**A Real- time Research Project/Societal Related Project Report**

**on**

**MOCK ME**

**A MOCK INTERVIEW WEBSITE**

**submitted in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

**by**

**23WH1A0598 Ms. D HARSHITHA**

**23WH1A0599 Ms. SK REHANA AZMEE**

**23WH1A05B9 Ms. M SRIJANI**

**23WH1A05C2 Ms. P GAYATHRI**

****

**Department of Computer Science & Engineering**

BVRIT HYDERABAD

College of Engineering for Women

**(UGC Autonomous Institution | Approved by AICTE | Affiliated to JNTUH)**

**(NAAC Accredited – A Grade | NBA Accredited B. Tech. (EEE, ECE, CSE and IT))**

**Bachupally, Hyderabad -500090**

**April, 2025**

BVRIT HYDERABAD

College of Engineering for Women

**(UGC Autonomous Institution | Approved by AICTE | Affiliated to JNTUH)**

**(NAAC Accredited – A Grade | NBA Accredited B. Tech. (EEE, ECE, CSE and IT))**

**Bachupally, Hyderabad -500090**

**Department of Computer Science and Engineering**

****

## CERTIFICATE

This is to certify that the Real-time Research Project/Societal Related Project entitled **“Mock Me”** is a bonafide work carried out by Ms. D Harshitha (23WH1A0598), Ms. Sk Rehana Azmee (23WH1A0599), Ms. M Srijani (23WH1A05B9), and Ms. P Gayathri (23WH1A05C2) in partial fulfillment of the award of the B.Tech. degree in **Computer Science and Engineering, BVRIT Hyderabad College of Engineering for Women, Bachupally, Hyderabad,** affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision .The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

##### Project Coordinator Head of Department

Ms. T Durga Devi, Dr. M Sree Vani,

Assistant Professor, Professor & HoD,

Department of CSE. Department of CSE.

## DECLARATION

We hereby declare that the work presented in this project entitled **“Mock Me”** submitted towards the completion of a Real-time Research Project/ Societal Related Project Work in the II-II B.Tech., CSE at BVRIT Hyderabad College of Engineering for Women, Hyderabad, is an authentic record of our original work.

**Sign with Date:**

**Ms. D HARSHITHA**

**(23WH1A0598)**

**Sign with Date:**

**Ms. SK REHANA AZMEE (23WH1A0599)**

**Sign with Date:**

**Ms. M SRIJANI**

**(23WH1A05B9)**

**Sign with Date: Ms. P GAYATHRI (23WH1A05C2)**

## ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. K V N Sunitha, Professor & Principal**, **BVRIT HYDERABAD College of Engineering for Women**, for her support by providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. M Sree Vani, Professor & HoD, Department of CSE, BVRIT HYDERABAD College of Engineering for Women,** for all timely support and valuable suggestions during the period of our project.

We are extremely thankful to our project coordinator, **Ms. T Durga Devi Assistant Professor, CSE, BVRIT HYDERABAD College of Engineering for Women,** for her constant guidance and encouragement throughout the project.

Finally, we would like to thank all faculty and Staff of CSE department who helped us directly or indirectly. Last but not least, we wish to acknowledge our Parents and Friends for giving moral strength and constant encouragement.

**Ms. D HARSHITHA (23WH1A0598)**

**Ms. SK REHANA AZMEE**

**(23WH1A0599)**

**Ms. M SRIJANI**

**(23WH1A05B9)**

## Ms. P GAYATHRI

## (23WH1A05C2)

## ABSTRACT

Mock Me is an innovative web-based solution designed to empower users in their career journey by providing realistic and personalized mock interview experiences. It offers a supportive and engaging environment where users can practice interviews, receive feedback, and track their progress, boosting their confidence and readiness for real-world opportunities. With modern web technologies and user-centered design, the platform delivers structured mock sessions, allowing users to navigate seamlessly, schedule sessions, for an effective preparation experience. This documentation provides a comprehensive overview of the platform’s development, covering key design decisions, implementation strategies, user experience considerations, and future improvements, making Mock Me a valuable resource for students, job seekers, and professionals aiming to succeed in interviews and advance their careers.

**INDEX**

**S. No Topic Page No.**

1 Introduction 1

* 1. Objectives 2
  2. Existing System 5
  3. Proposed System 7

2 Literature Survey 8

3 Methodology 15

3.1 Proposed Model/Architecture 17

3.2 Datasets 26

4 Results and Discussion 29

5 Conclusion & Future scope 30

6 References 32

## LIST OF FIGURES

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Fig. No** | **Fig. Name** | **Page No.** |
| 1 | 2.2 | Flow Chart | 4 |
| 2 | 2.3 | E-R Diagram | 6 |
| 3 | 2.4 | Sequential Diagram | 9 |
| 4 | 5.1 | Login Page | 20 |
| 5 | 5.2 | Sign Up Page | 21 |
| 6 | 5.3 | Available Locations Page | 22 |
| 7 | 5.4 | Home Page | 22 |
| 8 | 5.5 | Detailed view of Product | 24 |
| 9 | 5.6 | Dynamic Cart | 25 |

# INTRODUCTION

### Introduction

Mock Me Interview Portfolio Platform is a user-centered web application designed to enhance interview preparedness through personalized mock sessions. Users can create profiles, select interview types, and practice with mentors, refining their skills in a structured environment. The platform ensures a seamless experience with secure authentication, intuitive scheduling, and real-time analytics, allowing users to track progress efficiently. Built with HTML, CSS, JavaScript, Node.js, and MySQL, Mock Me leverages a robust backend to manage user accounts, session data, and performance metrics, providing a streamlined and effective way to build confidence for real-world interviews.

