We are having a problem billing the locqum organization. Please update your payment method or call your payment provider for details on why the transaction failed.
You can always contact support with any questions.

Commit c227a50

claude committed last month Verified

```
feat: auth con password y flujo de compra con autenticacion

Backend:
- Reemplaza OTP por autenticacion con password (BCrypt)
- POST /api/auth/register: email, password, firstName, lastName
- POST /api/auth/login: email, password
- Elimina endpoints OTP (otp/request, otp/verify)
- PasswordEncoder bean (BCryptPasswordEncoder) en SecurityConfig
- POST /api/payments ahora requiere autenticacion (usa email del JWT)
- User model/entity con campo password_hash
- DB migration V10: agrega columna password_hash

Frontend:
- LoginComponent: formulario email + password
- RegisterComponent: email (x2), nombres, apellidos, password (x2)
- AuthService: metodos login() y register() con password
- PurchaseComponent: al pagar, si no esta autenticado guarda
  carrito en localStorage y redirige a /auth/register.
  Al volver, restaura el carrito automaticamente.
- Eliminado formulario de email/tel/OTP del flujo de compra

https://claude.ai/code/session_01WTbEJHa9nd8kVVoKCirXH8
```

master

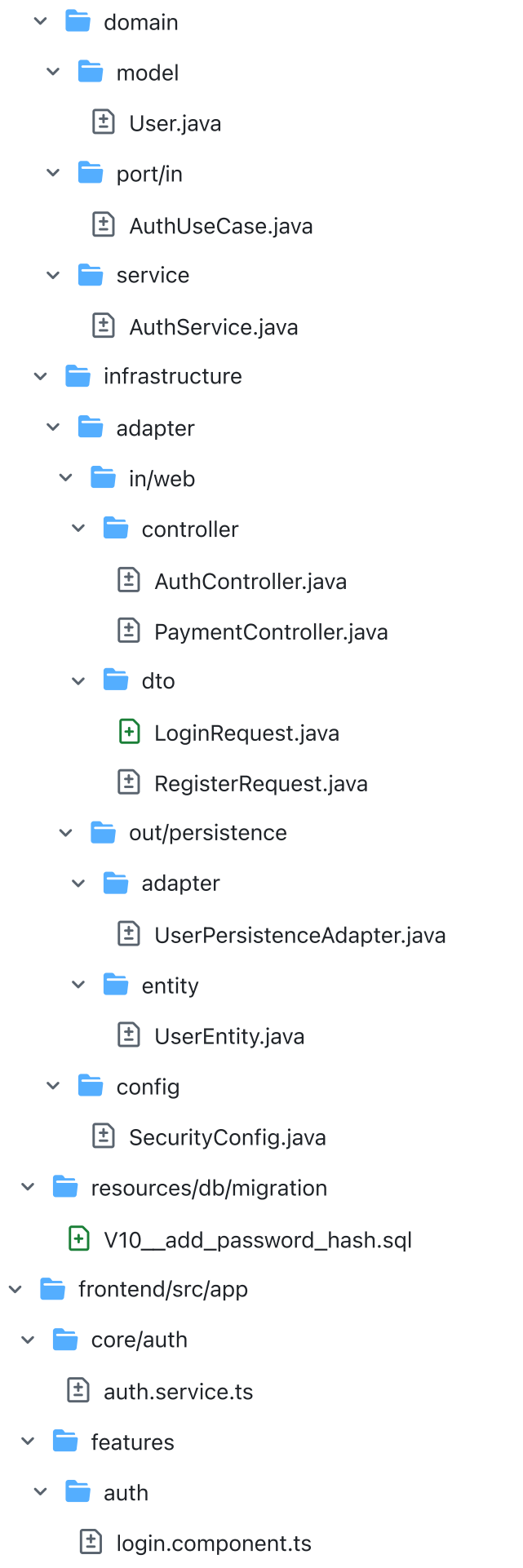
1 parent 0c67b77 commit c227a50

15 files changed +233 -459 lines changed ↑ Top

Filter files...

backend/src/main

java/co/gov/loteriabogota/loteriabackend


domain
model
User.java
port/in
AuthUseCase.java
service
AuthService.java
infrastructure
adapter
in/web
controller
AuthController.java
PaymentController.java
dto
LoginRequest.java
RegisterRequest.java
out/persistence
adapter
UserPersistenceAdapter.java
entity
UserEntity.java
config
SecurityConfig.java
resources/db/migration
V10__add_password_hash.sql
frontend/src/app
core/auth
auth.service.ts
features
auth
login.component.ts

register.component.ts

purchase

purchase.component.ts

15 files changed +233 -459 lines changed ↑ Top Search within code

…abogota/loteriabackend/domain/model/User.java

```
@@ -10,6 +10,7 @@ public class User {
10  10
11  11      private UUID id;
12  12      private Email email;
    13  +   private String passwordHash;
13  14      private String firstName;
14  15      private String lastName;
15  16      private String name;
@@ -34,8 +35,9 @@ public static User createBasic(Email email) {
34  35          return new User(UUID.randomUUID(), email, KycLevel.BASIC);
35  36      }
36  37
37      -   public static User createWithName(Email email, String firstName, String lastName) {
    38  +   public static User createWithPassword(Email email, String passwordHash, String firstName, String lastName) {
38  39          User user = new User(UUID.randomUUID(), email, KycLevel.BASIC);
    40  +       user.setPasswordHash(passwordHash);
39  41          user.setFirstName(firstName);
40  42          user.setLastName(lastName);
41  43          user.setName(firstName + " " + lastName);
@@ -55,6 +57,8 @@ public String getFullName() {
55  57      public void setId(UUID id) { this.id = id; }
56  58      public Email getEmail() { return email; }
57  59      public void setEmail(Email email) { this.email = email; }
    60  +   public String getPasswordHash() { return passwordHash; }
    61  +   public void setPasswordHash(String passwordHash) { this.passwordHash = passwordHash; }
58  62      public String getFirstName() { return firstName; }
59  63      public void setFirstName(String firstName) { this.firstName = firstName; }
60  64      public String getLastName() { return lastName; }
```

…oteriabackend/domain/port/in/AuthUseCase.java

```
@@ -4,11 +4,9 @@

 public interface AuthUseCase {

-    void requestOtp(Email email);
+    AuthResult register(Email email, String password, String firstName, String lastName);

-    AuthResult verifyOtp(Email email, String otpCode);
-
-    AuthResult register(Email email, String otpCode, String firstName, String lastName);
+    AuthResult login(Email email, String password);

     AuthResult refreshToken(String refreshToken);
```

