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 5f7e86f

Carlos A Sarmiento committed 2 weeks ago

```
Actualizar perfil – Cambiar Contraseña
```

master

1 parent bd706e0 commit 5f7e86f

26 files changed +728 -2 lines changed

Top

Filter files...

- .claude
  - settings.local.json
- backend
  - pom.xml
  - src/main
    - java/co/gov/loteriabogota/loteriabackend
      - domain
        - model
          - PasswordResetToken.java
        - port
          - in
            - PasswordResetUseCase.java
            - UserUseCase.java
          - out

- EmailGateway.java
- PasswordResetTokenRepository.java
- service
  - PasswordResetService.java
  - UserService.java
- infrastructure/adapter
  - in/web
    - controller
      - PasswordResetController.java
      - UserController.java
    - dto
      - ChangePasswordRequest.java
      - ForgotPasswordRequest.java
      - ResetPasswordRequest.java
  - out
    - email
      - SpringMailEmailGateway.java
    - persistence
      - adapter
        - PasswordResetTokenPersistenceAdapter.java
      - entity
        - PasswordResetTokenEntity.java
      - repository
        - JpaPasswordResetTokenRepository.java
- resources
  - application.yml
  - db/migration
    - V13__create_password_reset_tokens.sql
- frontend/src/app
  - app.routes.ts
  - core/auth

auth.service.ts

features

auth

forgot-password.component.ts

login.component.ts

reset-password.component.ts

profile

profile.component.ts

26 files changed +728 -2 lines changed

Search within code

.claude/settings.local.json

```
@@ -2,7 +2,9 @@
  "permissions": {
    "allow": [
      "Bash(mvn compile:*)",
-     "Bash(mvn test:*)"
+     "Bash(mvn test:*)",
+     "Bash(xargs cat:*)",
+     "Bash(xargs ls:*)"
    ]
  }
}
```

backend/pom.xml

```
@@ -117,6 +117,10 @@
            <groupId>org.apache.httpcomponents.client5</groupId>
            <artifactId>httpclient5</artifactId>
        </dependency>
+       <dependency>
+           <groupId>org.springframework.boot</groupId>
+           <artifactId>spring-boot-starter-mail</artifactId>
+       </dependency>
    </dependencies>

    <build>
```

…abackend/domain/model/PasswordResetToken.java

```
@@ -0,0 +1,48 @@
+ package co.gov.loteriabogota.loteriabackend.domain.model;
+
+ import java.time.Instant;
+ import java.util.UUID;
+
+ public class PasswordResetToken {
+
+     private UUID id;
+     private UUID userId;
+     private String token;
+     private Instant expiresAt;
+     private boolean used;
+     private Instant createdAt;
+
+     public PasswordResetToken() {}
+
+     public static PasswordResetToken create(UUID userId, String token, long ttlSeconds) {
+         PasswordResetToken t = new PasswordResetToken();
+         t.id = UUID.randomUUID();
+         t.userId = userId;
+         t.token = token;
+         t.expiresAt = Instant.now().plusSeconds(ttlSeconds);
+         t.used = false;
+         t.createdAt = Instant.now();
+         return t;
+     }
+
+     public boolean isExpired() {
+         return Instant.now().isAfter(expiresAt);
+     }
+
+     public boolean isValid() {
+         return !used && !isExpired();
+     }
+
+     public UUID getId() { return id; }
+     public void setId(UUID id) { this.id = id; }
```

```
38  +     public UUID getUserId() { return userId; }
39  +     public void setUserId(UUID userId) { this.userId = userId; }
40  +     public String getToken() { return token; }
41  +     public void setToken(String token) { this.token = token; }
42  +     public Instant getExpiresAt() { return expiresAt; }
43  +     public void setExpiresAt(Instant expiresAt) { this.expiresAt = expiresAt; }
44  +     public boolean isUsed() { return used; }
45  +     public void setUsed(boolean used) { this.used = used; }
46  +     public Instant getCreatedAt() { return createdAt; }
47  +     public void setCreatedAt(Instant createdAt) { this.createdAt = createdAt; }
48  + }
```

…kend/domain/port/in/PasswordResetUseCase.java

```
    @@ -0,0 +1,8 @@
1   + package co.gov.loteriabogota.loteriabackend.domain.port.in;
2   +
3   + public interface PasswordResetUseCase {
4   +
5   +     void requestPasswordReset(String email);
6   +
7   +     void resetPassword(String token, String newPassword);
8   + }
```

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

```
    @@ -10,6 +10,8 @@ public interface UserUseCase {
10  10
11  11      User updateProfile(UUID id, UpdateProfileCommand command);
12  12
    13  +     void changePassword(UUID id, String currentPassword, String newPassword);
    14  +
13  15      record UpdateProfileCommand(
14  16              String firstName,
15  17              String secondName,
```

…eriabackend/domain/port/out/EmailGateway.java

```
    @@ -0,0 +1,6 @@
1   + package co.gov.loteriabogota.loteriabackend.domain.port.out;
2   +
3   + public interface EmailGateway {
```

```
4  +
5  +     void sendPasswordResetEmail(String toEmail, String resetLink);
6  + }
```

