultase más difícil proporcionar las actualizaciones a los clientes con rapidez.

### ¿En qué consiste la integración continua?

Con la integración continua, los desarrolladores envían los cambios de forma periódica a un repositorio compartido con un sistema de control de versiones como Git. Antes de cada envío, los desarrolladores pueden elegir ejecutar pruebas de unidad local en el código como medida de verificación adicional antes de la integración. Un servicio de integración continua crea y ejecuta automáticamente pruebas de unidad en los nuevos cambios realizados en el código para identificar inmediatamente cualquier error.



# Beneficios de la integración continua

### Mejore la productividad de desarrollo

La integración continua mejora la productividad del equipo al liberar a los desarrolladores de las tareas manuales y fomentar comportamientos que ayudan a reducir la cantidad de errores y bugs enviados a los clientes.



### Encuentre y arregle los errores con mayor rapidez

Gracias a la realización de pruebas más frecuentes, el equipo puede descubrir y arreglar los errores antes de que se conviertan en problemas más graves.



### Entregue las actualizaciones con mayor rapidez

La integración continua le permite a su equipo entregar actualizaciones a los clientes con mayor rapidez y frecuencia.



# ¿Qué necesitas antes de empezar?



Que todo esté versionado

Build automatizado

Compromiso del equipo

# Procedimiento

En la práctica, la integración continua se aplica del siguiente modo: un ingeniero debe desarrollar una función. Antes de comenzar a trabajar, se descargará la versión completa del código de la aplicación, la denominada línea principal. En esta versión local, también llamada copia de trabajo, será sobre la que trabaje. Cuando haya concluido su tarea, probará el programa, solucionará los posibles errores y podrá volver a cargar la nueva versión en la línea principal.

No obstante, este ingeniero no trabaja solo. Mientras desarrollaba su tarea, otros integrantes del equipo han realizado las suyas y cada uno de ellos tendrá una versión diferente del software en su equipo. Asimismo, probablemente la versión de la línea principal también haya sido modificada por otros desarrolladores en este lapso de tiempo, por lo que tendrá que actualizar su copia local. Si surge algún problema, será su responsabilidad solucionarlo. A continuación, integrará el código en la línea principal y probará de nuevo el programa. Solo cuando no se vuelva a producir ningún error, lo cual podría ocurrir si no ha actualizado correctamente su copia local, habrá concluido el proceso y podrá dedicarse al desarrollo de la siguiente tarea.

# Condiciones para que funcione bien



- *Hacer check-in (commit, push, etc) al menos una vez al día.*
- *Crear una suite de pruebas automatizada.*
- *El build debe ser rápido.*
- *Nada es más importante que arreglar un build roto.*

# Comportamiento

Cuando se trabaja de este modo, deben acatarse una serie de reglas. En la mayoría de las ocasiones, los programadores se rigen por los principios que Martin Fowler describió para llevar a cabo con éxito una integración continua. En primer lugar, se ha de garantizar que todos los implicados se encuentran al mismo nivel, y que no hay nadie cuyo comportamiento pueda causar problemas.

- Un único código fuente
- Automatizar la compilación del proyecto
- Sistemas que realizan sus propias pruebas
- Integración diaria
- Main line operativa
- Reparación inmediata
- Integración rápida
- Pruebas en una réplica del entorno de producción
- Accesibilidad
- Buena comunicación
- Despliegue automatizado

# Ventajas e inconvenientes de la continuous integration

| Ventajas | Inconvenientes |
| --- | --- |
| Detección temprana de errores | Cambio de procesos habituales |
| Comunicación constante | Precisa servidores y entornos adicionales |
| Evita una integración final de gran envergadura | Elaboración de un plan de pruebas propio |
| Registro exacto de las modificaciones | Puede derivar en demoras cuando varios desarrolladores intentan integrar sus códigos al mismo tiempo |
| Disponibilidad constante de una versión funcional y actualizada | |
| Fomenta el trabajo minucioso | |

# Selección de herramientas de integración continua

En principio, la integración continua se puede aplicar sin necesidad de utilizar herramientas específicas, ya que todas las fases pueden llevarse a cabo de forma manual, pero esto requeriría mucho tiempo y disciplina. Con ayuda de las herramientas justas, se puede facilitar el trabajo. Estas herramientas suelen proveer un servidor y ayudan en la compilación del proyecto y en el control de versiones.

