OpenAEV-Platform / **openaev** Public

Code Issues 609 Pull requests 66 Discussions Actions Projects

Commit `c09a4e7`

richard-julien committed on Sep 27, 2022 Verified

```
[api/front] Be able to reset the password through a link on the login page (#135)
```

`master` · 2.3.3 ••• 1.0.0

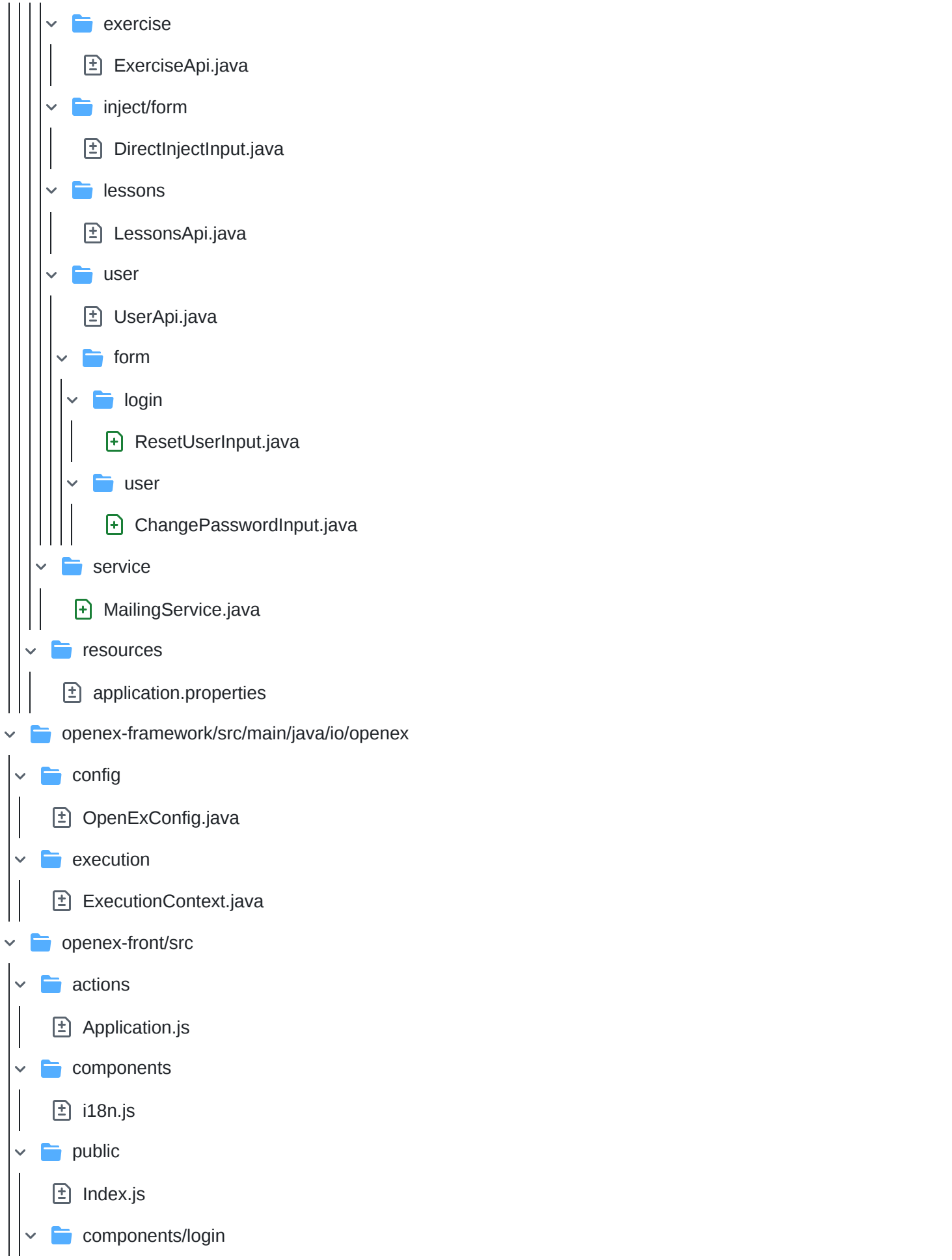
1 parent `b76c3c1` commit `c09a4e7`

26 files changed +426 -107 lines changed

Top

Filter files…

- openex-api
  - pom.xml
  - src/main
    - java/io/openex
      - config
        - AppSecurityConfig.java
      - injects
        - challenge
          - ChallengeExecutor.java
        - email
          - EmailExecutor.java
        - model
          - EmailContent.java
        - media
          - MediaExecutor.java
      - rest


exercise
ExerciseApi.java
inject/form
DirectInjectInput.java
lessons
LessonsApi.java
user
UserApi.java
form
login
ResetUserInput.java
user
ChangePasswordInput.java
service
MailingService.java
resources
application.properties
openex-framework/src/main/java/io/openex
config
OpenExConfig.java
execution
ExecutionContext.java
openex-front/src
actions
Application.js
components
i18n.js
public
Index.js
components/login

- Login.js
- Reset.js
- resources/css
  - main.css
- utils
  - Localization.js
- openex-model
  - pom.xml
  - src/main/java/io/openex/database/model
    - Exercise.java
    - Inject.java

26 files changed +426 -107 lines changed

Search within code

openex-api/pom.xml

```
@@ -154,6 +154,11 @@
154 154                 <artifactId>commons-email</artifactId>
155 155                 <version>1.5</version>
156 156             </dependency>
    157 +           <dependency>
    158 +               <groupId>org.apache.commons</groupId>
    159 +               <artifactId>commons-collections4</artifactId>
    160 +               <version>4.4</version>
    161 +           </dependency>
157 162             <dependency>
158 163                 <groupId>org.flywaydb</groupId>
159 164                 <artifactId>flyway-core</artifactId>
```

