Implementation Document

for

MapGo

Version 1.1

Prepared by

Group #: 6 **Group Name: Wall-E(s)**

Abhinav Garg	210029	abhinavg21@IITK.ac.in
Chouhan Jayesh	210296	chouhanj21@IITK.ac.in
Harshini Dola	210418	harshini21@IITK.ac.in
Mondem Shanwitha Yadav	210628	shanwitha21@IITK.ac.in
Nirmal Prajapati	210735	nirmal21@IITK.ac.in
Prachi Choudhary	210732	prachic21@IITK.ac.in
Priyanshu Meena	210785	priyanshum21@IITK.ac.in
Rohan Ravi	210870	rohanr21@IITK.ac.in
Sumit Kumar Bairwa	211074	sumitkb21@IITK.ac.in
Ujjwal Gautam	211122	ujjwalg21@IITK.ac.in
Yashas D	211199	yashasd21@IITK.ac.in

Course: CS253

Mentor TA: *Mr. Anirudh Nanduri*

Date: 20th March, 2023

CONTENTS II

REVISIONS II

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Group #6 Wall-E(s)	This document contains details about the implementation of the software MapGo.	20/03/23
1.1	Rohan Ravi	Updates in future development plans	23/04/23

1 Implementation Details

The software is developed using the MERN tech stack, which consists of MongoDB, Express, React, Node.js. The application hence is primarily written in JavaScript language, with Node.js and Express used for the backend and React for the frontend.

We utilize MongoDB as our database technology to store and manage data. It is a document oriented, non-relational database, with flexible data model. It provides high-scalability and performance as well. Mongo DB Atlas, the database hosting application provides features to deploy the database which would be beneficial at the deployment stage of the software.

Node.js is a server-side JavaScript runtime environment that enables building high-performance web applications.
Express is a lightweight flexible web application framework for Node.js, with simple and easy to use features for routing, middleware, and error handling.

React is a popular front-end library for building single-page user interfaces. It uses a virtual DOM, which makes it fast and efficient in rendering updates to the UI, giving the user a seamless experience. It provides a component-based architecture that makes it easy to reuse and maintain the UI code.

The development team prefers to use VScode as the code editor for writing and maintaining the software codebase, for its easy to store and manage environment. Vscode also provides a huge market of extensions which can be used to highlight, prettify the code giving the developers a better programming experience.

The development team prefers to use GitHub as the service to host our git repository, which provides ease while contributing.

2 Codebase

2.1 GitHub Link

https://github.com/r-dott/mapGo

2.2 How to navigate

The GitHub repository contains two folders: backend and view.

1 Backend
   1 Middleware: middleware functions are defined here which are used in certain routes.
   2 Models: contains the classes and methods for document objects that are stored in the mongo DB database.
   3 Routes: defines the routes that are accessed from the frontend of the application in order to get, post, or update any information such as that stored in the database.
   4 .env.example: is an example file format to be used by any developer to store variables to be used in the application. The actual .env file is not provided in the repository as it may credentials.
   5 .gitignore : details of files that are ignored when committing to the git repository.
   6 App.js is the main backend application.
   7 Package.json: contains, among other details, the details of all dependencies used in programming the backend of the application.

2 View
   This folder is structured in the way a react-app folder is structured.

   1 Public folder contains the index.html page which gets rendered whenever the react app is started.
   2 Src folder contains the component codes that are used to render the main index page.
   3 Frontend routes are also defined here.
   4 Package.json among other details contains the dependencies used in programming the frontend of the application.

For more details on how to run the application locally, or to contribute to the codebase, please follow the README.md file in the GitHub repository.

3 Completeness

3.1 Features completed

Functional requirements that are completed (please refer to SRS for more details*)
Numbering is done according to the software requirement specification document.

- 3.1.1 Login feature along with forgot password option.
- 3.1.2 Registering with a valid iitk email.
- 3.1.3 User can search and subscribe to hosts to follow the events announced by them.
- 3.1.4 Profile page with all upcoming events.
- 3.1.5 Homepage with campus map with event locations highlighted.
- 3.1.11 Super user creates verified host labels to organisations, as host creation can only be done by the super user, once request is mailed to them by the organisation.
- 3.1.12 Deletion of events once the end timing exceeds the current time.
- 3.1.13 Unsubscribe feature to allow users to not view the events of a particular host.

3.2 Future development ideas

Functional requirements that can shall be added in future versions: (refer to SRS for more details*)
Numbering is done according to the software requirement specification document.

- 3.1.6 Visitors' mode addition to allow outside user to use the application. Making registration possible even without IITK email, for users visiting the campus for specific events.
- 3.1.7 Addition of notifications feature to update the user about new addition of events by the host.
- 3.1.8 Chat box feature for each event for users to chat and enquire about an event.
- 3.1.9 Group feature to allow multiple users to announce events for members withing that group.

Additional improvements

- Improvement of UI design. Making page traversing easier for the end user, with clear instructions to the user of what each button, page, search bar, etc. is for.
- Optimized search algorithm of events and hosts, and user-specific suggestion, just like social media applications.
- Making each event public, so that a user may add it to their Events list without having to subscribe to the host. This requires host to create 2 types of events: "public to all" and "only to subscribers".

*: point number refers to the functional requirement mentioned in the Software Requirement Specifications document

Appendix A - Group Log

A.1 Group meetings

20th February 2023: group meeting to discuss language and framework to use.

3rd March 2023: meeting with TA, to discuss the finer details of carrying out implementation process and understanding what goes into software testing

5th March 2023: division of programming tasks.

15th March 2023: meeting with TA to seek help with issues while development.

A.2 Tasks

Application Backend	Rohan Ravi
Application Frontend	Ujjwal Gautam Rohan Ravi Abhinav Garg
Map Integration and features	Abhinav Garg
UI designing	Rohan Ravi
Implementation document	Rohan Ravi