…oteriabackend/domain/service/AuthService.java

```
@@ -1,87 +1,58 @@
 package co.gov.loteriabogota.loteriabackend.domain.service;

 import co.gov.loteriabogota.loteriabackend.domain.exception.DomainException;
-import co.gov.loteriabogota.loteriabackend.domain.exception.InvalidOtpException;
 import co.gov.loteriabogota.loteriabackend.domain.model.User;
 import co.gov.loteriabogota.loteriabackend.domain.port.in.AuthUseCase;
-import co.gov.loteriabogota.loteriabackend.domain.port.out.OtpStore;
 import co.gov.loteriabogota.loteriabackend.domain.port.out.UserRepository;
 import co.gov.loteriabogota.loteriabackend.domain.vo.Email;
 import co.gov.loteriabogota.loteriabackend.infrastructure.security.JwtProvider;
 import lombok.extern.slf4j.Slf4j;
+import org.springframework.security.crypto.password.PasswordEncoder;
 import org.springframework.stereotype.Service;

-import java.security.SecureRandom;
-
 @Slf4j
 @Service
 public class AuthService implements AuthUseCase {
```

```
19  16
20      -     private final OtpStore otpStore;
21  17        private final UserRepository userRepository;
22  18        private final JwtProvider jwtProvider;
23      -     private final SecureRandom random = new SecureRandom();
    19  +     private final PasswordEncoder passwordEncoder;
24  20
25      -     public AuthService(OtpStore otpStore, UserRepository userRepository,
            JwtProvider jwtProvider) {
26      -         this.otpStore = otpStore;
    21  +     public AuthService(UserRepository userRepository, JwtProvider jwtProvider,
            PasswordEncoder passwordEncoder) {
27  22            this.userRepository = userRepository;
28  23            this.jwtProvider = jwtProvider;
    24  +         this.passwordEncoder = passwordEncoder;
29  25        }
30  26
31  27        @Override
32      -     public void requestOtp(Email email) {
33      -         String otp = String.format("%06d", random.nextInt(1_000_000));
34      -         otpStore.save(email, otp);
35      -         log.info("OTP generated for {}: {}", email.value(), otp);
36      -     }
37      -
38      -     @Override
39      -     public AuthResult verifyOtp(Email email, String otpCode) {
40      -         String stored = otpStore.find(email)
41      -                 .orElseThrow(InvalidOtpException::new);
42      -
43      -         if (!stored.equals(otpCode)) {
44      -             throw new InvalidOtpException();
    28  +     public AuthResult register(Email email, String password, String firstName,
            String lastName) {
    29  +         if (userRepository.findByEmail(email).isPresent()) {
    30  +             throw new DomainException("Ya existe una cuenta con este correo. Usa
            iniciar sesion.");
45  31            }
46  32
47      -         otpStore.remove(email);
48      -
49      -         boolean newUser = false;
50      -         User user = userRepository.findByEmail(email).orElse(null);
```

```
51        -         if (user == null) {
52        -             user = User.createBasic(email);
53        -             user = userRepository.save(user);
54        -             newUser = true;
55        -         }
      33  +         String hash = passwordEncoder.encode(password);
      34  +         User user = User.createWithPassword(email, hash, firstName, lastName);
      35  +         user = userRepository.save(user);
56    36
57    37            String accessToken =
          jwtProvider.generateAccessToken(user.getId().toString());
58    38            String refreshToken =
          jwtProvider.generateRefreshToken(user.getId().toString());
59    39
60        -         return new AuthResult(accessToken, refreshToken, newUser);
      40  +         return new AuthResult(accessToken, refreshToken, true);
61    41        }
62    42
63    43        @Override
64        -     public AuthResult register(Email email, String otpCode, String firstName,
          String lastName) {
65        -         String stored = otpStore.find(email)
66        -                 .orElseThrow(InvalidOtpException::new);
      44  +     public AuthResult login(Email email, String password) {
      45  +         User user = userRepository.findByEmail(email)
      46  +                 .orElseThrow(() -> new DomainException("Correo o contraseña
          incorrectos"));
67    47
68        -         if (!stored.equals(otpCode)) {
69        -             throw new InvalidOtpException();
      48  +         if (user.getPasswordHash() == null || !passwordEncoder.matches(password,
          user.getPasswordHash())) {
      49  +             throw new DomainException("Correo o contraseña incorrectos");
70    50            }
71    51
72        -         otpStore.remove(email);
73        -
74        -         if (userRepository.findByEmail(email).isPresent()) {
75        -             throw new DomainException("Ya existe una cuenta con este correo. Usa
          iniciar sesion.");
76        -         }
77        -
```

```
78        -         User user = User.createWithName(email, firstName, lastName);
79        -         user = userRepository.save(user);
80        -
81    52            String accessToken =
          jwtProvider.generateAccessToken(user.getId().toString());
82    53            String refreshToken =
          jwtProvider.generateRefreshToken(user.getId().toString());
83    54
84        -         return new AuthResult(accessToken, refreshToken, true);
      55  +         return new AuthResult(accessToken, refreshToken, false);
85    56        }
86    57
87    58        @Override
```

…adapter/in/web/controller/AuthController.java

```
          @@ -17,29 +17,23 @@ public AuthController(AuthUseCase authUseCase) {
17    17            this.authUseCase = authUseCase;
18    18        }
19    19
20        -     @PostMapping("/otp/request")
21        -     public ResponseEntity<Void> requestOtp(@Valid @RequestBody OtpRequest request)
          {
22        -         authUseCase.requestOtp(new Email(request.email()));
23        -         return ResponseEntity.ok().build();
24        -     }
25        -
26        -     @PostMapping("/otp/verify")
27        -     public ResponseEntity<AuthResponse> verifyOtp(@Valid @RequestBody
          OtpVerifyRequest request) {
28        -         AuthUseCase.AuthResult result = authUseCase.verifyOtp(
29        -                 new Email(request.email()), request.otpCode());
30        -         return ResponseEntity.ok(new AuthResponse(
31        -                 result.accessToken(), result.refreshToken(), result.newUser()));
32        -     }
33        -
34    20        @PostMapping("/register")
35    21        public ResponseEntity<AuthResponse> register(@Valid @RequestBody
          RegisterRequest request) {
36    22            AuthUseCase.AuthResult result = authUseCase.register(
37        -                 new Email(request.email()), request.otpCode(),
```

```
                 new Email(request.email()), request.password(),
                 request.firstName(), request.lastName());
         return ResponseEntity.ok(new AuthResponse(
                 result.accessToken(), result.refreshToken(), result.newUser()));
     }

+    @PostMapping("/login")
+    public ResponseEntity<AuthResponse> login(@Valid @RequestBody LoginRequest request) {
+        AuthUseCase.AuthResult result = authUseCase.login(
+                new Email(request.email()), request.password());
+        return ResponseEntity.ok(new AuthResponse(
+                result.accessToken(), result.refreshToken(), result.newUser()));
+    }
+
     @PostMapping("/refresh")
     public ResponseEntity<AuthResponse> refresh(@Valid @RequestBody RefreshTokenRequest request) {
         AuthUseCase.AuthResult result = authUseCase.refreshToken(request.refreshToken());
```

