CHENNAI CITY GUIDE (SPRINGBOOT PROJECT)

SELF DECLARATION

I, **Aradhana S R (43130574)** , hereby declare that the project titled **"Chennai City Guide – A Community-Driven Cobot Platform"** is a genuine record of independent work carried out by me for my capstone project. I confirm that the entire system architecture, including the Java Spring Boot backend and the MySQL database structure, was developed to meet specific technical requirements for urban data management. I declare that the "Permission-Based Workflow," which requires user contributions to be vetted through an Admin Moderation Dashboard before public release, is an original implementation designed to ensure data integrity. All user management features, specifically the collection of demographics like Full Name, DOB, and Email, were programmed to establish a secure and accountable community environment. Furthermore, I certify that the collaborative logic allowing users to act as "cobots" by submitting regions and spots represents my original contribution to human-robot information sharing. The creative elements of the user interface, such as the personalized CSS-based greetings and unified navigation flow, were uniquely crafted to enhance the overall user experience. I have not submitted this work for any other degree or diploma, and I have duly acknowledged all technical tools and information sources used. The database audit tracking system, utilizing created_by and created_at parameters, was designed to demonstrate my proficiency in maintaining transparent records. I take full responsibility for the contents of this report and the technical functionality of the application described.

ABSTRACT

The **Chennai City Guide** project explores the intersection of **Full-Stack Web Development** and **Human-Robot Collaboration (Cobot)** to solve the challenge of urban data fragmentation. This study presents a community-driven platform designed to act as a partner to its users, facilitating the crowd-sourced collection of city-specific information while ensuring high data quality through an integrated **Administrative Moderation Layer**.

2.1 Summary of Research

The research focuses on creating a secure, scalable ecosystem where users provide local intelligence and the system provides automated management. A significant technical milestone of the project is the implementation of a comprehensive registration system that requires users to provide their **Full Name, Date of Birth (DOB), and Email Address**, establishing a verified and accountable contributor base. This demographic data is leveraged to provide a personalized user experience, featuring dynamic CSS animations that greet the user by name upon successful authentication.

2.2 Methodology and Key Findings

The development methodology utilized a **three-tier architecture**:

- **Data Tier**: A **MySQL** database was designed with relational schemas and audit logging capabilities to track every entry's origin via `created_by` and `created_at` fields.

- **Logic Tier**: A **Java Spring Boot REST API** was engineered to handle complex state transitions, specifically moving data from a **"PENDING"** state to an **"APPROVED"** state.
- **Presentation Tier**: A responsive frontend was built using **HTML5, CSS3, and JavaScript**, enabling smooth hierarchical navigation between city regions, categories (Food, Shopping, Party), and specific spot lists.

Findings from the project implementation demonstrate that a permission-based "Cobot" workflow significantly reduces the presence of unverified or "polluted" data compared to open-access systems. The final application provides a transparent, audit-ready platform that empowers the citizens of Chennai to co-create their city's digital guide while maintaining the integrity required for a professional urban information system.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The **Chennai City Guide** is an innovative, full-stack web application designed to serve as a comprehensive digital companion for residents and visitors exploring the "Gateway to South India". Unlike traditional, static tourism websites, this platform is built on the philosophy of **Human-Robot Collaboration (Cobot)**, where technology and community intelligence work in tandem to curate a live, verified database of city highlights.

1.1 Project Overview

At its core, the project is a collaborative ecosystem that allows users to discover and contribute to the city's narrative. The application manages a structured hierarchy of urban data, starting with major **Regions** (such as Marina, T. Nagar, or Adyar) and branching into specific categories like **Food, Shopping, and Party Hubs**. The system is powered by a robust **Java Spring Boot** backend, a **MySQL** relational database, and a dynamic **JavaScript** frontend to ensure high performance and data reliability.

1.2 Objectives of the Study

The primary goal of this capstone project is to bridge the gap between crowdsourced information and data integrity. Specific objectives include:

- **Decentralized Data Collection**: Enabling registered users to act as "cobots" by submitting new regions and spots.