…in/port/out/PasswordResetTokenRepository.java

```
@@ -0,0 +1,15 @@
1  + package co.gov.loteriabogota.loteriabackend.domain.port.out;
2  +
3  + import co.gov.loteriabogota.loteriabackend.domain.model.PasswordResetToken;
4  +
5  + import java.util.Optional;
6  + import java.util.UUID;
7  +
8  + public interface PasswordResetTokenRepository {
9  +
10 +     PasswordResetToken save(PasswordResetToken token);
11 +
12 +     Optional<PasswordResetToken> findByToken(String token);
13 +
14 +     void invalidateAllForUser(UUID userId);
15 + }
```

…kend/domain/service/PasswordResetService.java

```
@@ -0,0 +1,81 @@
1  + package co.gov.loteriabogota.loteriabackend.domain.service;
2  +
3  + import co.gov.loteriabogota.loteriabackend.domain.exception.DomainException;
4  + import co.gov.loteriabogota.loteriabackend.domain.model.PasswordResetToken;
5  + import co.gov.loteriabogota.loteriabackend.domain.model.User;
6  + import co.gov.loteriabogota.loteriabackend.domain.port.in.PasswordResetUseCase;
7  + import co.gov.loteriabogota.loteriabackend.domain.port.out.EmailGateway;
8  + import
     co.gov.loteriabogota.loteriabackend.domain.port.out.PasswordResetTokenRepository;
9  + import co.gov.loteriabogota.loteriabackend.domain.port.out.UserRepository;
10 + import co.gov.loteriabogota.loteriabackend.domain.vo.Email;
11 + import lombok.extern.slf4j.Slf4j;
12 + import org.springframework.beans.factory.annotation.Value;
13 + import org.springframework.security.crypto.password.PasswordEncoder;
14 + import org.springframework.stereotype.Service;
15 +
16 + import java.time.Instant;
```

```
17  + import java.util.UUID;
18  +
19  + @Slf4j
20  + @Service
21  + public class PasswordResetService implements PasswordResetUseCase {
22  +
23  +     private static final long TOKEN_TTL_SECONDS = 3600; // 1 hora
24  +
25  +     private final UserRepository userRepository;
26  +     private final PasswordResetTokenRepository tokenRepository;
27  +     private final EmailGateway emailGateway;
28  +     private final PasswordEncoder passwordEncoder;
29  +
30  +     @Value("${app.frontend-url:http://localhost:4200}")
31  +     private String frontendUrl;
32  +
33  +     public PasswordResetService(UserRepository userRepository,
34  +                                 PasswordResetTokenRepository tokenRepository,
35  +                                 EmailGateway emailGateway,
36  +                                 PasswordEncoder passwordEncoder) {
37  +         this.userRepository = userRepository;
38  +         this.tokenRepository = tokenRepository;
39  +         this.emailGateway = emailGateway;
40  +         this.passwordEncoder = passwordEncoder;
41  +     }
42  +
43  +     @Override
44  +     public void requestPasswordReset(String email) {
45  +         // Siempre respondemos OK para no revelar si el correo existe
46  +         userRepository.findByEmail(new Email(email)).ifPresent(user -> {
47  +             tokenRepository.invalidateAllForUser(user.getId());
48  +
49  +             String rawToken = UUID.randomUUID().toString();
50  +             PasswordResetToken token = PasswordResetToken.create(user.getId(),
    rawToken, TOKEN_TTL_SECONDS);
51  +             tokenRepository.save(token);
52  +
53  +             String resetLink = frontendUrl + "/auth/reset-password?token=" +
    rawToken;
54  +             try {
55  +                 emailGateway.sendPasswordResetEmail(email, resetLink);
56  +             } catch (Exception ex) {
```

```
57  +                log.error("Error enviando correo de restablecimiento a {}: {}",
    email, ex.getMessage());
58  +            }
59  +        });
60  +    }
61  +
62  +    @Override
63  +    public void resetPassword(String rawToken, String newPassword) {
64  +        PasswordResetToken token = tokenRepository.findByToken(rawToken)
65  +                .orElseThrow(() -> new DomainException("El enlace de recuperación
    es inválido o ha expirado"));
66  +
67  +        if (!token.isValid()) {
68  +            throw new DomainException("El enlace de recuperación es inválido o ha
    expirado");
69  +        }
70  +
71  +        User user = userRepository.findById(token.getUserId())
72  +                .orElseThrow(() -> new DomainException("Usuario no encontrado"));
73  +
74  +        user.setPasswordHash(passwordEncoder.encode(newPassword));
75  +        user.setUpdatedAt(Instant.now());
76  +        userRepository.save(user);
77  +
78  +        token.setUsed(true);
79  +        tokenRepository.save(token);
80  +    }
81  + }
```