…pter/in/web/controller/PaymentController.java

```
@@ -3,21 +3,21 @@
 import co.gov.loteriabogota.loteriabackend.domain.model.Lottery;
 import co.gov.loteriabogota.loteriabackend.domain.model.LotteryTicket;
 import co.gov.loteriabogota.loteriabackend.domain.model.PaymentOrder;
-import co.gov.loteriabogota.loteriabackend.domain.exception.DomainException;
+import co.gov.loteriabogota.loteriabackend.domain.model.User;
 import co.gov.loteriabogota.loteriabackend.domain.port.in.LotteryUseCase;
 import co.gov.loteriabogota.loteriabackend.domain.port.in.PaymentUseCase;
-import co.gov.loteriabogota.loteriabackend.domain.port.in.PhoneVerificationUseCase;
 import co.gov.loteriabogota.loteriabackend.domain.port.in.TicketUseCase;
+import co.gov.loteriabogota.loteriabackend.domain.port.in.UserUseCase;
 import co.gov.loteriabogota.loteriabackend.domain.port.out.TicketRepository;
 import co.gov.loteriabogota.loteriabackend.domain.vo.Email;
-import co.gov.loteriabogota.loteriabackend.domain.vo.PhoneNumber;
 import co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.CreatePaymentRequest;
```

```
15  14    import co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.CreatePaymentResponse;
16  15    import co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.PaymentStatusResponse;
17  16    import co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.TicketResponse;
18  17    import jakarta.servlet.http.HttpServletRequest;
19  18    import jakarta.validation.Valid;
20  19    import org.springframework.http.ResponseEntity;
    20  +  import org.springframework.security.core.Authentication;
21  21    import org.springframework.web.bind.annotation.*;
22  22
23  23    import java.util.List;
        @@ -28,25 +28,29 @@
28  28    public class PaymentController {
29  29
30  30        private final PaymentUseCase paymentUseCase;
31      -     private final PhoneVerificationUseCase phoneVerificationUseCase;
32  31        private final TicketRepository ticketRepository;
33  32        private final LotteryUseCase lotteryUseCase;
    33  +     private final UserUseCase userUseCase;
34  34
35  35        public PaymentController(PaymentUseCase paymentUseCase,
36      -                             PhoneVerificationUseCase phoneVerificationUseCase,
37  36                                 TicketRepository ticketRepository,
38      -                             LotteryUseCase lotteryUseCase) {
    37  +                             LotteryUseCase lotteryUseCase,
    38  +                             UserUseCase userUseCase) {
39  39            this.paymentUseCase = paymentUseCase;
40      -         this.phoneVerificationUseCase = phoneVerificationUseCase;
41  40            this.ticketRepository = ticketRepository;
42  41            this.lotteryUseCase = lotteryUseCase;
    42  +         this.userUseCase = userUseCase;
43  43        }
44  44
45  45        @PostMapping
46  46        public ResponseEntity<CreatePaymentResponse> createPayment(
    47  +             Authentication auth,
47  48                @Valid @RequestBody CreatePaymentRequest request,
```

```
48  49                HttpServletRequest httpRequest) {
49  50
    51  +         UUID userId = UUID.fromString(auth.getPrincipal().toString());
    52  +         User user = userUseCase.getUserById(userId);
    53  +
50  54            List<TicketUseCase.TicketItem> items = request.items().stream()
51  55                    .map(i -> new TicketUseCase.TicketItem(
52  56                            UUID.fromString(i.lotteryId()), i.number(), i.series(),
        i.fractions()))
        @@ -55,35 +59,8 @@ public ResponseEntity<CreatePaymentResponse> createPayment(
55  59            String ipAddress = httpRequest.getRemoteAddr();
56  60            String userAgent = httpRequest.getHeader("User-Agent");
57  61
58      -         String email = request.email() != null ? request.email().trim() : null;
59      -         if (email != null && email.isEmpty()) email = null;
60      -
61      -         String tel = request.tel() != null ? request.tel().trim() : null;
62      -         if (tel != null && tel.isEmpty()) tel = null;
63      -         String cleanTel = tel == null ? null : tel.replaceAll("\\s+", "");
64      -
65      -         String rawTel = request.tel() != null ? request.tel().trim() : null;
66      -         if (rawTel != null && rawTel.isEmpty()) rawTel = null;
67      -
68      -         String normalizedTel = (rawTel == null) ? null : normalize(rawTel);
69      -
70      -         String effectiveEmail = email;
71      -         if (effectiveEmail == null && normalizedTel != null) {
72      -           String digits = normalizedTel.replaceAll("[^0-9]", "");
73      -           effectiveEmail = "tel_" + digits + "@noemail.local";
74      -         }
75      -
76      -         PhoneNumber phoneNumber = (normalizedTel == null) ? null : new
        PhoneNumber(normalizedTel);
77      -
78      -         if (phoneNumber != null) {
79      -           boolean verified = phoneVerificationUseCase.isVerified(phoneNumber,
        request.telVerificationToken());
80      -           if (!verified) {
81      -             throw new DomainException("El telefono no esta verificado. Solicita y
        valida el codigo SMS antes de pagar.");
82      -           }
83      -         }
```

```
84        -
85    62              PaymentOrder order = paymentUseCase.createPayment(
86        -                   new Email(effectiveEmail), phoneNumber, items,
          request.returnUrl(), ipAddress, userAgent);
      63  +                   user.getEmail(), null, items, request.returnUrl(), ipAddress,
          userAgent);
87    64
88    65              return ResponseEntity.ok(new CreatePaymentResponse(
89    66                      order.getReference(),
          @@ -120,18 +97,4 @@ public ResponseEntity<PaymentStatusResponse>
          getPaymentStatus(@PathVariable Stri
120   97                      ticketResponses
121   98              ));
122   99          }
123       -     private String normalize(String tel) {
124       -     String trimmed = tel == null ? "" : tel.trim();
125       -
126       -     if (trimmed.matches("^3\\d{9}$")) return "+57" + trimmed;
127       -     if (trimmed.matches("^\\+\\d{10,15}$")) return trimmed;
128       -
129       -     String digits = trimmed.replaceAll("[^0-9]", "");
130       -     if (digits.matches("^3\\d{9}$")) return "+57" + digits;
131       -     if (digits.matches("^57\\d{10}$")) return "+" + digits;
132       -     if (digits.matches("^\\d{10,15}$")) return "+" + digits;
133       -
134       -     throw new IllegalArgumentException("Tel invalido");
135       - }
136       -
137   100     }
```

…tructure/adapter/in/web/dto/LoginRequest.java

```
      @@ -0,0 +1,8 @@
1     + package co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto;
2     +
3     + import jakarta.validation.constraints.NotBlank;
4     +
5     + public record LoginRequest(
6     +         @NotBlank @jakarta.validation.constraints.Email String email,
7     +         @NotBlank String password
8     + ) {}
```