#### Objectives

The primary objectives of the Mock Me are as follows:

#### Effective Interview Practice:

#### Provide a structured platform where users can simulate mock interviews tailored to different roles and industries.

#### Personalized Feedback:

#### Deliver detailed, actionable feedback by mentors to help users identify strengths and areas for improvement.

#### User Data Security:

#### Ensure the protection of user information, video responses, through strong security protocols.

#### Progress Tracking:

#### Enable users to track their interview performance and improvements.

#### User-Centric Design:

#### Offer a clean, responsive, and accessible interface that supports seamless navigation and ease

#### of use across devices.

#### Methodology

#### The development of Mock Me:

#### A Mock Interview Portfolio Platform follows a structured approach that integrates user research, iterative design, and robust technical implementation to deliver an effective and user-friendly interview practice tool.

#### User Research:

#### Conduct surveys and interviews to identify interview challenges and user needs.

#### Design Iterations:

#### Create wireframes and prototypes to refine usability, ensuring clarity, accessibility, and a seamless user experience.

#### Technical Development:

#### Build the frontend using HTML, CSS, JavaScript, and Node.js for a responsive and interactive experience. Connect to a backend database to manage user accounts and mock interview data. Implement secure login and data protection mechanisms to ensure user privacy and system integrity.

# DESIGN

### Introduction

The design phase of Mock Me: A Mock Interview Portfolio Platform was guided by a strategic approach aimed at fulfilling user needs and achieving functional excellence. Key design principles included:

* **User Interface (UI):** Crafting a clean, responsive, and accessible interface that accommodates users of varying technical backgrounds, ensuring smooth navigation and a focused interview practice experience.
* **Database Design:** Developing a structured and scalable database system to manage user profiles, mock interview sessions with efficiency and reliability.
* **Security Measures:** Incorporating strong security protocols to safeguard user data ensuring protection against unauthorized access and alignment with privacy standards.

#### Architecture Diagram

The architecture diagram illustrates the structural layout of Mock Me, highlighting the integration and interaction of key system components. Architectural diagrams play a crucial role in understanding, designing, and communicating the overall structure and workflow of a platform. They offer a high-level perspective that simplifies the visualization of system processes, data flow, and component relationships—making it easier to identify potential bottlenecks and opportunities for optimization. Below is a simplified architecture representation for the platform:

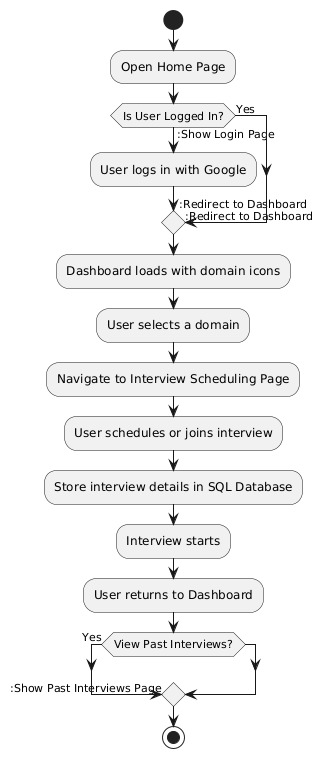


Fig. 2.2 flow chart

This flowchart outlines the user journey on an interview scheduling platform, starting from the home page. If logged in, the user is taken to the dashboard; otherwise, they log in first. On the dashboard, the user selects a domain and navigates to the interview scheduling page to schedule an interview. The user then returns to the dashboard, with options to view past interviews. The process ends when the user logs out and exits the site.

#### E-R DIAGRAM

An E-R diagram, or Entity-Relationship diagram, is a visual representation of the relationships between different data entities in a database. It helps to illustrate how data is structured and how different entities are related to each other. In simple terms, it's like a map that shows how different pieces of data connect and interact within a system.

An Entity-Relationship (E-R) diagram is a tool used in database design to visually depict the structure of a database. It consists of entities, relationships, and attributes:

1. Entities: These are objects or things in the real world that have distinct existence and are represented by rectangles. For example, in a university database, entities could be "Student," "Course," and "Professor."
2. **Attributes:** These are properties or characteristics of entities, represented by ovals. For instance, the "Student" entity might have attributes like "StudentID," "Name," and "DateOfBirth."
3. **Relationships:** These illustrate how entities are related to each other and are represented by diamonds. For example, a "Student" entity might have a relationship with a "Course" entity, which could be "enrolls in”.
4. **Cardinality:** This shows the numerical relationship between entities. For example, a single student can enroll in multiple courses (one-to-many relationship), or a course can have many students enrolled in it (many-to-one relationship). By using an E-R diagram, you can easily see how different pieces of data fit together, ensuring that the database is well-organized and accurately reflects the real-world relationships between data. This makes it easier for database designers and developers to understand the system and create an efficient database structure

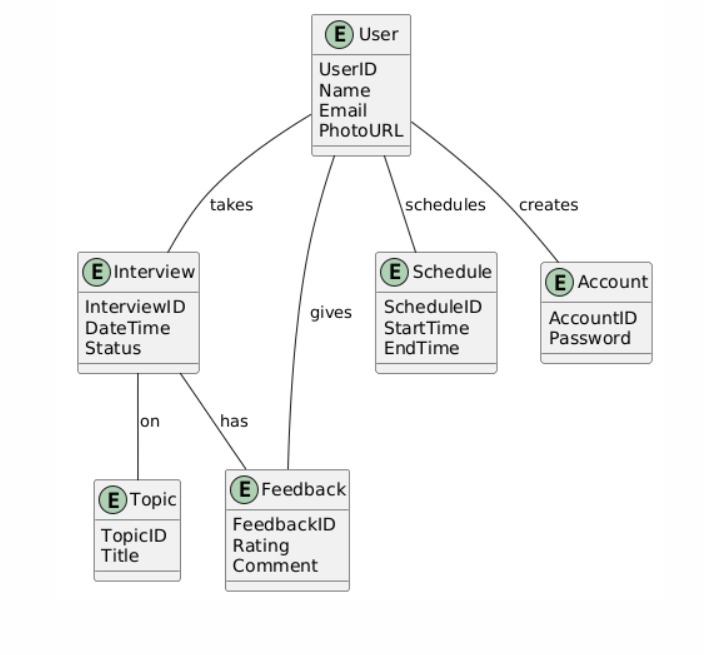


Fig. 2.3 E-R Diagram

In an Entity-Relationship (E-R) diagram, several symbols are commonly used to represent the various components of the diagram. Here are the most used symbols:

**Rectangle**: Represents an entity. An entity is an object or concept about which data is stored. For example, "Student," "Course," and "Teacher" are entities.