…oteriabackend/domain/service/UserService.java

```
    ...   @@ -1,10 +1,12 @@
1   1     package co.gov.loteriabogota.loteriabackend.domain.service;
2   2
      3   + import co.gov.loteriabogota.loteriabackend.domain.exception.DomainException;
3   4     import co.gov.loteriabogota.loteriabackend.domain.exception.UserNotFoundException;
4   5     import co.gov.loteriabogota.loteriabackend.domain.model.User;
5   6     import co.gov.loteriabogota.loteriabackend.domain.port.in.UserUseCase;
6   7     import co.gov.loteriabogota.loteriabackend.domain.port.out.UserRepository;
7   8     import co.gov.loteriabogota.loteriabackend.domain.vo.DocumentId;
      9   + import org.springframework.security.crypto.password.PasswordEncoder;
8   10    import org.springframework.stereotype.Service;
```

```
 9   11
10   12    import java.time.Instant;
          @@ -14,9 +16,11 @@
14   16    public class UserService implements UserUseCase {
15   17
16   18        private final UserRepository userRepository;
     19 +      private final PasswordEncoder passwordEncoder;
17   20
18        -     public UserService(UserRepository userRepository) {
     21 +      public UserService(UserRepository userRepository, PasswordEncoder
           passwordEncoder) {
19   22            this.userRepository = userRepository;
     23 +          this.passwordEncoder = passwordEncoder;
20   24        }
21   25
22   26        @Override
          @@ -49,4 +53,17 @@ public User updateProfile(UUID id, UpdateProfileCommand cmd)
          {
49   53            user.setUpdatedAt(Instant.now());
50   54            return userRepository.save(user);
51   55        }
     56 +
     57 +      @Override
     58 +      public void changePassword(UUID id, String currentPassword, String newPassword)
           {
     59 +          User user = getUserById(id);
     60 +
     61 +          if (user.getPasswordHash() == null ||
           !passwordEncoder.matches(currentPassword, user.getPasswordHash())) {
     62 +              throw new DomainException("La contraseña actual es incorrecta");
     63 +          }
     64 +
     65 +          user.setPasswordHash(passwordEncoder.encode(newPassword));
     66 +          user.setUpdatedAt(Instant.now());
     67 +          userRepository.save(user);
     68 +      }
52   69    }
```

...n/web/controller/PasswordResetController.java

```
          @@ -0,0 +1,31 @@
```

```
1  + package co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.controller;
2  +
3  + import co.gov.loteriabogota.loteriabackend.domain.port.in.PasswordResetUseCase;
4  + import co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.ForgotPasswordRequest;
5  + import co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.ResetPasswordRequest;
6  + import jakarta.validation.Valid;
7  + import org.springframework.http.ResponseEntity;
8  + import org.springframework.web.bind.annotation.*;
9  +
10 + @RestController
11 + @RequestMapping("/api/auth")
12 + public class PasswordResetController {
13 +
14 +     private final PasswordResetUseCase passwordResetUseCase;
15 +
16 +     public PasswordResetController(PasswordResetUseCase passwordResetUseCase) {
17 +         this.passwordResetUseCase = passwordResetUseCase;
18 +     }
19 +
20 +     @PostMapping("/forgot-password")
21 +     public ResponseEntity<Void> forgotPassword(@Valid @RequestBody ForgotPasswordRequest request) {
22 +         passwordResetUseCase.requestPasswordReset(request.email());
23 +         return ResponseEntity.noContent().build();
24 +     }
25 +
26 +     @PostMapping("/reset-password")
27 +     public ResponseEntity<Void> resetPassword(@Valid @RequestBody ResetPasswordRequest request) {
28 +         passwordResetUseCase.resetPassword(request.token(), request.newPassword());
29 +         return ResponseEntity.noContent().build();
30 +     }
31 + }
```

…adapter/in/web/controller/UserController.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.port.in.UserUseCase;
    5   + import
          co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.ChangePasswor
          dRequest;
5   6     import
          co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.UpdateProfile
          Request;
6   7     import
          co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto.UserResponse;
7   8     import jakarta.validation.Valid;
          @@ -49,6 +50,14 @@ public ResponseEntity<UserResponse>
          updateProfile(Authentication auth,
49  50             return ResponseEntity.ok(toResponse(user));
50  51         }
51  52
    53  +     @PutMapping("/me/password")
    54  +     public ResponseEntity<Void> changePassword(Authentication auth,
    55  +                                                @Valid @RequestBody
          ChangePasswordRequest request) {
    56  +         UUID userId = UUID.fromString(auth.getPrincipal().toString());
    57  +         userUseCase.changePassword(userId, request.currentPassword(),
          request.newPassword());
    58  +         return ResponseEntity.noContent().build();
    59  +     }
    60  +
52  61         private UserResponse toResponse(User user) {
53  62             return new UserResponse(
54  63                     user.getId(),
```

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

```
    @@ -0,0 +1,9 @@
1   + package co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto;
2   +
3   + import jakarta.validation.constraints.NotBlank;
4   + import jakarta.validation.constraints.Size;
5   +
6   + public record ChangePasswordRequest(
7   +         @NotBlank String currentPassword,
```

