

SportTrack

By Marek Szkutnik

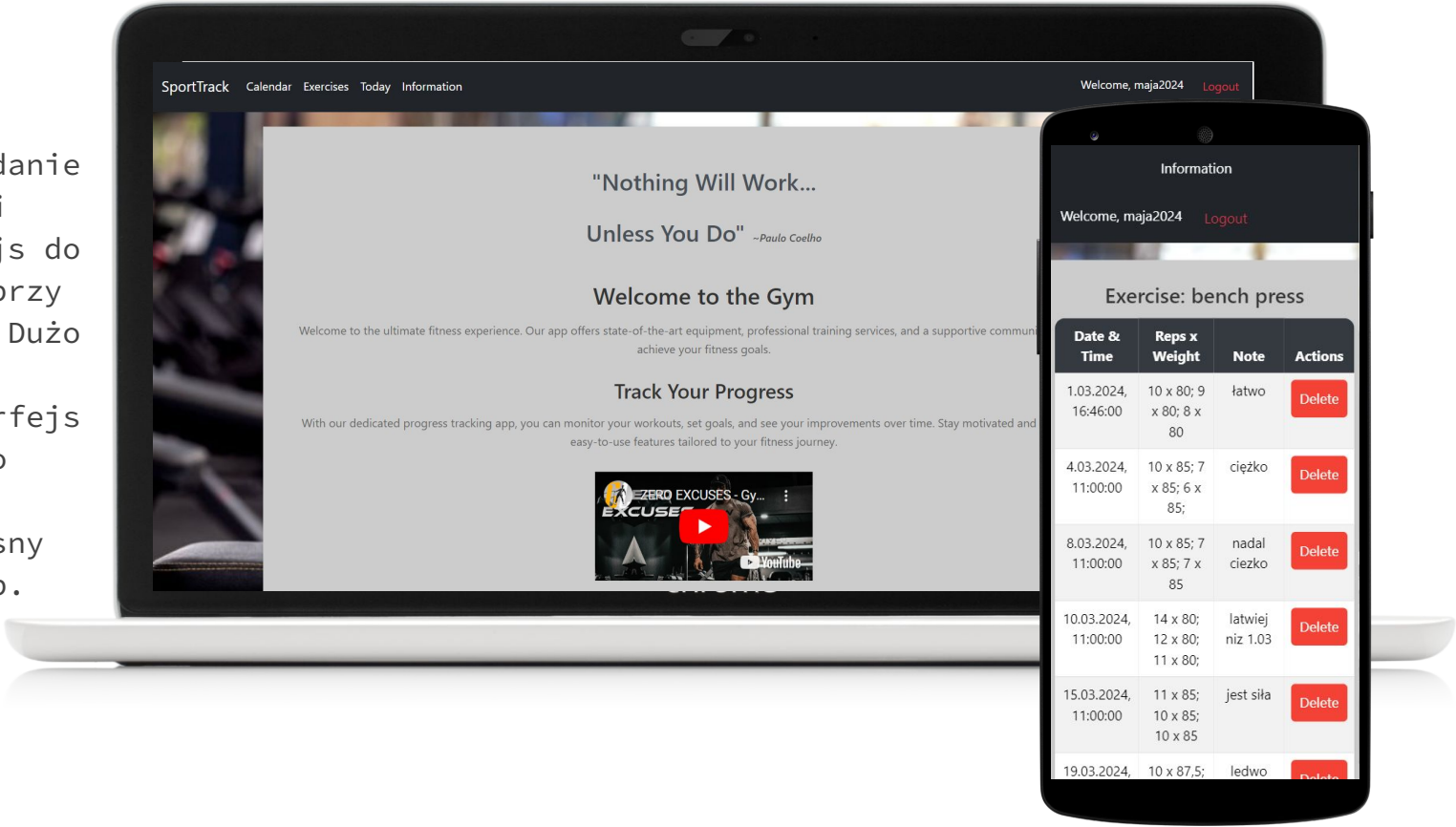
0 aplikaciji od frontu

SportTrack

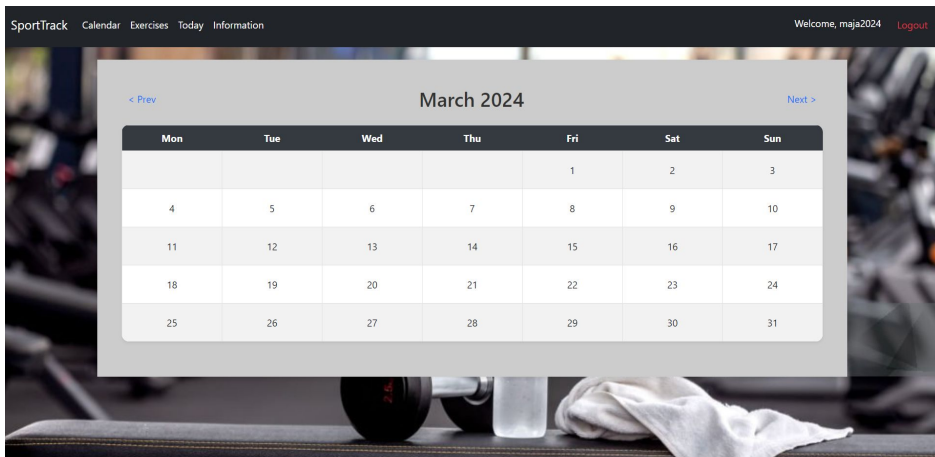
aplikacja do śledzenia postępu przy ćwiczeniach