**Oval**: Represents an attribute. An attribute is a property or characteristic of an entity. For example, "StudentID," "Name," and "DateOfBirth" are attributes of the "Student" entity

**Diamond**: Represents a relationship. A relationship shows how two entities are related to each other. For example, an "enrolls in" relationship between "Student" and "Course."

**Line**: Connects attributes to entities and entities to relationships. It indicates the association between different components of the E-R diagram

**Double** **Oval**: Represents a multi-valued attribute, which can have multiple values. For example, a "PhoneNumbers" attribute might store more than one phone number for a person.

**Dashed Oval:** Represents a derived attribute, which can be derived from other attributes. For example, "Age" can be derived from "DateOfBirth."

**Double Rectangle:** Represents a weak entity, which depends on another entity for its existence. For example, a "Dependent" entity might depend on an "Employee" entity.

**Double Diamond:** Represents an identifying relationship, which links a weak entity to its owning entity. This is used to show that the weak entity cannot exist without the strong entity

**Ellipses:** In some variations, attributes can be represented by ellipses instead of ovals.

##### SEQUENTIAL DIAGRAM

A sequence diagram is a type of interaction diagram in Unified Modeling Language (UML) that shows how objects interact in a particular sequence to achieve a specific goal or outcome. It is particularly useful for understanding and documenting the dynamic behavior of a system by visualizing the flow of messages, events, and interactions between various components over time. A sequence diagram is a type of chart used to show how different parts of a system interact over time. It helps visualize the order in which events happen and how objects or people (actors) communicate with each other to perform a function.

**Here's a simple breakdown:**

1. **Lifeline**: A vertical line that represents an object or person involved in the interaction.
2. **Actors**: People or external systems that interact with the system, shown as stick figures.
3. **Messages**: Arrows that show the communication between objects or actors. They indicate who is sending and receiving information.
4. **Activation** **Bar**: A thin rectangle on the lifeline that shows when an object is doing something.
5. **Fragments**: Boxes that show special conditions like loops or choices (like if-else statements).

Sequence diagrams are useful because they show the step-by-step process of how a task is completed, making it easier to understand and communicate how a system works.

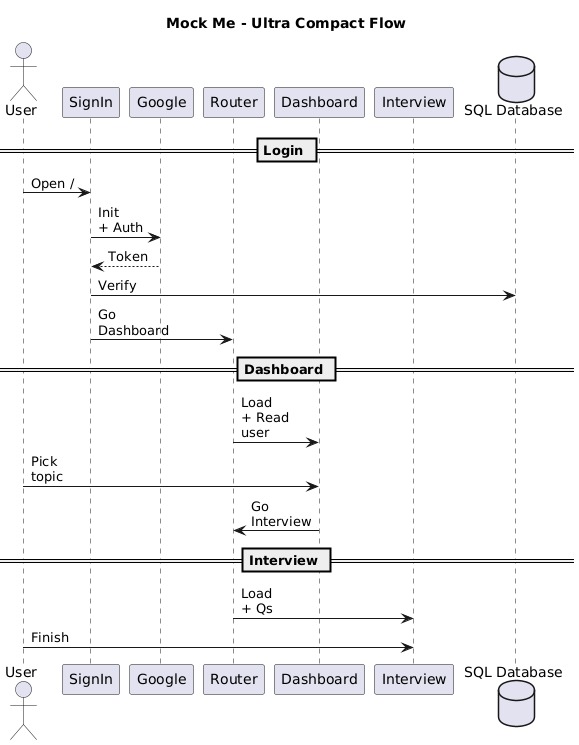


Fig. 2.4 SEQUENTIAL DIAGRAM

* + **Sequence Diagram Description**

This sequence diagram illustrates the interactions between a User and five core systems: the Login System, Dashboard System, Interview System and Firebase.

**1. User Login**

* The user opens the application.
* The SignIn component initializes and performs authentication via Google.
* Upon successful authentication, a token is generated.
* The Login System sends the token to Firebase for verification.
* Once verified, the user is redirected to the Dashboard.

**2. Viewing Dashboard**

* The Dashboard System loads and retrieves user data from Firebase.
* The user picks a topic for the interview.
* The system then routes the user to the Interview section.

**3. Interview Process**

* The Interview System loads and fetches the relevant questions from Firebase.
* The user completes the interview session.

.

## TECHNOLOGY STACK

#### INTRODUCTION

**Mock Me:** A Mock Interview Portfolio Platform is built using a modern and scalable technology stack to deliver a smooth, secure, and interactive user experience. The front-end is developed with HTML, CSS, JavaScript and Node.js ensuring a responsive and user-friendly interface across devices. For the back-end, the application leverages MySQL to manage user accounts, mock interview sessions, question banks, and performance analytics efficiently. SQL Workbench is utilized for structured database design and management. Security and data privacy are prioritized through robust authentication protocols and secure storage practices. This tech stack supports a seamless and reliable environment for users to practice, improve, and succeed in their interview journeys.

#### SOFTWARE REQUIREMENTS

**Operating System:** An operating system manages hardware and software resources, allowing applications to function smoothly and enabling interaction between users and the system. For this project, development and deployment were carried out on Windows environments.

**Web Server:** A web server processes client requests and delivers web content efficiently. This project uses Node.js and Express.js as the web server to handle API requests, manage user sessions, and facilitate backend communication within a scalable, JavaScript-based environment.

**Database**: A database is a structured collection of data that enables efficient storage, access, and management of information. It organizes data into tables with rows and columns, allowing easy retrieval and manipulation to support application functionality. For this project, we used MySQL, an open-source relational database management system (RDBMS) known for its speed, reliability, and simplicity. It uses Structured Query Language (SQL) to manage data related to users, interview sessions, making it ideal for dynamic and data-driven web applications like Mock Me.