```
8   +         @NotBlank @Size(min = 8, message = "La nueva contraseña debe tener al menos
      8 caracteres") String newPassword
9   + ) {}
```

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

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

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

```
•••   @@ -0,0 +1,9 @@
1   + package co.gov.loteriabogota.loteriabackend.infrastructure.adapter.in.web.dto;
2   +
3   + import jakarta.validation.constraints.NotBlank;
4   + import jakarta.validation.constraints.Size;
5   +
6   + public record ResetPasswordRequest(
7   +         @NotBlank String token,
8   +         @NotBlank @Size(min = 8, message = "La contraseña debe tener al menos 8
      caracteres") String newPassword
9   + ) {}
```

…adapter/out/email/SpringMailEmailGateway.java

```
•••   @@ -0,0 +1,40 @@
1   + package co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.email;
2   +
3   + import co.gov.loteriabogota.loteriabackend.domain.port.out.EmailGateway;
4   + import lombok.extern.slf4j.Slf4j;
5   + import org.springframework.mail.SimpleMailMessage;
6   + import org.springframework.mail.javamail.JavaMailSender;
7   + import org.springframework.beans.factory.annotation.Value;
8   + import org.springframework.stereotype.Component;
9   +
10  + @Slf4j
```

```
11  + @Component
12  + public class SpringMailEmailGateway implements EmailGateway {
13  +
14  +     private final JavaMailSender mailSender;
15  +
16  +     @Value("${spring.mail.username:noreply@loteriabogota.gov.co}")
17  +     private String fromAddress;
18  +
19  +     public SpringMailEmailGateway(JavaMailSender mailSender) {
20  +         this.mailSender = mailSender;
21  +     }
22  +
23  +     @Override
24  +     public void sendPasswordResetEmail(String toEmail, String resetLink) {
25  +         SimpleMailMessage message = new SimpleMailMessage();
26  +         message.setFrom(fromAddress);
27  +         message.setTo(toEmail);
28  +         message.setSubject("Restablecer contraseña – Lotería de Bogotá");
29  +         message.setText(
30  +                 "Hola,\n\n" +
31  +                 "Recibimos una solicitud para restablecer la contraseña de tu
      cuenta en Lotería de Bogotá.\n\n" +
32  +                 "Haz clic en el siguiente enlace para crear una nueva contraseña
      (válido por 1 hora):\n\n" +
33  +                 resetLink + "\n\n" +
34  +                 "Si no solicitaste este cambio, puedes ignorar este correo.\n\n" +
35  +                 "Lotería de Bogotá"
36  +         );
37  +         mailSender.send(message);
38  +         log.info("Correo de restablecimiento enviado a {}", toEmail);
39  +     }
40  + }
```

…ter/PasswordResetTokenPersistenceAdapter.java

```
•••   @@ -0,0 +1,57 @@
1   + package
      co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.persistence.adapter;
2   +
3   + import co.gov.loteriabogota.loteriabackend.domain.model.PasswordResetToken;
4   + import
      co.gov.loteriabogota.loteriabackend.domain.port.out.PasswordResetTokenRepository;
```

```
5  + import co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.persistence.entity.PasswordResetTokenEntity;
6  + import co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.persistence.repository.JpaPasswordResetTokenRepository;
7  + import org.springframework.stereotype.Component;
8  +
9  + import java.util.Optional;
10 + import java.util.UUID;
11 +
12 + @Component
13 + public class PasswordResetTokenPersistenceAdapter implements PasswordResetTokenRepository {
14 +
15 +     private final JpaPasswordResetTokenRepository jpaRepository;
16 +
17 +     public PasswordResetTokenPersistenceAdapter(JpaPasswordResetTokenRepository jpaRepository) {
18 +         this.jpaRepository = jpaRepository;
19 +     }
20 +
21 +     @Override
22 +     public PasswordResetToken save(PasswordResetToken token) {
23 +         return toDomain(jpaRepository.save(toEntity(token)));
24 +     }
25 +
26 +     @Override
27 +     public Optional<PasswordResetToken> findByToken(String token) {
28 +         return jpaRepository.findByToken(token).map(this::toDomain);
29 +     }
30 +
31 +     @Override
32 +     public void invalidateAllForUser(UUID userId) {
33 +         jpaRepository.markAllUsedByUserId(userId);
34 +     }
35 +
36 +     private PasswordResetToken toDomain(PasswordResetTokenEntity e) {
37 +         PasswordResetToken t = new PasswordResetToken();
38 +         t.setId(e.getId());
39 +         t.setUserId(e.getUserId());
40 +         t.setToken(e.getToken());
```

