Implementation Document

for

FindIIT

Version 1.0

Prepared by Team WebVortex

Group 2: **Group Name: WebVortex**

Rudra Sinha	**230880**	**rudrasinha23@iitk.ac.in**
Siddharth Miglani	**231006**	**siddharthm23@iitk.ac.in**
Dharajiya Yug Harshadbhai	**230362**	**dharajiya23@iitk.ac.in**
Pratul Koolwal	**230782**	**pratulk23@iitk.ac.in**
Rohan Gauranga Potukuchi	**230864**	**rohangp23@iitk.ac.in**
Nischay Agarwal	**230705**	**nischaya23@iitk.ac.in**
Vihaan Sapra	**231149**	**svihaan23@iitk.ac.in**
Suyash Kapoor	**231066**	**suyashk23@iitk.ac.in**
Aarush Narendra Ghate	**230017**	**aarushng23@iitk.ac.in**
Shravan Agrawal	**230984**	**ashravan23@iitk.ac.in**

Course: CS253

Mentor TA: Hemang Mohanlal Khatri

Date: 28 March 2025

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Initial Draft (v1.0)	Rudra Sinha Siddharth Miglani Dharajiya Yug Harshadbhai Pratul Koolwal Rohan Gauranga Potukuchi Nischay Agarwal Vihaan Sapra Suyash Kapoor Aarush Narendra Ghate Shravan Agarwal	First Draft	28/03/25

1 Implementation Details

FindIIT is built using a three-layer architecture that is widely used in interactive applications:

- **Presentation Layer:** This layer consists of the interface elements with which users directly engage. For FindIIT, this is a React-based front end that provides a dynamic and responsive experience in the browser.
- **Application Layer:** This layer handles the business logic that converts user actions into functional outcomes. In FindIIT, this is achieved using Node.js and Express.js to create a RESTful API, bridging the front end and the database while managing core operations.
- **Data Layer:** This layer is responsible for data management, storing all relevant information. FindIIT utilizes MongoDB for flexible, document-oriented data storage, ensuring efficient handling of lost and found records.

This structured approach allows for clear separation of concerns, enabling FindIIT to scale and maintain its features effectively.

1. Presentation Layer (Project Unit):

Technologies Used:

- **Programming Languages:**
  - **HTML & CSS:** For structuring and styling the web pages.
  - **JavaScript:** The primary language for interactive behavior in the browser.
- **Framework:**
  - **React:** A JavaScript library for building user interfaces.
- **Libraries and Tools:**
  - **React Router:** Manages navigation between different views in the single-page application.
  - **Axios (or Fetch API):** Handles asynchronous HTTP requests to interact with the backend.
  - **Google Maps API:** To implement map and location based functionality.
  - **Webpack & Babel:** For module bundling and transpiling modern JavaScript syntax into browser-compatible code.
  - **npm:** Package manager that facilitates dependency management and script automation.
  - **Vite:** To speed up and simplify our development and testing process.

Justification of Choices:

- **React** was selected for its component-based architecture, which promotes reusability and easier maintenance. The Virtual DOM mechanism ensures efficient rendering, providing a smooth user experience even in highly interactive applications. Its large ecosystem, extensive community support, and constant updates make it a superior choice compared to other libraries or frameworks like Angular or Vue for this project.

- **React Router** simplifies navigation in a single-page application, while **Axios** provides a clean and promise-based approach to handle HTTP requests, which is generally preferred over older approaches like jQuery AJAX.

- **Webpack and Babel** streamline the development process by allowing modern JavaScript to be used safely in a production environment, optimizing assets and ensuring cross-browser compatibility.