**Backend Development:** Backend development focuses on the server-side logic that powers web applications, handling user requests, managing data, and ensuring smooth communication between the frontend and database. In Mock Me, we used Node.js and Express.js for backend development, enabling efficient routing, API handling, and seamless integration with the MySQL database. This setup ensures structured data management, secure user authentication, and real-time interaction between components for a scalable and responsive experience.

**Frontend Development**: Frontend development focuses on designing and building the interactive elements of a web application that users engage with directly. It involves creating a visually appealing and responsive interface using HTML5 for structure, CSS3 for styling, and JavaScript for interactivity. HTML5 defines the layout and structure of web pages, while CSS3 enhances the design through color, fonts, and adaptive layouts for various screen sizes. JavaScript adds dynamic functionality, ensuring seamless user interaction. In this project, we used HTML5, CSS3, and JavaScript to create an engaging, accessible, and user-friendly interface that enhances the overall experience.

**Development Environment:**

We have used Visual Studio Code in our project. It is a lightweight and powerful code editor that supports JavaScript, Node.js, HTML, and CSS. With built-in debugging, Git integration, and an extensive extension library, VS Code provides a flexible and efficient development environment for modern web applications.

##### 3.2 HARDWARE REQUIREMENTS

**Processor**: We used a multi-core processor such as Intel Core i5, providing strong computing power for handling backend operations and user interactions.

**Memory**: The system requires a minimum of 8 GB RAM, with 16 GB or more recommended for optimal multitasking and faster application performance.

**Storage**: We utilized an SSD with at least 256 GB capacity, ensuring quick data access and reliable storage for application files and database management.

**Network**: A reliable internet connection is essential for seamless communication, data transfer, and interaction between frontend and backend components.

**Peripherals**: Standard keyboard and mouse were used for input, along with additional monitors to enhance workspace efficiency and facilitate development task.

##### PACKAGS AND MODULES

Packages and modules help structure and organize code efficiently, enabling modularity and reusability in development.

**Packages:** In Node.js, packages are managed using npm (Node Package Manager) and provide essential functionalities that simplify the development process.

**Modules:** JavaScript modules allow encapsulation of functions, variables, and components into separate files, improving maintainability and scalability. Using CommonJS (require/module.exports) or ES6 imports (import/export), modules help organize backend logic effectively.

The following packages and modules were used in this project:

1. Node.js (Runtime environment for executing JavaScript on the server)
2. Express.js (Framework for handling server-side routing and API requests)
3. MySQL (Relational database management for storing user data)
4. Cors (Middleware for managing cross-origin requests)
5. Body-parser (Middleware for processing JSON request bodies)
6. Path (Module for handling file system paths)
7. Bootstrap (Front-end library for styling and responsiveness)

## IMPLEMENTATION

##### INTRODUCTION

Mock Me is built with a modern and scalable tech stack to provide a seamless and efficient user experience. The front-end utilizes HTML, CSS, and JavaScript for responsiveness and interactivity, ensuring a smooth and engaging interface. The back-end is powered by Node.js and Express.js, handling server-side logic and facilitating secure interactions with the MySQL database for structured data management. Strong authentication mechanisms and secure storage practices prioritize user data privacy and system integrity. The platform is designed for scalability, reliability, and performance, helping users practice, improve, and succeed in their interviews.

##### TEST-CASES

Test cases define structured scenarios that verify specific functionalities in a software application. They help ensure expected behavior under various conditions, detect defects, and validate system integrity before deployment. Each test case includes inputs, expected outputs, and steps for execution, along with necessary assumptions.

-> If API endpoints are not retrieving data properly, possible reasons include:

- **Incorrect Request Parameters:** Ensure query parameters and body data match expected formats.

- **Database Query Issues:** Verify SQL queries are correctly structured and aligned with the database schema.

- **CORS Restrictions:** If requests are blocked, confirm CORS settings in the backend.

- **Middleware Configuration**: Ensure request parsing middleware (e.g., body-parser) is correctly implemented.

- **Server Logs:** Check logs for specific errors related to request handling or database interactions.

-> A 404 error may occur when accessing API endpoints. Troubleshooting steps include:

- **Check Route Definitions:** Verify Express.js routes and ensure they match the expected request paths.

- **Database Connectivity:** Ensure MySQL database is running and correctly linked to the backend.

- **Request Format:** Confirm correct HTTP method (GET, POST, etc.) and request structure.

- **Server Restart:** After modifying API routes, restart the Node.js server to apply changes.

- **Check Console Logs:** Debug errors using server and browser console logs.

-> If Node.js server fails to start, common solutions include:

- **Check Port Availability:** Ensure no conflicting process is using the server port.

- **Fix Syntax Errors:** Debug for missing imports, misplaced brackets, or incorrect configurations.

- **Restart Services:** Restart the Node.js process and database service.

- Verify Environment Variables: Ensure API keys, database credentials, and configurations are correctly loaded.

- **Check Dependencies:** Use npm install to reinstall missing or corrupted packages.

-> The "404 Not Found" error when accessing front-end resources can be resolved by:

- **Check File Paths:** Ensure correct links to JavaScript, CSS, or media files in HTML.

- **Server Setup:** Confirm static file serving is correctly configured in Express.js.

- **Browser Console:** Inspect console errors for missing resource URLs or incorrect paths.

- **Network Issues:** Debug network requests to confirm proper API communication.

- **Deployment Validation:** Ensure assets and dependencies are correctly packaged during deployment.

* 1. **TESTING**

Testing in Mock Me is crucial for ensuring functionality, reliability, and usability. Various methodologies are implemented to validate platform performance and maintain a seamless user experience.