...ava/io/openex/config/AppSecurityConfig.java

```
@@ -63,6 +63,7 @@ protected void configure(HttpSecurity http) throws
Exception {
63 63                  /**/.antMatchers("/api/player/**").permitAll()
64 64                  /**/.antMatchers("/api/settings").permitAll()
65 65                  /**/.antMatchers("/api/login").permitAll()
```

```
      66 +                 /**/.antMatchers("/api/reset/**").permitAll()
66    67                   /**/.antMatchers("/api/**").authenticated()
67    68                   .and()
68    69                   .logout()
```

...ex/injects/challenge/ChallengeExecutor.java

```
@@ -79,7 +79,7 @@ public List<Expectation> process(Execution execution,
ExecutableInject injection
79    79                   List<Document> documents =
injection.getInject().getDocuments().stream()
80    80
.filter(InjectDocument::isAttached).map(InjectDocument::getDocument).toList();
81    81                   List<DataAttachment> attachments = resolveAttachments(execution,
injection, documents);
82        -                String message = content.buildMessage(injection.getInject(),
imapEnabled);
      82 +                String message = content.buildMessage(injection, imapEnabled);
83    83                   boolean encrypted = content.isEncrypted();
84    84                   users.stream().parallel().forEach(userInjectContext -> {
85    85                       try {
```

.../io/openex/injects/email/EmailExecutor.java

```
@@ -71,7 +71,7 @@ public List<Expectation> process(Execution execution,
ExecutableInject injection
71    71           List<DataAttachment> attachments = resolveAttachments(execution,
injection, documents);
72    72           String inReplyTo = content.getInReplyTo();
73    73           String subject = content.getSubject();
74        -        String message = content.buildMessage(inject, imapEnabled);
      74 +        String message = content.buildMessage(injection, imapEnabled);
75    75           boolean mustBeEncrypted = content.isEncrypted();
76    76           // Resolve the attachments only once
77    77           List<ExecutionContext> users = injection.getUsers();
@@ -83,7 +83,7 @@ public List<Expectation> process(Execution execution,
ExecutableInject injection
83    83               users = users.stream().peek(context -> context.put("document_uri",
buildDocumentUri(context, inject))).toList();
```

```
84  84          }
85  85          Exercise exercise = injection.getSource().getExercise();
86      -       String replyTo = exercise.getReplyTo();
    86  +       String replyTo = exercise != null ? exercise.getReplyTo() :
            openExConfig.getDefaultMailer();
87  87          //noinspection SwitchStatementWithTooFewBranches
88  88          switch (contract.getId()) {
89  89              case EMAIL_GLOBAL -> sendMulti(execution, users, replyTo, inReplyTo,
            subject, message, attachments);
```

...penex/injects/email/model/EmailContent.java

```
    @@ -1,7 +1,7 @@
1   1   package io.openex.injects.email.model;
2   2
3   3   import com.fasterxml.jackson.annotation.JsonProperty;
4       - import io.openex.database.model.Inject;
    4   + import io.openex.execution.ExecutableInject;
5   5   import org.springframework.util.StringUtils;
6   6
7   7   import java.util.Objects;
    @@ -58,24 +58,21 @@ public void setEncrypted(boolean encrypted) {
58  58          this.encrypted = encrypted;
59  59      }
60  60
61      -   public String buildMessage(Inject inject, boolean imapEnabled) {
    61  +   public String buildMessage(ExecutableInject injection, boolean imapEnabled)
            {
62  62          // String footer = inject.getFooter();
63      -       String header = inject.getHeader();
    63  +       String header = injection.getInject().getHeader();
64  64          StringBuilder data = new StringBuilder();
65  65          if (StringUtils.hasLength(header)) {
66  66              data.append(HEADER_DIV).append(header).append(END_DIV);
67  67          }
68  68          data.append(START_DIV).append(body).append(END_DIV);
69      -       // if (StringUtils.hasLength(footer)) {
70      -       //    data.append(FOOTER_DIV).append(footer).append(END_DIV);
71      -       // }
```

```
72  69          // If imap is enable we need to inject the id marker
73      -       if (imapEnabled && !inject.isDirect()) {
    70  +       if (injection.isRuntime() && imapEnabled) {
74  71              data.append(START_DIV)
75  72                      .append("<br/><br/><br/><br/>")
76  73                      .append("------------------------------------------------
    ------------------------------<br/>")
77  74                      .append("OpenEx internal information, do not remove!<br/>")
78      -                   .append("[inject_id=").append(inject.getId()).append("]
    <br/>")
    75  +                   .append("
    [inject_id=").append(injection.getInject().getId()).append("]<br/>")
79  76                      .append("------------------------------------------------
    ------------------------------<br/>")
80  77                      .append(END_DIV);
81  78          }
```

.../io/openex/injects/media/MediaExecutor.java

```
    @@ -79,7 +79,7 @@ public List<Expectation> process(Execution execution,
    ExecutableInject injection
79  79                      List<Document> documents =
    injection.getInject().getDocuments().stream()
80  80
    .filter(InjectDocument::isAttached).map(InjectDocument::getDocument).toList();
81  81                      List<DataAttachment> attachments =
    resolveAttachments(execution, injection, documents);
82      -                   String message = content.buildMessage(injection.getInject(),
    imapEnabled);
    82  +                   String message = content.buildMessage(injection,
    imapEnabled);
83  83                      boolean encrypted = content.isEncrypted();
84  84                      users.stream().parallel().forEach(userInjectContext -> {
85  85                          try {
```