2. Application Layer (Logic Unit)

Technologies Used:

- **Programming Language:**
  - **JavaScript (ES6+):** Used for server-side development.
- **Framework:**
  - **Node.js:** A runtime environment that enables JavaScript to run on the server.
  - **Express:** A minimalistic and flexible web framework for Node.js to create robust RESTful APIs.
- **Libraries and Tools:**
  - **Mongoose:** An Object Data Modeling (ODM) library for MongoDB, which simplifies schema design and data validation.
  - **JWT + bcrypt:** For implementing authentication strategies for user login to ensure secure access.
  - **ESLint:** For enforcing code quality and consistency.
  - **npm:** For managing dependencies and scripts.
  - **socket.io:** For implementing chatting between users.
  -

Justification of Choices:

- **Node.js** was chosen because it allows full-stack JavaScript development, streamlining development efforts by using a single language across both client and server sides. Its non-blocking, event-driven architecture makes it well-suited for applications that need to handle many simultaneous connections efficiently.

- **Express** stands out due to its lightweight nature and extensive middleware ecosystem, which provides a straightforward way to build RESTful APIs. It's less opinionated than frameworks like NestJS, giving developers the flexibility to structure their applications as needed.

- **Mongoose** provides a structured way to interact with MongoDB by defining schemas and models. This layer of abstraction reduces boilerplate code and simplifies error handling and data validation, making it easier to manage complex data interactions.

3. Data Layer (Data Unit)

Technologies Used:

- **Database System:**
  - **MongoDB:** A NoSQL, document-oriented database that stores data in JSON-like documents.
- **Database Interaction Library:**
  - **Mongoose:** Serves as the interface between the Node.js application and MongoDB.
- **Deployment Tools:**
  - **MongoDB Atlas:** Cloud-based MongoDB hosting that offers scalability, backup, and monitoring features.

Justification of Choices:

- **MongoDB** was selected due to its flexibility in handling varying data structures without requiring a predefined schema. This is particularly advantageous in a lost and found application, where item attributes might differ significantly.

- **Mongoose** further simplifies data operations by providing a schema-based solution to model application data. It also offers built-in validation and middleware, which help maintain data integrity and streamline database interactions.

- **MongoDB Atlas** provides a managed database service that reduces the operational overhead associated with database maintenance, ensuring that the application can scale securely and reliably with minimal administrative burden.

3.1 Authentication and Authorization

FindIIT secures user interactions by relying exclusively on JSON Web Tokens (JWT) for both authentication and authorization. When a user logs in, their credentials are verified, and a JWT is issued that encodes their identity along with any necessary claims. This token is then sent with each subsequent request, allowing the server to authenticate the user without the need for server-side session storage. The self-contained nature of JWTs not only simplifies state management but also ensures that every request is validated against potential security threats, providing a robust layer of protection for our application.
If while sending a request JWT token is not found or is expired, the user is redirected to the Login Page.

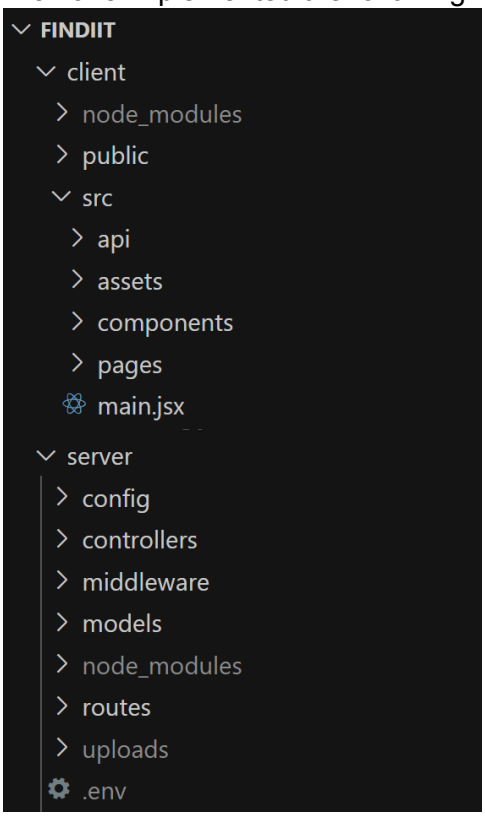
3.2 Version Control

We adopted Git for our version control needs, a tool renowned for its widespread use in software development. Git meticulously records every file change, providing a detailed history that allows for reverting to earlier versions when necessary. Its robust collaboration features enable multiple team members to integrate their contributions seamlessly. Additionally, we used GitHub as our web-based platform for repository management and team collaboration, streamlining our development workflow.

