**Titel**: Restaurant Tablet Voedsel Bestel App (Java)

**Technische Vereisten**: Spring Boot Java-app, MySQL-database.

**Volledige Opdracht Omschrijving**: We willen een applicatie ontwikkelen voor tablets in restaurants waarmee klanten eten en drinken naar hun tafel kunnen bestellen. De app moet gebruiksvriendelijk en gemakkelijk te gebruiken zijn. De app moet eerst worden geactiveerd met een werknemersinlogcode. De app gaat honderden gerechten en tientallen drankjes bevatten, dus er moeten een aantal filterfuncties zijn inbegrepen.

**Overzicht Functionele Vereisten:**

1. Maak een Spring Boot Java applicatie met een MySQL-database.

2. Maak een inlogscherm waar werknemers een code moeten invoeren om de app te activeren.

3. Maak een gebruikersinterface voor het bestellen van eten en drinken.

4. Voeg filterfuncties toe aan de app, zodat gebruikers gemakkelijk kunnen zoeken naar gerechten en drankjes.

**Vereisten:**

1. Java - U moet een goede kennis hebben van Java en het Spring Boot framework.

2. MySQL - U moet een goede kennis hebben van MySQL-databases.

3. UI/UX - U moet een goede kennis hebben van gebruikersinterfaces en gebruikerservaringen.

**Stappen:**

1. Installeer en configureer het Spring Boot framework en maak een nieuwe Spring Boot-applicatie.

2. Maak de MySQL-database en configureer deze in de Spring Boot-applicatie.

3. Ontwerp en implementeer het inlogscherm voor werknemers.

4. Ontwerp en implementeer de gebruikersinterface voor het bestellen van eten en drinken.

5. Implementeer de filterfuncties in de app.

6. Test de app grondig om er zeker van te zijn dat deze correct werkt.  
  
zie “Uitgebreide Functionele Vereisten” voor nadere details.

**Referenties:**

- Spring Boot Documentatie: https://spring.io/projects/spring-boot

- MySQL Documentatie: https://dev.mysql.com/doc/

**Uitgebreide Functionele Vereisten:**

**1. Werknemers Authenticatie:**

- Implementatie: Gebruik Spring Security voor het authenticatieproces. Dit kan worden bereikt door een aangepaste `UserDetailsService` te implementeren die werknemerscodes controleert tegen een database. Je moet een tabel in de MySQL-database maken voor werknemers met hun unieke codes en rollen.

**2. Tafelnummer Toewijzing door Werknemers:**

- Implementatie: Voeg een REST API endpoint toe die werknemers toestaat om tafelnummers toe te wijzen aan klanten. Dit kan worden gedaan door een `TableAssignmentController` te maken met een methode die een POST-verzoek afhandelt, waarbij het tafelnummer en de klantinformatie als parameters worden doorgegeven.

**3. Menu Weergave:**

- Implementatie: Maak een `Menu` entiteit en een bijbehorende `MenuRepository` om menu-items in de database op te slaan en op te halen. Gebruik Spring Data JPA voor database-interacties. Voor de weergave kun je een `MenuController` maken die een GET-verzoek afhandelt en de lijst met menu-items retourneert.

**4. Bestellen van Eten en Drinken:**

- Implementatie: Implementeer een bestelformulier op de frontend en een `OrderController` op de backend die bestellingen afhandelt. Gebruik een POST-verzoek om bestelgegevens van de klant naar de server te sturen. Sla de bestelling op in de database met een `Order` entiteit en een bijbehorende `OrderRepository`.

**5. Filterfuncties:**

- Implementatie: Voeg filterlogica toe in de `MenuController` om menu-items te filteren op basis van criteria zoals categorie, prijs, enz. Je kunt Spring Data JPA specificaties gebruiken of aangepaste query-methoden in de `MenuRepository` om dit te ondersteunen.

**6. Besteloverzicht:**

- Implementatie: Na het plaatsen van een bestelling, stuur de klant naar een overzichtspagina die de details van hun bestelling toont. Dit vereist het opslaan van de bestelgegevens in de sessie of database en het ophalen ervan voor weergave. Implementeer een `OrderSummaryController` die de bestelgegevens retourneert voor weergave.

**7. Bestelgeschiedenis:**

- Implementatie: Werknemers moeten toegang hebben tot een pagina of sectie in de app die alle geplaatste bestellingen toont. Dit kan worden bereikt door een `OrderHistoryController` te implementeren die een lijst van alle bestellingen ophaalt uit de database en deze toont.

**8. Tafelgeschiedenis:**

- Implementatie: Bewaar informatie over klantbezetting en bestellingen per tafel in de database. Implementeer een `TableHistoryController` die gegevens ophaalt over wanneer klanten aan een tafel zaten en wat ze hebben besteld, en toon deze informatie aan werknemers.

**9. Afhandelen van Tafels en Bestellingen:**

- Implementatie: Voeg functionaliteit toe voor werknemers om aan te geven dat een bestelling is afgehandeld en dat een tafel is vrijgegeven. Dit kan worden geïmplementeerd door statusvelden toe te voegen aan de `Order` en `Table` entiteiten en door een methode in de respectievelijke controllers om de status bij te werken.

**Technische Implementatie Details:**

- Gebruik Spring MVC voor het afhandelen van webverzoeken en Spring Data JPA voor databasetoegang.

- Implementeer RESTful services waar nodig, vooral voor interacties tussen de frontend en de backend.

- Gebruik bijvoorbeeld Thymeleaf of Angular/React voor de frontend, afhankelijk van de vereisten voor gebruikersinteractie en -ervaring.

- Zorg voor beveiliging en authenticatie met Spring Security, met speciale aandacht voor de bescherming van werknemers- en bestelgegevens.

- Implementeer unit- en integratietests met JUnit en Spring Boot Test om de betrouwbaarheid van de applicatie te garanderen.