E-Learning Resource

Objective:

E-Learning resource is an online application to be built education based website/software , helping students to get all resources & study materials of every courses available. It uses "E-Book" facility. It is reliable & time efficient approach compared to all links of the website provided by any search engine while searching for course materials.

Users of the System:

1. Admin
2. Professors
3. Users

Functional Requirements:

- To provide official & legal links of the website from which user(student) can download resources & study materials of relevant course.
- Only accessible after registering to that specific website.
- Getting associated with professors of esteemed institutes & colleges.
- To provide interaction between professor & students.
- Not all links provided by GOOGLE are virus free or recommended to download. This would be a better approach.
- **Resources link should not repeat.**

Output/ Post Condition:

- Records Persisted in Success & Failure Collections
- Standalone application / Deployed in an app Container

Non-Functional Requirements:

Security	• App Platform –UserName/Password-Based Credentials • Sensitive data has to be categorized and stored in a secure manner • Secure connection for transmission of any data
Performance	• Peak Load Performance • E-Learning Resource -< 3 Sec • Admin application < 2 Sec • Non Peak Load Performance
Availability	• 99.99 % Availability
Standard Features	• Scalability • Maintainability • Usability • Availability • Failover
Logging & Auditing	• The system should support logging(app/web/DB) & auditing at all levels
Monitoring	• Should be able to monitor via as-is enterprise monitoring tools
Cloud	• The Solution should be made Cloud-ready and should have a minimum impact when moving away to Cloud infrastructure

Browser Compatible	• IE 7+ • Mozilla Firefox Latest – 15 • Google Chrome Latest – 20 • Mobile Ready

Technology Stack

Front End	React Google Material Design Bootstrap / Bulma
Server Side	Spring Boot Spring Web (Rest Controller) Spring Security Spring AOP Spring Hibernate
Core Platform	OpenJDK 11
Database	MySQL or H2

Platform Pre-requisites (Do's and Don'ts):

1. The React app should run in port 8081. Do not run the React app in the port: 3000.
2. Spring boot app should run in port 8080.

Key points to remember:

1. The id (for frontend) and attributes(backend) mentioned in the SRS should not be modified at any cost. Failing to do may fail test cases.
2. Remember to check the screenshots provided with the SRS. Strictly adhere to id mapping and attribute mapping. Failing to do may fail test cases.
3. Strictly adhere to the proper project scaffolding (Folder structure), coding conventions, method definitions and return types.
4. Adhere strictly to the endpoints given below.

Application assumptions:

1. The login page should be the first page rendered when the application loads.
2. Manual routing should be restricted by using AuthGaurd by implementing the canActivate interface. For example, if the user enters as http://localhost:3000/signup or http://localhost:3000/home the page should not navigate to the corresponding page instead it should redirect to the login page.
3. Unless logged into the system, the user cannot navigate to any other pages.

4. Logging out must again redirect to the login page.
5. To navigate to the admin side, you can store a user type as admin in the database with a username and password as admin.
6. Use admin/admin as the username and password to navigate to the admin dashboard.

Validations:

1. Basic email validation should be performed.
2. Basic mobile validation should be performed.

Project Tasks:

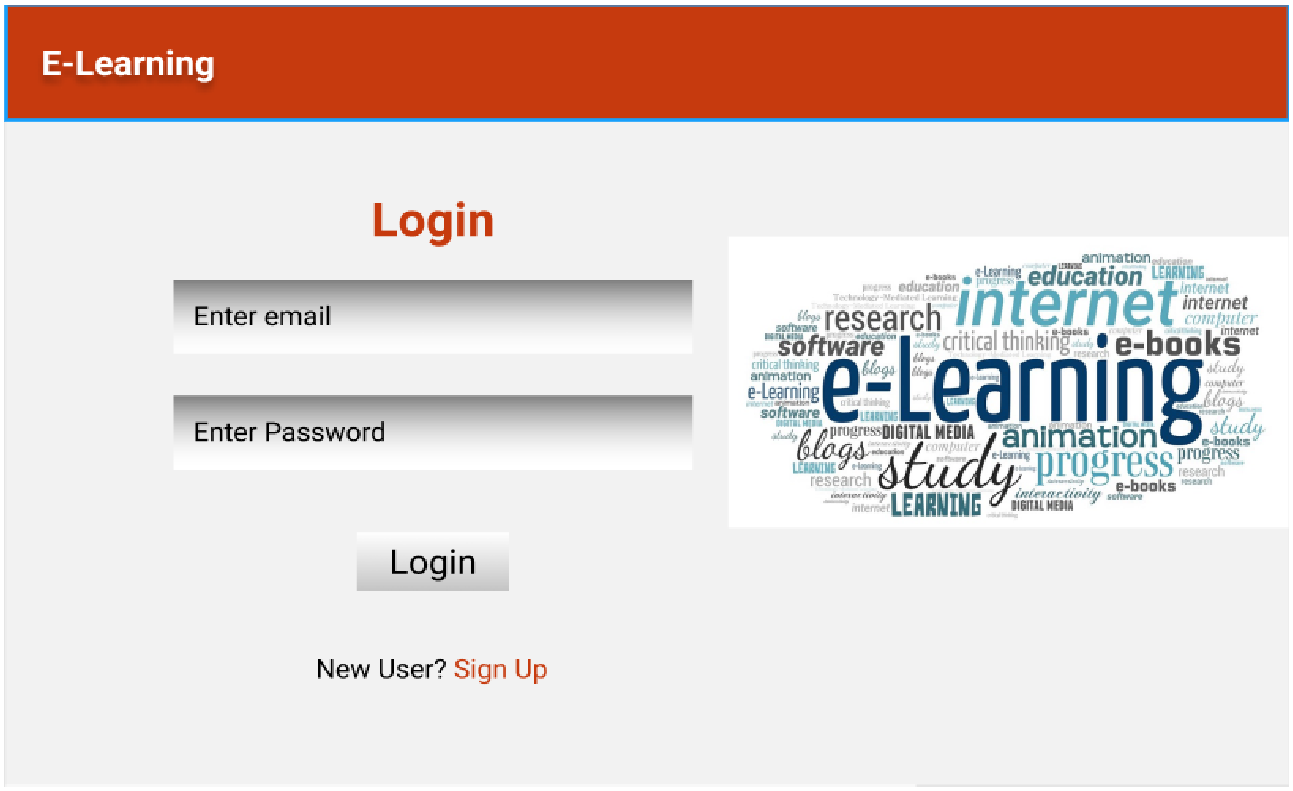
API Endpoints:

USER			
Action	URL	Method	Response
Login	/login	POST	true/false
Signup	/signup	POST	true/false
Get All Resource	/home	GET	Array of Resource
Update Resource	/home/{id}	PUT	Updated Success
Add Resourse	/home	POST	Added Successfully
Delete Resource	/home/{id}	DELETE	Resource Deleted
Start Chat	/chat/{id}	POST	Chat Started
Get Chat	/chat/{id}	GET	Array of Chat
Delete Chat	/chat/{id}	DELETE	Chat Deleted
ADMIN			
Action	URL	Method	Response
Get All User	/admin	GET	Array of Users
Approve User	/admin/verify	POST	User Verified
Delete User	/admin/delete/{id}	DELETE	User deleted
Update Resource	/admin/resourse/{id}	PUT	Updated Success

Frontend:

Login:

Output Screenshot:


E-Learning

Login

Enter email

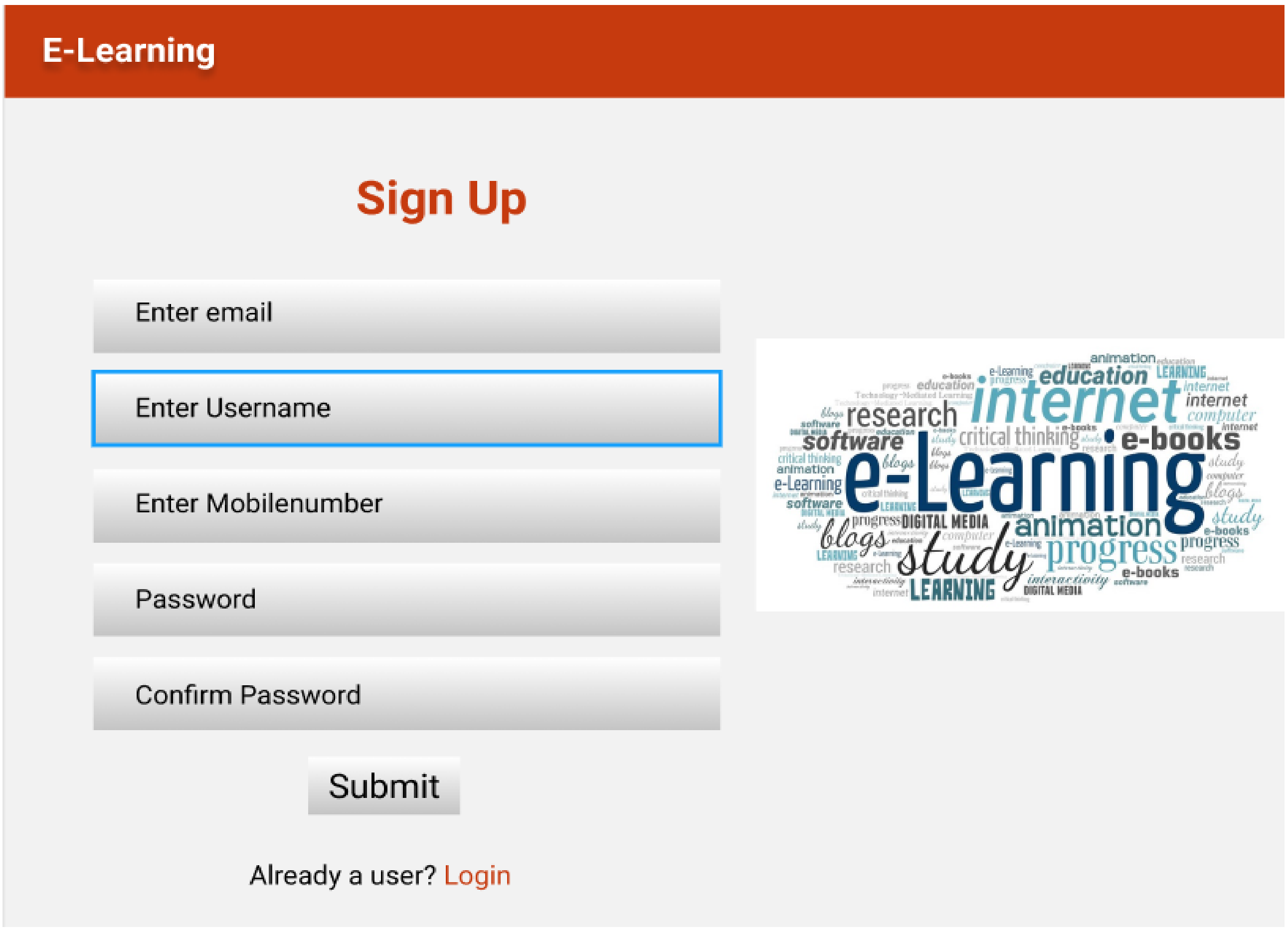
Enter Password

Login

New User? Sign Up

Signup:

Output Screenshot:


E-Learning

Sign Up

Enter email

Enter Username

Enter Mobilenumber

Password

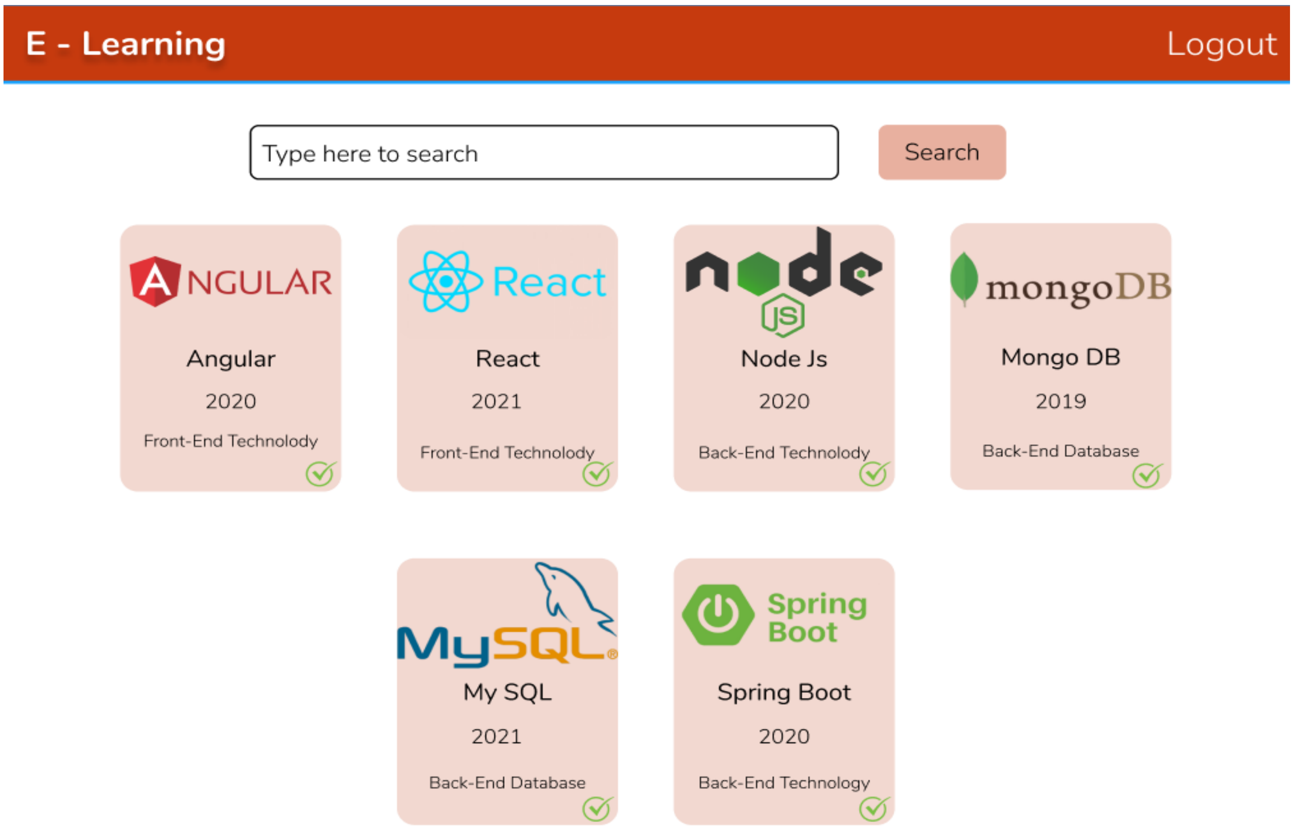
Confirm Password

Submit

Already a user? Login

Home:

