tactusoft / **loteriabogota**

Code Issues Pull requests Actions Projects Security and quality Insi

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 f2d23d1

claude committed last month Verified

```
feat: compra anonima con email (confirmacion) o celular (OTP)

- POST /api/payments ahora es publico (no requiere JWT)
- PaymentController acepta 3 flujos: autenticado, email+confirmacion, celular+OTP
- PaymentService resuelve/crea usuario por email o por telefono
- User.createBasicFromPhone() genera email sintetico para constraint DB
- UserRepository.findByPhone() en domain port, JPA repo y adapter
- CreatePaymentRequest: campos email, emailConfirmation, phone, phoneVerificationToken
- Flyway V11: email nullable en payment_orders
- Frontend purchase: paso de identificacion con toggle email/celular
- Frontend payment-result: muestra email o celular segun corresponda

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

master

1 parent fe7a7cc commit f2d23d1

13 files changed +530 -158 lines changed

Top

Filter files...

- backend/src/main
  - java/co/gov/loteriabogota/loteriabackend
    - domain
      - model
        - User.java
      - port/out
        - UserRepository.java

- service
  - PaymentService.java
- infrastructure
  - adapter
    - in/web
      - controller
        - PaymentController.java
      - dto
        - CreatePaymentRequest.java
        - PaymentStatusResponse.java
    - out/persistence
      - adapter
        - UserPersistenceAdapter.java
      - entity
        - PaymentOrderEntity.java
      - repository
        - JpaUserRepository.java
  - config
    - SecurityConfig.java
- resources/db/migration
  - V11__allow_anonymous_checkout.sql
- frontend/src/app/features
  - payment
    - payment-result.component.ts
  - purchase
    - purchase.component.ts

13 files changed +530 -158 lines changed

Search within code

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

```
@@ -2,6 +2,7 @@
2  2
```

```
3   3     import co.gov.loteriabogota.loteriabackend.domain.vo.DocumentId;
4   4     import co.gov.loteriabogota.loteriabackend.domain.vo.Email;
      5   + import co.gov.loteriabogota.loteriabackend.domain.vo.PhoneNumber;
5   6     import java.time.Instant;
6   7     import java.time.LocalDate;
7   8     import java.util.UUID;
          @@ -35,6 +36,14 @@ public static User createBasic(Email email) {
35  36            return new User(UUID.randomUUID(), email, KycLevel.BASIC);
36  37        }
37  38
      39  +     public static User createBasicFromPhone(PhoneNumber phoneNumber) {
      40  +         // Generate a synthetic email from the phone number to satisfy the email NOT NULL constraint
      41  +         String syntheticEmail = phoneNumber.value().replace("+", "") + "@phone.loteriabogota.gov.co";
      42  +         User user = new User(UUID.randomUUID(), new Email(syntheticEmail), KycLevel.BASIC);
      43  +         user.setPhone(phoneNumber.value());
      44  +         return user;
      45  +     }
      46  +
38  47        public static User createWithPassword(Email email, String passwordHash, String firstName, String lastName) {
39  48            User user = new User(UUID.randomUUID(), email, KycLevel.BASIC);
40  49            user.setPasswordHash(passwordHash);
```

…iabackend/domain/port/out/UserRepository.java

```
          @@ -2,6 +2,7 @@
2   2
3   3     import co.gov.loteriabogota.loteriabackend.domain.model.User;
4   4     import co.gov.loteriabogota.loteriabackend.domain.vo.Email;
      5   + import co.gov.loteriabogota.loteriabackend.domain.vo.PhoneNumber;
5   6     import java.util.Optional;
6   7     import java.util.UUID;
7   8
          @@ -11,5 +12,7 @@ public interface UserRepository {
11  12
12  13        Optional<User> findByEmail(Email email);
13  14
```

```
15  +     Optional<User> findByPhone(PhoneNumber phone);
16  +
14  17        User save(User user);
15  18    }
```

…riabackend/domain/service/PaymentService.java

