Calories Calculation App

Përmbledhje e Arkitekturës

1. Stili i Arkitektures

Aplikacioni është ndërtuar duke përdorur modelin **MVC (Model-View-Controller)**.Eshte perdorur arkitekturë e shtresezuar për të siguruar ndarje të qartë të secilit funksionalitet dhe mirëmbajtshmëri. Projekti gjithashtu integron **Spring Security** për autentifikim dhe autorizim, duke e bere me të sigurt.

---

2. Struktura e folders dhe funksionaliteti qe mbeshtesin

2.1. com.food.controller

- **Qëllimi:** Menaxhon HTTP request dhe i drejton ato tek services ose view e duhura.
- **Komponentët Kryesore:**
  - `ThymeLeafController.java`: Menaxhon GET dhe POST request për view e bazuara në Thymeleaf.
  - Përgjegjës për të renderuar view duke përdorur të dhëna dinamike dhe për të procesuar form submissions.

2.2. com.food.entity

- **Qëllimi:** Përfaqëson modelin e të dhënave të aplikacionit dhe skemën e databazës.
- **Komponentët Kryesore:**
  - `Food.java`: Entitet që përfaqëson artikujt ushqimorë, me fusha si emri, kaloritë, cmimi dhe koha e regjistrimit.
  - `User.java`: Entitet që përfaqëson përdoruesit, duke përshirë detaje si emri i përdoruesit, fjalëkalimi dhe email,roli.
- Këto klasa janë të shënuara me `@Entity` dhe përmbajnë lidhje si `@OneToMany` ose `@ManyToOne`.

2.3. com.food.exception

- **Qëllimi:** Menaxhimi i gabimeve për aplikacionin.
- **Karakteristikat Kryesore:**
  - Përcakton exceptions të personalizuara si `ResourceNotFoundException` ose `AppException,etj`.
  - Ofron feedback të kuptueshëm mbi gabimet.

2.4. com.food.repository

- **Qëllimi:** Ndërvepron me databazën duke përdorur Spring Data JPA.
- **Komponentët Kryesore:**
  - `FoodRepository.java`: Menaxhon operacionet CRUD për entitetet `Food`.
  - `UserRepository.java`: Menaxhon operacionet e databazës për entitetet `User`.
- Siguron një urë midis shtresës së shërbimeve dhe databazës.

2.5. com.food.request.payload

- **Qëllimi:** Përcakton Data Transfer Objects (DTOs) për kerkesat.
- **Komponentët Kryesore:**
  - `FoodDto.java`: Merr të dhënat e artikujve ushqimorë të dërguara nga përdoruesit.
  - `LoginDto.java`: Menaxhon payload për login me username dhe password.
- `UserDto.java`: Menaxhon të dhënat për regjistrimin e përdoruesve.

2.6. com.food.response.payload

- **Qëllimi:** Strukturon të dhënat e kthyer te klienti në format të konstruktuar.
- **Komponentët Kryesore:**
  - `FoodCaloriesResponseDto.java`: Përfaqëson dhe kthen te dhena ne lidhje me konsumin e ushqimit per nje user te caktuar.
  - `FoodConsumptionDto.java`: Gjurmon food entries dhe te dhena permbledhese ne pergjithesi(mujor dhe ditor), jo vetem per nje user specifik.
  - `WeeklyFoodReportResponseDto.java`: Përmbledh konsumin ushqimor javor.
  - `PriceLimitReachedResponseDto.java`: Jep informacion mbi perdoruesit qe kane tejkaluar limit e shpenzimit mujor.
  - `GenericMessage.java`: Dërgon mesazhe standarde për sukses ose error.
  - `UserResponseDto.java`:Perfaqeson dhe kthen te dhena mbi kredencialet e perdoruesit.

2.7. com.food.security.service

- **Qëllimi:** Menaxhon proceset e autentifikimit dhe autorizimit duke përdorur Spring Security.
- **Komponentët Kryesore:**
  - `UserDetailsImpl.java`: Implementon `UserDetails` për të siguruar informacion specifik për sigurinë e përdoruesve.
  - Integron `UserRepository` për të ngarkuar kredencialet e përdoruesve gjatë autentifikimit.

2.8. com.food.service

- **Qëllimi:** Përmban logjikën e biznesit të aplikacionit.
- **Komponentët Kryesore:**
  - `AuthService.java`: Menaxhon login, regjistrimin dhe detyrat e autentifikimit të përdoruesve.
  - `FoodService.java`: Menaxhon operacionet e lidhura me artikujt ushqimorë, gjurmimin e kalorive dhe analizën e konsumit dhe shpenzimit.
  - `UserService.java`: Menaxhon operacionet specifike për përdoruesit.
- Services veprojnë si ndërfaqe midis controllers dhe repositories.

---

3. Karakteristikat Kryesore

3.1. Model

- Enkapsulon të dhënat e aplikacionit dhe përfshin klasat e entiteteve (`com.food.entity`) dhe repositories (`com.food.repository`).
- Ndërvepron drejtpërdrejt me databazën duke përdorur Spring Data JPA.

3.2. View

- Implementuar duke përdorur **Thymeleaf** për rendering në server.
- Thymeleaf templates renderojnë faqe HTML dinamike me të dhëna nga controllers.
- Mbështet form submissions për inputet e përdoruesve, si shtimi i food entries ose perditesimi i te dhenave qe shfaqen.

3.3. Controller

- Menaxhon interaktivitetin e përdoruesit dhe drejton request tek services.
- Proceson GET requests për të marrë të dhëna për view dhe POST requests për form submissions.
- Kthen përgjigje në formën e pamjeve të renderuara.

3.4. Security

- Integron **Spring Security** për autentifikimin e përdoruesve dhe kontrollin e aksesit bazuar në role.
- Implementon `UserDetailsService` për autentifikimin e përdoruesve kundrejt kredencialeve të ruajtura.
- Mbron endpoints sensitive.