…cture/adapter/in/web/dto/RegisterRequest.java

```
@@ -5,7 +5,7 @@
5  5
6  6   public record RegisterRequest(
7  7           @NotBlank @jakarta.validation.constraints.Email String email,
8     -         @NotBlank @Size(min = 6, max = 6) String otpCode,
   8  +         @NotBlank @Size(min = 6) String password,
9  9           @NotBlank @Size(max = 100) String firstName,
10 10          @NotBlank @Size(max = 100) String lastName
11 11  ) {}
```

…rsistence/adapter/UserPersistenceAdapter.java

```
@@ -40,6 +40,7 @@ private User toDomain(UserEntity entity) {
40 40          User user = new User();
41 41          user.setId(entity.getId());
42 42          user.setEmail(new Email(entity.getEmail()));
   43 +        user.setPasswordHash(entity.getPasswordHash());
43 44          user.setName(entity.getName());
44 45          user.setFirstName(entity.getFirstName());
45 46          user.setLastName(entity.getLastName());
@@ -58,6 +59,7 @@ private UserEntity toEntity(User user) {
58 59          UserEntity entity = new UserEntity();
59 60          entity.setId(user.getId());
60 61          entity.setEmail(user.getEmail().value());
   62 +        entity.setPasswordHash(user.getPasswordHash());
61 63          entity.setName(user.getName());
62 64          entity.setFirstName(user.getFirstName());
63 65          entity.setLastName(user.getLastName());
```

…dapter/out/persistence/entity/UserEntity.java

```
@@ -16,6 +16,9 @@ public class UserEntity {
16 16      @Column(nullable = false, unique = true, length = 255)
17 17      private String email;
18 18
   19 +    @Column(name = "password_hash", length = 255)
   20 +    private String passwordHash;
   21 +
19 22      @Column(length = 150)
```

```
20  23          private String name;
21  24
                @@ -62,6 +65,8 @@ void onUpdate() {
62  65          public void setId(UUID id) { this.id = id; }
63  66          public String getEmail() { return email; }
64  67          public void setEmail(String email) { this.email = email; }
    68  +       public String getPasswordHash() { return passwordHash; }
    69  +       public void setPasswordHash(String passwordHash) { this.passwordHash =
                passwordHash; }
65  70          public String getName() { return name; }
66  71          public void setName(String name) { this.name = name; }
67  72          public String getFirstName() { return firstName; }
```

…end/infrastructure/config/SecurityConfig.java

```
                @@ -7,6 +7,8 @@
7   7       import org.springframework.security.config.annotation.web.builders.HttpSecurity;
8   8       import
                org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
9   9       import org.springframework.security.config.http.SessionCreationPolicy;
    10  +   import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
    11  +   import org.springframework.security.crypto.password.PasswordEncoder;
10  12      import org.springframework.security.web.SecurityFilterChain;
11  13      import
                org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilte
                r;
12  14
                @@ -20,6 +22,11 @@ public SecurityConfig(JwtAuthenticationFilter jwtFilter) {
20  22              this.jwtFilter = jwtFilter;
21  23          }
22  24
    25  +       @Bean
    26  +       public PasswordEncoder passwordEncoder() {
    27  +           return new BCryptPasswordEncoder();
    28  +       }
    29  +
23  30          @Bean
24  31          public SecurityFilterChain securityFilterChain(HttpSecurity http) throws
                Exception {
25  32              return http
```

```
@@ -29,7 +36,7 @@ public SecurityFilterChain securityFilterChain(HttpSecurity
http) throws Excepti
29  36                          .requestMatchers("/api/auth/**").permitAll()
30  37                          .requestMatchers(HttpMethod.GET,
"/api/lotteries/**").permitAll()
31  38                          .requestMatchers(HttpMethod.POST,
"/api/tickets/checkout").permitAll()
32      -                       .requestMatchers(HttpMethod.POST,
"/api/payments").permitAll()
    39  +                       .requestMatchers(HttpMethod.POST,
"/api/payments").authenticated()
33  40                          .requestMatchers(HttpMethod.GET,
"/api/payments/*/status").permitAll()
34  41                          .requestMatchers("/swagger-ui/**", "/swagger-ui.html",
"/v3/api-docs/**").permitAll()
35  42                          .requestMatchers("/api/otp/**").permitAll()
```

…urces/db/migration/V10__add_password_hash.sql

```
...  @@ -0,0 +1 @@
1   + ALTER TABLE users ADD COLUMN password_hash VARCHAR(255);
```

frontend/src/app/core/auth/auth.service.ts

```
@@ -24,23 +24,19 @@ export class AuthService {
24  24      return this.accessToken();
25  25    }
26  26
27      -   requestOtp(email: string): Observable<void> {
28      -     return this.http.post<void>(`${environment.apiUrl}/auth/otp/request`, { email
});
29      -   }
30      -
31      -   verifyOtp(email: string, otpCode: string): Observable<AuthResponse> {
32      -     return this.http.post<AuthResponse>(`${environment.apiUrl}/auth/otp/verify`, {
    27  +   login(email: string, password: string): Observable<AuthResponse> {
    28  +     return this.http.post<AuthResponse>(`${environment.apiUrl}/auth/login`, {
33  29          email,
34      -       otpCode,
    30  +       password,
35  31        }).pipe(
36  32          tap((res) => this.setTokens(res)),
```

```
37  33        );
38  34      }
39  35
40      -   register(email: string, otpCode: string, firstName: string, lastName: string): Observable<AuthResponse> {
    36  +   register(email: string, password: string, firstName: string, lastName: string): Observable<AuthResponse> {
41  37        return this.http.post<AuthResponse>(`${environment.apiUrl}/auth/register`, {
42  38          email,
43      -       otpCode,
    39  +       password,
44  40          firstName,
45  41          lastName,
46  42        }).pipe(
```