```
@@ -51,11 +51,26 @@ public PaymentOrder createPayment(Email email, PhoneNumber phoneNumber, List<Tic
51  51            if (items == null || items.isEmpty()) {
52  52                throw new DomainException("Debe seleccionar al menos un billete");
53  53            }
54  +         if (email == null && phoneNumber == null) {
55  +             throw new DomainException("Debe proporcionar un correo electronico o un numero de celular");
56  +         }
54  57
55      -         User user = userRepository.findByEmail(email).orElseGet(() -> {
56      -             User newUser = User.createBasic(email);
57      -             return userRepository.save(newUser);
58      -         });
58  +         // Resolve or create user based on email or phone
59  +         User user;
60  +         if (email != null) {
61  +             user = userRepository.findByEmail(email).orElseGet(() -> {
62  +                 User newUser = User.createBasic(email);
63  +                 if (phoneNumber != null) {
64  +                     newUser.setPhone(phoneNumber.value());
65  +                 }
66  +                 return userRepository.save(newUser);
67  +             });
68  +         } else {
69  +             user = userRepository.findByPhone(phoneNumber).orElseGet(() -> {
70  +                 User newUser = User.createBasicFromPhone(phoneNumber);
71  +                 return userRepository.save(newUser);
72  +             });
73  +         }
59  74
60  75            // Calculate total and validate items
61  76            BigDecimal totalAmount = BigDecimal.ZERO;
@@ -77,7 +92,8 @@ public PaymentOrder createPayment(Email email, PhoneNumber phoneNumber, List<Tic
```

```
77  92
78  93              // Create payment order
79  94              String reference = "LB-" + System.currentTimeMillis();
80      -           PaymentOrder order = new PaymentOrder(user.getId(), email.value(),
        reference, Money.cop(totalAmount));
    95  +           String orderEmail = email != null ? email.value() : null;
    96  +           PaymentOrder order = new PaymentOrder(user.getId(), orderEmail, reference,
        Money.cop(totalAmount));
81  97
82  98              if (phoneNumber != null) {
83  99                  order.setPhone(phoneNumber.value());
        @@ -86,9 +102,12 @@ public PaymentOrder createPayment(Email email, PhoneNumber
        phoneNumber, List<Tic
86  102             String description = "Loteria de Bogota - " + items.size() + "
        billete(s)";
87  103             String fullReturnUrl = returnUrl + "?reference=" + reference;
88  104
    105 +           // PlaceToPay requires an email; for phone-only purchases use the
        synthetic email
    106 +           String gatewayEmail = email != null ? email.value() :
        user.getEmail().value();
    107 +
89  108             PaymentGateway.SessionResult session = paymentGateway.createSession(
90  109                     reference, description, totalAmount, "COP",
91      -               email.value(), phoneNumber != null ? phoneNumber.value() : null,
    110 +               gatewayEmail, phoneNumber != null ? phoneNumber.value() : null,
92  111                 fullReturnUrl, ipAddress, userAgent);
93  112
94  113             order.setPlacetopayRequestId(session.requestId());
```

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

```
    ...     @@ -1,15 +1,18 @@
1   1       package
        co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.controller;
2   2
    3   +   import co.gov.loteriabogota.loteriabackend.domain.exception.DomainException;
3   4       import co.gov.loteriabogota.loteriabackend.domain.model.Lottery;
4   5       import co.gov.loteriabogota.loteriabackend.domain.model.LotteryTicket;
5   6       import co.gov.loteriabogota.loteriabackend.domain.model.PaymentOrder;
6   7       import co.gov.loteriabogota.loteriabackend.domain.model.User;
```

```
7   8     import co.gov.loteriabogota.loteriabackend.domain.port.in.LotteryUseCase;
8   9     import co.gov.loteriabogota.loteriabackend.domain.port.in.PaymentUseCase;
    10  + import
          co.gov.loteriabogota.loteriabackend.domain.port.in.PhoneVerificationUseCase;
9   11    import co.gov.loteriabogota.loteriabackend.domain.port.in.TicketUseCase;
10  12    import co.gov.loteriabogota.loteriabackend.domain.port.in.UserUseCase;
11  13    import co.gov.loteriabogota.loteriabackend.domain.port.out.TicketRepository;
12  14    import co.gov.loteriabogota.loteriabackend.domain.vo.Email;
    15  + import co.gov.loteriabogota.loteriabackend.domain.vo.PhoneNumber;
13  16    import
          co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.CreatePaymen
          tRequest;
14  17    import
          co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.CreatePaymen
          tResponse;
15  18    import
          co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.PaymentStatu
          sResponse;
        @@ -31,15 +34,18 @@ public class PaymentController {
31  34        private final TicketRepository ticketRepository;
32  35        private final LotteryUseCase lotteryUseCase;
33  36        private final UserUseCase userUseCase;
    37  +     private final PhoneVerificationUseCase phoneVerificationUseCase;
34  38
35  39        public PaymentController(PaymentUseCase paymentUseCase,
36  40                                 TicketRepository ticketRepository,
37  41                                 LotteryUseCase lotteryUseCase,
38      -                              UserUseCase userUseCase) {
    42  +                              UserUseCase userUseCase,
    43  +                              PhoneVerificationUseCase phoneVerificationUseCase) {
39  44            this.paymentUseCase = paymentUseCase;
40  45            this.ticketRepository = ticketRepository;
41  46            this.lotteryUseCase = lotteryUseCase;
42  47            this.userUseCase = userUseCase;
    48  +         this.phoneVerificationUseCase = phoneVerificationUseCase;
43  49        }
44  50
45  51        @PostMapping
        @@ -48,8 +54,35 @@ public ResponseEntity<CreatePaymentResponse> createPayment(
48  54                @Valid @RequestBody CreatePaymentRequest request,
49  55                HttpServletRequest httpRequest) {
50  56
```

