Project Report

On

SAFAR
(Travel-log-web application)

Submitted in partial fulfillment for the award of Diploma in Advance Computing PG-DAC

Guided by:
Mr. Bhanu Pratap Singh & Mr. Ashish Sharma

Presented By

PRN	NAME
230930920001	Abhijit Bhangale
230930920005	Gopal Patil

Center for Development of Advance Computing (CDAC), C-DAC Pune

CERTIFICATE

This is to certify that the project work under the title **‘*SAFAR (Travel- log Web Application)*’** is done by **Abhijit Bhangale** and **Gopal Patil** in partial fulfilment of the requirement for award of Diploma in AdvancedComputing Course.

Mr. Bhanu Pratap Singh
Project Guide

Mr. Manoj Sharma
Course Coordinator

Mr. Ashish Sharma

Project Mentor

Date:
20/02/2024

ACKNOWLEDGEMENT

I wish to express my profound gratitude toward my mentors and all those who have helped me during this projectfor extending their support tirelessly. Owing to the massive encouragement provided my teachers, I have achieved fulfillment of this project.

I express my deep gratitude **Mr. Bhanu Pratap Singh** for his mentor- ship as well as ongoing support, I am thankful to his reviewing the entire manuscript with painstaking attention as well.

I would also like to express my profound gratitude to **Mr. Ashish Sharma** for giving his invaluable guidance and time enriching ideas throughout this project. And last but not the least, I also feel a deep sense of gratitude towards our Co-ordination **Mr. Manoj Sharma** for his affectionate encouragement and cooperation in accomplishing the project.

INDEX

INTRODUCTION

The SAFAR is a comprehensive platform designed to facilitate travel blogging, trip planning, and user engagement. It is built with React for the frontend, Spring Boot for the backend, and MySQL for the database. The app incorporates a range of features targeting both users and bloggers, enhancing the experience of sharing and exploring travel stories.

I. PURPOSE

Blogger Profiles: Create personalized profiles on SAFAR to showcase your travel experiences.

Multiple Blogs: Bloggers can write and manage multiple blogs, allowing for a diverse collection of travel stories.

Detailed Journals: Each blog can contain multiple logs, enabling bloggers to document their journeys with rich detail and immersive content.

Sharing Experiences: Bloggers can share their travel adventures with a global audience, providing a platform to inspire and connect with fellow enthusiasts.

Interactive Platform: Normal viewers can explore a variety of blogs, read through detailed logs, and engage with the content.

Comments and Interaction: Normal viewers have the ability to leave comments on blogs, fostering interaction and communication between the audience and bloggers.

Community Engagement: SAFAR serves as a hub for a community of travelers, allowing both bloggers and viewers to connect, share, and appreciate diverse travel experiences.

User-Friendly Interface: The platform is designed to be user-friendly, ensuring a seamless experience for both bloggers and viewers.

Global Exploration: Normal viewers can virtually explore different parts of the world through the eyes of various bloggers, broadening their understanding of diverse cultures and destinations.

Inspiration and Connection: The SAFAR is not just a platform, it's a source of inspiration and connection for the travel community, bringing together individuals who share a passion for exploration.

FEASIBILITY STUDY AND REQUIREMENTS

I. Technical Feasibility:

Assess the availability of necessary technologies and tools (React, Spring Boot, MySQL) for development. Ensure the compatibility of selected technologies for the intended features. Evaluate the technical expertise of the development team.

II. Operational Feasibility:

Analyze the practicality of the system in real-world scenarios. Consider the ease of system maintenance and updates. Evaluate the operational costs associated with hosting, maintenance, and user support.

III. Legal and Ethical Feasibility:

Ensure compliance with data protection laws and regulations. Address ethical considerations related to user privacy and content sharing. Assess potential legal challenges and develop strategies for compliance.

Requirements Description:

Modularity:

Achieved through the use of modular components and reusable code. Each module addresses a specific functionality, enhancing maintainability and scalability.

User Experience:

Emphasis on an intuitive and visually appealing interface for a positive user experience. User-centric design principles to facilitate ease of navigation and engagement.

MVC Architecture:

Adherence to the Model-View-Controller (MVC) architectural pattern. Separation of concerns for efficient code organization and better maintainability.

Ease of Access:

Accessibility features to ensure usability for users with disabilities. Responsive design for seamless access across various devices and screen sizes

Wide Availability:

The application offers a diverse range of travel-related features, serving the needs of both individual bloggers and travel enthusiasts.

Safar Explorer minimizes costs associated with traditional travel agencies or physical establishments, avoiding expenses such as office rent, inventory, and travel agents, while offering a comprehensive and accessible travel experience. Safar Explorer aspires to revolutionize the travel industry by providing a user-friendly, accessible, and cost-effective platform for bloggers and travel enthusiasts alike.

WHY REACT AND SPRINGBOOT

Benefits of Using Spring Boot with ReactJs :

Spring Boot and ReactJs offer multiple benefits when building full stack web applications:

High performance and scalability:

They are a powerful duo for high performance and scalability. Spring Boot's lightweight container is ideal for deploying and running applications, while ReactJs excels at rendering complex user interfaces efficiently.

Robust backend:

Spring Boot is ideal for developing enterprise-level applications as it offers a powerful and scalable backend for building APIs and microservices. It has extensive support for various data sources and allows easy integration with other projects, making it simpler to build microservices based architectures.

Efficient frontend development:

ReactJs simplifies frontend development by utilizing a component-based architecture, which allows for code reusability. This leads to faster development, easier maintenance, and improved user experience.

Easy integration:

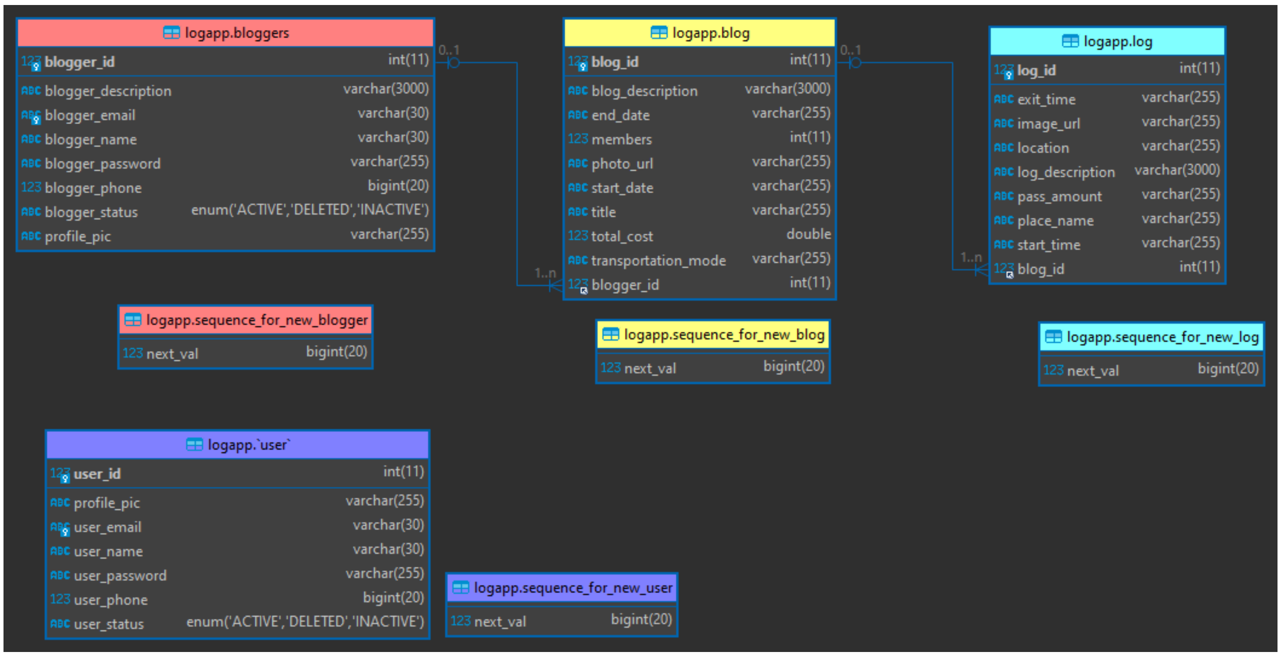
ReactJs has the ability to consume RESTful APIs from a Spring Boot backend using HTTP libraries like Axios, fetch, and superagent simplifies data communication.