...va/io/openex/rest/exercise/ExerciseApi.java

```
    @@ -1,6 +1,7 @@
1   1   package io.openex.rest.exercise;
```

```
2   2
3   3    import com.fasterxml.jackson.databind.ObjectMapper;
    4  +  import io.openex.config.OpenExConfig;
4   5    import io.openex.database.model.*;
5   6    import io.openex.database.model.Exercise.STATUS;
6   7    import io.openex.database.repository.*;
         @@ -21,6 +22,7 @@
21  22   import org.springframework.web.multipart.MultipartFile;
22  23   import reactor.util.function.Tuples;
23  24
    25 +  import javax.annotation.Resource;
24  26   import javax.annotation.security.RolesAllowed;
25  27   import javax.servlet.http.HttpServletResponse;
26  28   import javax.transaction.Transactional;
         @@ -61,6 +63,9 @@ public class ExerciseApi extends RestBehavior {
61  63
62  64       @Value("${openex.mail.imap.username}")
63  65       private String imapUsername;
    66 +
    67 +     @Resource
    68 +     private OpenExConfig openExConfig;
64  69       // endregion
65  70
66  71       // region repositories
         @@ -292,6 +297,8 @@ public Exercise createExercise(@Valid @RequestBody
         ExerciseCreateInput input) {
292 297
                 exercise.setTags(fromIterable(tagRepository.findAllById(input.getTagIds())));
293 298              if (imapEnabled) {
294 299                  exercise.setReplyTo(imapUsername);
    300 +            } else {
    301 +                exercise.setReplyTo(openExConfig.getDefaultMailer());
295 302              }
296 303              // Find automatic groups to grants
297 304              List<Group> groups = fromIterable(groupRepository.findAll());
```

...nex/rest/inject/form/DirectInjectInput.java

```
         @@ -81,7 +81,6 @@ public Inject toInject() {
```

```
81  81              inject.setDescription(getDescription());
82  82              inject.setContent(getContent());
83  83              inject.setContract(getContract());
84      -           inject.setDirect(true);
85  84              return inject;
86  85          }
87  86      }
```

...java/io/openex/rest/lessons/LessonsApi.java

```
    ...     @@ -1,26 +1,17 @@
1   1       package io.openex.rest.lessons;
2   2
3       -   import io.openex.config.OpenExConfig;
4       -   import io.openex.contract.Contract;
5   3       import io.openex.database.model.*;
6   4       import io.openex.database.repository.*;
7   5       import io.openex.database.specification.LessonsAnswerSpecification;
8   6       import io.openex.database.specification.LessonsCategorySpecification;
9   7       import io.openex.database.specification.LessonsQuestionSpecification;
10      -   import io.openex.execution.ExecutableInject;
11      -   import io.openex.execution.ExecutionContext;
12      -   import io.openex.execution.Injector;
13      -   import io.openex.injects.email.EmailContract;
14      -   import io.openex.injects.email.model.EmailContent;
15  8       import io.openex.rest.helper.RestBehavior;
16  9       import io.openex.rest.lessons.form.*;
17      -   import io.openex.service.ContractService;
    10  +   import io.openex.service.MailingService;
18  11      import org.springframework.beans.factory.annotation.Autowired;
19      -   import org.springframework.context.ApplicationContext;
20  12      import org.springframework.security.access.prepost.PreAuthorize;
21  13      import org.springframework.web.bind.annotation.*;
22  14
23      -   import javax.annotation.Resource;
24  15      import javax.validation.Valid;
25  16      import java.util.List;
26  17      import java.util.Optional;
    ...     @@ -32,19 +23,19 @@
32  23
33  24      @RestController
```

```
34  25    public class LessonsApi extends RestBehavior {
35        -
36        -     @Resource
37        -     private OpenExConfig openExConfig;
38        -
39  26        private ExerciseRepository exerciseRepository;
40  27        private AudienceRepository audienceRepository;
41  28        private LessonsTemplateRepository lessonsTemplateRepository;
42  29        private LessonsCategoryRepository lessonsCategoryRepository;
43  30        private LessonsQuestionRepository lessonsQuestionRepository;
44  31        private LessonsAnswerRepository lessonsAnswerRepository;
45        -     private ContractService contractService;
46        -     private ApplicationContext context;
47  32        private UserRepository userRepository;
    33    +     private MailingService mailingService;
    34    +
    35    +     @Autowired
    36    +     public void setMailingService(MailingService mailingService) {
    37    +         this.mailingService = mailingService;
    38    +     }
48  39
49  40        @Autowired
50  41        public void setUserRepository(UserRepository userRepository) {
          @@ -81,16 +72,6 @@ public void
          setLessonsAnswerRepository(LessonsAnswerRepository lessonsAnswerRepo
81  72            this.lessonsAnswerRepository = lessonsAnswerRepository;
82  73        }
83  74
84        -     @Autowired
85        -     public void setContractService(ContractService contractService) {
86        -         this.contractService = contractService;
87        -     }
88        -
89        -     @Autowired
90        -     public void setContext(ApplicationContext context) {
91        -         this.context = context;
92        -     }
93        -
94  75        @GetMapping("/api/exercises/{exerciseId}/lessons_categories")
95  76        @PreAuthorize("isExerciseObserver(#exerciseId)")
```

```
96  77      public Iterable<LessonsCategory> exerciseLessonsCategories(@PathVariable
            String exerciseId) {
        @@ -220,28 +201,9 @@ public void deleteExerciseLessonsQuestion(@PathVariable
            String lessonsQuestionId
220 201     public void sendExerciseLessons(@PathVariable String exerciseId, @Valid
            @RequestBody LessonsSendInput input) {
221 202         Exercise exercise =
            exerciseRepository.findById(exerciseId).orElseThrow();
222 203         List<LessonsCategory> lessonsCategories =
            lessonsCategoryRepository.findAll(LessonsCategorySpecification.fromExercise(exe
            rciseId)).stream().toList();
223     -         EmailContent emailContent = new EmailContent();
224     -         emailContent.setSubject(input.getSubject());
225     -         emailContent.setBody(input.getBody());
226     -         Inject inject = new Inject();
227     -         inject.setTitle("Lessons learned campaign");
228     -         inject.setDescription("Direct inject for lessons learned
            questionnaire");
229     -         inject.setContent(mapper.valueToTree(emailContent));
230     -         inject.setContract(EmailContract.EMAIL_DEFAULT);
231     -         inject.setUser(currentUser());
232     -         inject.setDirect(true);
233     -         Contract contract = contractService.resolveContract(inject);
234     -         if (contract == null) {
235     -             throw new UnsupportedOperationException("Unknown inject contract "
            + inject.getContract());
236     -         }
237     -         inject.setType(contract.getConfig().getType());
238     -         inject.setExercise(exercise);
239     -         List<ExecutionContext> userInjectContexts =
            lessonsCategories.stream().flatMap(lessonsCategory ->
            lessonsCategory.getAudiences().stream()
240     -                         .flatMap(audience ->
            audience.getUsers().stream())).distinct()
241     -                 .map(user -> new ExecutionContext(openExConfig, user, inject,
            "Direct execution")).toList();
242     -         ExecutableInject injection = new ExecutableInject(true, true, inject,
            contract, List.of(), userInjectContexts);
243     -         Injector executor = context.getBean(contract.getConfig().getType(),
            Injector.class);
```

```
244       -         executor.executeInjection(injection);
      204 +         List<User> users = lessonsCategories.stream().flatMap(lessonsCategory -
          > lessonsCategory.getAudiences().stream()
      205 +                 .flatMap(audience ->
          audience.getUsers().stream())).distinct().toList();
      206 +         mailingService.sendEmail(input.getSubject(), input.getBody(), users,
          Optional.of(exercise));
245   207         }
246   208
247   209         @GetMapping("/api/exercises/{exerciseId}/lessons_answers")
```