```
41  +         t.setExpiresAt(e.getExpiresAt());
42  +         t.setUsed(e.isUsed());
43  +         t.setCreatedAt(e.getCreatedAt());
44  +         return t;
45  +     }
46  +
47  +     private PasswordResetTokenEntity toEntity(PasswordResetToken t) {
48  +         PasswordResetTokenEntity e = new PasswordResetTokenEntity();
49  +         e.setId(t.getId());
50  +         e.setUserId(t.getUserId());
51  +         e.setToken(t.getToken());
52  +         e.setExpiresAt(t.getExpiresAt());
53  +         e.setUsed(t.isUsed());
54  +         e.setCreatedAt(t.getCreatedAt());
55  +         return e;
56  +     }
57  + }
```

…sistence/entity/PasswordResetTokenEntity.java

```
...   @@ -0,0 +1,46 @@
1   + package
      co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.persistence.entity;
2   +
3   + import jakarta.persistence.*;
4   + import java.time.Instant;
5   + import java.util.UUID;
6   +
7   + @Entity
8   + @Table(name = "password_reset_tokens")
9   + public class PasswordResetTokenEntity {
10  +
11  +     @Id
12  +     private UUID id;
13  +
14  +     @Column(name = "user_id", nullable = false)
15  +     private UUID userId;
16  +
17  +     @Column(nullable = false, unique = true, length = 255)
18  +     private String token;
19  +
20  +     @Column(name = "expires_at", nullable = false)
```

```
21  +     private Instant expiresAt;
22  +
23  +     @Column(nullable = false)
24  +     private boolean used;
25  +
26  +     @Column(name = "created_at", nullable = false, updatable = false)
27  +     private Instant createdAt;
28  +
29  +     @PrePersist
30  +     void onCreate() {
31  +         if (createdAt == null) createdAt = Instant.now();
32  +     }
33  +
34  +     public UUID getId() { return id; }
35  +     public void setId(UUID id) { this.id = id; }
36  +     public UUID getUserId() { return userId; }
37  +     public void setUserId(UUID userId) { this.userId = userId; }
38  +     public String getToken() { return token; }
39  +     public void setToken(String token) { this.token = token; }
40  +     public Instant getExpiresAt() { return expiresAt; }
41  +     public void setExpiresAt(Instant expiresAt) { this.expiresAt = expiresAt; }
42  +     public boolean isUsed() { return used; }
43  +     public void setUsed(boolean used) { this.used = used; }
44  +     public Instant getCreatedAt() { return createdAt; }
45  +     public void setCreatedAt(Instant createdAt) { this.createdAt = createdAt; }
46  + }
```

…pository/JpaPasswordResetTokenRepository.java

```
•••   @@ -0,0 +1,20 @@
1   + package
      co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.persistence.reposito
      ry;
2   +
3   + import
      co.gov.loteriabogota.loteriabackend.infrastructure.adapter.out.persistence.entity.P
      asswordResetTokenEntity;
4   + import org.springframework.data.jpa.repository.JpaRepository;
5   + import org.springframework.data.jpa.repository.Modifying;
6   + import org.springframework.data.jpa.repository.Query;
7   + import org.springframework.transaction.annotation.Transactional;
8   +
```

```
+ import java.util.Optional;
+ import java.util.UUID;
+
+ public interface JpaPasswordResetTokenRepository extends JpaRepository<PasswordResetTokenEntity, UUID> {
+
+     Optional<PasswordResetTokenEntity> findByToken(String token);
+
+     @Modifying
+     @Transactional
+     @Query("UPDATE PasswordResetTokenEntity t SET t.used = true WHERE t.userId = :userId AND t.used = false")
+     void markAllUsedByUserId(UUID userId);
+ }
```

backend/src/main/resources/application.yml

```
@@ -4,6 +4,17 @@ spring:
    threads:
      virtual:
        enabled: true
+   mail:
+     host: smtp.gmail.com
+     port: 587
+     username: noreply@loteriabogota.gov.co
+     password: your-smtp-password
+     properties:
+       mail:
+         smtp:
+           auth: true
+           starttls:
+             enable: true
    jpa:
      open-in-view: false
      hibernate:
@@ -22,6 +33,7 @@ app:
      base-url: https://checkout-test.placetopay.com
    cors:
      allowed-origins: http://localhost:4200
+   frontend-url: http://localhost:4200
    comercial:
      base-url: https://192.172.200.44:8443
```

```
27  39    springdoc:
```

...gration/V13__create_password_reset_tokens.sql

```
    @@ -0,0 +1,11 @@
1   + CREATE TABLE password_reset_tokens (
2   +     id UUID PRIMARY KEY,
3   +     user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
4   +     token VARCHAR(255) NOT NULL UNIQUE,
5   +     expires_at TIMESTAMP WITH TIME ZONE NOT NULL,
6   +     used BOOLEAN NOT NULL DEFAULT FALSE,
7   +     created_at TIMESTAMP WITH TIME ZONE NOT NULL DEFAULT CURRENT_TIMESTAMP
8   + );
9   +
10  + CREATE INDEX idx_password_reset_tokens_token ON password_reset_tokens(token);
11  + CREATE INDEX idx_password_reset_tokens_user_id ON password_reset_tokens(user_id);
```

frontend/src/app/app.routes.ts