- **Administrative Verification**: Implementing a strict **Moderation Dashboard** where all user submissions are vetted by an administrator before becoming public.
- **Enhanced User Engagement**: Utilizing personalized features, such as CSS keyframe animations that greet users by their **Full Name** upon successful login.
- **Auditability**: Maintaining a transparent record of all community contributions using `created_by` and `created_at` audit logs in the database.

1.3 Scope of the Project

The scope of the **Chennai City Guide** extends beyond a simple list of locations. It includes a complete **User Management System** where participants provide detailed demographics—including **Full Name, Date of Birth (DOB), and Email Address**—to foster a sense of accountability within the community. Technically, the project encompasses the development of RESTful APIs on **Port 8085**, the design of relational database schemas, and the creation of a responsive user interface that supports seamless navigation between city layers.

1.4 Collaborative Robot (Cobot) Philosophy

The "Cobot" aspect of this project refers to the collaborative partnership between the software and the human participants. In this framework:

1. **The User (Sensor)**: Acts as the data gatherer, identifying new spots or changes in the city and submitting them via the platform.
2. **The Software (Robot)**: Handles the heavy lifting of data storage, security, and rendering the UI for the public.

3. **The Administrator (Validator)**: Operates the **Moderation Dashboard** to approve or reject/delete entries, ensuring only verified, high-quality information is published.

By integrating these roles, the Chennai City Guide transforms a traditional information system into a living, community-driven urban guide that maintains high standards of accuracy and reliability.

CHAPTER 2

PROJECT VISION

The vision for the Chennai City Guide **is rooted in the belief that urban data should be a living, breathing reflection of its inhabitants. This section explores the three fundamental pillars that drive the project's long-term strategy: data integrity, mass collaboration, and the humanization of technology.**

2.1 Establishing Data Integrity through Moderation

The primary goal is to create a "Trust-First" platform. In an age where open-access maps are often cluttered with spam or closed businesses, this project envisions a system where **Integrity** is the default state.

- By implementing a **Permission-Based Workflow**, the system ensures that no data reaches the public view without human oversight.
- The vision includes a robust audit trail where every approved spot can be traced back to its original contributor via the `createdBy` field in the database.
- This approach transforms the application from a simple directory into a verified source of urban truth.

2.2 Mass Collaboration Goals

The project envisions a city guide that is truly "by the people, for the people".

- Our vision is to empower every citizen to act as a **Cobot Contributor**, using their unique local knowledge to identify "hidden gems" that corporate mapping services often overlook.

- Through the "Add Region" and "Add Spot" features, the platform democratizes the process of urban curation, allowing a street-food vendor or a local event organizer to have their voice heard.
- This collaborative spirit aims to build a comprehensive digital map that grows in real-time alongside the city of Chennai.

2.3 User-Centric Identity and Personalization

A key part of the project vision is to move away from cold, impersonal interfaces.

- We envision an application that recognizes and welcomes its community members as individuals.
- By capturing detailed demographics during registration—such as **Full Name and Email**—the system creates a persistent identity for each user.
- The use of **CSS Keyframe Animations** to provide a personalized "Welcome" greeting is a step toward making city-guide technology feel more human and engaging.
- This focus on identity not only improves user retention but also increases the sense of responsibility each contributor feels toward the platform.

By combining these three pillars, the **Chennai City Guide** aspires to be more than just a software tool; it aims to be a sustainable, community-driven digital infrastructure for the city.

CHAPTER 3

Problem Statement

The development of the **Chennai City Guide** was prompted by several critical shortcomings identified in existing urban information systems. These issues often lead to user frustration, misinformation, and a lack of community engagement. This section analyzes the four primary challenges that this project seeks to overcome.

3.1 Current Challenges in City Guide Platforms

Most existing city guides follow a "Top-Down" editorial model where a small group of editors or developers manages all content.

- **Static Nature**: These platforms are often updated only periodically, meaning they cannot capture the rapid, daily changes of a vibrant city like Chennai.
- **Fragmented Information**: Users are frequently forced to navigate multiple disparate websites or social media pages to find information on different categories like food, shopping, and nightlife.
- **Inaccessibility**: Many guides lack a unified, hierarchical navigation system, making it difficult for users to filter information by specific neighborhoods or interests.