…tend/src/app/features/auth/login.component.ts

```
        @@ -2,16 +2,13 @@ import { Component, signal } from '@angular/core';
2   2     import { Router, RouterLink } from '@angular/router';
3   3     import { FormsModule } from '@angular/forms';
4   4     import { AuthService } from '@core/auth/auth.service';
5       - import { OtpInputComponent } from '@shared/components/otp-input/otp-input.component';
6   5     import { SpinnerComponent } from '@shared/components/spinner/spinner.component';
7   6     import { ErrorMessageComponent } from '@shared/components/error-message/error-message.component';
8   7
9       - type Step = 'email' | 'otp';
10      -
11  8     @Component({
12  9       selector: 'app-login',
13  10      standalone: true,
14      -   imports: [FormsModule, RouterLink, OtpInputComponent, SpinnerComponent, ErrorMessageComponent],
    11  +   imports: [FormsModule, RouterLink, SpinnerComponent, ErrorMessageComponent],
15  12      template: `
16  13        @if (loading()) {
17  14          <app-spinner />
        @@ -20,58 +17,48 @@ type Step = 'email' | 'otp';
20  17            <app-error-message [message]="error()!" />
21  18          }
```

```
22  19
23      -       @if (step() === 'email') {
24      -         <div class="space-y-4">
25      -           <h2 class="text-xl font-bold text-center">Ingresa tu correo</h2>
26      -           <p class="text-sm text-gray-500 text-center">Te enviaremos un codigo de
            verificacion</p>
27      -           <div class="relative">
28      -             <input
29      -               type="email"
30      -               [(ngModel)]="email"
31      -               placeholder="tu@correo.com"
32      -               class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl text-lg
33      -                      focus:border-loteria-red focus:outline-none transition-
            colors"
34      -               (keydown.enter)="requestOtp()"
35      -             />
36      -           </div>
37      -           <button
38      -             (click)="requestOtp()"
39      -             [disabled]="!email().includes('@')"
40      -             class="w-full bg-loteria-red text-white py-3 rounded-xl font-semibold
41      -                    hover:bg-loteria-red-dark transition-colors disabled:opacity-50
            disabled:cursor-not-allowed"
42      -           >
43      -             Enviar codigo
44      -           </button>
45      -           <p class="text-center text-sm text-gray-500">
46      -             No tienes cuenta?
47      -             <a routerLink="/auth/register" class="text-loteria-red font-semibold
            hover:underline">
48      -               Registrate
49      -             </a>
50      -           </p>
51      -         </div>
52      -       }
        20  +       <div class="space-y-4">
        21  +         <h2 class="text-xl font-bold text-center">Iniciar sesion</h2>
        22  +         <p class="text-sm text-gray-500 text-center">Ingresa tus credenciales</p>
53  23
54      -       @if (step() === 'otp') {
55      -         <div class="space-y-4">
56      -           <h2 class="text-xl font-bold text-center">Codigo de verificacion</h2>
```

```
57        -           <p class="text-sm text-gray-500 text-center">
58        -             Ingresa el codigo enviado a {{ email() }}
59        -           </p>
60        -           <app-otp-input (otpComplete)="verifyOtp($event)" />
61        -           <button
62        -             (click)="step.set('email')"
63        -             class="w-full text-sm text-loteria-red hover:underline text-center"
64        -           >
65        -             Cambiar correo
66        -           </button>
67        -         </div>
68        -       }
      24  +         <input
      25  +           type="email"
      26  +           [(ngModel)]="email"
      27  +           placeholder="Correo electronico"
      28  +           class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
      29  +                  focus:border-loteria-red focus:outline-none transition-colors"
      30  +         />
      31  +         <input
      32  +           type="password"
      33  +           [(ngModel)]="password"
      34  +           placeholder="Contraseña"
      35  +           class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
      36  +                  focus:border-loteria-red focus:outline-none transition-colors"
      37  +           (keydown.enter)="doLogin()"
      38  +         />
      39  +
      40  +         <button
      41  +           (click)="doLogin()"
      42  +           [disabled]="!email().includes('@') || !password()"
      43  +           class="w-full bg-loteria-red text-white py-3 rounded-xl font-semibold
      44  +                  hover:bg-loteria-red-dark transition-colors disabled:opacity-50
          disabled:cursor-not-allowed"
      45  +         >
      46  +           Iniciar sesion
      47  +         </button>
      48  +
      49  +         <p class="text-center text-sm text-gray-500">
      50  +           No tienes cuenta?
      51  +           <a routerLink="/auth/register" class="text-loteria-red font-semibold
          hover:underline">
```

```
      52  +             Registrate
      53  +           </a>
      54  +         </p>
      55  +       </div>
69    56        }
70    57      `,
71    58    })
72    59    export class LoginComponent {
73        -   readonly step = signal<Step>('email');
74    60      readonly email = signal('');
      61  +   readonly password = signal('');
75    62      readonly loading = signal(false);
76    63      readonly error = signal<string | null>(null);
77    64
          @@ -80,27 +67,13 @@ export class LoginComponent {
80    67        private readonly router: Router,
81    68      ) {}
82    69
83        -   requestOtp(): void {
84        -     this.loading.set(true);
85        -     this.error.set(null);
86        -
87        -     this.auth.requestOtp(this.email()).subscribe({
88        -       next: () => {
89        -         this.loading.set(false);
90        -         this.step.set('otp');
91        -       },
92        -       error: (err) => {
93        -         this.loading.set(false);
94        -         this.error.set(err.error?.message ?? 'Error al enviar el codigo');
95        -       },
96        -     });
97        -   }
      70  +   doLogin(): void {
      71  +     if (!this.email().includes('@') || !this.password()) return;
98    72
99        -   verifyOtp(code: string): void {
100   73        this.loading.set(true);
101   74        this.error.set(null);
102   75
103       -     this.auth.verifyOtp(this.email(), code).subscribe({
      76  +     this.auth.login(this.email(), this.password()).subscribe({
```

```
104  77           next: () => {
105  78             this.auth.loadProfile().subscribe({
106  79               next: () => {
     @@ -115,7 +88,7 @@ export class LoginComponent {
115  88           },
116  89           error: (err) => {
117  90             this.loading.set(false);
118      -          this.error.set(err.error?.message ?? 'Codigo invalido');
     91 +          this.error.set(err.error?.message ?? 'Correo o contraseña incorrectos');
119  92           },
120  93         });
121  94       }
     ...
```