.../main/java/io/openex/rest/user/UserApi.java

```
          @@ -5,21 +5,26 @@
5     5     import io.openex.database.repository.OrganizationRepository;
6     6     import io.openex.database.repository.TagRepository;
7     7     import io.openex.database.repository.UserRepository;
      8   + import io.openex.rest.exception.InputValidationException;
8     9     import io.openex.rest.helper.RestBehavior;
9     10    import io.openex.rest.user.form.login.LoginUserInput;
      11  + import io.openex.rest.user.form.login.ResetUserInput;
      12  + import io.openex.rest.user.form.user.ChangePasswordInput;
10    13    import io.openex.rest.user.form.user.CreateUserInput;
11        - import io.openex.rest.user.form.user.UpdatePasswordInput;
12    14    import io.openex.rest.user.form.user.UpdateUserInput;
      15  + import io.openex.service.MailingService;
13    16    import io.openex.service.UserService;
      17  + import org.apache.commons.collections4.map.PassiveExpiringMap;
      18  + import org.apache.commons.lang3.RandomStringUtils;
14    19    import org.springframework.beans.factory.annotation.Autowired;
15        - import org.springframework.http.ResponseEntity;
16    20    import org.springframework.security.access.AccessDeniedException;
17    21    import org.springframework.web.bind.annotation.*;
18    22
19    23    import javax.annotation.Resource;
20    24    import javax.annotation.security.RolesAllowed;
21    25    import javax.transaction.Transactional;
22    26    import javax.validation.Valid;
      27  + import java.util.List;
23    28    import java.util.Optional;
```

```
24  29
25  30    import static io.openex.database.model.User.ROLE_ADMIN;
        @@ -28,14 +33,19 @@
28  33
29  34    @RestController
30  35    public class UserApi extends RestBehavior {
31      -
    36  +     PassiveExpiringMap<String, String> resetTokenMap = new PassiveExpiringMap<>
          (1000 * 60 * 10);
32  37        @Resource
33  38        private SessionManager sessionManager;
34      -
35  39        private OrganizationRepository organizationRepository;
36  40        private UserRepository userRepository;
37  41        private TagRepository tagRepository;
38  42        private UserService userService;
    43  +     private MailingService mailingService;
    44  +
    45  +     @Autowired
    46  +     public void setMailingService(MailingService mailingService) {
    47  +         this.mailingService = mailingService;
    48  +     }
39  49
40  50        @Autowired
41  51        public void setTagRepository(TagRepository tagRepository) {
        @@ -58,18 +68,67 @@ public void setUserRepository(UserRepository
        userRepository) {
58  68        }
59  69
60  70        @PostMapping("/api/login")
61      -     public ResponseEntity<User> login(@Valid @RequestBody LoginUserInput input)
          {
    71  +     public User login(@Valid @RequestBody LoginUserInput input) {
62  72            Optional<User> optionalUser =
          userRepository.findByEmail(input.getLogin());
63  73            if (optionalUser.isPresent()) {
64  74                User user = optionalUser.get();
65  75                if (userService.isUserPasswordValid(user, input.getPassword())) {
66  76                    userService.createUserSession(user);
67      -                 return ResponseEntity.ok().body(user);
```

```
                return user;
            }
        }
        throw new AccessDeniedException("Invalid credentials");
    }

    @PostMapping("/api/reset")
    public void passwordReset(@Valid @RequestBody ResetUserInput input) {
        Optional<User> optionalUser = userRepository.findByEmail(input.getLogin());
        if (optionalUser.isPresent()) {
            User user = optionalUser.get();
            String resetToken = RandomStringUtils.randomNumeric(8);
            String username = user.getName() != null ? user.getName() : user.getEmail();
            if ("fr".equals(input.getLang())) {
                String subject = resetToken + " est votre code de récupération de compte OpenEx";
                String body = "Bonjour " + username + ",</br>" +
                        "Nous avons reçu une demande de réinitialisation de votre mot de passe OpenEx.</br>" +
                        "Entrez le code de réinitialisation du mot de passe suivant : " + resetToken;
                mailingService.sendEmail(subject, body, List.of(user));
            } else {
                String subject = resetToken + " is your recovery code of your OpenEx account";
                String body = "Hi " + username + ",</br>" +
                        "A request has been made to reset your OpenEx password.</br>" +
                        "Enter the following password recovery code: " + resetToken;
                mailingService.sendEmail(subject, body, List.of(user));
            }
            // Store in memory reset token
            resetTokenMap.put(resetToken, user.getId());
        }
    }

    @PostMapping("/api/reset/{token}")
```

```
109 + public User changePasswordReset(@PathVariable String token, @Valid
    @RequestBody ChangePasswordInput input) throws InputValidationException {
110 +     String userId = resetTokenMap.get(token);
111 +     if (userId != null) {
112 +         String password = input.getPassword();
113 +         String passwordValidation = input.getPasswordValidation();
114 +         if (!passwordValidation.equals(password)) {
115 +             throw new InputValidationException("password_validation", "Bad
    password validation");
116 +         }
117 +         User changeUser = userRepository.findById(userId).orElseThrow();
118 +         changeUser.setPassword(userService.encodeUserPassword(password));
119 +         User savedUser = userRepository.save(changeUser);
120 +         resetTokenMap.remove(token);
121 +         return savedUser;
122 +     }
123 +     // Bad token or expired token
124 +     throw new AccessDeniedException("Invalid credentials");
125 + }
126 +
127 + @GetMapping("/api/reset/{token}")
128 + public boolean validatePasswordResetToken(@PathVariable String token) {
129 +     return resetTokenMap.get(token) != null;
130 + }
131 +
```

```
73  132     @RolesAllowed(ROLE_ADMIN)
74  133     @GetMapping("/api/users")
75  134     public Iterable<User> users() {
        @@ -79,7 +138,7 @@ public Iterable<User> users() {
79  138     @RolesAllowed(ROLE_ADMIN)
80  139     @PutMapping("/api/users/{userId}/password")
81  140     public User changePassword(@PathVariable String userId,
82      -                               @Valid @RequestBody UpdatePasswordInput input) {
        141 +                           @Valid @RequestBody ChangePasswordInput input) {
83  142         User user = userRepository.findById(userId).orElseThrow();
84  143         user.setPassword(userService.encodeUserPassword(input.getPassword()));
85  144         return userRepository.save(user);
```