Large community:

They both have large and active developer communities that provide useful resources, support, and up-to-date information.

ERD AND DFD

1. Entity Relationship Diagram:



1. Blogger Entity
Relationships:
One-to-Many with Blog: Each blogger can have multiple blogs.

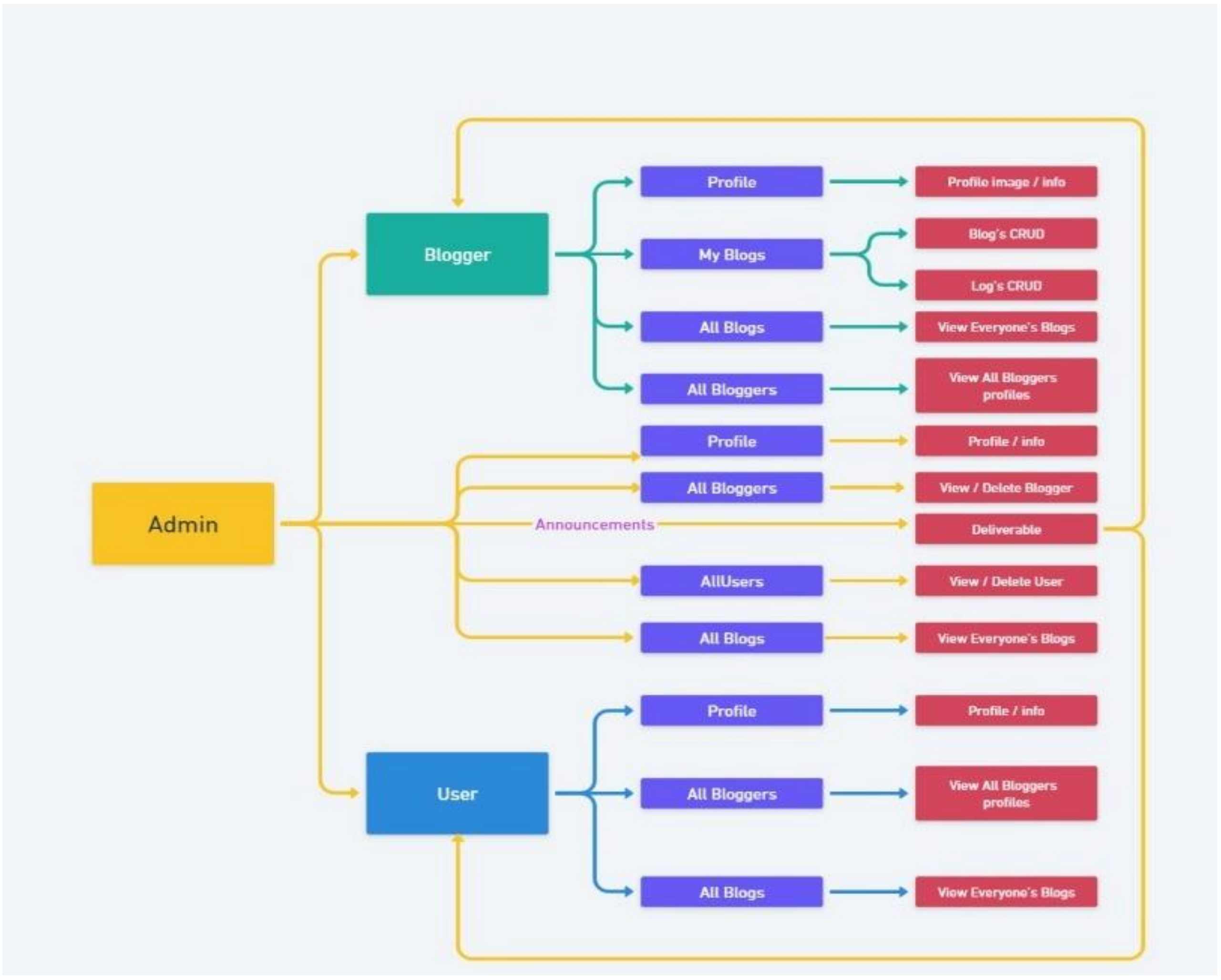
2. Blog Entity
Relationships:
Many-to-One with Blogger: Many blogs can be associated with one blogger.
One-to-Many with Log: Each blog can have multiple logs.

3. Log Entity
Relationships:
Many-to-One with Blog: Many logs can be associated with one blog.

4. User Entity

Entity	**Relationships**
Blogger	One-to-Many with Blog
Blog	Many-to-One with Blogger, One-to-Many with Log
Log	Many-to-One with Blog
User	

I. Data Flow Diagram:



<u>Deliverable</u>:

Broadcasts:
Admins can send broadcasts to all users, bloggers, or a specific group.
Broadcasting important announcements, updates, or news related to the application.

Announcements:
Feature for creating and publishing announcements visible to all users.
Announcements can include upcoming events, new features, or general information.

Content Moderation:
Tools for content moderation to ensure compliance with community guidelines.
The ability to flag, review, and take action on inappropriate content.

Collaborations and Partnerships:
Managing collaborations with other entities or organizations.
Exploring partnership opportunities to enhance the application's offerings.

DATABASE DESCRIPTION

Database Design and Creation:

MySQL was chosen as the database management system for the TravelLogApp. MySQL is a popular relational database known for its performance, scalability, and reliability. It offers several advantages over NoSQL databases, such as:

1. ACID Compliance: MySQL ensures Atomicity, Consistency, Isolation, and Durability, which guarantees data integrity and consistency.

2. Relational Structure: MySQL follows a relational structure, allowing for efficient data organization and retrieval through the use of tables, rows, and columns.

3. Scalability: MySQL can handle large amounts of data and can be easily scaled to accommodate growing application needs.

4. Mature and Stable: MySQL has been around for a long time and has a large community of users and developers, making it a reliable choice for database management.

Java Database Connectivity:
To establish a connection between the Springboot backend and the MySQL database, the application.properties file was used. This file contains the necessary configurations, including the database URL, username, password, and dialect. The dialect specifies the MySQL database driver to be used instead of the default my-sql-connector jar.

Choice of Java for Backend:
Java is widely recognized for its robustness, security, and versatility. Here are key reasons for choosing Java for the backend:

Security: Java provides a secure runtime environment, making it a preferred choice for applications that handle sensitive data.

Portability: Java's "Write Once, Run Anywhere" (WORA) principle allows the backend code to be executed on any platform, enhancing the application's portability.

Scalability: Java's scalability features make it suitable for handling increasing workloads, ensuring optimal performance as the application grows.

Community Support: Java has a vast and active community, providing ample resources, libraries, and support for developers.