…d/src/app/features/auth/register.component.ts

```
     @@ -2,16 +2,13 @@ import { Component, signal } from '@angular/core';
2    2   import { Router, RouterLink } from '@angular/router';
3    3   import { FormsModule } from '@angular/forms';
4    4   import { AuthService } from '@core/auth/auth.service';
5        - import { OtpInputComponent } from '@shared/components/otp-input/otp-input.component';
6    5   import { SpinnerComponent } from '@shared/components/spinner/spinner.component';
7    6   import { ErrorMessageComponent } from '@shared/components/error-message/error-message.component';
8    7
9        - type Step = 'form' | 'otp';
10       -
11   8   @Component({
12   9     selector: 'app-register',
13   10    standalone: true,
14       -   imports: [FormsModule, RouterLink, OtpInputComponent, SpinnerComponent, ErrorMessageComponent],
     11 +   imports: [FormsModule, RouterLink, SpinnerComponent, ErrorMessageComponent],
15   12    template: `
16   13      @if (loading()) {
17   14        <app-spinner />
     @@ -20,87 +17,89 @@ type Step = 'form' | 'otp';
20   17          <app-error-message [message]="error()!" />
21   18        }
22   19
23       -       @if (step() === 'form') {
```

```
24        -         <div class="space-y-4">
25        -           <h2 class="text-xl font-bold text-center">Crear cuenta</h2>
26        -           <p class="text-sm text-gray-500 text-center">Completa tus datos para
            registrarte</p>
      20  +       <div class="space-y-4">
      21  +         <h2 class="text-xl font-bold text-center">Crear cuenta</h2>
      22  +         <p class="text-sm text-gray-500 text-center">Completa tus datos para
            registrarte</p>
27    23
28        -           <input
29        -             type="text"
30        -             [(ngModel)]="firstName"
31        -             placeholder="Nombres"
32        -             class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
33        -                    focus:border-loteria-red focus:outline-none transition-colors"
34        -           />
35        -           <input
36        -             type="text"
37        -             [(ngModel)]="lastName"
38        -             placeholder="Apellidos"
39        -             class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
40        -                    focus:border-loteria-red focus:outline-none transition-colors"
41        -           />
42        -           <input
43        -             type="email"
44        -             [(ngModel)]="email"
45        -             placeholder="Correo electronico"
46        -             class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
47        -                    focus:border-loteria-red focus:outline-none transition-colors"
48        -           />
49        -           <input
50        -             type="email"
51        -             [(ngModel)]="emailConfirm"
52        -             placeholder="Confirmar correo electronico"
53        -             class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
54        -                    focus:border-loteria-red focus:outline-none transition-colors"
55        -             (keydown.enter)="requestOtp()"
56        -           />
      24  +         <input
      25  +           type="text"
      26  +           [(ngModel)]="firstName"
      27  +           placeholder="Nombres"
```

```
28  +           class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
29  +                  focus:border-loteria-red focus:outline-none transition-colors"
30  +         />
31  +         <input
32  +           type="text"
33  +           [(ngModel)]="lastName"
34  +           placeholder="Apellidos"
35  +           class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
36  +                  focus:border-loteria-red focus:outline-none transition-colors"
37  +         />
38  +         <input
39  +           type="email"
40  +           [(ngModel)]="email"
41  +           placeholder="Correo electronico"
42  +           class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
43  +                  focus:border-loteria-red focus:outline-none transition-colors"
44  +         />
45  +         <input
46  +           type="email"
47  +           [(ngModel)]="emailConfirm"
48  +           placeholder="Confirmar correo electronico"
49  +           class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
50  +                  focus:border-loteria-red focus:outline-none transition-colors"
51  +         />
52
58  -           @if (email() && emailConfirm() && email() !== emailConfirm()) {
59  -             <p class="text-red-500 text-xs">Los correos no coinciden</p>
60  -           }
53  +         @if (email() && emailConfirm() && email() !== emailConfirm()) {
54  +           <p class="text-red-500 text-xs">Los correos no coinciden</p>
55  +         }
56
62  -           <button
63  -             (click)="requestOtp()"
64  -             [disabled]="!isFormValid()"
65  -             class="w-full bg-loteria-red text-white py-3 rounded-xl font-semibold
66  -                    hover:bg-loteria-red-dark transition-colors disabled:opacity-50
      disabled:cursor-not-allowed"
67  -           >
68  -             Enviar codigo de verificacion
69  -           </button>
57  +         <input
```

```
58  +           type="password"
59  +           [(ngModel)]="password"
60  +           placeholder="Contraseña (minimo 6 caracteres)"
61  +           class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
62  +                  focus:border-loteria-red focus:outline-none transition-colors"
63  +         />
64  +         <input
65  +           type="password"
66  +           [(ngModel)]="passwordConfirm"
67  +           placeholder="Confirmar contraseña"
68  +           class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
69  +                  focus:border-loteria-red focus:outline-none transition-colors"
70  +           (keydown.enter)="doRegister()"
71  +         />
70  72
71      -           <p class="text-center text-sm text-gray-500">
72      -             Ya tienes cuenta?
73      -             <a routerLink="/auth/login" class="text-loteria-red font-semibold
        hover:underline">
74      -               Iniciar sesion
75      -             </a>
76      -           </p>
77      -         </div>
78      -       }
73  +         @if (password() && passwordConfirm() && password() !== passwordConfirm())
        {
74  +           <p class="text-red-500 text-xs">Las contraseñas no coinciden</p>
75  +         }
79  76
80      -       @if (step() === 'otp') {
81      -         <div class="space-y-4">
82      -           <h2 class="text-xl font-bold text-center">Codigo de verificacion</h2>
83      -           <p class="text-sm text-gray-500 text-center">
84      -             Ingresa el codigo enviado a {{ email() }}
85      -           </p>
86      -           <app-otp-input (otpComplete)="verifyAndRegister($event)" />
87      -           <button
88      -             (click)="step.set('form')"
89      -             class="w-full text-sm text-loteria-red hover:underline text-center"
90      -           >
91      -             Volver al formulario
92      -           </button>
```

```
93        -         </div>
94        -       }
      77  +         <button
      78  +           (click)="doRegister()"
      79  +           [disabled]="!isFormValid()"
      80  +           class="w-full bg-loteria-red text-white py-3 rounded-xl font-semibold
      81  +                  hover:bg-loteria-red-dark transition-colors disabled:opacity-50
          disabled:cursor-not-allowed"
      82  +         >
      83  +           Registrarse
      84  +         </button>
      85  +
      86  +         <p class="text-center text-sm text-gray-500">
      87  +           Ya tienes cuenta?
      88  +           <a routerLink="/auth/login" class="text-loteria-red font-semibold
          hover:underline">
      89  +             Iniciar sesion
      90  +           </a>
      91  +         </p>
      92  +       </div>
95    93        }
96    94      `,
97    95    })
98    96    export class RegisterComponent {
99        -   readonly step = signal<Step>('form');
100   97      readonly firstName = signal('');
101   98      readonly lastName = signal('');
102   99      readonly email = signal('');
103  100      readonly emailConfirm = signal('');
     101  +   readonly password = signal('');
     102  +   readonly passwordConfirm = signal('');
104  103      readonly loading = signal(false);
105  104      readonly error = signal<string | null>(null);
106  105
         @@ -114,33 +113,19 @@ export class RegisterComponent {
114  113          this.firstName().trim().length > 0 &&
115  114          this.lastName().trim().length > 0 &&
116  115          this.email().includes('@') &&
117       -       this.email() === this.emailConfirm()
     116  +       this.email() === this.emailConfirm() &&
     117  +       this.password().length >= 6 &&
     118  +       this.password() === this.passwordConfirm()
```

```
118  119        );
119  120      }
120  121
121       -  requestOtp(): void {
     122  +  doRegister(): void {
122  123        if (!this.isFormValid()) return;
123  124
124  125        this.loading.set(true);
125  126        this.error.set(null);
126  127
127       -    this.auth.requestOtp(this.email()).subscribe({
128       -      next: () => {
129       -        this.loading.set(false);
130       -        this.step.set('otp');
131       -      },
132       -      error: (err) => {
133       -        this.loading.set(false);
134       -        this.error.set(err.error?.message ?? 'Error al enviar el codigo');
135       -      },
136       -    });
137       -  }
138       -
139       -  verifyAndRegister(code: string): void {
140       -    this.loading.set(true);
141       -    this.error.set(null);
142       -
143       -    this.auth.register(this.email(), code, this.firstName().trim(),
           this.lastName().trim()).subscribe({
     128  +    this.auth.register(this.email(), this.password(), this.firstName().trim(),
           this.lastName().trim()).subscribe({
144  129          next: () => {
145  130            this.auth.loadProfile().subscribe({
146  131              next: () => {
```