...ex/rest/user/form/login/ResetUserInput.java

```
@@ -0,0 +1,29 @@
+ package io.openex.rest.user.form.login;
+
+ import javax.validation.constraints.NotBlank;
+
+ import static io.openex.config.AppConfig.MANDATORY_MESSAGE;
+
+ public class ResetUserInput {
+
+     @NotBlank(message = MANDATORY_MESSAGE)
+     private String login;
+
+     private String lang;
+
+     public String getLogin() {
+         return login;
+     }
+
+     public void setLogin(String login) {
+         this.login = login;
+     }
+
+     public String getLang() {
+         return lang;
+     }
+
+     public void setLang(String lang) {
+         this.lang = lang;
+     }
+ }
```

...est/user/form/user/ChangePasswordInput.java

```
@@ -0,0 +1,34 @@
+ package io.openex.rest.user.form.user;
+
+ import com.fasterxml.jackson.annotation.JsonProperty;
+
+ import javax.validation.constraints.NotBlank;
+
+ import static io.openex.config.AppConfig.MANDATORY_MESSAGE;
```

```
+
+ public class ChangePasswordInput {
+
+     @NotBlank(message = MANDATORY_MESSAGE)
+     @JsonProperty("password")
+     private String password;
+
+     @NotBlank(message = MANDATORY_MESSAGE)
+     @JsonProperty("password_validation")
+     private String passwordValidation;
+
+     public String getPassword() {
+         return password;
+     }
+
+     public void setPassword(String password) {
+         this.password = password;
+     }
+
+     public String getPasswordValidation() {
+         return passwordValidation;
+     }
+
+     public void setPasswordValidation(String passwordValidation) {
+         this.passwordValidation = passwordValidation;
+     }
+ }
```

.../java/io/openex/service/MailingService.java

```
@@ -0,0 +1,71 @@
+ package io.openex.service;
+
+ import com.fasterxml.jackson.databind.ObjectMapper;
+ import io.openex.config.OpenExConfig;
+ import io.openex.contract.Contract;
+ import io.openex.database.model.Exercise;
+ import io.openex.database.model.Inject;
+ import io.openex.database.model.User;
+ import io.openex.execution.ExecutableInject;
+ import io.openex.execution.ExecutionContext;
```

```
11 + import io.openex.execution.Injector;
12 + import io.openex.injects.email.EmailContract;
13 + import io.openex.injects.email.model.EmailContent;
14 + import org.springframework.beans.factory.annotation.Autowired;
15 + import org.springframework.context.ApplicationContext;
16 + import org.springframework.stereotype.Service;
17 +
18 + import javax.annotation.Resource;
19 + import java.util.List;
20 + import java.util.Optional;
21 +
22 + import static io.openex.helper.UserHelper.currentUser;
23 +
24 + @Service
25 + public class MailingService {
26 +
27 +     @Resource
28 +     protected ObjectMapper mapper;
29 +
30 +     @Resource
31 +     private OpenExConfig openExConfig;
32 +
33 +     private ContractService contractService;
34 +
35 +     private ApplicationContext context;
36 +
37 +     @Autowired
38 +     public void setContext(ApplicationContext context) {
39 +         this.context = context;
40 +     }
41 +
42 +     @Autowired
43 +     public void setContractService(ContractService contractService) {
44 +         this.contractService = contractService;
45 +     }
46 +
47 +     public void sendEmail(String subject, String body, List<User> users,
   Optional<Exercise> exercise) {
48 +         EmailContent emailContent = new EmailContent();
49 +         emailContent.setSubject(subject);
```

```
50  +         emailContent.setBody(body);
51  +         Inject inject = new Inject();
52  +         inject.setContent(mapper.valueToTree(emailContent));
53  +         inject.setContract(EmailContract.EMAIL_DEFAULT);
54  +         inject.setUser(currentUser());
55  +         Contract contract = contractService.resolveContract(inject);
56  +         if (contract == null) {
57  +             throw new UnsupportedOperationException("Unknown inject contract " +
      inject.getContract());
58  +         }
59  +         inject.setType(contract.getConfig().getType());
60  +         exercise.ifPresent(inject::setExercise);
61  +         List<ExecutionContext> userInjectContexts = users.stream().distinct()
62  +                 .map(user -> new ExecutionContext(openExConfig, user, inject,
      "Direct execution")).toList();
63  +         ExecutableInject injection = new ExecutableInject(false, true, inject,
      contract, List.of(), userInjectContexts);
64  +         Injector executor = context.getBean(contract.getConfig().getType(),
      Injector.class);
65  +         executor.executeInjection(injection);
66  +     }
67  +
68  +     public void sendEmail(String subject, String body, List<User> users) {
69  +         sendEmail(subject, body, users, Optional.empty());
70  +     }
71  + }
```