```
    @@ -33,6 +33,14 @@ export const routes: Routes = [
33  33            path: 'register',
34  34            loadComponent: () => import('@features/auth/register.component').then(m =>
        m.RegisterComponent),
35  35          },
36  +         {
37  +           path: 'forgot-password',
38  +           loadComponent: () => import('@features/auth/forgot-
        password.component').then(m => m.ForgotPasswordComponent),
39  +         },
40  +         {
41  +           path: 'reset-password',
42  +           loadComponent: () => import('@features/auth/reset-
        password.component').then(m => m.ResetPasswordComponent),
43  +         },
36  44        ],
37  45      },
38  46      { path: '**', redirectTo: '' },
```

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

```
    @@ -92,6 +92,18 @@ export class AuthService {
```

```
92  92          );
93  93        }
94  94
    95  +     forgotPassword(email: string): Observable<void> {
    96  +       return this.http.post<void>(`${environment.apiUrl}/auth/forgot-password`, { email });
    97  +     }
    98  +
    99  +     resetPassword(token: string, newPassword: string): Observable<void> {
    100 +       return this.http.post<void>(`${environment.apiUrl}/auth/reset-password`, { token, newPassword });
    101 +     }
    102 +
    103 +     changePassword(currentPassword: string, newPassword: string): Observable<void> {
    104 +       return this.http.put<void>(`${environment.apiUrl}/users/me/password`, { currentPassword, newPassword });
    105 +     }
    106 +
95  107       loadProfile(): Observable<UserProfile> {
96  108         return this.http.get<UserProfile>(`${environment.apiUrl}/users/me`).pipe(
97  109           tap((profile) => this.user.set(profile)),
```

…pp/features/auth/forgot-password.component.ts

```
    @@ -0,0 +1,88 @@
1   + import { Component, signal } from '@angular/core';
2   + import { RouterLink } from '@angular/router';
3   + import { FormsModule } from '@angular/forms';
4   + import { AuthService } from '@core/auth/auth.service';
5   + import { SpinnerComponent } from '@shared/components/spinner/spinner.component';
6   + import { ErrorMessageComponent } from '@shared/components/error-message/error-message.component';
7   +
8   + @Component({
9   +   selector: 'app-forgot-password',
10  +   standalone: true,
11  +   imports: [FormsModule, RouterLink, SpinnerComponent, ErrorMessageComponent],
12  +   template: `
13  +     @if (loading()) {
14  +       <app-spinner />
15  +     } @else {
```

```
16  +       <div class="space-y-4">
17  +         <h2 class="text-xl font-bold text-center">Recuperar contraseña</h2>
18  +         <p class="text-sm text-gray-500 text-center">
19  +           Ingresa tu correo y te enviaremos un enlace para restablecer tu
      contraseña.
20  +         </p>
21  +
22  +         @if (success()) {
23  +           <div class="bg-green-50 border border-green-200 rounded-xl p-4 text-
      center">
24  +             <p class="text-green-700 font-semibold text-sm">¡Correo enviado!</p>
25  +             <p class="text-green-600 text-sm mt-1">
26  +               Si el correo está registrado, recibirás un enlace para restablecer tu
      contraseña.
27  +             </p>
28  +           </div>
29  +         } @else {
30  +           @if (error()) {
31  +             <app-error-message [message]="error()!" />
32  +           }
33  +
34  +           <input
35  +             type="email"
36  +             [(ngModel)]="email"
37  +             placeholder="Correo electronico"
38  +             class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
39  +                    focus:border-loteria-red focus:outline-none transition-colors"
40  +             (keydown.enter)="submit()"
41  +           />
42  +
43  +           <button
44  +             (click)="submit()"
45  +             [disabled]="!email().includes('@')"
46  +             class="w-full bg-loteria-red text-white py-3 rounded-xl font-semibold
47  +                    hover:bg-loteria-red-dark transition-colors disabled:opacity-50
      disabled:cursor-not-allowed"
48  +           >
49  +             Enviar enlace
50  +           </button>
51  +         }
52  +
53  +         <p class="text-center text-sm text-gray-500">
```

```
54  +           <a routerLink="/auth/login" class="text-loteria-red font-semibold
    hover:underline">
55  +             Volver a iniciar sesion
56  +           </a>
57  +         </p>
58  +       </div>
59  +     }
60  +   `,
61  + })
62  + export class ForgotPasswordComponent {
63  +   readonly email = signal('');
64  +   readonly loading = signal(false);
65  +   readonly error = signal<string | null>(null);
66  +   readonly success = signal(false);
67  +
68  +   constructor(private readonly auth: AuthService) {}
69  +
70  +   submit(): void {
71  +     if (!this.email().includes('@')) return;
72  +
73  +     this.loading.set(true);
74  +     this.error.set(null);
75  +
76  +     this.auth.forgotPassword(this.email()).subscribe({
77  +       next: () => {
78  +         this.loading.set(false);
79  +         this.success.set(true);
80  +       },
81  +       error: () => {
82  +         this.loading.set(false);
83  +         // Mostramos éxito igual para no revelar si el correo existe
84  +         this.success.set(true);
85  +       },
86  +     });
87  +   }
88  + }
```

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

