https://github.com/ujjwaltiwari07/capstonemAadhar.git

WriteUp

Problem statement:

Develop an application to automate the process of applying for an Aadhar Card by making it smoother for Indian citizens.

Scenario:

Varniraj Service PVT. LTD is closely working with "The Government of India" to help them get a solution for processing applications for Aadhaar Card. Application is intended to register citizens and let them display ID to process their Aadhar Card application.

Expected Deliverables:

Features of the application:

- Registration
- Login
- Apply for a new Aadhar Card
- Place a request for updating Aadhar details
- Apply for a duplicate Aadhar Card
- Admin: Approve Aadhar Application and issue new Aadhar number
- Apply to close Aadhaar card (due to death)

Recommended technologies:

- **Database:** MySQL
- **Backend:** Java Programming (Spring Boot, JPA, Hibernate)
- **Frontend:** Angular, Bootstrap, and HTML/CSS
- **Automation and testing technologies:** Selenium and TestNG
- **DevOps tools/technologies:** Git, GitHub, Jenkins, and Docker
- **Optional implementation:** Kubernetes, AWS

Project development guidelines:

- The project will be delivered within four sprints with every sprint delivering a Minimal Viable Product.
- It is mandatory to perform proper sprint planning with user stories to develop all project components.

- The learner should use the above-mentioned technologies for different layers of the project.
- The web application should be responsive and should fetch or send data dynamically without hard-coded values.
- The learner must maintain the version of the application over GitHub, and every new change should be sent to the repository.
- The learner must implement a CI/CD pipeline using Jenkins.
- The learner should also deploy and host the application on an AWS EC2 instance.
- The learner should also implement automation testing before the application enters the CI/CD pipeline.
- The learner should use Git branching to perform basic automation testing of the application in it separately.
- The learner should make a rich frontend of the application, which is user-friendly and easy for the user to navigate through the application.
- There will be two portals in the application, the admin and user portal.

Admin Portal:

The admin portal deals with all the backend data generation. The admin user should be able to:

- Login through admin credentials
- Approve new Aadhaar Card request
- Verify request for duplicate Aadhaar
- Display all issued Aadhaar Card
- Delete Aadhaar card details for dead citizen

User Portal:

It deals with user activities. The end-user should be able to:

- Sign in to apply for a new Aadhar Card
- Login to see the Aadhar number assigned by the admin
- Update address, phone number, and date of birth of Aadhaar Card
- Request duplicate Aadhaar Card

Frontend Validation:

- For admin: The password should have at least: One Uppercase, one lowercase, one special character (@,#,&….), and one number.
- For citizens: The password should consist of only digits.

Backend Validation:

- Mobile number validation should be applied.

}

- Password length should not be less than 6 characters.
- For citizens: The home page would authenticate only if the mobile number provided in Aadhaar is matching with the password.

Sample Input data for Backend REST API:

- To register new citizens:

  HTTP Method: POST

  URL: http://localhost:6789/AadharApp/citizens

  Request Body:

  ```
  {
      "name": "Uttam Patel",
      "dob": "2011-08-23",
      "address": "2/5 Heerabagh Flats",
      "emailId": "uttampatel0811@gmail.com",
      "mobileNo": "7976694711",
      "gender": "Male"
  }
  ```

- Apply for Aadhar Card using an existing citizen ID:

  HTTP Method: POST

  URL: http://localhost:6789/AadharApp/issueAadhar

  RequestBody:

  ```
  {
      "citizenId": 1002,
      "passportId": null,
      "issueDate": "2020-04-25"
  }
  ```

Sprint 1:

1. Requirement Gathering & analysis
2. System Design
3. System setup
4. System Configured

Sprint 2:

1. Configured Github to machine.
2. Start frontend with angular
3. Commit to github
4. Start backend with springboot, jpa, hibernate in eclipse
5. Connect frondend to backend.
6. Check connection is established or not.
7. Create databse and configured to application.properties

Sprint 3:

1. Create different types of component in angular for each module.
2. Create service for getting and setting the data to datbse through springboot.
3. Create controller . beans, repository in eclipse to operation.

Step 4:

1. Run both fronted and backend.
2. Checked the application has any bug, fixed it
3. Create java application with selenium and TestNg

4. Perform Unit & Performance testing.
5. Automate that application
6. Everything is working, push to github.
7. Take snapshots of each activities and complete documentations.