— — —

Aplikacja ma za zadanie dostarczać prosty i intuicyjny interfejs do śledzenia postępu przy swoich treningach. Dużo aplikacji ma skomplikowany interfejs i nie zostawia dużo miejsca na swobodę użytkownika we własny indywidualny sposób.

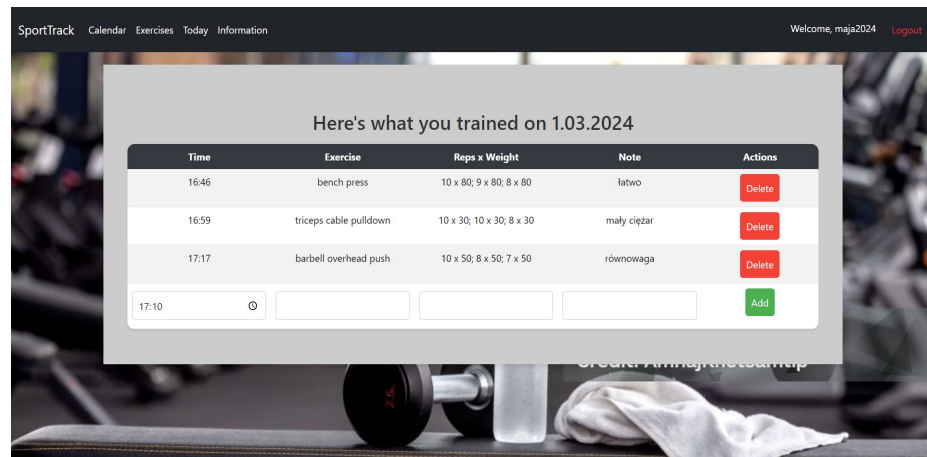


Kalendarz



Kalendarz pozwala nam wybrać konkretny dzień w konkretnym miesiącu

Ćwiczenia w danym dniu



Ta strona pozwala nam przeglądać dodawać oraz usuwać treningi wykonane w konkretnym dniu

Treningi na ćwiczenie

SportTrack Calendar Exercises Today Information Welcome, maja2024 Logout

Exercise: bench press

Date & Time	Reps x Weight	Note	Actions
1.03.2024, 16:46:00	10 x 80; 9 x 80; 8 x 80	łatwo	Delete
4.03.2024, 11:00:00	10 x 85; 7 x 85; 6 x 85;	ciężko	Delete
8.03.2024, 11:00:00	10 x 85; 7 x 85; 7 x 85	nadal ciężko	Delete
10.03.2024, 11:00:00	14 x 80; 12 x 80; 11 x 80;	łatwiej niż 1.03	Delete
15.03.2024, 11:00:00	11 x 85; 10 x 85; 10 x 85	jest siła	Delete
19.03.2024, 11:00:00	10 x 87,5; 8 x 87,5; 7 x 87,5	ledwo ledwo	Delete
2024.03.18			Add

Kolejna strona z tabelą, ta pokazuje nam każdy raz gdy wykonywaliśmy dane ćwiczenie. Świetna do śledzenia progresu. Możemy do niej przejść z treningów na dzień albo z wszystkich ćwiczeń

Wszystkie ćwiczenia

SportTrack Calendar Exercises Today Information Welcome, maja2024 Logout

Here are all exercises you saved

#	Exercise:
1	bench press
2	triceps cable pulldown
3	barbell overhead push
4	squats
5	wzmacnianie kolan

Ta strona wyświetla, każde unikalne ćwiczenia jakie sami zapisaliśmy do aplikacji.