```
@@ -46,6 +46,12 @@ import { ErrorMessageComponent } from
'@shared/components/error-message/error-me
46  46            Iniciar sesion
```

```
47  47            </button>
48  48
    49  +         <p class="text-center text-sm">
    50  +           <a routerLink="/auth/forgot-password" class="text-gray-500
        hover:underline">
    51  +             ¿Olvidaste tu contraseña?
    52  +           </a>
    53  +         </p>
    54  +
49  55            <p class="text-center text-sm text-gray-500">
50  56              No tienes cuenta?
51  57              <a routerLink="/auth/register" class="text-loteria-red font-semibold
        hover:underline">
```

…app/features/auth/reset-password.component.ts

```
•••   @@ -0,0 +1,113 @@
1   + import { Component, OnInit, signal } from '@angular/core';
2   + import { Router, RouterLink, ActivatedRoute } from '@angular/router';
3   + import { FormsModule } from '@angular/forms';
4   + import { AuthService } from '@core/auth/auth.service';
5   + import { SpinnerComponent } from '@shared/components/spinner/spinner.component';
6   + import { ErrorMessageComponent } from '@shared/components/error-message/error-
      message.component';
7   +
8   + @Component({
9   +   selector: 'app-reset-password',
10  +   standalone: true,
11  +   imports: [FormsModule, RouterLink, SpinnerComponent, ErrorMessageComponent],
12  +   template: `
13  +     @if (loading()) {
14  +       <app-spinner />
15  +     } @else {
16  +       <div class="space-y-4">
17  +         <h2 class="text-xl font-bold text-center">Nueva contraseña</h2>
18  +         <p class="text-sm text-gray-500 text-center">Ingresa tu nueva contraseña.
      </p>
19  +
20  +         @if (success()) {
21  +           <div class="bg-green-50 border border-green-200 rounded-xl p-4 text-
      center">
```

```
22  +            <p class="text-green-700 font-semibold text-sm">Contraseña
    restablecida</p>
23  +            <p class="text-green-600 text-sm mt-1">Ya puedes iniciar sesion con tu
    nueva contraseña.</p>
24  +          </div>
25  +          <a routerLink="/auth/login"
26  +             class="block w-full text-center bg-loteria-red text-white py-3
    rounded-xl font-semibold
27  +                    hover:bg-loteria-red-dark transition-colors">
28  +            Iniciar sesion
29  +          </a>
30  +        } @else {
31  +          @if (!token()) {
32  +            <div class="bg-red-50 border border-red-200 rounded-xl p-4 text-
    center">
33  +              <p class="text-red-600 text-sm">Enlace inválido. Solicita uno nuevo.
    </p>
34  +            </div>
35  +          } @else {
36  +            @if (error()) {
37  +              <app-error-message [message]="error()!" />
38  +            }
39  +
40  +            <input
41  +              type="password"
42  +              [(ngModel)]="newPassword"
43  +              placeholder="Nueva contraseña (mínimo 8 caracteres)"
44  +              class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
45  +                     focus:border-loteria-red focus:outline-none transition-
    colors"
46  +            />
47  +            <input
48  +              type="password"
49  +              [(ngModel)]="confirmPassword"
50  +              placeholder="Confirmar contraseña"
51  +              class="w-full px-4 py-3 border-2 border-gray-200 rounded-xl
52  +                     focus:border-loteria-red focus:outline-none transition-
    colors"
53  +              (keydown.enter)="submit()"
54  +            />
55  +
56  +            <button
```

```
57  +              (click)="submit()"
58  +              [disabled]="newPassword().length < 8 || newPassword() !==
      confirmPassword()"
59  +              class="w-full bg-loteria-red text-white py-3 rounded-xl font-
      semibold
60  +                     hover:bg-loteria-red-dark transition-colors disabled:opacity-
      50 disabled:cursor-not-allowed"
61  +            >
62  +              Restablecer contraseña
63  +            </button>
64  +          }
65  +        }
66  +
67  +        <p class="text-center text-sm text-gray-500">
68  +          <a routerLink="/auth/login" class="text-loteria-red font-semibold
      hover:underline">
69  +            Volver a iniciar sesion
70  +          </a>
71  +        </p>
72  +      </div>
73  +    }
74  +  `,
75  + })
76  + export class ResetPasswordComponent implements OnInit {
77  +   readonly newPassword = signal('');
78  +   readonly confirmPassword = signal('');
79  +   readonly loading = signal(false);
80  +   readonly error = signal<string | null>(null);
81  +   readonly success = signal(false);
82  +   readonly token = signal<string | null>(null);
83  +
84  +   constructor(
85  +     private readonly auth: AuthService,
86  +     private readonly route: ActivatedRoute,
87  +   ) {}
88  +
89  +   ngOnInit(): void {
90  +     const token = this.route.snapshot.queryParamMap.get('token');
91  +     this.token.set(token);
92  +   }
93  +
94  +   submit(): void {
```