Enterprise-Level Development: Java is extensively used in enterprise-level applications, making it a reliable and proven technology for building robust backend systems.

Introduction to Spring:
Spring is a comprehensive framework for Java development, addressing various concerns such as dependency injection, aspect-oriented programming, and modularization.

Dependency Injection (DI): Spring promotes the use of Dependency Injection, enhancing modularization and making components loosely coupled.

Aspect-Oriented Programming (AOP): AOP in Spring allows the separation of cross-cutting concerns, promoting cleaner and more maintainable code.

Modularity: Spring's modular architecture enables developers to use only the components they need, promoting a lightweight and efficient application design.

IoC Container: Inversion of Control (IoC) in Spring manages object creation and dependencies, simplifying the application's configuration and improving testability.

Introduction to Spring Boot:
Spring Boot is an extension of the Spring framework that simplifies the development of production-ready applications. Key features include:

Convention over Configuration: Spring Boot follows a convention-over-configuration approach, reducing the need for developers to specify configurations explicitly.

Embedded Servers: Spring Boot includes embedded servers (like Tomcat), eliminating the need for external server configurations.

Auto-Configuration: Spring Boot automatically configures application components based on the project's dependencies, reducing manual setup efforts.

Microservices Architecture: Spring Boot is well-suited for microservices architecture, allowing developers to build and deploy independent, scalable services.

Java Persistence API (JPA):
JPA is a Java specification for object-relational mapping (ORM) that simplifies database interactions in Java applications.

Object-Relational Mapping (ORM): JPA simplifies the translation of Java objects into relational database structures and vice versa, minimizing the need for manual SQL queries.

Database Abstraction: JPA provides a database-agnostic interface, allowing developers to switch between different databases without changing the application code.

Entity Relationship Mapping (ERM): JPA supports the mapping of Java entities to database tables, streamlining database operations.

Transaction Management: JPA handles transactions, ensuring data consistency and integrity in a multi-user environment.

By combining Java, Spring, Spring Boot, and JPA, the backend development benefits from a secure, scalable, and maintainable architecture, laying the foundation for building a robust and efficient travel log application.

Sr.No.	Annotations	Description
1.	@SpringBootApplication	@Configuration + @ComponentScan + @EnableAutoConfiguration
2.	@Autowired	Used to automatically inject dependencies
3.	@RestController	Indicates that the class is a controller that handles HTTP requests
4.	@RequestMapping	Maps HTTP requests to handler methods in a controller
5.	@CrossOrigin	Allowing HTTP requests from different origins to access the resources
6.	@PostMapping, @GetMapping	Used to map HTTP POST requests, used to map HTTP GET requests
7.	@PutMapping, @DeleteMapping	Used to map HTTP PUT requests, used to map HTTP DELETE requests
8.	@ModelAttribute	Binds method parameters or method return values to model attributes
9.	@PathVariable	Used to extract values from URL
10.	@Service	To mark class as a service
11.	@Transactional	It is used to perform units of work in a transaction meaning that they either complete successfully or leave the system in a consistent state if an error occurs.
12.	@Query	To provide user defined query

13.	@RequestParam, @Param	Used to bind request parameters to method parameters
14.	@RequestBody	Used to bind the body of an HTTP request to a method parameter
15.	@NoArgsConstructors, @AllArgsConstructors	To create a default and parameterized constructors respectively
16.	@Entity	Used for accessing data between Java objects and relational databases
17.	@GeneratedValue, @SequenceGenerator	To create auto-incrementing user-defined ID
18.	@JsonBackReference, @JsonManagedReference	Is used to prevent infinite recursion in bi-directional relationships, used to indicate the forward part of a bidirectional relationship

Dependencies

React dependencies:

react-router-dom	Used for navigation and URL management
Axios	fetching and sending data between client and server
Bootstrap and React bootstrap	CSS Framework
File Saver	saving files from the browser and exporting as CSV/PDF
React Cookie Consent	For managing cookie
react-toastify	displaying notifications or toasts in React application
react-speech-recognition	For speech-to-text functionality
react-to-print	Use for printing content from react component

Spring Boot Dependencies:

Spring Web	Useful for creating Restful API's and handling HTTP requests
Spring boot Dev Tools	enhance the development experience
Spring Data JPA	simplifies data access
Lombok	helps reduce boilerplate code
MySQL Driver	Used to connect a Spring application to a MySQL database.
Spring security	Spring-security defines an Enum BCryptVersion inside the BCryptPasswordEncoder class. In spring-security, the default strength of the Bcrypt algorithm is 10. The salt is random, and the default version is dollar 2a.

Coding and Implementation

API:

During the implementation phase, several components were created in the backend using Springboot and JPA. These components include:

1. Entities: The entities represent the database tables and their relationships. In the case of the Travel Log App(SAFAR), entities like Blogger, Blog, and Log were created using annotations like @Entity, @OneToMany, and @ManyToOne to define the relationships between them.

2. DTOs (Data Transfer Objects): DTOs were created to transfer data between the frontend and backend. They help in decoupling the frontend and backend, ensuring that only the required data is transferred.

3. Repositories: Repositories were created to handle database operations like CRUD (Create, Read, Update, Delete). They provide an interface to interact with the database using JPA methods.

4. Services: Services were created to handle the business logic of the application. They encapsulate the logic and interact with the repositories to perform database operations.

5. Controllers: Controllers were created to handle API requests and responses. They define the API endpoints and map them to the corresponding methods in the services.

To ensure robustness and error handling, exceptions were handled using appropriate exception handling mechanisms in the controllers. This helps in providing meaningful error messages to the frontend and handling exceptions gracefully.

Parallel API Calls using Postman:
To test the API endpoints, Postman was used to make parallel API calls. Postman is a popular tool for testing APIs and allows developers to send requests, view responses, and analyze the performance of the APIs. By making parallel API calls, the performance and scalability of the backend can be tested, ensuring that the application can handle multiple requests simultaneously.

Roles-Based Authentication:
The TravelLogApp implemented roles-based authentication, with three roles: admin, blogger, and user. Admin has full access to the application, bloggers can create and manage their blogs, and users can view and interact with the blogs. This ensures that the application is secure and restricts access to certain functionalities based on user roles.

TESTING :

Functional Test Cases

- Verify that all mandatory fields are marked as such and cannot be left blank.
- Validate that the registration form accepts valid and unique email addresses.
- Test if the password field contains a requirement for minimum input length.
- Test if the registration form displays the right error messages for invalid or wrong inputs.
- Test if the registration process allows users to enter special characters in fields where applicable.
- Validate that the form prevents registration with an already registered email address.
- Test if the registration form supports different input formats, such as uppercase, lowercase, and mixed case.
- Validate if the registration form handles leading and trailing spaces appropriately.
- Test if the registration page has proper validation for phone number formats.
- Test if the registration process includes encryption of sensitive user information.

Non-Functional Test Cases

Usability Testing:

- Test the user-friendliness of the registration page by assessing the clarity and readability of instructions displayed on the page.
- Validate the registration form for its ease of navigation.
- Test the consistency of the page design and layout.
- Verify if the error messages are clear with the right information.
- Test the accessibility of the registration page for users with special needs, adhering to accessibility guidelines.

Security Testing:

- Test if the registration page securely handles sensitive user data, such as passwords and personal information (password encryption applied).

Compatibility Testing:

- Validate that the registration page is responsive and displays correctly on multiple devices.
- Validate the functionality and compatibility of any external APIs or services used in the registration process.

