**GITHUB LINK:** [**https://github.com/UmeraNilofer/FSD-Java.git**](https://github.com/UmeraNilofer/FSD-Java.git)

**algorithm:**

1. Data Modeling:

- Define the entities: `Citizen`, `User`, and `VaccineCenter`, along with their respective attributes and relationships.

- Create repository interfaces (`CitizenRepository`, `UserRepository`, `VaccineCenterRepository`) to handle data access and CRUD operations for the entities.

2. Business Logic:

- Create service interfaces (`CitizenService`, `UserService`, `VaccineCenterService`) to define the operations and business logic related to citizens, users, and vaccine centers.

- Implement the service interfaces in the corresponding service implementation classes (`CitizenServiceImpl`, `UserService`, `VaccineCenterServiceImpl`).

- Within the service implementation classes, use the repository interfaces to perform the required CRUD operations and other business logic.

3. Exception Handling:

- Create custom exception classes (`NotFoundException`) to handle exceptional scenarios, such as when a requested entity is not found.

- Throw these exceptions when necessary, and handle them appropriately in the service implementation classes or in the controller layer.

4. User Interface:

- Create HTML templates (`errorpage.html`, `login.html`, `register.html`) for rendering error messages and user interfaces for login and registration pages.

- Customize the templates based on your specific requirements and design preferences.

5. Configuration:

- Configure the application properties in the `application.properties` file, such as the database connection details and Hibernate settings.

6. Testing:

- Write unit tests using the `VaccinationCenterApplicationTests` class to ensure the proper functioning of the application.

- Perform tests on various components, including the repository interfaces, service interfaces, and other critical functionalities.

7. Deployment:

- Package the application into a deployable format (e.g., JAR or WAR file) using the appropriate build tools (e.g., Maven or Gradle).

- Deploy the application to a web server or a cloud platform, depending on your deployment strategy.