```
51        -         UUID userId = UUID.fromString(auth.getPrincipal().toString());
52        -         User user = userUseCase.getUserById(userId);
      57  +         Email email = null;
      58  +         PhoneNumber phoneNumber = null;
      59  +
      60  +         if (auth != null && auth.getPrincipal() != null) {
      61  +             // Authenticated user - use their email
      62  +             UUID userId = UUID.fromString(auth.getPrincipal().toString());
      63  +             User user = userUseCase.getUserById(userId);
      64  +             email = user.getEmail();
      65  +         } else if (request.email() != null && !request.email().isBlank()) {
      66  +             // Anonymous with email
      67  +             if (request.emailConfirmation() == null
      68  +                     ||
            !request.email().trim().equalsIgnoreCase(request.emailConfirmation().trim())) {
      69  +                 throw new DomainException("Los correos no coinciden");
      70  +             }
      71  +             email = new Email(request.email().trim());
      72  +         } else if (request.phone() != null && !request.phone().isBlank()) {
      73  +             // Anonymous with phone
      74  +             String normalizedPhone = normalizePhone(request.phone());
      75  +             phoneNumber = new PhoneNumber(normalizedPhone);
      76  +
      77  +             if (request.phoneVerificationToken() == null ||
            request.phoneVerificationToken().isBlank()) {
      78  +                 throw new DomainException("Debe verificar su numero de celular");
      79  +             }
      80  +             if (!phoneVerificationUseCase.isVerified(phoneNumber,
            request.phoneVerificationToken())) {
      81  +                 throw new DomainException("El token de verificacion es invalido o
            expiro");
      82  +             }
      83  +         } else {
      84  +             throw new DomainException("Debe proporcionar un correo electronico o
            un numero de celular");
      85  +         }
53    86
54    87            List<TicketUseCase.TicketItem> items = request.items().stream()
55    88                    .map(i -> new TicketUseCase.TicketItem(
          @@ -60,7 +93,7 @@ public ResponseEntity<CreatePaymentResponse> createPayment(
60    93            String userAgent = httpRequest.getHeader("User-Agent");
61    94
```

```
62  95            PaymentOrder order = paymentUseCase.createPayment(
63      -                 user.getEmail(), null, items, request.returnUrl(), ipAddress,
            userAgent);
        96  +             email, phoneNumber, items, request.returnUrl(), ipAddress,
            userAgent);
64  97
65  98            return ResponseEntity.ok(new CreatePaymentResponse(
66  99                    order.getReference(),
        @@ -91,10 +124,22 @@ public ResponseEntity<PaymentStatusResponse>
        getPaymentStatus(@PathVariable Stri
91  124           return ResponseEntity.ok(new PaymentStatusResponse(
92  125                   order.getReference(),
93  126                   order.getEmail(),
        127 +             order.getPhone(),
94  128                   order.getStatus().name(),
95  129                   order.getTotalAmount().amount(),
96  130                   order.getCreatedAt(),
97  131                   ticketResponses
98  132           ));
99  133       }
        134 +
        135 +     private String normalizePhone(String tel) {
        136 +         String trimmed = tel == null ? "" : tel.trim();
        137 +         if (trimmed.matches("^3\\d{9}$")) return "+57" + trimmed;
        138 +         if (trimmed.matches("^\\+\\d{10,15}$")) return trimmed;
        139 +         String digits = trimmed.replaceAll("[^0-9]", "");
        140 +         if (digits.matches("^3\\d{9}$")) return "+57" + digits;
        141 +         if (digits.matches("^57\\d{10}$")) return "+" + digits;
        142 +         if (digits.matches("^\\d{10,15}$")) return "+" + digits;
        143 +         throw new DomainException("Numero de celular invalido");
        144 +     }
100 145   }
```

…/adapter/in/web/dto/CreatePaymentRequest.java

