RESUMEN SESIÓN EN LÍNEA

Fecha de la Sesión:	12/11/2024
Competencia:	TÉCNICA
Tema Central:	GRABACIÓN Y RESUMEN CONFERENCIA CONTROL DE VERSIONES DE SOFTWARE
Instructor Responsable:	ILMER CUELLO

Desarrollo de la Sesión

- ✓ **COMPETENCIA: 220501096 - DESARROLLAR LA SOLUCIÓN DE SOFTWARE DE ACUERDO CON EL DISEÑO Y METODOLOGÍAS DE DESARROLLO**

- ✓ **RESULRADO DE APRENDIZAJE:** 220501096- 01 planear actividades de construcción del software de acuerdo con el diseño establecido.

  - ✓ Se inicia la sesión en línea a las 6:30 pm con los aprendices de las fichas: 2868399, del programa ANÁLISIS Y DESARROLLO DE SOFTWARE. Posterior a la solicitud de aprobación para grabar la sesión en línea, se inicia explicando las evidencias:

  - ✓ RAP 15_ GA7-220501096-AA1-EV01_IVO_ Informe técnico de plan de trabajo para construcción de software.

  - ✓ RAP 15_ GA7-220501096-AA1-EV02_ITE_ Definir estándares de codificación de acuerdo a plataforma de desarrollo elegida.

  - ✓ RAP 15_ GA7-220501096-01-AA1-EV03_IC_ Identifica herramientas de versionamiento.

  - ✓ RAP 15_ GA7-220501096-AA1-EV04_RM_ Instalación y configuración de herramienta de versionamiento (Local / Web).

  - ✓ RAP 15_ GA7-220501096-AA1-EV05_IC_ Herramientas de versionamiento (GIT) instalada y configurada.

ENLACE GRABACIÓN:

https://drive.google.com/file/d/1IYD9qA9kC0nAZ23sHR-tpCeo3oBaYFEw/view?usp=sharing

GRABACIÓN Y RESUMEN CONFERENCIA WEB RAP 15 – VERSIONAMEINTO 12/01/2024

Desarrollo de las evidencias "RAP 15"

- ✓ **RAP 15_ GA7-220501096-AA1-EV01_IVO_ Informe técnico de plan de trabajo para construcción de software.**

Para estructurar un informe técnico de un plan de trabajo para la construcción de software, considera los siguientes apartados

Nombre del proyecto	
Objetivo del proeycto	
Alcance del proyecto	
Cronograma: Diagrama de Gantt u otro cronograma que detalle las fases y tiempos del proyecto.	
Requerimientos Funcionales:	
Requerimientos No funcionales:	
Recursos Necesarios	
Metodología de Trabajo	
Gestión de Riesgos	
Control y Seguimiento	

- ✓ **RAP 15_ GA7-220501096-AA1-EV02_ITE_ Definir estándares de codificación de acuerdo a plataforma de desarrollo elegida.**

Definir estándares de codificación según la plataforma de desarrollo elegida implica establecer un conjunto de reglas y buenas prácticas para escribir el código, adaptadas a las características específicas de la plataforma o tecnología en uso (como Java, .NET, Python, entre otras). Estos estándares ayudan a mantener la coherencia, legibilidad y calidad del código a lo largo del desarrollo del software, facilitando la colaboración entre los desarrolladores y reduciendo la posibilidad de errores. Algunos aspectos comunes incluyen:

Convenciones de Nombres: Reglas sobre cómo nombrar variables, métodos, clases y otros elementos.
Estructura de Archivos y Carpetas: Organización de los archivos y directorios del proyecto.
Formato del Código: Indentación, espacios, uso de llaves, líneas en blanco, etc.
Manejo de Errores y Excepciones: Cómo capturar y manejar los errores en el código.
Documentación: Estándares para comentarios y documentación en el código.
Pruebas y Validaciones: Definición de prácticas de prueba, como pruebas unitarias y de integración.

- ✓ **RAP 15_ GA7-220501096-01-AA1-EV03_IC_ Identifica herramientas de versionamiento.**

Existen diversas herramientas de control de versiones que se utilizan para gestionar cambios en el código fuente a lo largo del tiempo. Aquí te presento algunas de las más populares y utilizadas en la industria del desarrollo de software:

Hacer un análisis de las diferentes herramientas, y concluir cual va a ser la herramienta para seleccionar.