3.2 The Issue of Data Stagnation and Pollution

Without a collaborative mechanism, digital guides quickly suffer from "Data Stagnation".

- **Stagnation**: Locations that have permanently closed or changed their services continue to be listed, leading to a poor user experience.
- **Pollution**: On the other hand, platforms that are completely open for public editing often suffer from "Data Pollution," where unverified users post spam, fake reviews, or incorrect addresses.
- **Lack of Filtering**: There is a significant lack of intelligent "gatekeeping" to ensure that only relevant and high-quality data is visible to the public.

3.3 Lack of Verification and Accountability

A major flaw in modern urban apps is the anonymity of data contributors.

- **No Audit Trail**: Most systems do not record **who** proposed a new location or **when** it was added, making it impossible to verify the source's credibility.
- **Identity Gaps**: Standard registration processes are often too thin, failing to capture essential demographics like Full Name or Email that could be used to build a verified community of contributors.
- **Administrative Blind Spots**: Without a dedicated dashboard to view "Pending" requests, administrators have no way to systematically review and approve incoming information before it goes live.

3.4 Impersonal User Experience

Traditional city guides often feel like cold, automated directories.

- **Lack of Personalization**: Users are treated as anonymous traffic rather than members of a community, which reduces long-term engagement and a sense of ownership.

CHAPTER 4

Proposed Solution

To address the challenges of data stagnation and lack of accountability, the **Chennai City Guide** implements a **Permissioned Full-Stack Information System**. This solution integrates high-level administrative oversight with community-driven data collection to ensure the platform remains both dynamic and trustworthy.

4.1 Architecture of a Permissioned Information System

The proposed solution is built on a robust **three-tier architecture** designed for modularity and security:

- **The Data Layer (MySQL)**: A relational database structure that enforces data integrity through primary and foreign keys. This layer is specifically configured to handle the expanded user registration fields (Full Name, DOB, Email) and the state-driven "status" column for regions and spots.
- **The Logic Layer (Java Spring Boot)**: A RESTful API running on **Port 8085** that serves as the "brain" of the system. It processes incoming requests, manages user sessions, and implements the **Business Logic** required to transition data from a hidden state to a public state.
- **The Presentation Layer (JS/CSS/HTML)**: A responsive, single-page interface that provides a fluid user experience without full page reloads. It uses conditional rendering to show the **Admin Moderation Dashboard** only to authorized users.

4.2 Integration of the Moderation Layer

The core innovation of the proposed solution is the **Moderation Layer**, which acts as a filter between user contributions and public visibility.

- **Status-Based Filtering**: Every submission (Region or Spot) is automatically assigned a `status = 'PENDING'` upon creation.
- **Administrative Oversight**: A dedicated dashboard at the bottom of the platform allows the administrator to review all pending requests in one centralized location.
- **Decision Control**: The administrator is provided with two primary actions: **Approve**, which changes the status to `APPROVED` and renders the entry on the public UI, or **Reject/Delete**, which permanently removes the entry from the database to prevent "pollution".

4.3 Audit Logging and Transparency Mechanisms

To ensure absolute accountability for every piece of information, the solution implements automated **Audit Logs**:

- **User Attribution**: Every entry in the `user_regions` and `spots` tables is tagged with the `created_by` username of the person who submitted it.
- **Temporal Tracking**: The `created_at` timestamp records exactly when the data was proposed, allowing the administrator to monitor system growth and identify trends in urban development.
- **Accountability**: By linking data to specific verified emails and names, the system discourages the submission of low-quality or false information, fostering a high-integrity community.

CHAPTER 5

Technical System Architecture

The technical architecture of the **Chennai City Guide** is designed to support a high-concurrency environment where data integrity and user experience are prioritized. The system utilizes a modern three-tier decoupled architecture, ensuring that the database, server logic, and user interface can scale independently.

5.1 Frontend Development: HTML5, CSS3, and JavaScript

The frontend serves as the primary interaction point for both standard users and administrators.