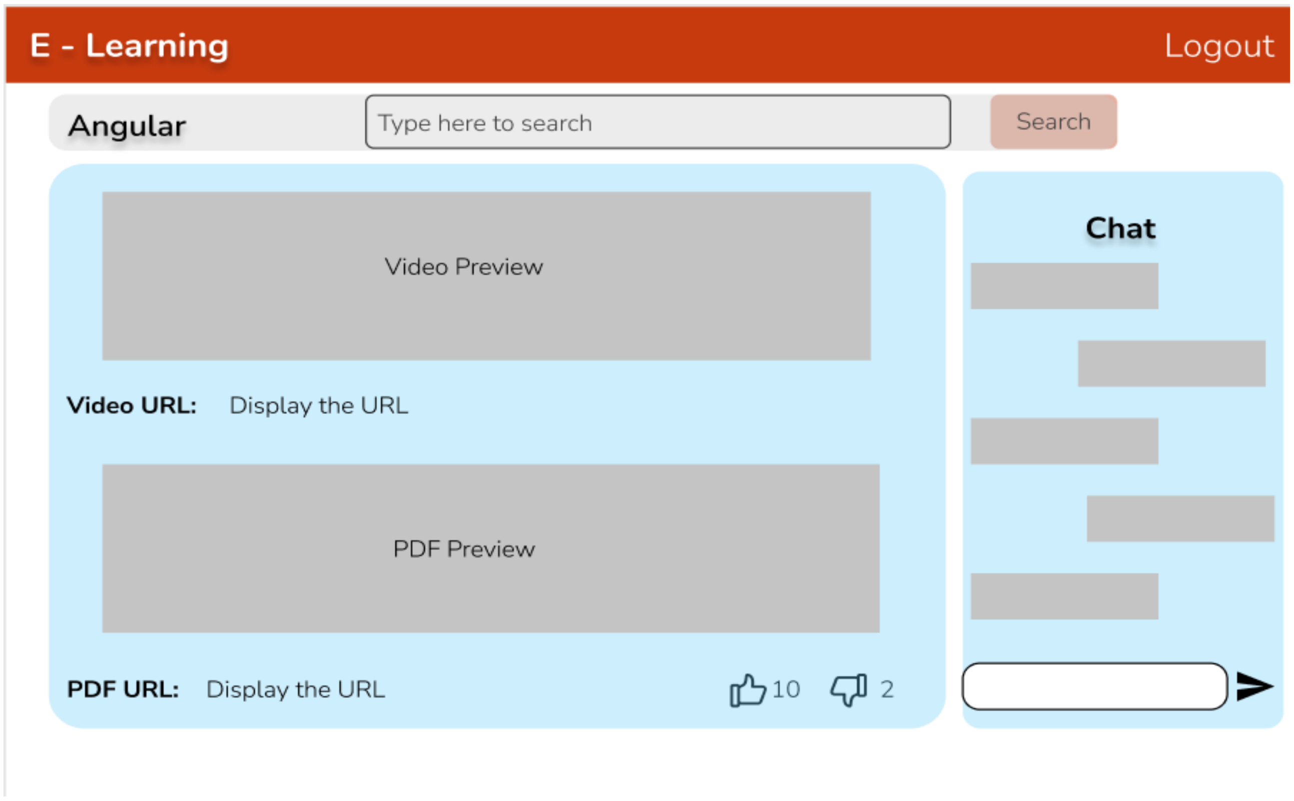
Output Screenshot:



Angular	React	Node Js	Mongo DB
2020	2021	2020	2019
Front-End Technolody	Front-End Technolody	Back-End Technolody	Back-End Database
My SQL	Spring Boot		
2021	2020		
Back-End Database	Back-End Technology		

Resourse By Id:

Output Screenshot:



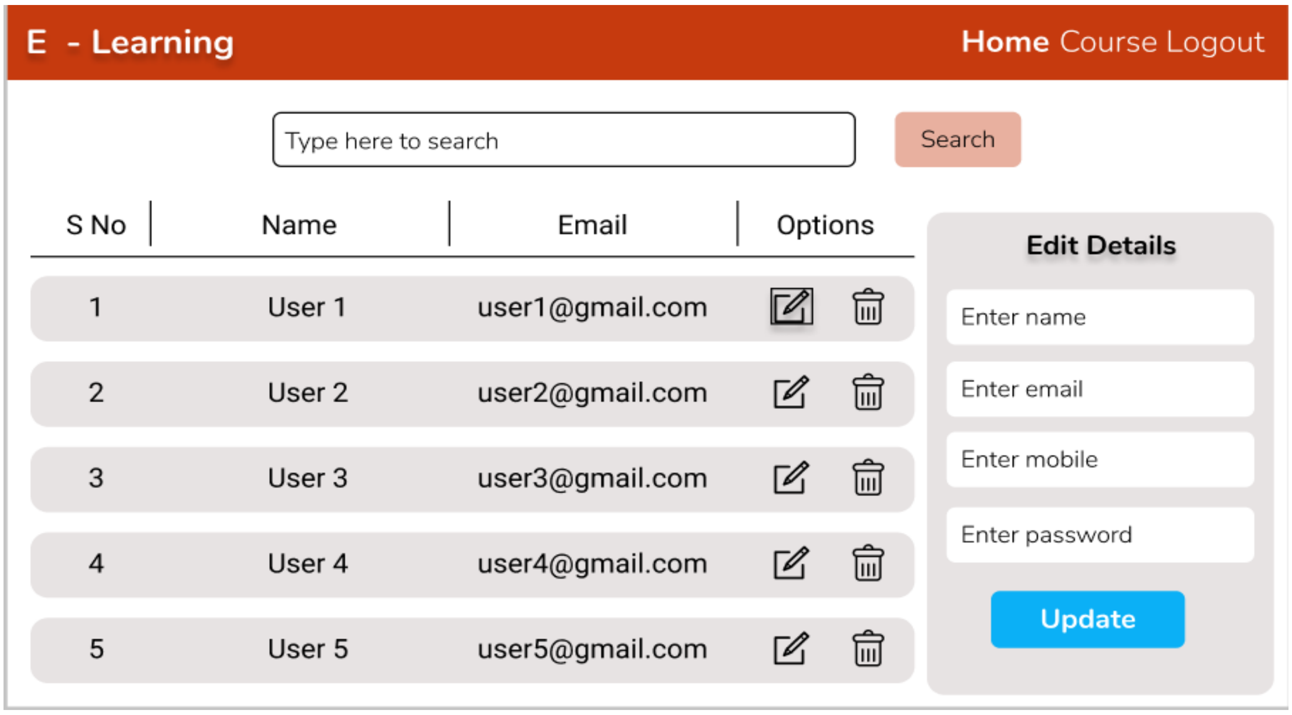
Video URL: Display the URL

PDF URL: Display the URL

Admin:

Home

Output Screenshot:



E - Learning | Home Course Logout

Type here to search | Search

S No	Name	Email	Options
1	User 1	user1@gmail.com	
2	User 2	user2@gmail.com	
3	User 3	user3@gmail.com	
4	User 4	user4@gmail.com	
5	User 5	user5@gmail.com	