1. Git

- **Descripción**: Es una de las herramientas de control de versiones más populares y utilizadas. Es un sistema distribuido, lo que significa que cada desarrollador tiene una copia completa del repositorio en su propia máquina.
- **Características**:
  - Control de versiones distribuido.
  - Compatibilidad con múltiples flujos de trabajo (ramas, fusiones).
  - Amplio soporte en la comunidad.
- **Plataformas**: GitHub, GitLab, Bitbucket.

2. Subversion (SVN)

- **Descripción**: Es una herramienta centralizada de control de versiones. Es conocida por su simplicidad y facilidad de uso.
- **Características**:
  - Control de versiones centralizado.
  - Administración sencilla de permisos.
  - Integración con diversas herramientas de desarrollo.
- **Plataformas**: TortoiseSVN, VisualSVN.

3. Mercurial

- **Descripción**: Similar a Git, Mercurial es un sistema de control de versiones distribuido. Es conocido por su rendimiento y facilidad de uso.
- **Características**:
  - Control de versiones distribuido.
  - Alta performance y manejo eficiente de grandes proyectos.
  - Facilidad de aprendizaje y uso.
- **Plataformas**: Bitbucket (también soporta Git).

4. Perforce (Helix Core)

- **Descripción**: Es una herramienta de control de versiones centralizada, utilizada principalmente en grandes empresas debido a su capacidad para manejar proyectos a gran escala.
- **Características**:
  - Control de versiones centralizado.
  - Alta escalabilidad y rendimiento.
  - Soporte robusto para grandes binarios y archivos.
- **Plataformas**: Helix Core, Helix Swarm.

5. CVS (Concurrent Versions System)

- **Descripción**: Es una de las herramientas de control de versiones más antiguas y ha sido ampliamente reemplazada por herramientas más modernas como Git y SVN.
- **Características**:
    - Control de versiones centralizado.
    - Sistema maduro y probado.
    - Menos común en nuevos proyectos.

6. Bazaar

- **Descripción**: Similar a Git y Mercurial, es un sistema de control de versiones distribuido. Fue desarrollado por Canonical, la compañía detrás de Ubuntu.
- **Características**:
    - Control de versiones distribuido.
    - Fácil integración con Launchpad.
    - Simplicidad y facilidad de uso.

7. Plastic SCM

- **Descripción**: Es una herramienta de control de versiones distribuida y centralizada que se enfoca en manejar la complejidad de grandes proyectos y equipos distribuidos.
- **Características**:
    - Soporte tanto para flujos de trabajo distribuidos como centralizados.
    - Herramientas visuales avanzadas para la gestión de ramas y fusiones.
    - Alta performance y escalabilidad.

8. ClearCase

- **Descripción**: Desarrollada por IBM, es una herramienta de control de versiones centralizada, utilizada principalmente en entornos corporativos grandes.
- **Características**:
    - Control de versiones centralizado.
    - Soporte robusto para configuraciones complejas.
    - Integración con otras herramientas de IBM.

- ✓ **RAP 15_ GA7-220501096-AA1-EV04_RM_ Instalación y configuración de herramienta de versionamiento (Local / Web).**

  Crear un vídeo haciendo la instalación de la herramienta de versionamiento

Recurso:
Descarga e Instalación de Git | Cómo descargar e Instalar Git en Windows | Explicación paso a paso
https://www.youtube.com/watch?v=jdXKwLNUfmg

- ✓ **RAP 15_ GA7-220501096-AA1-EV05_IC_ Herramientas de versionamiento (GIT) instalada y configurada.**

  Crear un vídeo haciendo la configuración de la herramienta de versionamiento

EVIDENCIA.

Documento Word o pdf.

- ✓ Portada
- ✓ Introducción
- ✓ Tabla de contenido
- ✓ Objetivos
- ✓ Desarrollo de las evidencias “RAP 15”
- ✓ Conclusiones
- ✓ Referencias bibliográficas.

Nota: enviar archivo .zip con la evidencia y los archivos ejecutables de la actividad