...i/src/main/resources/application.properties

```
@@ -104,6 +104,7 @@ logging.logback.rollingpolicy.max-history=7
104 104  #############
105 105
106 106  # Mail sending config (Always available, mandatory)
    107 + openex.default-mailer=no-reply@openex.io
107 108  spring.mail.host=smtp.mail.com
108 109  spring.mail.port=587
109 110  spring.mail.username=<username@mail.com>
```

...ain/java/io/openex/config/OpenExConfig.java

```
@@ -36,6 +36,9 @@ public class OpenExConfig {
36  36      @JsonProperty("auth_kerberos_enable")
37  37      private boolean authKerberosEnable;
38  38
    39  +   @JsonProperty("default_mailer")
    40  +   private String defaultMailer;
    41  +
39  42      @JsonIgnore
40  43      private String cookieName = "openex_token";
41  44
@@ -140,4 +143,12 @@ public boolean isAuthKerberosEnable() {
140 143     public void setAuthKerberosEnable(boolean authKerberosEnable) {
141 144         this.authKerberosEnable = authKerberosEnable;
142 145     }
    146 +
    147 +   public String getDefaultMailer() {
    148 +       return defaultMailer;
    149 +   }
    150 +
    151 +   public void setDefaultMailer(String defaultMailer) {
    152 +       this.defaultMailer = defaultMailer;
    153 +   }
143 154 }
```

...a/io/openex/execution/ExecutionContext.java

```
@@ -33,12 +33,14 @@ private ExecutionContext(User user, Exercise exercise,
List<String> audiences) {
33  33
34  34      public ExecutionContext(OpenExConfig config, User user, Injection injection,
List<String> audiences) {
35  35          this(user, injection.getExercise(), audiences);
36      -       String exerciseId = injection.getExercise().getId();
37      -       String queryParams = "?user=" + user.getId() + "&inject=" +
injection.getId();
38      -       this.put(PLAYER_URI, config.getBaseUrl() + "/private/" + exerciseId +
queryParams);
39      -       this.put(CHALLENGES_URI, config.getBaseUrl() + "/challenges/" +
exerciseId + queryParams);
```

```
40      -         this.put(SCOREBOARD_URI, config.getBaseUrl() + "/scoreboard/" +
          exerciseId + queryParams);
41      -         this.put(LESSONS_URI, config.getBaseUrl() + "/lessons/" + exerciseId +
          queryParams);
        36  +         if (injection.getExercise() != null) {
        37  +             String exerciseId = injection.getExercise().getId();
        38  +             String queryParams = "?user=" + user.getId() + "&inject=" +
              injection.getId();
        39  +             this.put(PLAYER_URI, config.getBaseUrl() + "/private/" + exerciseId
              + queryParams);
        40  +             this.put(CHALLENGES_URI, config.getBaseUrl() + "/challenges/" +
              exerciseId + queryParams);
        41  +             this.put(SCOREBOARD_URI, config.getBaseUrl() + "/scoreboard/" +
              exerciseId + queryParams);
        42  +             this.put(LESSONS_URI, config.getBaseUrl() + "/lessons/" + exerciseId
              + queryParams);
        43  +         }
42      44          }
43      45
44      46          public ExecutionContext(User user, Exercise exercise, String audience) {
```

openex-front/src/actions/Application.js

```
            @@ -20,6 +20,29 @@ export const updateParameters = (data) => (dispatch) => {
20      20        )(dispatch);
21      21    };
22      22
        23  + export const askReset = (username, locale) => (dispatch) => {
        24  +   const data = { login: username, lang: locale };
        25  +   return postReferential(schema.user, '/api/reset', data)(dispatch);
        26  + };
        27  +
        28  + export const resetPassword = (token, values) => (dispatch) => {
        29  +   const data = { password: values.password, password_validation:
              values.password_validation };
        30  +   const ref = postReferential(schema.user, `/api/reset/${token}`, data)
              (dispatch);
        31  +   return ref.then((finalData) => {
        32  +     if (finalData[FORM_ERROR]) {
        33  +       return finalData;
```

```
      34 +     }
      35 +     return dispatch({
      36 +       type: Constants.IDENTITY_LOGIN_SUCCESS,
      37 +       payload: finalData,
      38 +     });
      39 +   });
      40 + };
      41 +
      42 + export const validateResetToken = (token) => (dispatch) => {
      43 +   return getReferential(null, `/api/reset/${token}`)(dispatch);
      44 + };
      45 +
23    46   export const askToken = (username, password) => (dispatch) => {
24    47     const data = { login: username, password };
25    48     const ref = postReferential(schema.user, '/api/login', data)(dispatch);
```

openex-front/src/components/i18n.js

```
@@ -275,6 +275,7 @@ export const useFormatter = () => {
275   275     };
276   276     return {
277   277       t: translate,
      278 +     locale: intl.locale ?? intl.defaultLocale,
278   279       tPick: (label) => (label ? label[intl.locale] ?? label[intl.defaultLocale]
          : ''),
279   280       n: formatNumber,
280   281       b: formatBytes,
```