Edit Details

Enter name

Enter email

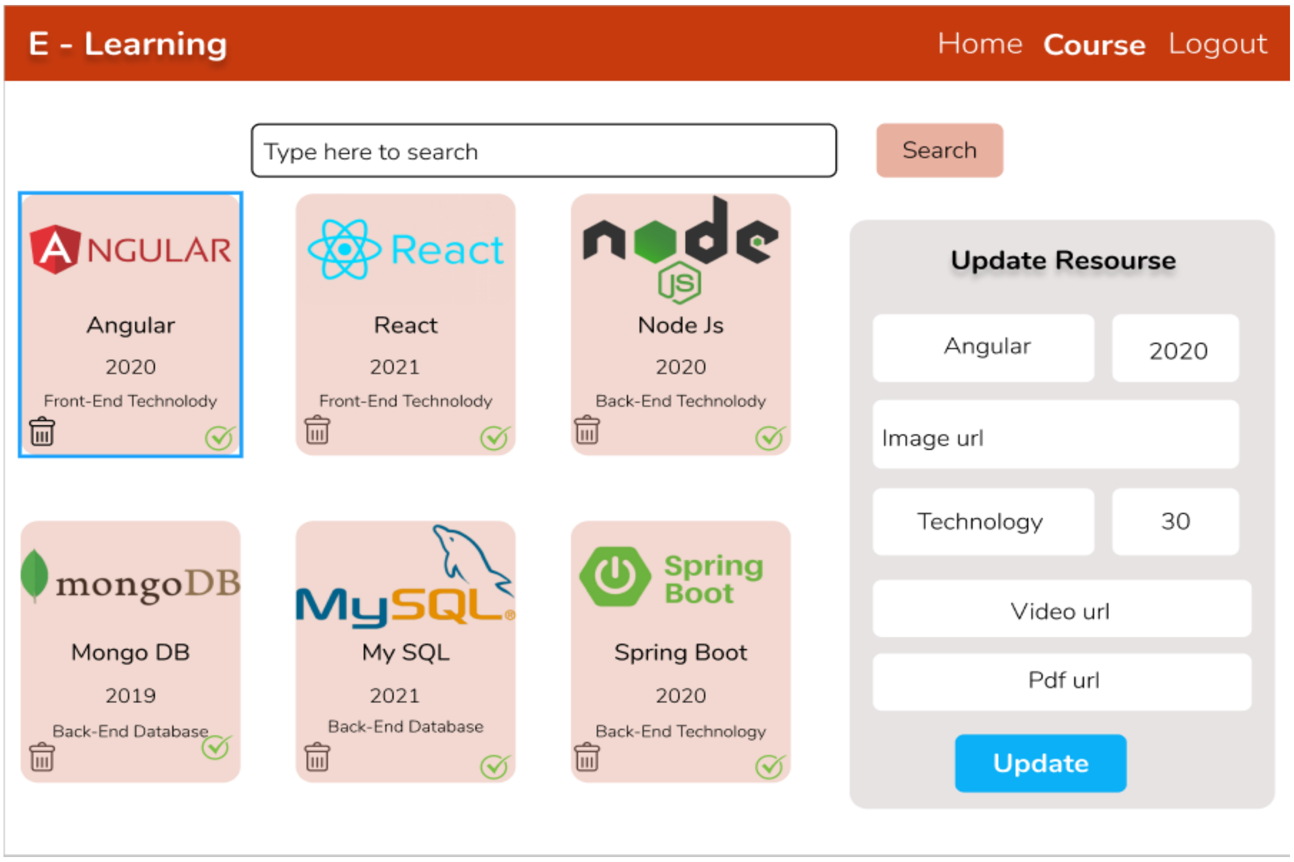
Enter mobile

Enter password

Update

Resourse By Id:

Output Screenshot:



E - Learning | Home Course Logout

Type here to search | Search

Angular
2020
Front-End Technolody

React
2021
Front-End Technolody

Node Js
2020
Back-End Technolody

Mongo DB
2019
Back-End Database

My SQL
2021
Back-End Database

Spring Boot
2020
Back-End Technology

Update Resourse

Angular | 2020

Image url

Technology | 30

Video url

Pdf url

Update

Backend:

Class and Method description:

Model Layer:

1. UserModel: This class stores the user type (admin or the User) and all user information.
    a. Attributes:
        i. email: String
        ii. password: String
        iii. username: String
        iv. mobileNumber: String
        v. qualification: String
        vi. active: Boolean
        vii. role: String
    b. Methods: -
2. LoginModel: This class contains the email and password of the user.
    a. Attributes:
        i. email: String
        ii. password: String
    b. Methods: -
3. ResourceModel: This class stores the all the resource provided by the verified User.
    a. Attributes:
        i. resourceId: String
        ii. resourceName: String
        iii. resourceLink: String
        iv. imageUrl: String
        v. resourceCategory: String
        vi. createdOn: Date
        vii. createdBy: UserModel
        viii. verified: Boolean
        ix. active: Boolean

   b. Methods: -

4. chatModel: This class stores the communication between two user.
   a. Attributes:
      i. chatId: String
      ii. primaryUser: UserModel
      iii. secondaryUser: UserModel
      iv. chatHistory: List<String>
      v. status: Boolean
      vi. lastSeen: Date
   b. Methods: -

Controller Layer:

5. SignupController: This class control the user signup
   a. Attributes: -
   b. Methods:
      i. saveUser(UserModel user): This method helps to store users in the database and return true or false based on the database transaction.
6. LoginController: This class controls the user login.
   a. Attributes: -
   b. Methods:
      i. checkUser(LoginModel data): This method helps the user to sign up for the application and must return true or false
7. ResourceController: This class controls the add/edit/update/view Resource.
   a. Attributes: -
   b. Methods:
      i. List<ResourceModel> getResource(): This method helps the User to fetch all resource from the database.
      ii. ResourceModel resourceById(String id): This method helps to retrieve a Resourse from the database based on the resource id.
      iii. resourceEditSave(ResourceModel data): This method helps to edit a resource and save it to the database.
      iv. resourceSave(ResourceModel data): This method helps to add a new resource to the database.
      v. resourceDelete (String id): This method helps to delete a resource from the database.
8. ChatController: This class helps to store the communication between the students and Professors.

a. Attributes: -

b. Methods:

    i. startChat(String id): This method helps the user to communicate with the professor.

    ii. List<ChatModel> showChat(String id): This method helps to view the chat based on chatId.

    iii. deleteChatItem(String id): This method helps to delete a chat from the database.