…c/app/features/purchase/purchase.component.ts

```
     ...  @@ -1,8 +1,9 @@
1         - import { Component, inject, signal, computed } from '@angular/core';
     1    + import { Component, inject, signal, computed, OnInit } from '@angular/core';
2    2      import { HttpClient } from '@angular/common/http';
3         - import { FormsModule } from '@angular/forms';
```

```
      3  + import { Router } from '@angular/router';
4     4    import { rxResource } from '@angular/core/rxjs-interop';
5     5    import { environment } from '@env';
      6  + import { AuthService } from '@core/auth/auth.service';
6     7    import { Lottery } from '../games/game.models';
7     8    import { SpinnerComponent } from '@shared/components/spinner/spinner.component';
8     9    import { ErrorMessageComponent } from '@shared/components/error-message/error-
           message.component';
           @@ -22,7 +23,7 @@ interface CartItem {
22    23    @Component({
23    24      selector: 'app-purchase',
24    25      standalone: true,
25          -   imports: [FormsModule, SpinnerComponent, ErrorMessageComponent, CurrencyPipe],
      26  +   imports: [SpinnerComponent, ErrorMessageComponent, CurrencyPipe],
26    27      template: `
27    28        @if (lotteriesResource.isLoading()) {
28    29          <div class="text-center py-12">
           @@ -310,77 +311,21 @@ interface CartItem {
310   311                         <span class="text-loteria-red">{{ cartTotal() | currency:
           'COP' : 'symbol-narrow' : '1.0-0' }}</span>
311   312                       </div>
312   313
313         -                   <!-- Email + Pay with PlaceToPay -->
314         -                   @if (!showEmailForm()) {
315         -                     <button (click)="showEmailForm.set(true)"
316         -                       class="w-full bg-loteria-red text-white py-3 rounded-xl
           font-semibold
317         -                              hover:bg-loteria-red-dark transition-colors">
      314  +                   <!-- Pay button -->
      315  +                   <button (click)="pay()"
      316  +                     [disabled]="purchasing()"
      317  +                     class="w-full bg-loteria-red text-white py-3 rounded-xl
           font-semibold
      318  +                            hover:bg-loteria-red-dark transition-colors
           disabled:opacity-50
      319  +                            flex items-center justify-center gap-2">
      320  +                     @if (purchasing()) {
      321  +                       <span class="material-icons text-sm animate-
           spin">refresh</span>
      322  +                       Redirigiendo a pasarela...
      323  +                     } @else {
```

```
      324  +                         <span class="material-icons text-sm">lock</span>
318   325                           Pagar
319         -                       </button>
320         -                     } @else {
321         -                       <div class="space-y-3">
322         -                         <div>
323         -                           <label class="block text-sm font-medium text-gray-600
              mb-1">Tu correo electronico</label>
324         -                           <input type="email" [(ngModel)]="email"
325         -                             placeholder="correo&#64;ejemplo.com"
326         -                             class="w-full px-4 py-2.5 border-2 border-gray-200
              rounded-xl
327         -                                    focus:border-loteria-red focus:outline-none
              text-sm" />
328         -                         </div>
329         -                         <div>
330         -                           <label class="block text-sm font-medium text-gray-600
              mb-1">O tu numero de celular</label>
331         -                           <input type="tel" [(ngModel)]="tel"
332         -                             placeholder="3022974899"
333         -                             class="w-full px-4 py-2.5 border-2 border-gray-200
              rounded-xl
334         -                                    focus:border-loteria-red focus:outline-none
              text-sm" />
335         -                           @if (tel().trim()) {
336         -                             <div class="flex gap-2 items-center mt-2">
337         -                               <button type="button" (click)="requestSmsOtp()"
338         -                                 [disabled]="otpSending() || phoneVerified()"
339         -                                 class="flex-1 bg-gray-100 text-gray-800 py-2.5
              rounded-xl font-semibold
340         -                                        hover:bg-gray-200 transition-colors
              disabled:opacity-50 text-sm">
341         -                                 @if (phoneVerified()) {
342         -                                   Celular verificado
343         -                                 } @else if (otpSending()) {
344         -                                   Enviando...
345         -                                 } @else {
346         -                                   Enviar codigo SMS
347         -                                 }
348         -                               </button>
349         -                             </div>
350         -
```

```
351 -                             @if (!phoneVerified()) {
352 -                               <div class="flex gap-2 mt-2">
353 -                                 <input type="text" inputmode="numeric"
    maxlength="6" [(ngModel)]="otpCode"
354 -                                   placeholder="Codigo (6 digitos)"
355 -                                   class="flex-1 px-4 py-2.5 border-2 border-gray-
    200 rounded-xl
356 -                                          focus:border-loteria-red focus:outline-
    none text-sm" />
357 -                                 <button type="button" (click)="verifySmsOtp()"
358 -                                   [disabled]="otpVerifying() ||
    otpCode().trim().length !== 6"
359 -                                   class="bg-loteria-red text-white px-4 py-2.5
    rounded-xl font-semibold
360 -                                          hover:bg-loteria-red-dark transition-
    colors disabled:opacity-50 text-sm">
361 -                                   @if (otpVerifying()) { Verificando... } @else {
    Verificar }
362 -                                 </button>
363 -                               </div>
364 -                               <p class="text-xs text-gray-400 mt-1">Si usas
    celular para pagar, debes verificarlo por SMS.</p>
365 -                             }
366 -                           }
367 -                         </div>
368 -                         <button (click)="pay()"
369 -                           [disabled]="purchasing() || (!email() && (!tel() ||
    !phoneVerified()))"
370 -                           class="w-full bg-loteria-red text-white py-3 rounded-xl
    font-semibold
371 -                                  hover:bg-loteria-red-dark transition-colors
    disabled:opacity-50
372 -                                  flex items-center justify-center gap-2">
373 -                           @if (purchasing()) {
374 -                             <span class="material-icons text-sm animate-
    spin">refresh</span>
375 -                             Redirigiendo a pasarela...
376 -                           } @else {
377 -                             <span class="material-icons text-sm">lock</span>