```
95  +     if (this.newPassword().length < 8 || this.newPassword() !==
        this.confirmPassword()) return;
96  +     const token = this.token();
97  +     if (!token) return;
98  +
99  +     this.loading.set(true);
100 +     this.error.set(null);
101 +
102 +     this.auth.resetPassword(token, this.newPassword()).subscribe({
103 +       next: () => {
104 +         this.loading.set(false);
105 +         this.success.set(true);
106 +       },
107 +       error: (err) => {
108 +         this.loading.set(false);
109 +         this.error.set(err.error?.message ?? 'El enlace es inválido o ha
        expirado');
110 +       },
111 +     });
112 +   }
113 + }
```

…src/app/features/profile/profile.component.ts

```
@@ -194,6 +194,43 @@ interface Ticket {
194 194           }
195 195         </div>
196 196
197 +       <!-- Cambiar contraseña -->
198 +       <div class="bg-white rounded-xl shadow-sm p-4 space-y-3">
199 +         <h2 class="font-semibold">Cambiar contraseña</h2>
200 +
201 +         <div>
202 +           <label class="text-xs text-gray-500 mb-1 block">Contraseña
        actual</label>
203 +           <input type="password" [(ngModel)]="currentPassword"
204 +             class="w-full px-3 py-2 border-2 border-gray-200 rounded-lg text-sm
        focus:border-loteria-red focus:outline-none" />
205 +         </div>
206 +         <div>
207 +           <label class="text-xs text-gray-500 mb-1 block">Nueva contraseña (mínimo
        8 caracteres)</label>
```

```
208 + <input type="password" [(ngModel)]="newPassword"
209 + class="w-full px-3 py-2 border-2 border-gray-200 rounded-lg text-sm focus:border-loteria-red focus:outline-none" />
210 + </div>
211 + <div>
212 + <label class="text-xs text-gray-500 mb-1 block">Confirmar nueva contraseña</label>
213 + <input type="password" [(ngModel)]="confirmNewPassword"
214 + class="w-full px-3 py-2 border-2 border-gray-200 rounded-lg text-sm focus:border-loteria-red focus:outline-none" />
215 + </div>
216 +
217 + @if (passwordSuccess()) {
218 + <p class="text-green-600 text-sm">Contraseña actualizada correctamente</p>
219 + }
220 + @if (passwordError()) {
221 + <p class="text-red-500 text-sm">{{ passwordError() }}</p>
222 + }
223 +
224 + <button
225 + (click)="changePassword()"
226 + [disabled]="changingPassword() || !currentPassword() || newPassword().length < 8 || newPassword() !== confirmNewPassword()"
227 + class="bg-loteria-red text-white px-6 py-2 rounded-lg text-sm font-semibold
228 + hover:bg-loteria-red-dark transition-colors disabled:opacity-50"
229 + >
230 + {{ changingPassword() ? 'Guardando...' : 'Cambiar contraseña' }}
231 + </button>
232 + </div>
233 +
197 234 <!-- Logout -->
198 235 <button
199 236 (click)="auth.logout()"
```

```
@@ -224,6 +261,13 @@ export class ProfileComponent implements OnInit {
224 261 readonly saveSuccess = signal(false);
225 262 readonly saveError = signal<string | null>(null);
226 263
264 + readonly currentPassword = signal('');
265 + readonly newPassword = signal('');
```

```
      266  +   readonly confirmNewPassword = signal('');
      267  +   readonly changingPassword = signal(false);
      268  +   readonly passwordSuccess = signal(false);
      269  +   readonly passwordError = signal<string | null>(null);
      270  +
227   271       readonly ticketsResource = rxResource({
228   272         loader: () => this.http.get<Ticket[]>(`${environment.apiUrl}/tickets`),
229   273       });
          @@ -254,6 +298,26 @@ export class ProfileComponent implements OnInit {
254   298         return 'Usuario';
255   299       }
256   300
      301  +   changePassword(): void {
      302  +     this.changingPassword.set(true);
      303  +     this.passwordSuccess.set(false);
      304  +     this.passwordError.set(null);
      305  +
      306  +     this.auth.changePassword(this.currentPassword(),
          this.newPassword()).subscribe({
      307  +       next: () => {
      308  +         this.changingPassword.set(false);
      309  +         this.passwordSuccess.set(true);
      310  +         this.currentPassword.set('');
      311  +         this.newPassword.set('');
      312  +         this.confirmNewPassword.set('');
      313  +       },
      314  +       error: (err) => {
      315  +         this.changingPassword.set(false);
      316  +         this.passwordError.set(err.error?.message ?? 'Error al cambiar la
          contraseña');
      317  +       },
      318  +     });
      319  +   }
      320  +
257   321       updateProfile(): void {
258   322         this.saving.set(true);
259   323         this.saveSuccess.set(false);
```

Comments 0

Lock conversation



Comment

Subscribe You're not receiving notifications from this thread.