- Jenkins
- Travis CI
- Bamboo
- Gitlab CI
- CircleCI
- CruiseControl
- Codeship
- TeamCity



# Referencias bibliograficas

https://aws.amazon.com/es/devops/continuous-integration/

https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/integracion-continua/

GC-F-004v.06

El empleo es de todos

Mintrabajo

# Análisis y Desarrollo de Sistemas de Información

SENA

**Aplicar las técnicas de recolección de datos, diseñando los instrumentos necesarios para el procesamiento de información, de acuerdo con la situación planteada**



Datos

## DATOS

- Es un conjunto: discreto, de factores objetivos sobre un hecho real o bien un registro de transacciones que por sí mismo tiene poca o ninguna relevancia o propósito.

- Un dato es un símbolo o símbolos que describen condiciones, hechos, situaciones o valores.

- Un dato puede significar un número, una letra, un signo ortográfico o cualquier símbolo que represente una cantidad, una medida, una palabra o una descripción. Tienen la capacidad de asociarse dentro de un contexto para convertirse en información. Por si mismos los datos no tienen capacidad de comunicar un significado y por tanto no pueden afectar el comportamiento de quien los recibe.

- Para que un dato sea útil debe convertirse en información, de tal manera que ofrece un significado, conocimiento, ideas o conclusiones.



## TIPOS DE DATOS

### Cuantitativos

Son aquellos que se pueden contar o medir, ejemplo:
Edad,
peso, fecha de transacción.

- Asociados con números
- Se implementan cuando los datos son numéricos.
- Los datos recolectados pueden ser analizados estadísticamente.

Los datos cuantitativos son rápidos y económicos de recopilar, fáciles de analizar y se pueden visualizar en gráficos, diagramas y tablas. Facilitan que las partes interesadas conozcan los números asociados al negocio para hacer estimaciones. Con ellos se realizan conjeturas para entender realmente el valor que aportan.

### Cualitativos

No se pueden contar, ni medir. Ejemplo: sexo, descripción de un artículo, nombre.

- Asociado a los detalles
- Se implementan cuando los datos pueden ser segregados en grupos bien definidos.
- Los datos recogidos solo pueden ser observados y no evaluados

Los datos cualitativos de calidad son muy importantes, ya que nos ayudan a comprender el «¿por qué?» detrás del comportamiento de un usuario. Pueden ayudar a pintar una imagen completa de la persona y nos ayudan a generar empatía con ella. Valoramos cómo razona o piensa, lo que siente y probablemente qué necesita.




Imagen tomada de https://www.questionpro.com/es/cualitativa-vs-cuantitativa.html

## DATOS -> INFORMACIÓN -> CONOCIMIENTO

### Integridad de los datos

Significa que todos los datos requeridos para responder a una pregunta específica están disponibles.

Por ejemplo, un marcador de un partido de futbol, debe incluir el marcador de goles de los dos equipos, si se anuncia el marcador de un solo equipo este será incompleto y no tendría sentido.

Los datos pueden ser: inequívocos cuando el contexto no es claro. Por ejemplo, la expresión 2-x puede parecer:
•"La cantidad 2 menos la cantidad desconocida llamada x" para un estudiante de algebra.
•Puede significar "2 barra x" a un vaquero que marca ganado.

Tenemos que conocer el contexto de estos símbolos antes de poder conocer su significado..

### Operaciones Sobre los datos

- **Captura:** se registran los datos antes de ser procesados. Puede ser manual o mediante cualquier dispositivo de entrada directa (teclado, caja registradora, lector óptico, captación sonora).
- **Validación**: se verifican y corrigen los datos durante la captura o después de ésta con la finalidad de minimizar el número de errores cometidos en su trascripción. Verifica que los datos capturados cumplan con ciertos parámetros previamente establecidos para el control de los mismos desde el punto de vista de su consistencia.
- **Almacenamiento**: se guardan los datos previamente capturados para su conservación en cualquier dispositivo físico (papel, magnético, microfilm, etc.)
- **Recuperación**: se logra el acceso posterior a los datos almacenados.
- **Reproducción**: se copian o trasladar los datos de un dispositivo a otro. Ejemplo: impresión de un
- documento, envío por internet,
- etc.

