Documentation for MLSA-IIIT-Bh Chapter Website

View GitHub Repository: https://github.com/SumitPanda03/MLSA_IIITBH

View Website Workflow: http://surl.li/pvbcm

NOTE: Members are kindly reminded to ensure that their uploaded images do not exceed 500 KB in size to optimize the performance and functionality of the website.

Frontend(Client):

1. Created using Vite.js + ReactJS

2. Components of the website:
   Carousel.jsx: Used for the **Members** page on the website to display the details of the members in a carousel format. The front- of the carousel displays the image of the member with his/her details and the back displays the mlsa milestone of the member.
   Footer.jsx: Manages the footer of the website.
   Navbar.jsx: Manages the navigation bar of the website.
   Title.jsx: Homepage page of the website.
   Zero.jsx: Welcome page that takes the user to the website.

3. Pages of the website:
   a. Chapter. jsx: Manages the carousel and the details of the Members.
   b. Application. jsx: Details regarding the complete application process.
   c. Milestones.jsx: Details regarding different milestones in MLSA(new, alpha, beta, gold)
   d. Opening.jsx: Loads the admin page to log in and enter details of a new member
   e. Upload.jsx: Opens the form to enter all the new entry details.

4. Structure folder:
   a. RenderNavigation.jsx: Render the navigation file to facilitate the routes.
   b. Navigation.jsx: Routing of the various pages(zero, title, application, chapter, milestones, opening, register) are linked through this file.

5. App.jsx: Calls the AuthWrapper routing file (used for the login and upload functionality).

6. AuthWrapper.jsx: Manages the various routes in the application. The admin route(Opening.jsx) is password-protected and can't be entered by anyone other than the

application admins. This password protection is achieved by using the features of React Router V6.

7. Bootstrap Version used: 4.6.2

8. Google fonts applied: Catamaran (Refer to index.css file) + Montserrat + Roboto Condensed

9. Animations used: .json format (Source: Lottie-Files, Package: Lottie-react and react-Lottie)

10. Background gradient used: radial-gradient(circle, #457b9d 35%, rgb(1, 1, 2) 99%, rgb(55, 152, 172) 100%) [Source: https://cssgradient.io/]

11. Other vital dependencies:
    a. axios: 1.5.1
    b. cors: 2.8.5
    c. dotenv: 16.3.1
    d. Jsonwebtoken: 9.0.2
    e. lottie-react: 2.4.0
    f. nodemon: 3.0.1
    g. react-bootstrap: 2.9.1
    h. react-icons: 4.11.0
    i. react-lottie: 1.2.4
    j. slick-carousel: 1.8.1
    k. typed.js: 2.0.16
    l. eslint: 5.0.8

Backend(Server):

1. Created using Node.js, Express.js, MongoDB, AWS.
2. Components of the website:

   App.js:

   s3.js:

   model: userSchema.js

   routes: router.js

Db: db.js

3. All the data except images of all the members are stored in MongoDB Atlas.
4. This document provides an overview of the backend architecture for the project. The backend is developed using Node.js, Express.js, MongoDB, and AWS services. It handles data storage, authentication, and authorization for the MLSA members' applications.
5. The images of the various members are stored in an AWS bucket.
6. Technologies Used:
    a. Node.js: JavaScript runtime for server-side development.
    b. Express.js: Web application framework for Node.js.
    c. MongoDB: NoSQL database for storing members' data.
    d. AWS S3: Storage service for storing images of MLSA members.
    e. JWT (JSON Web Token): Used for authentication and authorization.
    f. Cors: Middleware for enabling cross-origin resource sharing.
    g. Cookie-parser: Middleware for parsing cookies.
    h. Mongoose: MongoDB object modeling for Node.js.
    i. AWS SDK: Software development kit for interacting with AWS services.

7. Dependencies:
    a. "@aws-sdk/client-s3": "^3.490.0",
    b. "@aws-sdk/lib-storage": "^3.490.0",
    c. "@aws-sdk/s3-request-presigner": "^3.490.0",
    d. "aws-sdk": "^2.1534.0",
    e. "axios": "^1.6.0",
    f. "cookie-parser": "^1.4.6",

g. "cors": "^2.8.5",

h. "dotenv": "^16.3.1",

i. "express": "^4.18.2",

j. "jsonwebtoken": "^9.0.2",

k. "moment": "^2.29.4",

l. "mongoose": "^7.6.3",

m. "multer": "^1.4.5-lts.1",

n. "nodemon": "^3.0.1",

o. "uuid": "^9.0.1"

8. conn.js: Present in the db folder, handles the connection to the MongoDB account storing all the members' data.
9. userSchema.js: Present in the model folder describing the schema used in this application, containing the details of all the mlsa members.
10. JWT is employed for proper authentication and authorization of users.
11. .env file: Stores secret credentials like API keys, MongoDB connection strings, etc.
12. .gitignore file: Ensures that sensitive information and node modules are not pushed into version control.
13. The backend is deployed on onrender.com.
14. S3.js is responsible for displaying website images and storing newly uploaded images on the AWS S3 bucket.
15. Router.js handles all routes of the website.
16. Endpoints:
    a) /register: POST route. It uploads all the member data and images to an AWS S3 bucket, generating image URLs, and saving the member data to a MongoDB database.

    b) /getdata : GET route. It retrieves member data from the MongoDB database, sorting the results in ascending order based on the “date” field, such that the oldest date data is displayed first.