Localization and Internationalization Testing:

- Validate if the page correctly handles special characters, date formats, and numerical representations based on the selected language/locale.

Error Handling and Recovery Testing:

- Test the registration page's ability to handle unexpected errors and exceptions efficiently.
- Validate that error messages are displayed timely and clearly with the right information.

Stress Testing:

- Validate the behavior of the registration page when subjected to a high volume of simultaneous registration requests.

Frontend Development with React:

React was chosen as the frontend technology for the TravelLog App(SAFAR). React is a JavaScript library that allows developers to build user interfaces efficiently. It provides a component-based architecture, allowing for the creation of reusable UI components.

To enhance the frontend design, Bootstrap and React-Bootstrap were used. Bootstrap is a popular CSS framework that provides pre-styled components and a responsive grid system. React-Bootstrap is a library that integrates Bootstrap components into React, making it easier to use and customize them.

FontAwesome icons were added to the frontend using CDN links. FontAwesome provides a wide range of icons that can be easily integrated into the application, enhancing the visual appeal and user experience.

Routing through react-router-dom:

React-router-dom was used for routing in the TravelLogApp. It allows for navigation between different pages or components within the application. By defining routes and associating them with specific components, users can navigate through the application seamlessly.

API Calls using Axios:

Axios, a popular JavaScript library, was used for making API calls in the TravelLog App(SAFAR). Axios provides a simple and intuitive API for sending HTTP requests and handling responses. It offers advantages over the fetch method, such as automatic JSON parsing, support for request cancellation, and better error handling.

Async & Await or Promises:

To handle asynchronous requests and responses, the TravelLog App(SAFAR) utilized the async/await feature in JavaScript. Async/await allows for writing asynchronous code in a synchronous manner, making it easier to handle promises and avoid callback hell. This ensures a smoother and more readable code flow, especially when making API calls and handling responses.

Future Enhancements

Enhanced User-Friendly Environment:
Continuous improvement of the user interface for both bloggers and regular users.
Intuitive search functionalities to quickly access relevant travel information.
User-centric design updates for a seamless and enjoyable experience.

Offline Accessibility:
Implementation of a feature allowing users to save travel logs and blogs as downloadable PDFs or other user-friendly file formats.
Offline access to travel information becomes crucial for users in areas with limited or no network connectivity.

Community Engagement Features:
Integration of features that foster community engagement, such as forums, discussion boards, or Q&A sections.
User ratings and reviews to highlight exceptional blogs and travel logs.

Smart Recommendations:
Incorporation of smart algorithms for personalized travel recommendations based on user preferences and travel history.
Intelligent suggestions for destinations, accommodations, and activities.

Interactive Maps:
Integration of interactive maps to visualize travel routes, points of interest, and geographic details within blogs and logs.
Geotagging for easy navigation and exploration.

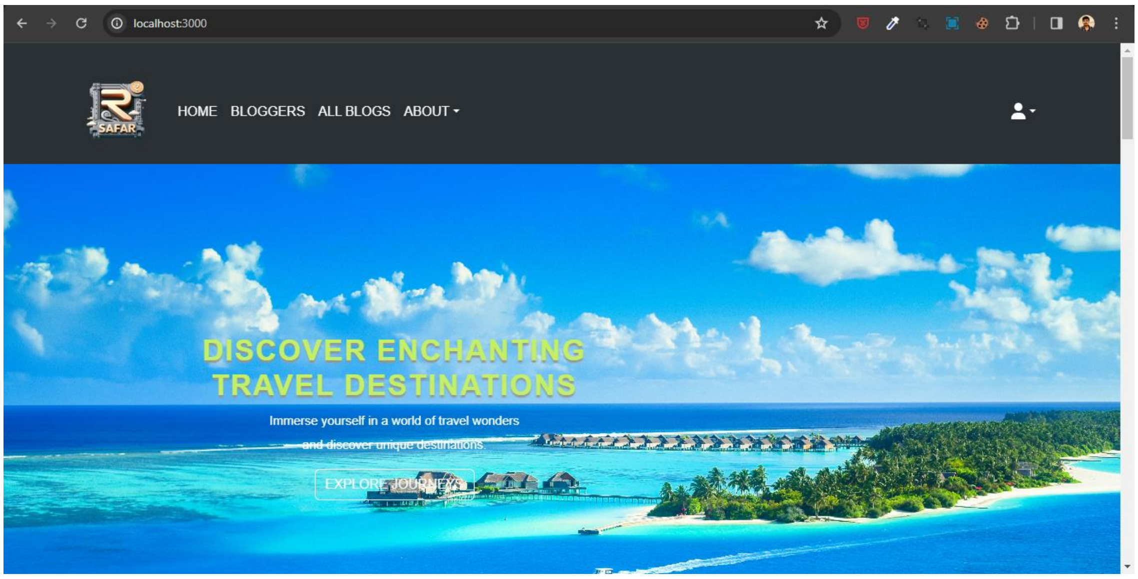
Mobile Application:
Development of a dedicated mobile application for seamless on-the-go access.
Mobile app features to enhance user engagement and provide notifications for new content.

Monetization Opportunities:
Exploration of potential monetization avenues, such as sponsored content, affiliate marketing, or premium features.
Collaboration with travel-related businesses for mutually beneficial partnerships.

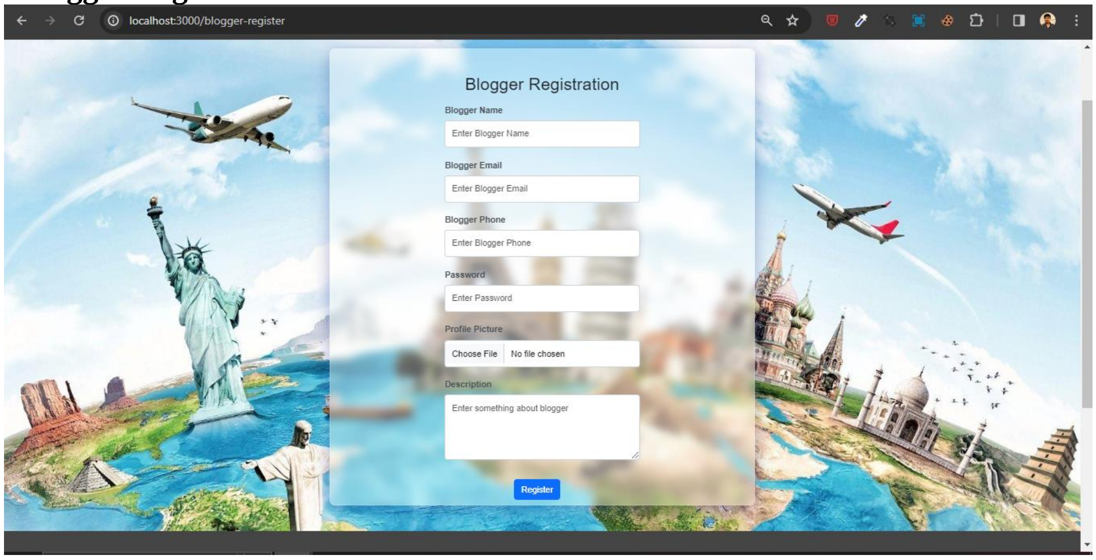
Privacy and Security Measures:
Implementation of robust privacy settings to ensure users have control over the visibility of their content.
Integration of advanced security measures to protect user data and maintain trust within the community.
These future enhancements aim to elevate the Travel Logs App into a comprehensive and indispensable tool for both bloggers and users, promoting seamless access to travel information in various scenarios, and fostering a vibrant community of travelers.