2 Codebase

We have made a repository on GitHub through which we can store and collaborate on the source code of this project FindIIT.

We have implemented the following file structure in the repository:



3 Completeness

FindIIT is a lost and found website tailored for the Indian Institute of Technology Kanpur (IITK) community. It is designed to be a centralized portal for reporting misplaced objects. Users can add category tags, photos, text description and location tags to ensure efficient matching of lost and found items. Users can also communicate via website to share additional details. This application is focused on making the reporting process as seamless and hassle free as possible.

Features Implemented:

I. **User Authentication:**
   - A. **Login**
   - B. **Register:**
     1. OTP generation and verification
     2. Set password
     3. Login
   - C. **Forgot password**

II. **Profile:**
   - A. **View Profile:**
     1. User information like Photo, Name, Roll number, Contact details, etc.
   - B. **First-time user:**
     1. Option to add Contact details, Photo, Room number, etc.
   - C. **Update/Edit Profile:**
     1. Modify fields like Photo, Room Number, Contact details
   - D. **Chat:**
     1. A chatting page where users can view their previous chats and also start new chats

III. **Homepage:**
   - A. **Navigation Bar:**
     1. Report Items:
        a) Prompt to create a report about a lost/found item
           (1) Includes fields such as Item Description, Time, and Location
           (2) Add tags to the report for easier searching
           (3) Option to upload image
     2. My Items:
        a) List of items reported as lost/found by the current user
     3. Map:
        a) Opens a map of the campus where items lost in a particular location can be viewed by clicking on a location
     4. View Profile
   - B. **Main Feed:**
     1. Search bar:
        a) Allows users to search to look for items in the feed
     2. Item Feed:
        a) Shows a list of lost and found items with their photo, tags and a short description
        b) On clicking an entry, users can see a detailed description of the item along with a comments section where people can talk with each other

The following Non-Functional Requirements have been met:

A. Memory Requirements:
   a. The application has a dedicated database to store all the user information and item details

B. Security Constraints:
   a. IITK email based user authentication has been used to ensure only people in the campus community can access the application. This ensures the legitimacy of the claims
   b. For enhanced security, passwords are stored in hashes

C. Communication Protocols:
   a. HTTP protocol has been used to communicate between servers, thus ensuring smooth transfer of information within the system.
   b. IITK Webmail has been used to send the OTPs to the users' email.

D. Software Quality Attributes:
   a. Reliability
   b. Performance
   c. Usability

The Following Features are Planned for Future Updates:

1. Matching similar items based on tags, and description (photos, location, etc)
2. Sending a mail notification to the user when similar items have been found
3. Automatic population/fetching of User Details based on IITK credentials
4. Adding an Admin user class to manage users
5. Expanding the scope of the website to include buying/selling and renting of products to make a comprehensive resource reallocation platform

Appendix A - Group Log

- We have made a WhatsApp group for effective communication amongst ourselves.
- We have created a private repository on GitHub on which we would be regularly pushing our codes to maintain efficiency and collaboration.
- We have broadly divided the team into parts, all members would be working on different aspects of frontend and backend so that all members understand the mechanism of software development and are able to implement it.

The following is the group log for the development of this Implementation Document

Date	Meet type	Final outcome of meet
11 February 2025	Offline	Decided the basic structure and theme of the webpages
3 March 2025	Offline	Fixed on a common database and communication framework
18 March 2025	Offline	Took an update on the progress
24 March 2025	Online	Integrated all features and worked on fixing bugs/improving UI
26 March 2025	Offline	Worked on Implementation Document and final touch ups on the website