**Full Stack Java Developer**

Capstone Project Problem Statement



**Capstone Project 2**

**Bus Pass Management System**

**Problem statement:**

To design and develop a system to manage a system that will maintain complete information about the bus pass details of employees for transport service

**Scenario:**

Pebble Technologies wishes to implement a system that will maintain complete information about the bus pass details of every employee who has ever used the transportation service.

Managing the pass requests traditionally and keeping track of them is a difficult job. So, they decided to go online to have control over their transportation services and provide a better experience for employees. The company has hired you as a full stack developer to build a bus pass application.

**The flow and features of the application:**

**Flow:**

Users can register an account with the Bus Pass Management System within an organization. With this account, users can view routes, vehicles on the route, stops on the route, and apply for a bus pass. Admin users can manage the necessary details in the system and approve or cancel bus pass requests from users.

**Expected deliverables:**

**Features of the application:**

**Admin Module:**

* Admin authentication page
* Approve or reject a bus pass request
* Appoint a bus coordinator
* Change bus coordinator
* Generate reports that can be sent to the finance team or admin team for billing
* Add a bus route
* Remove a bus route
* Manage the bus: Change the type of bus, the number of buses on each route, the route of the bus, and information about each bus
* Navigation option to access all features with ease

**User Module:**

* Register a user
* Apply for a bus pass
* View all the bus route stops, type of vehicles, number of vehicles on each route, and percentage of seats occupied on each route
* Update the bus pass, suspend a bus pass for a few months, and request to change to another route
* Request for a new bus route
* Feedback on bus facilities, including improvements and complaints about the driver
* Request a bus pass cancellation
* Update the user's contact details
* Get a snapshot of the user's details: bus route, timing, type, stops, and so on
* Write to the administrator and request the addition of a new route (if one is not already present)
* Navigation option to access all features with ease

**Validations (Frontend/Backend):**

* Login password validation
* Bus pass validation at the backend
* User type creation validation with a set of permissions. For example, the user belongs to the organization.

**Business Logic Exceptional Scenarios:**

* **BusPassException:** An unexpected situation that occurs while trying to create a new bus pass that already exists or is requested.
* **BusPassSuspendException:** This is a situation when the user is trying to book through a suspended or expired bus pass.
* **AuthenticationException**: This is a situation where the user tries to log in as an Admin.

**The following requirements should be met:**

A student must achieve the following steps in each phase:

* Install the required tools (Java JDK, Apache Maven, Spring Tool Suite, STS)
* Create a starter project
* Configure file storage properties to enable multipart file loads to be uploaded for sharing across the organization
* Configure the directory in which the files will be stored
* Store the uploaded files in the target directory
* Bind the properties defined in the properties file to a POJO class
* Enable configuration properties
* Build a REST API for uploading and downloading files
* Build a service for storing files in the system and retrieving them as and when needed
* Optional: Build containers using Docker and Spring Cloud
* Optional: Implement Kubernetes-based orchestration