USER INTERFACE

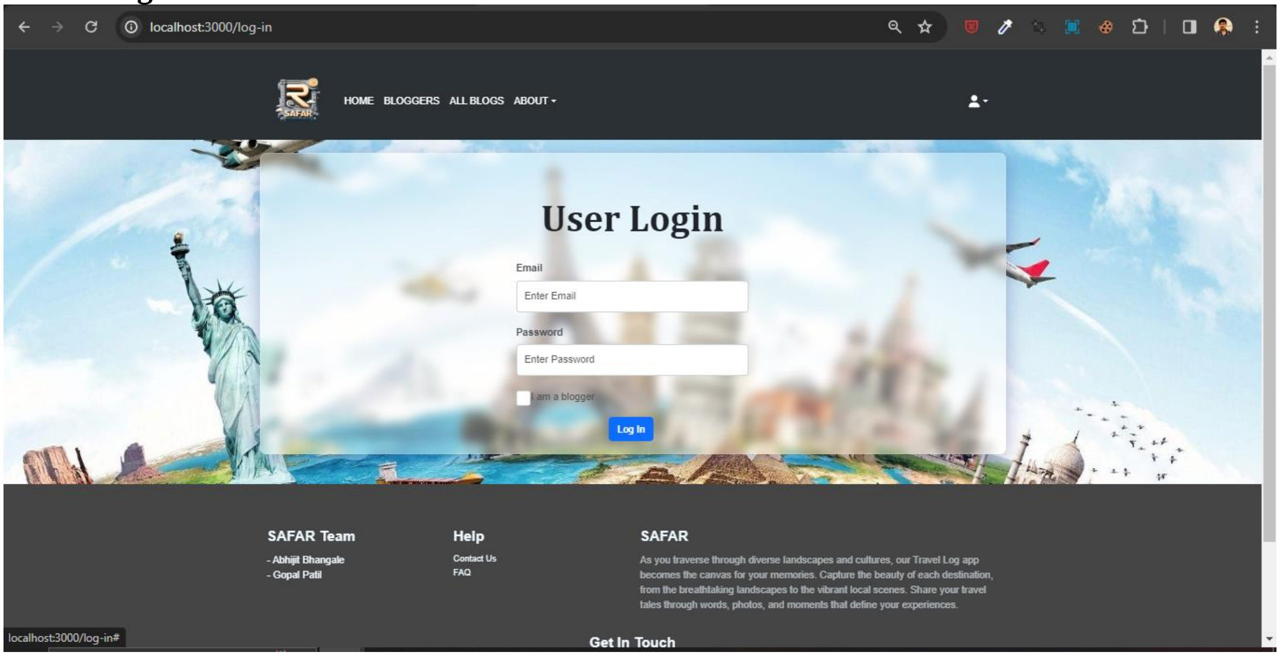
1. Home Page


localhost:3000
SAFAR
HOME
BLOGGERS
ALL BLOGS
ABOUT
DISCOVER ENCHANTING TRAVEL DESTINATIONS
Immerse yourself in a world of travel wonders and discover unique destinations.
EXPLORE JOURNEYS

2.Blogger Registration:

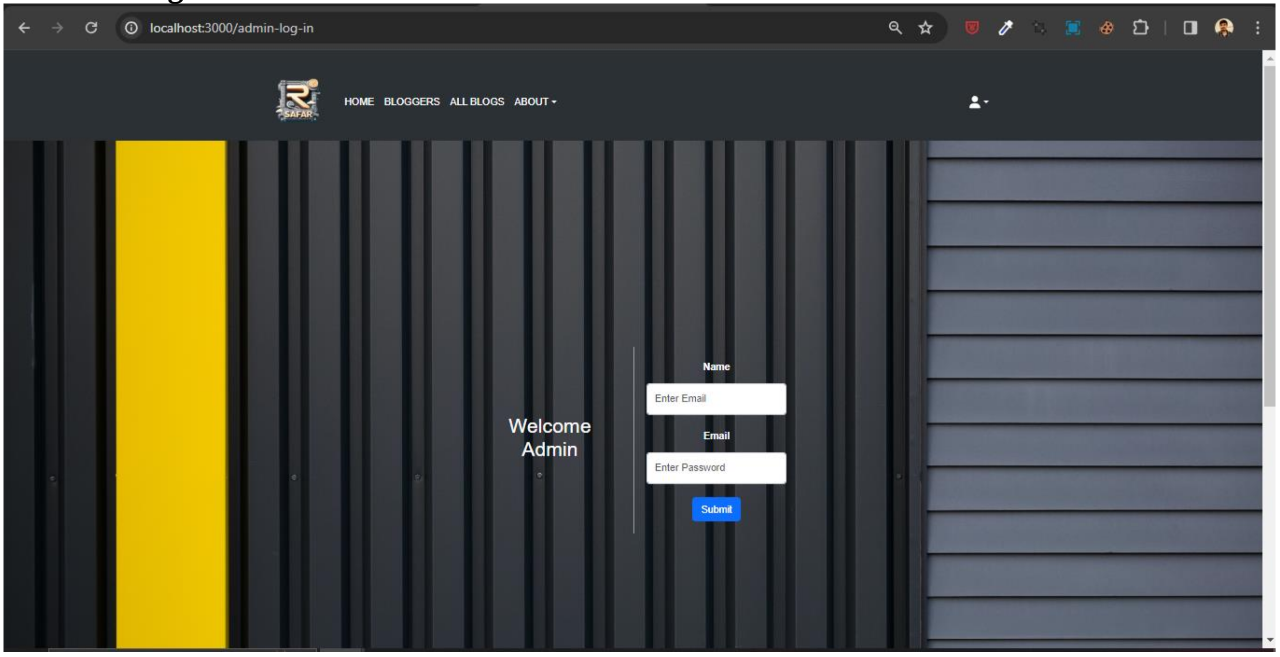


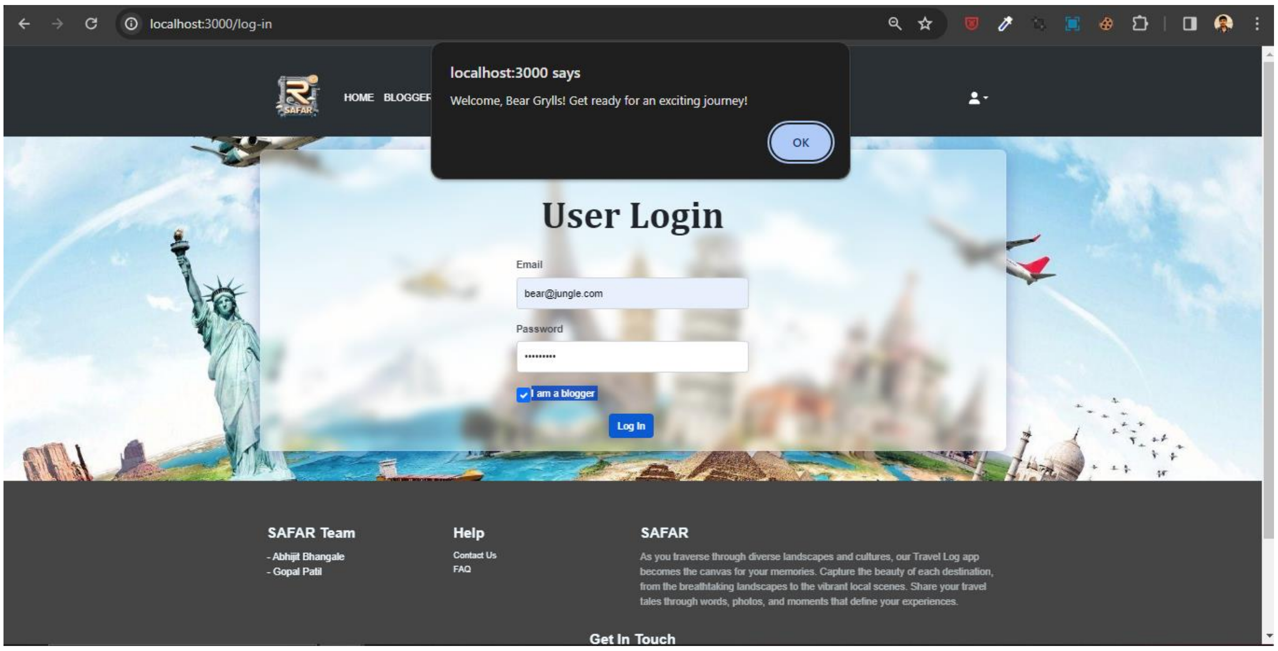
3.User Login:



As you traverse through diverse landscapes and cultures, our Travel Log app becomes the canvas for your memories. Capture the beauty of each destination, from the breathtaking landscapes to the vibrant local scenes. Share your travel tales through words, photos, and moments that define your experiences.

4.Admin Login:




localhost:3000/log-in

localhost:3000 says

Welcome, Bear Grylls! Get ready for an exciting journey!

OK

User Login

Email: bear@jungle.com

Password: ••••••••••

[✓ I am a blogger]

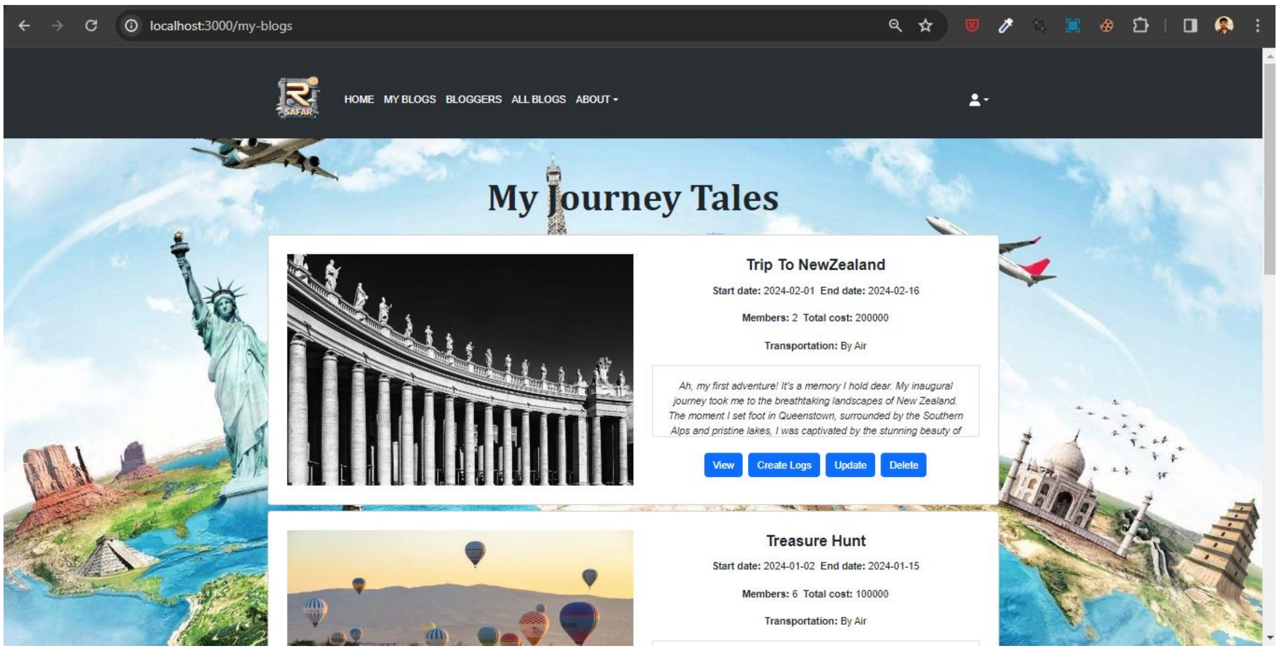
Log In

SAFAR Team
- Abhijit Bhangale
- Gopal Patil

Help
Contact Us
FAQ

SAFAR
As you traverse through diverse landscapes and cultures, our Travel Log app becomes the canvas for your memories. Capture the beauty of each destination, from the breathtaking landscapes to the vibrant local scenes. Share your travel tales through words, photos, and moments that define your experiences.

Get In Touch


localhost:3000/my-blogs

HOME MY BLOGS BLOGGERS ALL BLOGS ABOUT

My Journey Tales

Trip To NewZealand

Start date: 2024-02-01 **End date:** 2024-02-16

Members: 2 **Total cost:** 200000

Transportation: By Air

Ah, my first adventure! It's a memory I hold dear. My inaugural journey took me to the breathtaking landscapes of New Zealand. The moment I set foot in Queenstown, surrounded by the Southern Alps and pristine lakes, I was captivated by the stunning beauty of