1. **Unit Testing:** Individual components, such as API endpoints, authentication flows, and user dashboards, are tested in isolation to ensure correct functionality and catch bugs early in development.
2. **Integration Testing:** The interaction between modules—including database operations, request handling, and API connectivity—is validated to ensure proper data flow and system behaviour.
3. **System Testing:** The complete platform is tested holistically, verifying user workflows, navigation consistency, mobile responsiveness, and performance across different devices and browsers.
4. **User Acceptance Testing (UAT):** End-users, including interviewers and candidates, participate in mock sessions to validate usability, feedback accuracy, and overall functionality, ensuring a seamless experience.
5. **Performance Testing:** The system is tested under various load conditions to confirm stability, scalability, and efficient handling of multiple concurrent interview sessions without delays.
6. **Security Testing:** Rigorous security assessments detect and resolve vulnerabilities. This includes testing authentication mechanisms, data encryption, access controls, and protection against SQL injection and cross-s

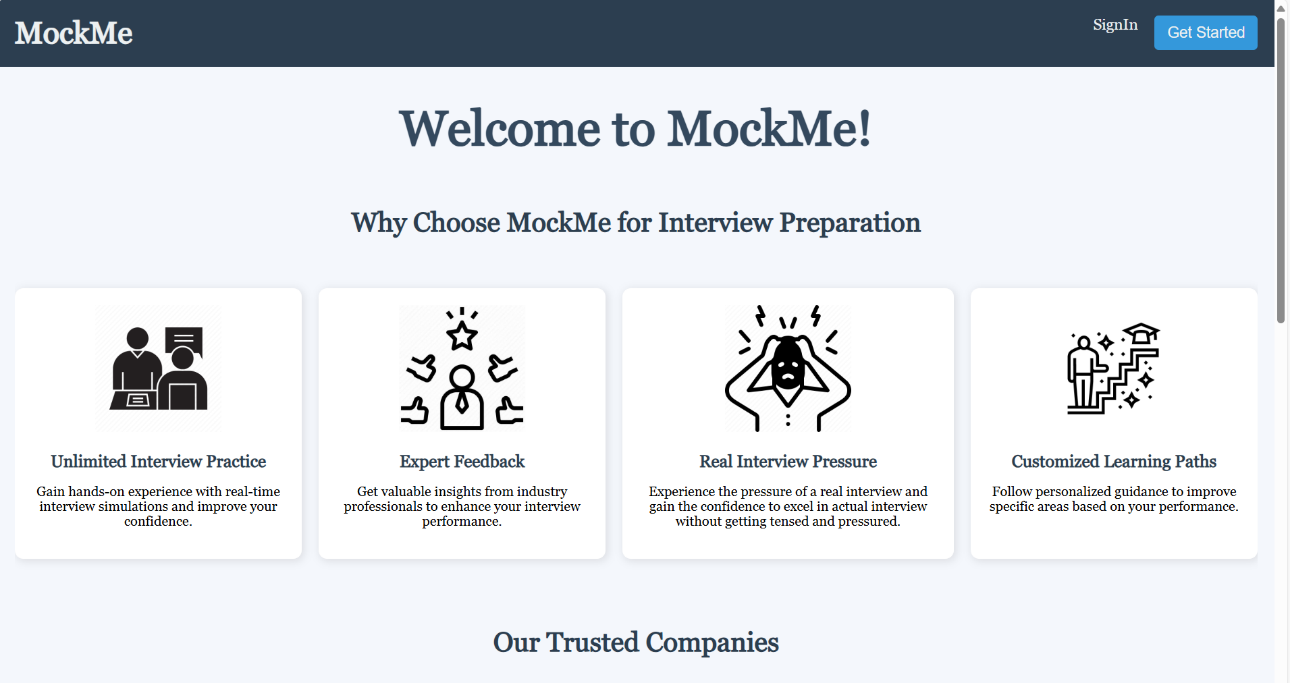


Fig.5.1 Home page

1.**Unlimited Interview Practice**

Gain hands-on experience with real-time interview simulations to improve your confidence and communication skills.

#### 2. Expert Feedback

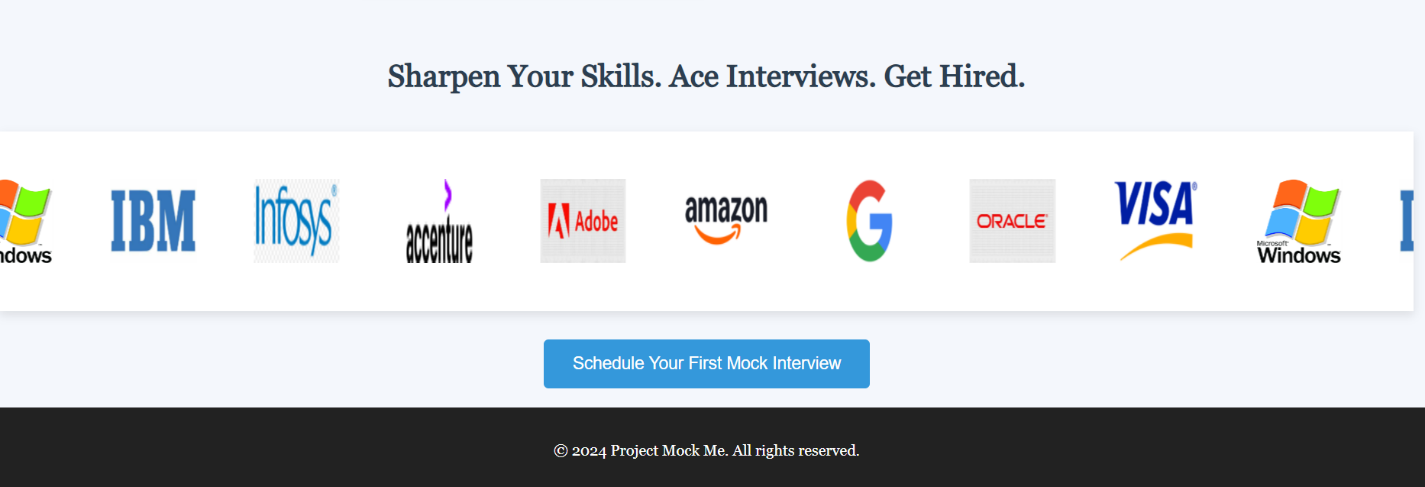
Get valuable insights from industry professionals to fine-tune your performance and understand what employers look for.