**Recommended Technologies:**

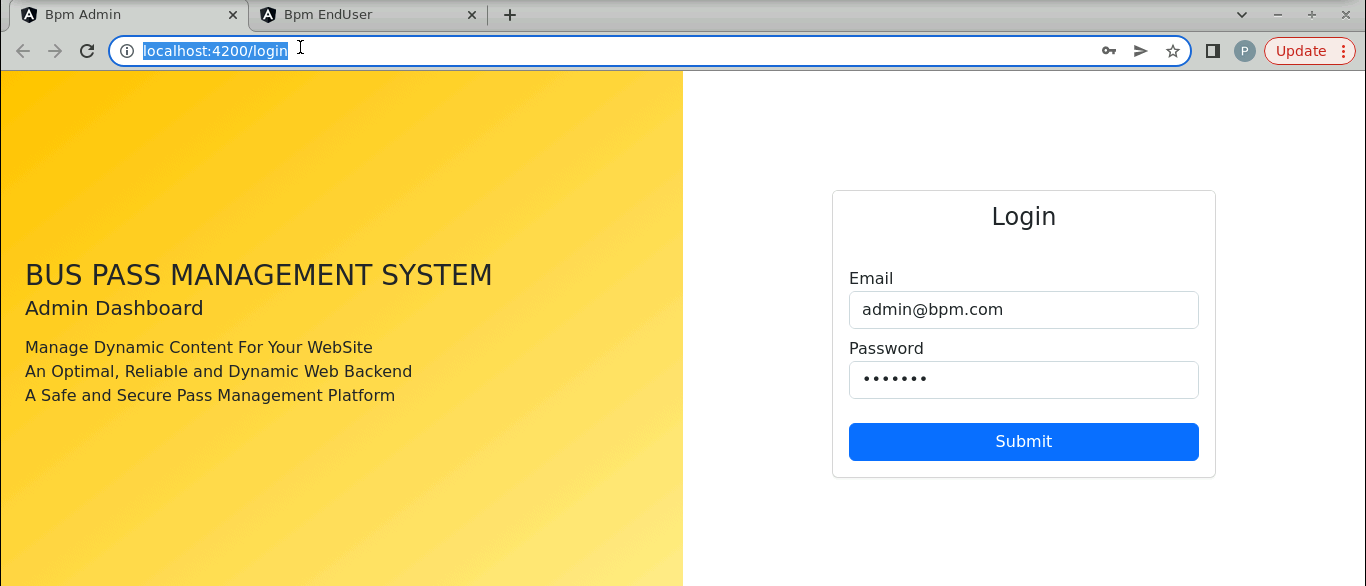
* Database management: MySQL
* Backend logic: Java Programming (Spring Boot, JPA, and Hibernate)
* Front-end development: Angular, HTML/CSS, and Bootstrap
* Automation and testing technologies: Selenium and TestNG
* DevOps and production technologies: Git, GitHub, Docker, and Jenkins
* Optional implementation: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

**Project development guidelines:**

* The project will be delivered within three sprints, with every sprint delivering a minimally viable product.
* It is mandatory to perform proper sprint planning with user stories to develop all the components of the project.
* The learner can 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 hardcoding values.
* The learner must maintain the version of the application on 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 the basic automation testing on the application.
* The learner should make a rich frontend for the application that is user-friendly and easy for the user to navigate through.
* There will be two portals in the application, namely the admin and user portal.
* The learner should implement validation within the backend Spring Boot layer, which will ensure data passed by the API meets the constraints of the system.
* The learner should host the backend application on a local server, and the hosted API should be used by Angular to communicate with the backend.

**Expected Output:**

**Admin user** **login page:**



**Login Successful page:**

A computer screen shot of a computer

Description automatically generated

**Adding user details:**

A screenshot of a computer

Description automatically generated

**Adding drivers’ details:**

A screenshot of a computer

Description automatically generated

**Adding vehicle details:**

A screenshot of a computer

Description automatically generated

**Adding bus routes:**

A screenshot of a computer

Description automatically generated

**Approve or reject bus pass request:**

A screenshot of a computer

Description automatically generated

**Display bus pass requests page (if the user applies for any pass, it will reflect here):**

A screenshot of a computer

Description automatically generated

**Feedback page (if the user gives any feedback, it will reflect here):**

A computer screen shot of a person holding a blue bin

Description automatically generated

**End-user functionalities:**

**Registration page:**

A screenshot of a computer

Description automatically generated

**Bus routes:**

A computer screen shot of a computer

Description automatically generated

**Bus details:**

A screenshot of a computer

Description automatically generated

**Raise a new bus pass request:**

A screenshot of a computer

Description automatically generated

**User feedback page (popup):**

A screenshot of a computer

Description automatically generated

**User feedback page:**

A screenshot of a computer

Description automatically generated