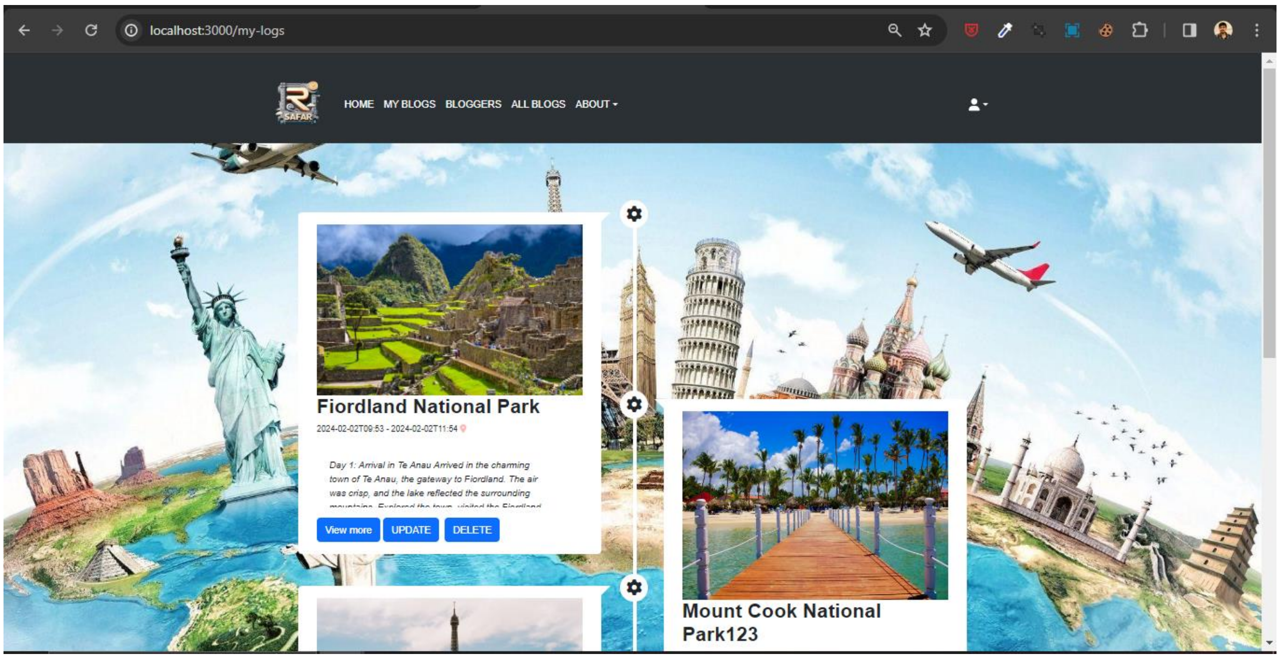
View Create Logs Update Delete

Treasure Hunt

Start date: 2024-01-02 **End date:** 2024-01-15

Members: 6 **Total cost:** 100000

Transportation: By Air

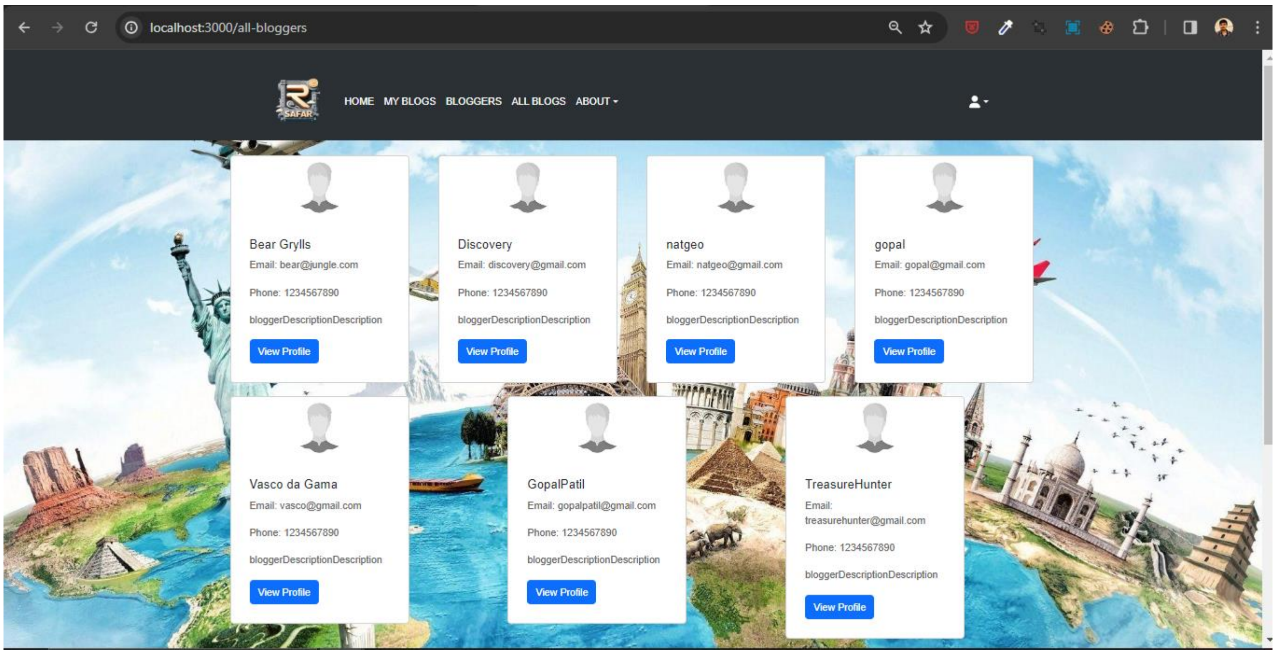


Fiordland National Park

2024-02-02T09:53 - 2024-02-02T11:54

Day 1: Arrival in Te Anau Arrived in the charming town of Te Anau, the gateway to Fiordland. The air was crisp, and the lake reflected the surrounding

Mount Cook National Park123



Bear Grylls
Email: bear@jungle.com
Phone: 1234567890
bloggerDescriptionDescription

Discovery
Email: discovery@gmail.com
Phone: 1234567890
bloggerDescriptionDescription

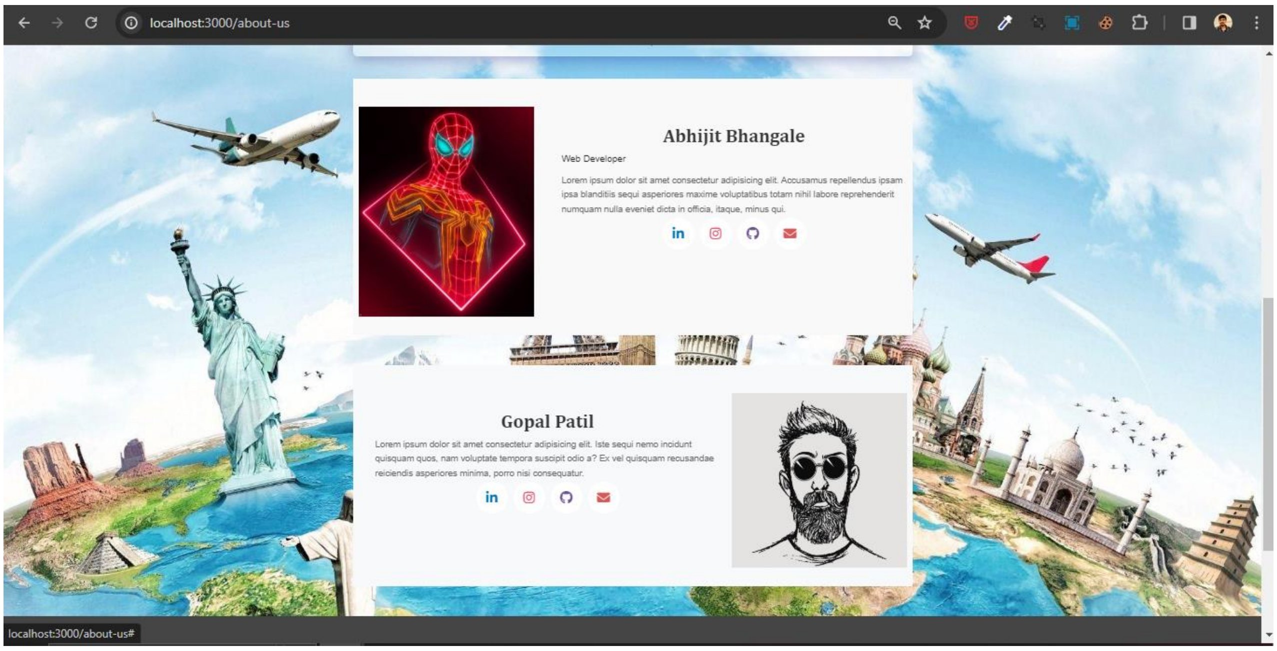
natgeo
Email: natgeo@gmail.com
Phone: 1234567890
bloggerDescriptionDescription

gopal
Email: gopal@gmail.com
Phone: 1234567890
bloggerDescriptionDescription

Vasco da Gama
Email: vasco@gmail.com
Phone: 1234567890
bloggerDescriptionDescription

GopalPatil
Email: gopalpatil@gmail.com
Phone: 1234567890
bloggerDescriptionDescription

TreasureHunter
Email: treasurehunter@gmail.com
Phone: 1234567890
bloggerDescriptionDescription



