**Learning Management System - LMS**

**Project Writeup**

“LMS” is an educational-based java web application. This web portal helps new Joiner to get all the resources & study materials of every course available on the online portal. All the new Joiner can access this website portal anytime and from anywhere on the internet. LMS is a better online e-learning portal compared to all web links given by Google, Bing, Yahoo, ChatGPT or any website while searching for course materials.

The system maintains the information on the Courses completed by new Joiners and overcome the obstacle during their preparations and resolving their queries by their assigned mentors.

**Project Planning**

1. **Defining Objective:**
   1. Preparing new Joiners for corporate world by inducing technology in use, hence making them productive from the earliest days of joining the organization.
2. **Identifying Key Entities:**
   1. **Employee (Admin):** Maintain and measure the activities for all entities
   2. **Courses:** The learning path assigned to new joiners
   3. **New Joiners:** Fresh recruits. Ready to onboard in organization
   4. **Employee (Mentors):** Employees assigned as a mentor to new joiners
3. **Technology Stack**

|  |  |
| --- | --- |
| **Stack** | **Technologies** |
| **Frontend** | Angular v17, Bootstrap |
| **Backend** | Java, SpringBoot, Hibernate |
| **Version Control** | Git, Github |
| **Database** | RDBMS (MySQL, Postgres, h2) |
| **DevOps** | Docker, Kubernetes |
| **Cloud** | Google Cloud Platform (GCP) |
| **Testing** | Postman, Junit, Selenium, Jasmine, Karma |
| **Project Management Tool** | Jira |

1. **System Design**
   1. **Architecture**
      * **Microservice**: Each module as a separate service
      * **API Gateway:** Central point for API Request
      * **Database:** Centralized / Distributed with proper indexing
      * **Security:** Data encryption, Authentication, Authorization
   2. **Database Schema**: Tables as needed in the system
2. **Development**
   1. **Frontend** 
      * Use of Angular Framework
      * Responsive UI using Bootstrap
      * Authentication and Role based access
   2. **Backend**
      * RESTful APIs for modules
      * Business Logic and Data Validation
   3. **Integration**
      * Seamless integration with frontend and backend
      * Develop APIs for communication between different microservices
      * Data synchronization and consistency checks
3. **Testing**
   1. **Unit Testing**

* Write tests for individual components and functions

**b.** **Integration Testing**

* Test interactions between different modules and services
* Ensure end-to-end functionality

1. **Deployment**
   1. **Environment Setup**

* Configure cloud infrastructure (servers, databases, storage)
  1. **Monitoring and Maintenance**
* Use monitoring tools (e.g., Splunk, Grafana) to track system performance
* Implement logging, performance metrics and distributed tracing

1. **Documentation**
   1. **User Manuals**

* Create detailed documentation for HR managers and new Joiners

1. **Conclusion**

**Project Overview**

1. **Project Title:** Learning Management System
2. **Duration:** 10 Days
3. **Team Size:** 05 full-stack developers
4. **Project Goals:**
5. Develop a comprehensive Learning Management System focusing on Human Capital Management (HCM).
6. Deploy the system as a *Software as a Service (SaaS)* to ensure scalability, accessibility, and maintainability.
7. Implement core HR functionalities including bridging the gap between the business requirements and available workforce by supplying productive new joiners In the organization.
8. **Key Features**
9. **Admin: Tracks all the information of the new joiners, courses and assigned mentors.**
10. **Admin can perform all types of CRUD Operation on Courses.**
11. **Employee is a role based module where *Admin* can perform each and every operation on data, but the new joiners will be able to view his/her data. Therefore access level restrictions has to be implemented in this project.**
12. **Admin and Mentors can upload course material. Admin permission required to upload new course.**
13. **Admin can view the queries which are sent by the new Joiners to the mentors. The admin can also check status for how many solved queries and unsolved queries.**
14. **Each new joiner must and should be registered before login. Here we have user authentication every time.**
15. **The student can send a query to the mentor if he/she has any doubts regarding courses.**
16. **One mentor can be assigned to multiple new joiners**
17. **System Features**

|  |  |
| --- | --- |
| **Key Feature 1** | **User (Admin/Mentor) should be able to login in the system** |
| Trigger | User clicks on “Login” Link on the home page |
| Pre-condition | User should be available in the database |
| Post-condition | The user is successfully login in the system |
| UI Screen | The form containing the following fields –   |  |  |  | | --- | --- | --- | | Email | Text value | Verified from DB | | Password | Text value | Verified from DB | | Login | Button |  | | Cancel | Button |  | |
| Main Flow | The user provides the required details in the form and Clicks “Login” button. On successfully login, user is forwarded to the dashboard page. |
| Data Validations | - All fields are required  - Input Data has to be validated at frontend |
| Business Rule | - Email and Password to be verified from Database |
| Exception Flow | In case of any failure in Data Validation or Business Rules, user should be brought back to the login screen along with the relevant error message. |
| Alternate Flow | When the user selects “Cancel” button, s/he should be taken back to the home Page |

|  |  |
| --- | --- |
| **Key Feature 2** | **Admin/Mentor should be able to create new Course** |
| Trigger | User clicks on “Add Course” Link on the Dashboard page |
| Pre-condition | All the courses should be displayed at Dashboard page |
| Post-condition | The complete information about the new course is stored in the database |
| UI Screen | The form containing the following fields –   |  |  |  | | --- | --- | --- | | Course Title | Text value |  | | Course Duration | Text value |  | | Course Id | Text value | Auto generated | | Uploaded By | Drop Down | Pre-populated | | Category | Drop Down |  | | Add Course | Button |  | | Cancel | Button |  |   \* Add more fields, if required. |
| Main Flow | The user provides the required details in the form and Clicks “Add Course” button. On successfully saving the details, user is forwarded to the home page along with the message “Details have been saved successfully.”.  Once successfully applied, the notification should dispatch to the Admin for further action/permission. |
| Data Validations | - All fields are required  - Input Data has to be validated at frontend |
| Business Rule | - Course ID is auto generated |
| Exception Flow | In case of any failure in Data Validation or Business Rules, user should be brought back to the add course screen along with the relevant error message. |
| Alternate Flow | When the user selects “Cancel” button, s/he should be taken back to the home Page |

|  |  |
| --- | --- |
| **Key Feature 3** | **New Joiner registration** |
| Trigger | **User clicks on “Sign-up” Link on the Home Page** |
| Pre-condition | - |
| Post-condition | User details should be stored in database |
| UI Screen | The form containing the following fields –   |  |  |  | | --- | --- | --- | | First Name | Text value |  | | Last Name | Text value |  | | Email | Text value | Verified from DB | | Password | Text value |  | | Confirm Password | Text value |  | | Mobile number | Number | Verified from DB | | Sign-up | Button |  | | Cancel | Button |  |   \* Add more fields, if required. |
| Main Flow | The user provides the required details in the form and Clicks “Sign-up” button. On success, the user details should be stored in the database. |
| Data Validations | - Input Data has to be validated at frontend  - Password and confirm password should match |
| Business Rule | - Should provide unique email and mobile number |
| Exception Flow | In case of any failure in Data Validation or Business Rules, user should be brought back to the Registration screen along with the relevant error message. |
| Alternate Flow | When the user selects “Cancel” button, s/he should be taken back to the Dashboard Page |