openex-front/src/public/Index.js

```
@@ -8,6 +8,7 @@ import { errorWrapper } from '../components/Error';
8     8   import Media from './components/medias/Media';
9     9   import Challenges from './components/challenges/Challenges';
10    10  import Lessons from './components/lessons/Lessons';
      11 + import Reset from './components/login/Reset';
11    12
12    13  const useStyles = makeStyles((theme) => ({
13    14    root: {
@@ -37,19 +38,10 @@ const Index = () => {
```

```
37  38        <div className={classes.root}>
38  39          <main className={classes.content}>
39  40            <Switch>
40      -           <Route
41      -             exact
42      -             path="/comcheck/:statusId"
43      -             render={errorWrapper(Comcheck)}
44      -           />
45      -           <Route
46      -             path="/medias/:exerciseId/:mediaId"
47      -             render={errorWrapper(Media)}
48      -           />
49      -           <Route
50      -             path="/challenges/:exerciseId"
51      -             render={errorWrapper(Challenges)}
52      -           />
        41  +           <Route exact path="/comcheck/:statusId" render=
              {errorWrapper(Comcheck)}/>
        42  +           <Route exact path="/reset" render={errorWrapper(Reset)}/>
        43  +           <Route path="/medias/:exerciseId/:mediaId" render=
              {errorWrapper(Media)}/>
        44  +           <Route path="/challenges/:exerciseId" render=
              {errorWrapper(Challenges)}/>
53  45            <Route path="/lessons/:exerciseId" render={errorWrapper(Lessons)} />
54  46            <Route component={Login} />
55  47          </Switch>
```

...-front/src/public/components/login/Login.js

```
    @@ -15,6 +15,7 @@ import {
15  15  import LoginForm from './LoginForm';
16  16  import inject18n from '../../../components/i18n';
17  17  import { storeHelper } from '../../../actions/Schema';
    18  + import Reset from './Reset';
18  19
19  20  const styles = () => ({
20  21    container: {
    @@ -36,13 +37,9 @@ const Login = (props) => {
36  37    const { classes, parameters, t } = props;
```

```
37  38    const { auth_openid_enable: isOpenId, auth_local_enable: isLocal } =
          parameters;
38  39    const { platform_providers: providers } = parameters;
39      - const [dimension, setDimension] = useState({
40      -   width: window.innerWidth,
41      -   height: window.innerHeight,
42      - });
43      - const updateWindowDimensions = () => {
44      -   setDimension({ width: window.innerWidth, height: window.innerHeight });
45      - };
        40 + const [reset, setReset] = useState(false);
        41 + const [dimension, setDimension] = useState({ width: window.innerWidth, height:
          window.innerHeight });
        42 + const updateWindowDimensions = () => setDimension({ width: window.innerWidth,
          height: window.innerHeight });
46  43    useEffect(() => {
47  44      window.addEventListener('resize', updateWindowDimensions);
48  45      return () => window.removeEventListener('resize', updateWindowDimensions);
        @@ -62,11 +59,15 @@ const Login = (props) => {
62  59    return (
63  60      <div className={classes.container} style={{ marginTop }}>
64  61        <img src={`/${logo}`} alt="logo" className={classes.logo} />
65      -       {isLocal && (
        62 +       {isLocal && !reset && (
66  63          <Paper variant="outlined">
67  64            <LoginForm onSubmit={onSubmit} />
        65 +          <div style={{ marginBottom: 10 }}>
        66 +            <a onClick={() => setReset(true)}>{t('I forgot my password')}</a>
        67 +          </div>
68  68          </Paper>
69  69        )}
        70 +       {isLocal && reset && <Reset onCancel={() => setReset(false)}/>}
70  71        {isOpenId
71  72          && (providers ?? []).map((provider) => (
72  73            <div key={provider.provider_name}>
```