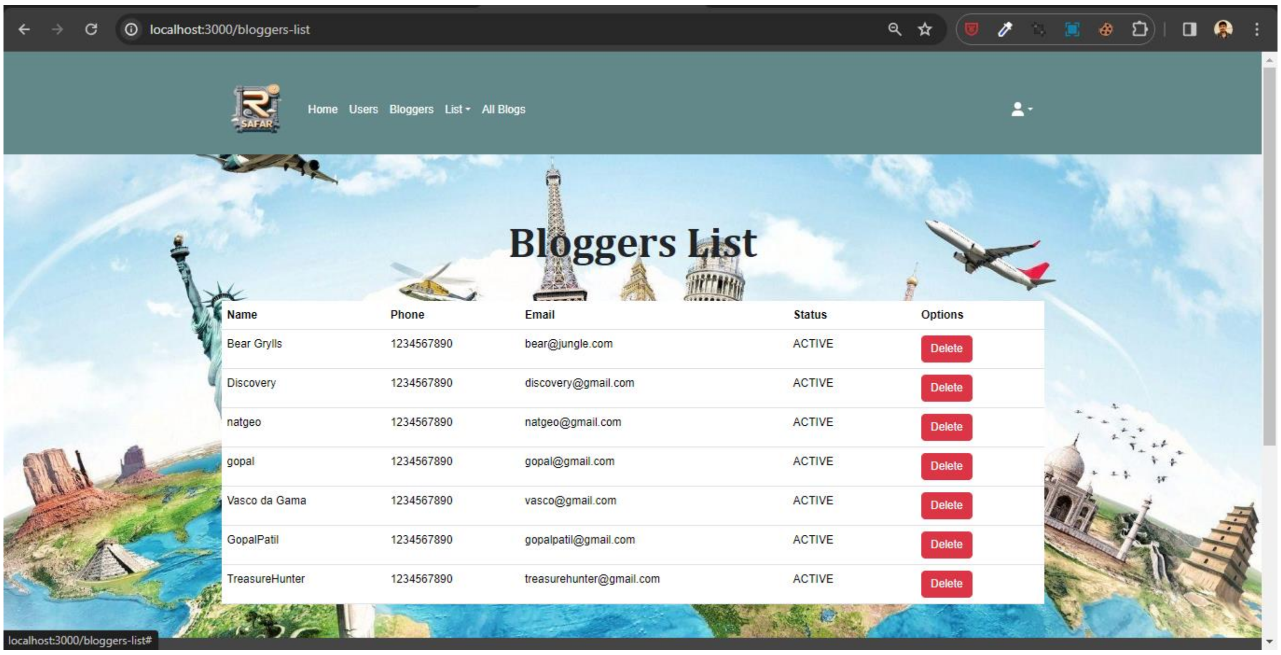
Abhijit Bhangale

Web Developer

Lorem ipsum dolor sit amet consectetur adipisicing elit. Accusamus repellendus ipsam ipsa blanditiis sequi asperiores maxime voluptatibus totam nihil labore reprehenderit numquam nulla eveniet dicta in officia, itaque, minus qui.

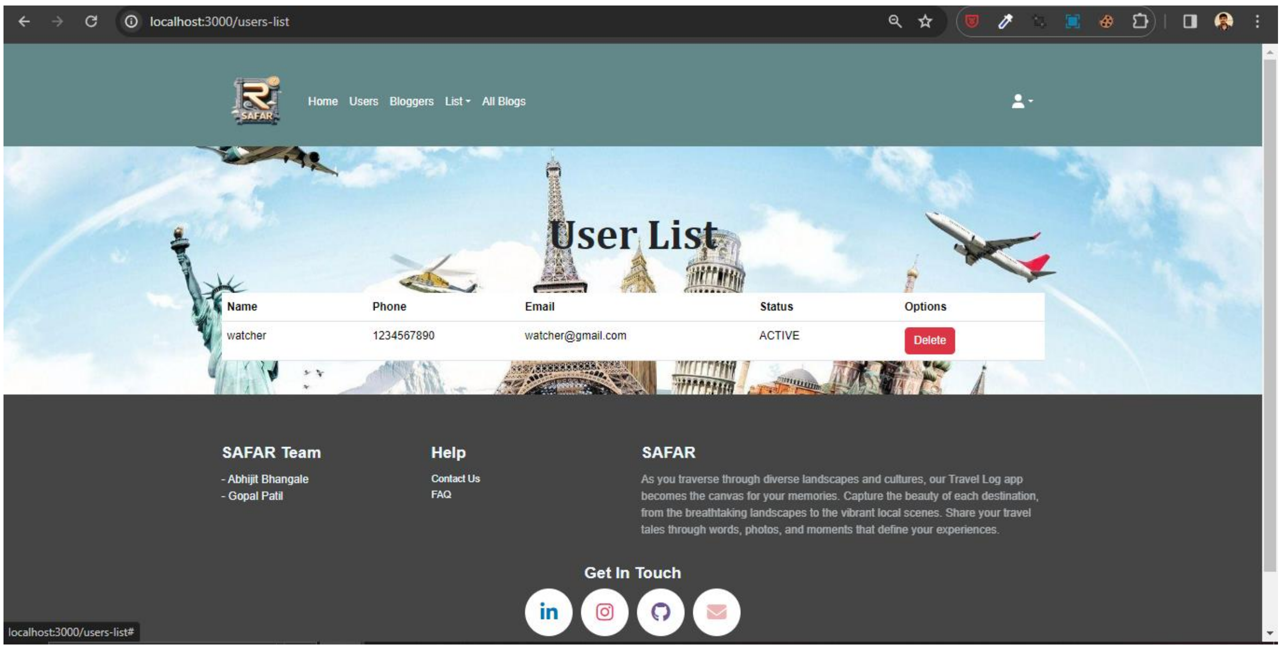
Gopal Patil

Lorem ipsum dolor sit amet consectetur adipisicing elit. Iste sequi nemo incidunt quisquam quos, nam voluptate tempora suscipit odio a? Ex vel quisquam recusandae reiciendis asperiores minima, porro nisi consequatur.



Bloggers List

Name	Phone	Email	Status	Options
Bear Grylls	1234567890	bear@jungle.com	ACTIVE	Delete
Discovery	1234567890	discovery@gmail.com	ACTIVE	Delete
natgeo	1234567890	natgeo@gmail.com	ACTIVE	Delete
gopal	1234567890	gopal@gmail.com	ACTIVE	Delete
Vasco da Gama	1234567890	vasco@gmail.com	ACTIVE	Delete
GopalPatil	1234567890	gopalpatil@gmail.com	ACTIVE	Delete
TreasureHunter	1234567890	treasurehunter@gmail.com	ACTIVE	Delete



User List

Name	Phone	Email	Status	Options
watcher	1234567890	watcher@gmail.com	ACTIVE	Delete

SAFAR Team
- Abhijit Bhangale
- Gopal Patil

Help
Contact Us
FAQ

SAFAR
As you traverse through diverse landscapes and cultures, our Travel Log app becomes the canvas for your memories. Capture the beauty of each destination, from the breathtaking landscapes to the vibrant local scenes. Share your travel tales through words, photos, and moments that define your experiences.

Get In Touch

CONCLUSION

In conclusion, the SAFAR is a comprehensive application that leverages the power of React, Spring boot, and MySQL to provide a seamless and interactive experience for travel bloggers and viewers. By following a systematic approach and incorporating various technologies and techniques, the app ensures a robust and scalable solution for travel blogging.

BIBLIOGRAPHY

https://getbootstrap.com/

https://react.io/

https://react-bootstrap.netlify.app/

https://www.w3schools.com/

https://www.google.com

https://spring.io/projects/spring-boot

https://www.npmjs.com/