0 aplikacji od zalecanych

Technologie



Java
Spring

React
JavaScript

HTML,
CSS

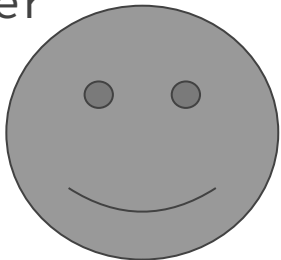
Spring

— — —

Aplikacja Springowa wystawia na zewnątrz Api komunikujące się z bazą danych. Obsługuje kilka prostych zapytań związanych z logiką treningów. Możemy je dodawać, usuwać pobierać – wszystkie, w zależności od dnia, w zależności od ćwiczenia.

```
17 @RestController
18 @AllArgsConstructor
19 public class TrainingController {
20
21     private final TrainingFacade trainingsFacade;
22
23     @GetMapping("/trainings/ForDay")
24     public ResponseEntity<List<TrainingResponseDto>> getTrainingsForUserOnDay(
25         @RequestParam int year,
26         @RequestParam int month,
27         @RequestParam int day) {...}
28
29     @GetMapping("/trainings/ForExercise")
30     public ResponseEntity<List<TrainingResponseDto>> getTrainingsForUserOnExercise(
31         @RequestParam String exercise) {...}
32
33     @PostMapping("/trainings/add")
34     public ResponseEntity<TrainingResponseDto> addOneTraining(@RequestBody @Valid AddTrainingRequestDto addTrainingRequest) {...}
35
36     @DeleteMapping("/trainings/delete")
37     public void deleteOneTraining(@RequestBody @Valid DeleteTrainingRequestDto deleteTrainingRequest) {...}
38
39     @GetMapping("/trainings/user/exercises")
40     public List<String> getUsersExercises(){...}
41 }
```


User



GET /trainings/ForDay?year=YYYY&month=MM&day=DD
GET /trainings/ForExercise?exercise=XXXXX
POST /trainings/add
DELETE /trainings/delete
GET @/trainings/user/exercises

getTrainingsForUserOnDay()
getTrainingsForUserOnExercise()
getDistinctExercisesForUser()
addTraining()
deleteTraining()

TrainingFacde

Spring Security

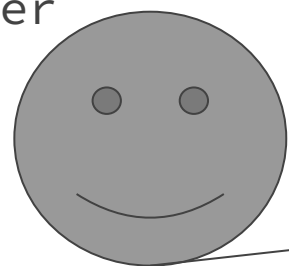
— — —

Aplikacja Springowa wystawia również endpointy służące logowaniu oraz rejestracji. Po udanej rejestracji użytkownik może się zalogować co zwróci mu wygenerowany token który następnie będzie służył do weryfikacji użytkownika.

```
© LoginController.java x
@RestController
@AllArgsConstructor
15
16 public class LoginController {
17     JwtAuthenticator jwtAuthenticator;
18     @ kaczko
19     @PostMapping("/token")
20     ResponseEntity<LoginResponseDto> authenticateAndGenerateToken(@RequestBody @Valid Login
21         LoginResponseDto loginResponseDto = jwtAuthenticator.authenticateAndGenerateToken(t
22     return ResponseEntity.ok(loginResponseDto);
23 }
24 }
25

© RegisterController.java x
18 @RestController
19 @AllArgsConstructor
20 public class RegisterController {
21     LoginAndRegisterFacade loginAndRegisterFacade;
22     private final PasswordEncoder bCryptPasswordEncoder;
23     @ kaczko
24     @PostMapping("/register")
25     ResponseEntity<RegisterResponseDto> RegisterUser(@RequestBody @Valid RegisterRequestDt
26         RegisterRequestDto userEncodedRequestDto = RegisterRequestDto.builder()
27             .password(bCryptPasswordEncoder.encode(userRequestDto.password()))
28             .username(userRequestDto.username())
29             .build();
30     UserResultDto registerResult = loginAndRegisterFacade.register(userEncodedRequestD
31     return ResponseEntity.status(HttpStatus.CREATED).body(new RegisterResponseDto(reg
```