378 -                             Ir a pagar
379 -                           }
380 -                         </button>
```

```
381        -                        <p class="text-xs text-gray-400 text-center">Pago seguro
           con PlaceToPay</p>
382        -                      </div>
383        -                    }
      326  +                    }
      327  +                  </button>
      328  +                  <p class="text-xs text-gray-400 text-center">Pago seguro con
           PlaceToPay</p>
384   329                   }
385   330                 </div>
386   331               </div>
          @@ -399,8 +344,10 @@ interface CartItem {
399   344       }
400   345     `,
401   346   })
402        - export class PurchaseComponent {
      347  + export class PurchaseComponent implements OnInit {
403   348     private readonly http = inject(HttpClient);
      349  +   private readonly router = inject(Router);
      350  +   readonly auth = inject(AuthService);
404   351
405   352     readonly selectedLottery = signal<Lottery | null>(null);
406   353
          @@ -415,19 +362,23 @@ export class PurchaseComponent {
415   362     readonly selectedFractions = signal(0);
416   363
417   364     readonly cart = signal<CartItem[]>([]);
418        -   readonly showEmailForm = signal(false);
419        -   readonly email = signal('');
420        -   readonly tel = signal('');
421        -   readonly telVerificationToken = signal<string | null>(null);
422        -   readonly otpCode = signal('');
423        -   readonly otpSending = signal(false);
424        -   readonly otpVerifying = signal(false);
425        -   readonly phoneVerified = signal(false);
426   365     readonly purchasing = signal(false);
427   366     readonly error = signal<string | null>(null);
428        -   readonly otpSent = signal(false);
429        -   readonly otpError = signal<string | null>(null);
430        -   readonly otpInfo = signal<string | null>(null);
      367  +
      368  +   private static readonly CART_KEY = 'loteria_cart';
```

```
      369  +
      370  +   ngOnInit(): void {
      371  +     const saved = localStorage.getItem(PurchaseComponent.CART_KEY);
      372  +     if (saved) {
      373  +       try {
      374  +         const items = JSON.parse(saved) as CartItem[];
      375  +         if (items.length > 0) {
      376  +           this.cart.set(items);
      377  +           localStorage.removeItem(PurchaseComponent.CART_KEY);
      378  +         }
      379  +       } catch { /* ignore */ }
      380  +     }
      381  +   }
431   382
432   383       readonly currentNumber = computed(() => this.digits().join(''));
433   384       readonly currentSeries = computed(() =>
          @@ -526,22 +477,16 @@ export class PurchaseComponent {
526   477         const current = [...this.cart()];
527   478         current.splice(index, 1);
528   479         this.cart.set(current);
529         -     if (current.length === 0) {
530         -       this.showEmailForm.set(false);
531         -     }
532   480       }
533   481
534   482       pay(): void {
535         -     const hasEmail = !!this.email().trim();
536         -     const hasTel = !!this.tel().trim();
537         -     if (!hasEmail && !hasTel) {
538         -       this.error.set('Debes ingresar correo o celular (al menos uno).');
539         -       return;
540         -     }
541         -     if (hasTel && !this.phoneVerified()) {
542         -       this.error.set('Debes verificar el celular por SMS antes de continuar.');
      483  +     if (!this.auth.isAuthenticated()) {
      484  +       // Save cart and redirect to register
      485  +       localStorage.setItem(PurchaseComponent.CART_KEY,
          JSON.stringify(this.cart()));
      486  +       this.router.navigate(['/auth/register']);
543   487         return;
544   488       }
```

```
      489  +
545   490        this.purchasing.set(true);
546   491        this.error.set(null);
547   492
           @@ -555,9 +500,6 @@ export class PurchaseComponent {
555   500        const returnUrl = window.location.origin + '/payment/result';
556   501
557   502        this.http.post<{ processUrl: string }>(`${environment.apiUrl}/payments`, {
558         -        email: this.email().trim() || null,
559         -        tel: this.tel().trim() || null,
560         -        telVerificationToken: this.telVerificationToken(),
561   503          items,
562   504          returnUrl,
563   505        }).subscribe({
           @@ -571,81 +513,6 @@ export class PurchaseComponent {
571   513        });
572   514      }
573   515
574         -    private normalizeTel(raw: string): string {
575         -      const t = (raw ?? '').trim();
576         -      if (!t) return '';
577         -      return t.replace(/\s+/g, '');
578         -    }
579         -
580         -    requestSmsOtp(): void {
581         -      const tel = this.normalizeTel(this.tel());
582         -      this.otpError.set(null);
583         -      this.otpInfo.set(null);
584         -
585         -      if (!tel) {
586         -        this.otpError.set('Ingresa un numero de celular primero.');
587         -        return;
588         -      }
589         -
590         -      this.otpSending.set(true);
591         -
592         -      this.http.post<void>('/api/otp/sms/request', { tel }).subscribe({
593         -        next: () => {
594         -          this.otpSent.set(true);
595         -          this.otpInfo.set('Codigo enviado por SMS.');
596         -          this.phoneVerified.set(false);
597         -          this.telVerificationToken.set(null);
```

```
598 -       },
599 -       error: (err) => {
600 -         this.otpError.set(err?.error ?? 'No se pudo enviar el codigo.');
601 -       },
602 -       complete: () => this.otpSending.set(false),
603 -     });
604 -   }
605 -
606 -   verifySmsOtp(): void {
607 -     const tel = this.normalizeTel(this.tel());
608 -     const code = (this.otpCode() ?? '').trim();
609 -
610 -     this.otpError.set(null);
611 -     this.otpInfo.set(null);
612 -
613 -     if (!tel) {
614 -       this.otpError.set('Debes ingresar el numero de celular.');
615 -       return;
616 -     }
617 -
618 -     if (!/^\d{6}$/.test(code)) {
619 -       this.otpError.set('El codigo debe tener 6 digitos.');
620 -       return;
621 -     }
622 -
623 -     this.otpVerifying.set(true);
624 -
625 -     this.http.post<{ verificationToken: string }>(
626 -       '/api/otp/sms/verify',
627 -       { tel, code }
628 -     ).subscribe({
629 -       next: (response) => {
630 -         this.telVerificationToken.set(response.verificationToken);
631 -         this.phoneVerified.set(true);
632 -         this.otpInfo.set('Celular verificado correctamente.');
633 -       },
634 -       error: (err) => {
635 -         this.phoneVerified.set(false);
636 -         this.telVerificationToken.set(null);
637 -         this.otpError.set(
638 -           err?.error?.message ??
639 -           err?.error ??
```

```
640        -           'Codigo invalido o expirado.'
641        -         );
642        -       },
643        -       complete: () => {
644        -         this.otpVerifying.set(false);
645        -       }
646        -     });
647        -   }
648        -
649  516       formatDate(dateStr: string): string {
650  517         const months = ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio',
651  518           'Julio', 'Agosto', 'Septiembre', 'Octubre', 'Noviembre', 'Diciembre'];
```

Comments 0

Lock conversation

Comment

Subscribe You're not receiving notifications from this thread.