#### 3. Real Interview Pressure

Simulate the intensity of real interviews so you're fully prepared—no more surprises, just results.

#### 4. Customized Learning Paths

Receive personalized guidance to target your weak areas and enhance your strengths based on performance data.



**Sharpen your skills. Ace Interviews.Get Hired.**

Step into the shoes of a top candidate and practice interviews tailored to real hiring standards of leading companies like Google, Amazon, Infosys, IBM, Adobe, and more. Our platform offers a seamless experience that helps user build confidence, sharpen your skills, and get interview-ready. Whether user id aiming for internships or full-time roles, it's time to take that first step toward your career goals. “Schedule your first mock Interview” button takes the user to signin page with which user can choose domain and schedule the interview.

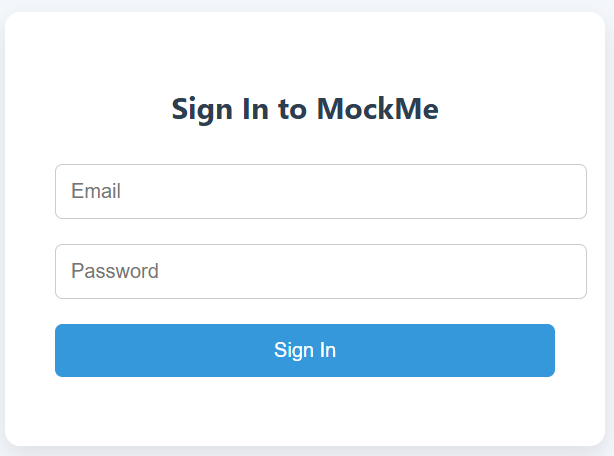


Fig 5.2 Sign-in page

##### Login Section:

* + **Email ID**: A text input field where users enter their email address.
  + **Password**: A text input field where users enter their password.
  + **Sign In button**: A button labeled "Sign In" that users click to submit their login credentials.

##### Registration Link:

* + At the bottom, there's a link for users who don't have an account, prompting them to "Register here."

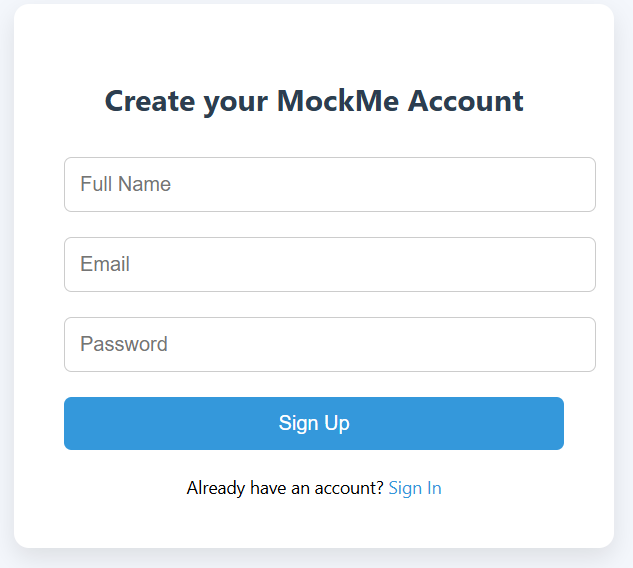


Fig 5.3 Sign Up page

Mock Me website signup page includes:

##### Sign Up Section:

* + **Email ID**: A text input field where users enter their email address.
  + **Your Name**: A text input field where users enter their name.
  + **Password**: A text input field where users enter their password.
  + **Sign Up Button**: A button labeled "Sign Up" that users click to submit their sign up credentials and move to the login page.

Server-Side Processing:

##### Form Submission:

* + The server receives the form data via a POST request.
  + The data includes the email, name, and password entered by the user.

##### Database Connection:

* + The server connects to the database where user details are stored.
  + The connection is usually established using a database driver (e.g., JDBC for Java, PDO for PHP).

##### Data Insertion:

* + The server checks if the email already exists in the database to prevent duplicate registrations.
  + If the email is unique, the server inserts the new user's details into the database.