User



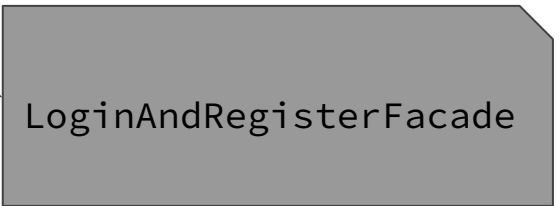
POST /token



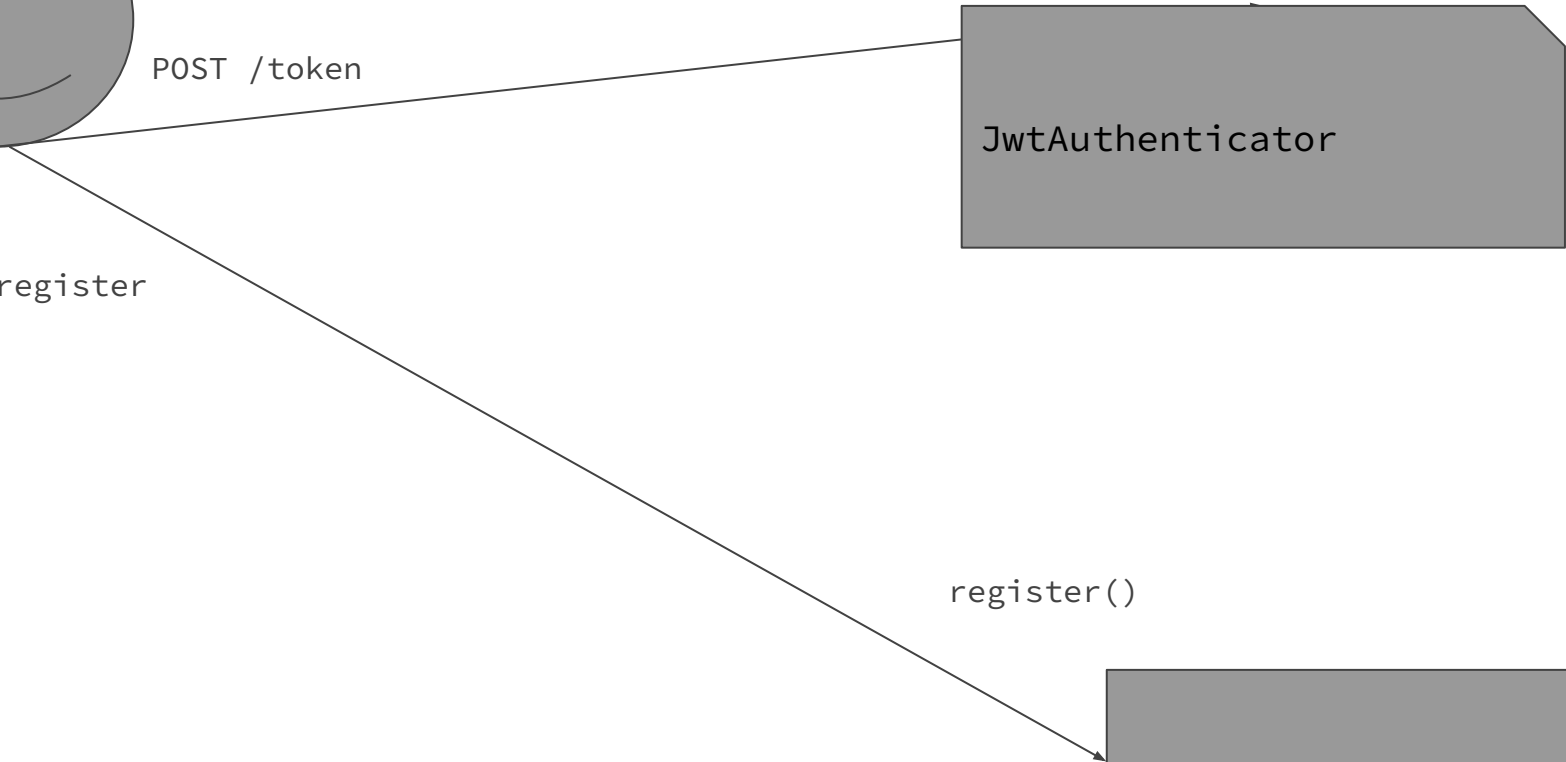
JwtAuthenticator

POST /register

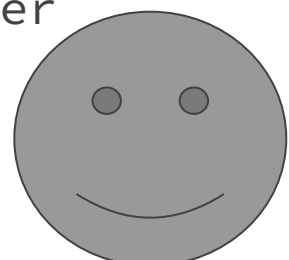
register()



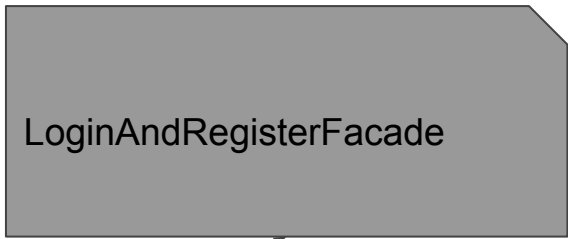
LoginAndRegisterFacade



User



GET /trainings/ForDay?year=YYYY&month=MM&day=DD
GET /trainings/ForExercise?exercise=XXXXX
POST /trainings/add
DELETE /trainings/delete
GET @/trainings/user/exercises



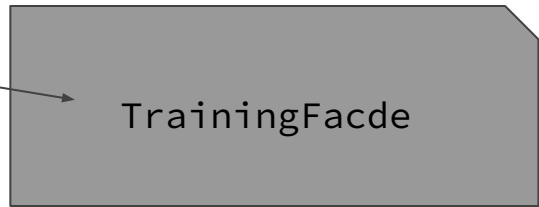
LoginAndRegisterFacade

findByUsername()



JwtAuthTokenFilter

getTrainingsForUserOnDay()
getTrainingsForUserOnExercise()
getDistinctExercisesForUser()
addTraining()
deleteTraining()



TrainingFacde