## DATOS -> INFORMACIÓN -> CONOCIMIENTO

### Procesamiento de los datos

Son las acciones sobre cualquier tipo de datos, para luego obtener información oportuna y útil en el logro de un mayor control y mejor toma de decisiones.
Se pueden clasificar una serie de pasos. A continuación se describen los pasos.

1. **Entrada:** Registro de datos en un medio adecuado para su manejo y procesamiento.
2. **Proceso:** Pueden organizarse de diferentes maneras:
    - Clasificación: Establecer un orden lógico para los datos según uno de sus atributos. Ejemplo: número de cédula, apellido, nivel de ventas, etc.
    - Agrupación: Consiste en separar sistemáticamente los datos por categorías. Ejemplos: tipo de productos, departamentos, rangos, etc.
    - Cálculos: Se refieren a los procesos aritméticos (+, -, * y /) y operaciones lógicas para convertir los datos en una forma significativa.
    - Síntesis: Consiste en sustituir grandes volúmenes de datos en información más fácil de interpretar. Ejemplos: gráficos de barras, líneas, tablas, etc.
3. **Salidas y/o resultados:** Información procesada que se obtiene del ciclo de procesamiento de datos en un medio de salida (papel o dispositivo magnético).
4. **Evaluación de resultados:** análisis las salidas de acuerdo a los objetivos y metas. De acuerdo con esto se puede ejercer nuevas acciones sobre los datos de entrada, en caso de ser necesario.

## DATOS -> INFORMACIÓN -> CONOCIMIENTO

### Concepto de Información

Se define como “un mensaje, normalmente bajo la forma de un documento o algún tipo de comunicación audible, gráfica o visible, que tiene un emisor y un receptor;
capaz de impactar sobre sus juicios de valor y comportamiento al proporcionar ciertas diferencias en
su interior o exterior. “ [3].
Es una colección de hechos significativos pertinentes, para una organización. Se puede afirmar que información
es un conjunto de datos significativos y pertinentes que describan sucesos o entidades.

### Características de la información

Se clasifican según:

- **Significado (semántica):** ¿Qué quiere decir?
- Importancia (relativa al receptor): ¿Trata sobre alguna cuestión importante?
- **Vigencia (en la dimensión espacio-tiempo):** ¿Es actual o desfasada?
- **Validez (relativa al emisor):** ¿El emisor es fiable o puede proporcionar información no válida (falsa)?
- **Valor (activo intangible volátil):** ¿Resulta útil para el destinatario?

# DATOS -> INFORMACIÓN -> CONOCIMIENTO

## DIFERENCIAS ENTRE DATO E INFORMACIÓN

Los Datos a diferencia de la información son utilizados como diversos métodos para comprimir la información a fin de permitir una transmisión o almacenamiento más eficaz.

- La cantidad de información de un mensaje puede ser entendida como el número de símbolos posibles que representan el mensaje. Los símbolos que representan el mensaje no son más que datos significativos.
- La información es un mensaje con un contenido determinado emitido por una persona hacia otra y, como tal, representa un papel primordial en el proceso de la comunicación, a la vez que posee una evidente función social.

A diferencia de los datos, la información tiene significado para quien la recibe, por eso, los seres humanos siempre han tenido la necesidad de cambiar entre sí información que luego transforman en acciones.
La información es, entonces, conocimiento basado en los datos, a los cuales mediante un procesamiento, se les ha dado significado, propósito y utilidad.

Una empresa de telefonía celular recibe los siguientes datos cada mes entre varios documentos de sus clientes:
7684444, Pedro Martínez, marzo 5 de 2012, abril 4 de 2012, abril 25 de 2012, 3005763812, 65000, Calle 4 No. 2-80, Popayán.
De este modo, cada dato por sí solo, no nos dice nada, por ello tienen que pasar por un proceso (análisis, ordenamiento, agregado, etc.), el cuál le da sentido a los datos anteriores:

# Referencias bibliograficas

Senn, James. (1986). Sistemas de información para la administración. Editorial Iberoamericana.

Gispert , Carlos.”Enciclopedia Autodidáctica Interactiva “. Sexta Edición, Océano, pp.1540 Tomo 6, México, 2002.

“Gran Diccionario Enciclopédico Ilustrado”. Séptima edición, Selecciones del Reader´s Digest, pp.1050, México, 2000.

http://www.ucla.edu.ve/dac/departamentos/informatica%20I/sesion%20no.%201.pdf