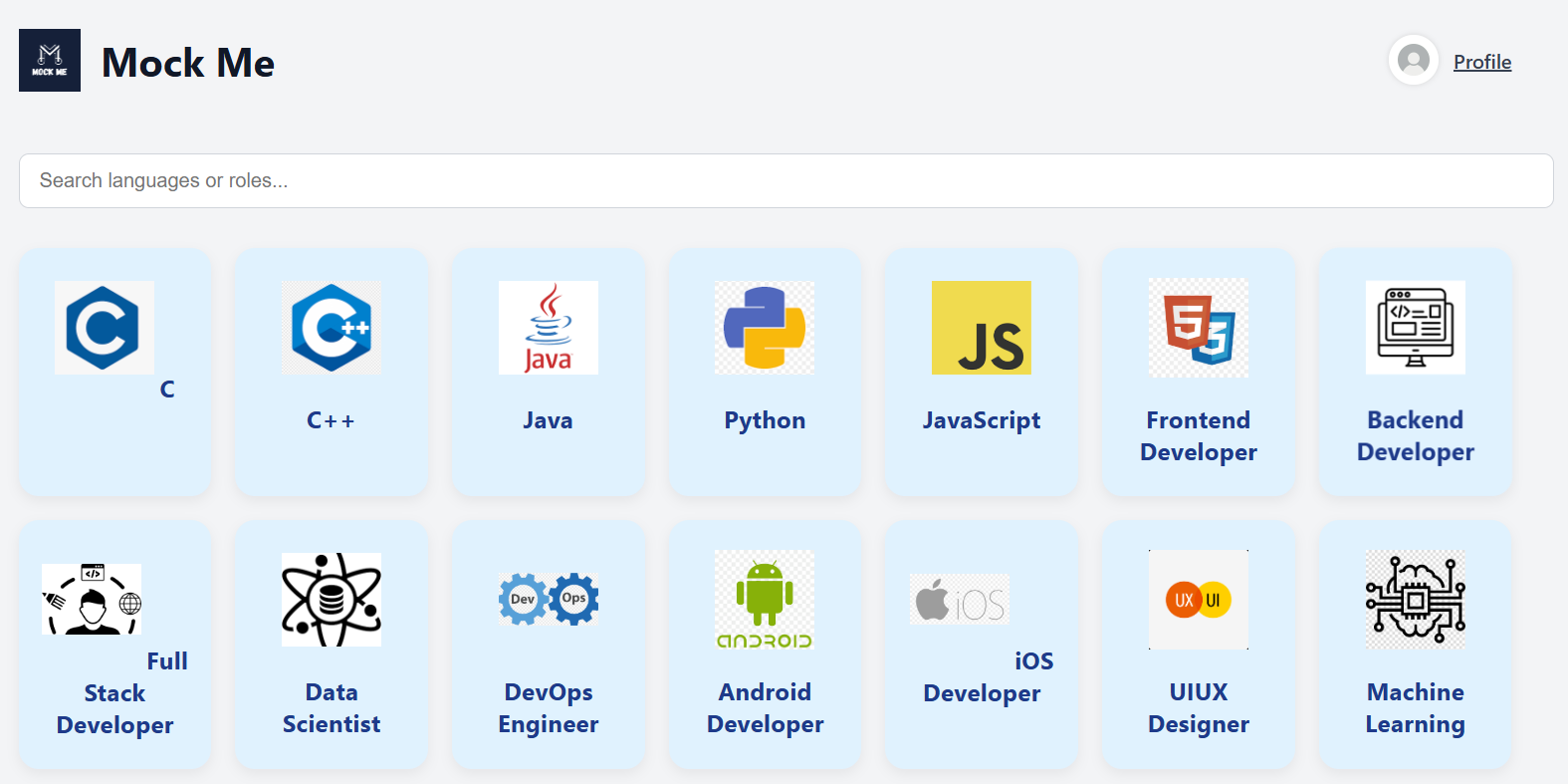


Fig. 5.4 Dashboard page

* Header Section:
* Logo: Top-left corner features the "Mock Me" logo with text.
* Profile Icon: Top-right corner has a user profile icon with the label "Profile" (likely for user account options like logout or settings).
* Search Bar:
* A central search input box labeled “Search languages or roles…”
* Allows users to quickly find a particular language, framework, or job role.
* Main Grid - Role Selection Tiles:
* Grid layout of clickable tiles, each representing a programming language or developer role.
* Each tile contains:
* An icon/logo for visual identification.
* The name of the language or role below the icon.
* These tiles are interactive and likely redirect to a mock interview or learning page.
* Displayed Roles/Languages:
* Languages: C, C++, Java, Python, JavaScript
* Developer Roles: Frontend Developer, Backend Developer, Full Stack Developer, Android Developer, iOS Developer, DevOps Engineer, Data Scientist, UI/UX Designer, Machine Learning Engineer, Cybersecurity, Cloud Engineering, AI, Database Developer, Game Developer
* Design Elements:
* Light, modern UI with soft blue background tiles and shadow effect.
* Clean, minimalistic layout designed for clarity and user-friendliness.
* Uniform tile sizes give the dashboard a structured look.

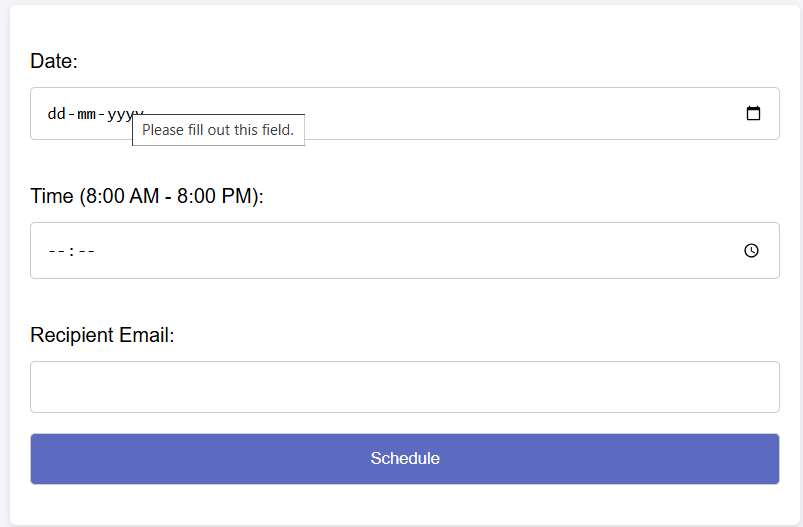


Fig.5.5 Scheduling Page

The screenshot displays the **Interview Scheduling** section of the "Mock Me” platform, designed to help users book a session with ease and clarity. Here's a breakdown of the elements present on this page:

**1. Date Picker**

* Users are required to select a **date** for scheduling their session.
* The input follows the format dd-mm-yyyy and includes a **calendar icon** to facilitate date selection.
* If left empty, the field prompts the user with a validation message: *"Please fill out this field."*

**2. Time Picker**

* Users must choose a **time slot** between **8:00 AM to 8:00 PM**.
* A clock icon is included to open a time selection widget for better accuracy and user experience.

**3. Recipient Email**

* This field is for entering the **email address** of the recipient who will receive the scheduling details.
* It ensures that the confirmation and meeting link (if any) are delivered directly to the recipient’s inbox.

**4. Schedule Button**

* Once all required fields are completed, users can click the **“Schedule”** button, styled in a clean purple background, to confirm their booking.
* The form ensures all fields are validated before submission, maintaining data accuracy and proper workflow.