```
        @@ -8,5 +8,9 @@
8   8
9   9     public record CreatePaymentRequest(
10  10            @NotEmpty @Valid List<CheckoutRequest.CheckoutItem> items,
11      -         @NotBlank String returnUrl
        11  +         @NotBlank String returnUrl,
        12  +         String email,
```

```
13  +         String emailConfirmation,
14  +         String phone,
15  +         String phoneVerificationToken
16      ) {}
```

…adapter/in/web/dto/PaymentStatusResponse.java

```
@@ -7,6 +7,7 @@
7   7     public record PaymentStatusResponse(
8   8             String reference,
9   9             String email,
10  +         String phone,
10  11            String status,
11  12            BigDecimal totalAmount,
12  13            Instant createdAt,
```

…rsistence/adapter/UserPersistenceAdapter.java

```
@@ -4,6 +4,7 @@
4   4     import co.gov.loteriabogota.loteriabackend.domain.port.out.UserRepository;
5   5     import co.gov.loteriabogota.loteriabackend.domain.vo.DocumentId;
6   6     import co.gov.loteriabogota.loteriabackend.domain.vo.Email;
7   +     import co.gov.loteriabogota.loteriabackend.domain.vo.PhoneNumber;
7   8     import
          co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.persistence.entity.U
          serEntity;
8   9     import
          co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.persistence.reposito
          ry.JpaUserRepository;
9   10    import org.springframework.stereotype.Component;
@@ -30,6 +31,11 @@ public Optional<User> findByEmail(Email email) {
30  31            return jpaRepository.findByEmail(email.value()).map(this::toDomain);
31  32        }
32  33
34  +     @Override
35  +     public Optional<User> findByPhone(PhoneNumber phone) {
36  +         return jpaRepository.findByPhone(phone.value()).map(this::toDomain);
37  +     }
38  +
33  39        @Override
34  40        public User save(User user) {
35  41            UserEntity entity = toEntity(user);
```

…ut/persistence/entity/PaymentOrderEntity.java

```
@@ -17,7 +17,7 @@ public class PaymentOrderEntity {
17  17      @Column(name = "user_id", nullable = false)
18  18      private UUID userId;
19  19
20      -   @Column(nullable = false)
    20  +   @Column
21  21      private String email;
22  22
23  23      @Column(name = "phone", length = 30)
```

…persistence/repository/JpaUserRepository.java

```
@@ -9,4 +9,6 @@
9   9   public interface JpaUserRepository extends JpaRepository<UserEntity, UUID> {
10  10
11  11      Optional<UserEntity> findByEmail(String email);
    12  +
    13  +   Optional<UserEntity> findByPhone(String phone);
12  14  }
```

…end/infrastructure/config/SecurityConfig.java

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

…b/migration/V11__allow_anonymous_checkout.sql

```
@@ -0,0 +1,2 @@
+ -- Allow anonymous checkout: make email nullable on payment_orders
+ ALTER TABLE payment_orders ALTER COLUMN email VARCHAR(255) NULL;
```

…/features/payment/payment-result.component.ts

```
@@ -8,7 +8,8 @@ import { CurrencyPipe, DatePipe } from '@angular/common';

 interface PaymentStatus {
   reference: string;
-  email: string;
+  email: string | null;
+  phone: string | null;
   status: 'PENDING' | 'APPROVED' | 'REJECTED';
   totalAmount: number;
   createdAt: string;
@@ -87,10 +88,18 @@ interface PaymentStatus {
               <span class="text-gray-500">Referencia</span>
               <span class="font-mono font-bold">{{ payment.reference }}</span>
             </div>
-            <div class="flex justify-between text-sm">
-              <span class="text-gray-500">Correo</span>
-              <span class="font-semibold">{{ payment.email }}</span>
-            </div>
+            @if (payment.email) {
+              <div class="flex justify-between text-sm">
+                <span class="text-gray-500">Correo</span>
+                <span class="font-semibold">{{ payment.email }}</span>
+              </div>
+            }
+            @if (payment.phone) {
+              <div class="flex justify-between text-sm">
+                <span class="text-gray-500">Celular</span>
+                <span class="font-semibold">{{ payment.phone }}</span>
+              </div>
+            }
             <div class="flex justify-between text-sm">
               <span class="text-gray-500">Total</span>
```

```
96   105                <span class="font-bold text-loteria-red">{{ payment.totalAmount |
          currency: 'COP' : 'symbol-narrow' : '1.0-0' }}</span>
```

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

Load Diff

Large diffs are not rendered by default.

Comments 0

Lock conversation

Comment

Subscribe You're not receiving notifications from this thread.