…-front/src/public/components/login/Reset.js

```
        @@ -0,0 +1,121 @@
    1 + import React, { useState } from 'react';
```

```
+ import { useDispatch } from 'react-redux';
+ import Paper from '@mui/material/Paper';
+ import Button from '@mui/material/Button';
+ import { Form } from 'react-final-form';
+ import { makeStyles } from '@mui/styles';
+ import { askReset, resetPassword, validateResetToken } from
  '../../../actions/Application';
+ import { useFormatter } from '../../../components/i18n';
+ import { TextField } from '../../../components/TextField';
+
+ const useStyles = makeStyles(() => ({
+   container: {
+     textAlign: 'center',
+     margin: '0 auto',
+     width: 400,
+   },
+   appBar: {
+     borderTopLeftRadius: '10px',
+     borderTopRightRadius: '10px',
+   },
+   logo: {
+     width: 200,
+     margin: '0px 0px 50px 0px',
+   },
+ }));
+
+ const validateFields = (t, values, requiredFields) => {
+   const errors = {};
+   requiredFields.forEach((field) => {
+     if (!values[field]) {
+       errors[field] = t('This field is required.');
+     }
+   });
+   return errors;
+ };
+
+ const STEP_ASK_RESET = 'ask';
+ const STEP_VALIDATE_TOKEN = 'validate';
+ const STEP_RESET_PASSWORD = 'reset';
+ const Reset = ({ onCancel }) => {
```

```
41 +   const classes = useStyles();
42 +   const { t, locale } = useFormatter();
43 +   const dispatch = useDispatch();
44 +   const [step, setStep] = useState(STEP_ASK_RESET);
45 +   const [token, setToken] = useState();
46 +   const onSubmitAskToken = (data) => {
47 +     dispatch(askReset(data.username, locale)).then(() => {
48 +       setStep(STEP_VALIDATE_TOKEN);
49 +     });
50 +   };
51 +   const onSubmitValidateToken = (data) => {
52 +     dispatch(validateResetToken(data.code)).then((response) => {
53 +       if (response) {
54 +         setToken(data.code);
55 +         setStep(STEP_RESET_PASSWORD);
56 +       }
57 +     });
58 +   };
59 +   const onSubmitValidatePassword = (data) => dispatch(resetPassword(token,
    data));
60 +   return (
61 +     <div className={classes.container}>
62 +       <Paper variant="outlined">
63 +         <div style={{ padding: 15 }}>
64 +           {step === STEP_ASK_RESET && <Form onSubmit={onSubmitAskToken}
65 +                                       validate={(values) => validateFields(t,
    values, ['username'])}>
66 +             {({ handleSubmit, submitting, pristine }) => (
67 +                 <form onSubmit={handleSubmit}>
68 +                   <TextField name="username" type="text"
69 +                       variant="standard" label={t('Email address')}
70 +                       fullWidth={true} style={{ marginTop: 5 }}/>
71 +                   <Button type="submit" variant="contained"
72 +                       disabled={pristine || submitting} onClick={handleSubmit}
73 +                       style={{ marginTop: 30 }}>
74 +                     {t('Send reset code')}
75 +                   </Button>
76 +                 </form>
77 +             )}
78 +           </Form>}
```

```
79  +           {step === STEP_VALIDATE_TOKEN && <Form onSubmit=
      {onSubmitValidateToken}
80  +                                           validate={(values) => validateFields(t,
      values, ['code'])}>
81  +             {({ handleSubmit, submitting, pristine }) => (
82  +                 <form onSubmit={handleSubmit}>
83  +                   <TextField name="code" type="text"
84  +                              variant="standard" label={t('Enter code')}
85  +                              fullWidth={true} style={{ marginTop: 5 }}/>
86  +                   <Button type="submit" variant="contained"
87  +                           disabled={pristine || submitting} onClick=
      {handleSubmit}
88  +                           style={{ marginTop: 30 }}>
89  +                     {t('Continue')}
90  +                   </Button>
91  +                 </form>
92  +             )}
93  +           </Form>}
94  +           {step === STEP_RESET_PASSWORD && <Form onSubmit=
      {onSubmitValidatePassword}
95  +                                                validate={(values) =>
      validateFields(t, values, ['password', 'password_validation'])}>
96  +             {({ handleSubmit, submitting, pristine }) => (
97  +                 <form onSubmit={handleSubmit}>
98  +                   <TextField name="password" type="password"
99  +                              variant="standard" label={t('Password')}
100 +                              fullWidth={true} style={{ marginTop: 5 }}/>
101 +                   <TextField name="password_validation" type="password"
102 +                              variant="standard" label={t('Password
      validation')}
103 +                              fullWidth={true} style={{ marginTop: 5 }}/>
104 +                   <Button type="submit" variant="contained"
105 +                           disabled={pristine || submitting} onClick=
      {handleSubmit}
106 +                           style={{ marginTop: 30 }}>
107 +                     {t('Change your password')}
108 +                   </Button>
109 +                 </form>
110 +             )}
111 +           </Form>}
```

```
112 +           <div style={{ marginTop: 10 }}>
113 +             <a onClick={() => onCancel()}>{t('Back to login')}</a>
114 +           </div>
115 +         </div>
116 +       </Paper>
117 +     </div>
118 +   );
119 + };
120 +
121 + export default Reset;
```

openex-front/src/resources/css/main.css

```
@@ -14,6 +14,7 @@ a,
14 14   a:hover,
15 15   a:visited,
16 16   a:focus {
17 +   cursor: pointer;
17 18     text-decoration: none;
18 19   }
19 20
```

openex-front/src/utils/Localization.js

```
@@ -6,7 +6,14 @@ const i18n = {
6  6            'OpenEx - Plateforme d’exercices de crise',
7  7          'Email address': 'Adresse email',
8  8          Password: 'Mot de passe',
9 +        'Password validation': 'Validation du mot de passe',
10 +       'Change your password': 'Changer votre mot de passe',
11 +       'I forgot my password': 'J\'ai oublié mon mot de passe',
12 +       'Send reset code': 'Envoyer le code',
13 +       'Enter code': 'Entrer le code',
14 +       'Back to login': 'Retour à l\'identification',
9  15         'Sign in': "S'identifier",
16 +       Continue: 'Continuer',
10 17         Dashboard: 'Tableau de bord',
11 18         Exercises: 'Exercices',
12 19         Players: 'Joueurs',
```

openex-model/pom.xml

```
@@ -34,6 +34,12 @@
34 34                <artifactId>spring-boot-starter-security</artifactId>
35 35                <version>${spring.version}</version>
36 36            </dependency>
   37 +         <dependency>
   38 +             <groupId>org.springframework.boot</groupId>
   39 +             <artifactId>spring-boot-configuration-processor</artifactId>
   40 +             <version>${spring.version}</version>
   41 +             <optional>true</optional>
   42 +         </dependency>
37 43            <dependency>
38 44                <groupId>org.springframework.boot</groupId>
39 45                <artifactId>spring-boot-starter-oauth2-client</artifactId>
```

...java/io/openex/database/model/Exercise.java

```
@@ -77,7 +77,7 @@ public enum STATUS {
77 77
78 78        @Column(name = "exercise_mail_from")
79 79        @JsonProperty("exercise_mail_from")
80    -     private String replyTo = "planners@openex.io";
   80 +     private String replyTo;
81 81
82 82        @ManyToOne(fetch = FetchType.LAZY)
83 83        @JoinColumn(name = "exercise_logo_dark")
```

...n/java/io/openex/database/model/Inject.java

```
@@ -147,9 +147,6 @@ public class Inject implements Base, Injection {
147 147         private List<InjectExpectation> expectations = new ArrayList<>();
148 148
149 149         // region transient
150    -     @Transient
151    -     private boolean direct = false;
152    -
153 150         @Transient
154 151         public String getHeader() {
```

```
155  152          return ofNullable(getExercise()).map(Exercise::getHeader).orElse("");
     @@ -248,14 +245,6 @@ public void setEnabled(boolean enabled) {
248  245          this.enabled = enabled;
249  246      }
250  247
251      -    public boolean isDirect() {
252      -        return direct;
253      -    }
254      -
255      -    public void setDirect(boolean direct) {
256      -        this.direct = direct;
257      -    }
258      -
259  248      public Instant getCreatedAt() {
260  249          return createdAt;
261  250      }
```

Comments 0

Please sign in to comment.