**Purpose of the Page:**

This scheduling interface is crafted to provide a simple and intuitive experience for users to book interview sessions. It ensures:

* Clear time constraints for availability
* Input validation for accuracy
* Immediate scheduling through a responsive UI

This form is likely part of a larger system that sends confirmation emails and possibly generates calendar invites or links for interview session

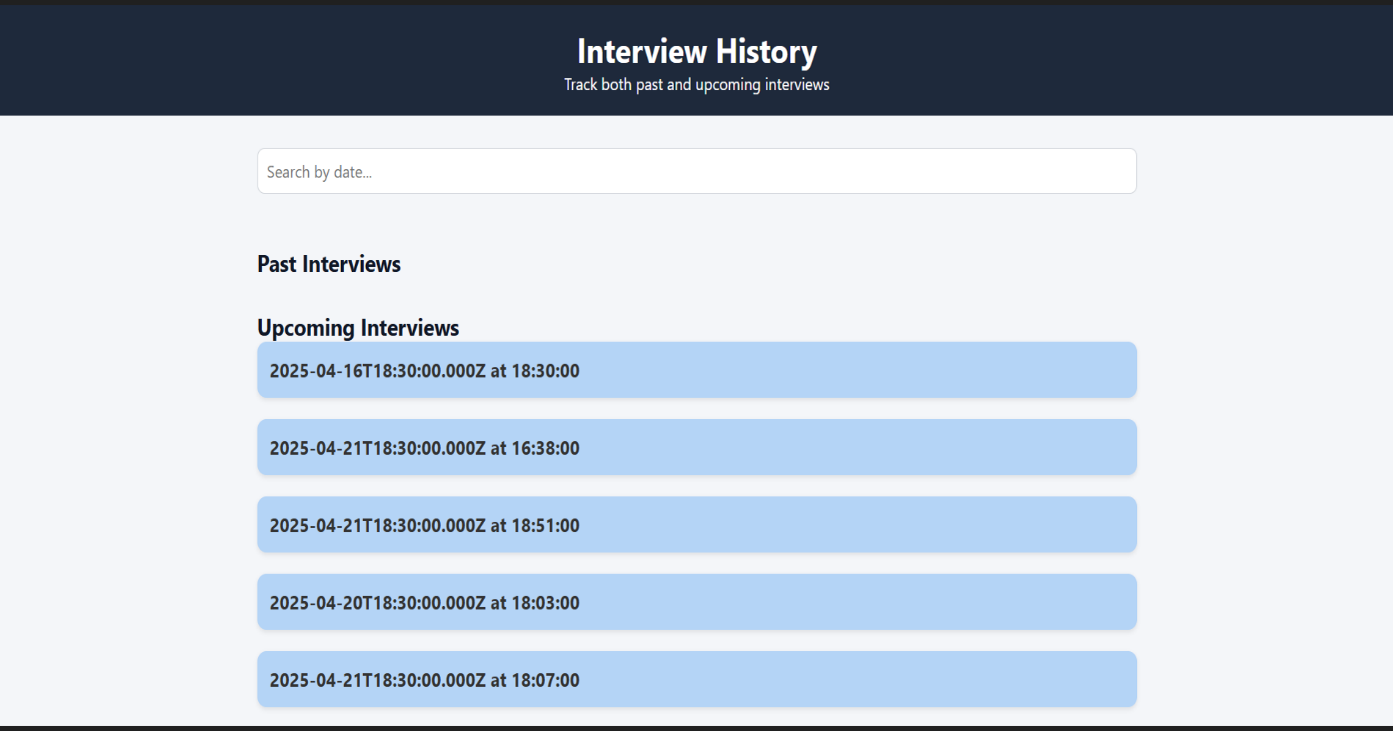


Fig 5.6 Interview History

The **Interview History** page is designed to help users track both **past** and **upcoming interviews** in a structured and user-friendly manner. It provides a clear timeline of interviews, allowing users to stay organized and prepared.

**Features:**

* **Search Functionality**:  
  A search bar at the top allows users to filter interviews by date, making it easier to locate specific entries.
* **Section Breakdown**:
* **Past Interviews**:  
  This section will display all interviews that have already occurred.
* **Upcoming Interviews**:  
  Displays a list of all scheduled interviews that are yet to occur. Each entry includes:
  + The **date and time** in ISO format.
  + The **readable time** for quick viewing.
* **Visual Design**:
* Upcoming interviews are shown in blue-highlighted cards for easy visibility.
* Clean and minimal layout with a dark header and light background for a pleasant user experience.

**5.** **CONCLUSION**

* 1. **CONCLUSION**

In conclusion, the Mock Me successfully integrates features designed to simulate real-world interview experiences while providing structured feedback to candidates. By incorporating functionalities such as manual interviewer evaluations, structured feedback forms, and an intuitive user dashboard, the platform delivers an interactive and practical approach to skill refinement. With a focus on security, accessibility, and seamless user interactions, Mock Me serves as an effective tool for job seekers looking to improve their interview performance. This initiative underscores the commitment to creating a technologically advanced yet user-centric platform that meets the evolving needs of candidates and recruiters, ultimately enhancing interview preparedness and career success.

**5.2 Future Scope for Mock Me**

Future improvements for Mock Me could include advanced analytics for tracking candidate progress across multiple mock interviews. Enhancing mobile accessibility and refining UI/UX design for better feedback visualization will be key priorities. Additionally, improving authentication systems to maintain evaluation integrity and expanding peer-to-peer mock interview collaboration will further enhance the platform’s effectiveness. Customizable interview templates for various industries will ensure Mock Me remains a versatile resource for professionals across different domains, continuously supporting career growth and skill development.

1. **REFERENCES**

• Node.js Documentation - [Node.js Official Docs](https://nodejs.org/en/docs/)  
• Express.js Documentation - [Express.js Official Guide](https://expressjs.com/en/guide/)  
• MySQL Documentation - [MySQL Official Docs](https://dev.mysql.com/doc/)  
• W3C HTML & CSS Standards - [W3C Web Design Standards](https://www.w3.org/standards/webdesign/)  
• Bootstrap Framework Documentation - [Bootstrap Docs](https://getbootstrap.com/docs/)  
• jQuery Documentation - [jQuery Official Docs](https://api.jquery.com/)  
• Popper.js Documentation - [Popper.js Official Guide](https://popper.js.org/)  
• Postman API Documentation - [Postman API Docs](https://learning.postman.com/docs/)