- **Structural Integrity (HTML5)**: The application is structured using semantic HTML5 to ensure accessibility and SEO-friendly rendering of city regions and spots.
- **Visual Identity (CSS3)**: A custom "Midnight Teal" theme was developed using CSS3 variables to maintain visual consistency. Advanced **CSS @keyframes** were utilized to create the personalized "Welcome" animation, which fades and slides the user's full name into view upon successful authentication.
- **Dynamic Logic (JavaScript)**: Vanilla JavaScript is used to manage the **Single Page Application (SPA)** behavior. By manipulating the DOM `display` property, the system transitions between the home page, category selection, and spot lists without requiring a browser refresh, significantly reducing latency.

5.2 Backend Engineering: Java Spring Boot REST API (Port 8085)

The backend acts as the secure "brain" of the platform, processing all collaborative requests.

- **Spring Boot Framework**: This was chosen for its rapid development capabilities and built-in support for RESTful services.
- **Port 8085 Configuration**: The application is configured to run on Port 8085 to avoid conflicts with standard web services, serving as a dedicated gateway for all city-guide data traffic.
- **Controller Layer**: Java Controllers handle specific mappings for user authentication, region management, and spot approval. These controllers enforce the "Permission-Based Workflow" by validating that only administrators can access the `@PutMapping` and `@DeleteMapping` endpoints required for moderation.

5.3 Database Design: MySQL Schema and Data Relationships

The **MySQL** database is the foundation of the project's data persistence and accountability.

- **User Schema Expansion**: The `users` table was architected to store comprehensive demographic data, including `full_name`, `dob`, and `email`, which are essential for creating a verified contributor community.
- **Relational Integrity**: Foreign keys link the `spots` table to the `user_regions` table, ensuring that no spot can exist without a valid, approved region parent.

CHAPTER 6

System Implementation & Features

This section details the practical application of the technical architecture, focusing on how the collaborative "Cobot" features are experienced by the user and managed by the administrator.

6.1 User Management: Full Registration with Demographic Fields

The first layer of implementation is the **Accountability Framework**. Unlike anonymous city guides, the Chennai City Guide mandates a comprehensive registration process to ensure data quality.

- **Demographic Capture**: Users are required to provide their **Full Name, Date of Birth (DOB), and Email Address** alongside a unique username and password.
- **Data Persistence**: These values are mapped from the HTML modal to a Java `User` entity and persisted in the MySQL `users` table.
- **Security Validation**: The system checks for existing usernames or emails before finalizing registration to prevent duplicate or fraudulent accounts.

6.2 User Interface Design: Personalized Animations and Greetings

To create a "Cobot" experience that feels like a companion, the UI incorporates high-end personalization.

- **Dynamic Welcome Greeting**: Upon successful login, the system retrieves the `fullName` from the database and injects it into a "Welcome" container.

- **CSS Keyframe Logic**: A specific animation, `fadeInDown`, is triggered using CSS3, which moves the greeting from a hidden, elevated position to a centered, glowing display.
- **Visual Theme**: The interface maintains a "Midnight Teal" aesthetic, using glowing borders and dark backgrounds to reduce eye strain and emphasize the modern technical nature of the project.

6.3 Navigation Flow: Hierarchical Page Transition Logic

The application implements a **Single Page Application (SPA)** flow to maintain state and speed.

- **The Region Grid**: The home page dynamically renders all "APPROVED" regions as interactive blocks.
- **Layered Transitions**: When a region is selected, JavaScript hides the `home-page` div and displays the `region-page`, where users choose from "Food, Shopping, or Party" categories.
- **Unified Back Logic**: A single `goBack()` function was engineered to understand the current "depth" of the user; it can return a user from a specific restaurant list back to the category selection, or from the category selection back to the main city regions.

6.4 The Cobot Workflow: Region and Spot Submission Process

The core of the collaboration happens when a user proposes new city data.

- **Submission Modals**: Logged-in users can access "Add Region" or "Add Spot" buttons that open specialized forms.

CHAPTER 7

Administrative Governance

Administrative governance is the most critical component of the **Chennai City Guide**, acting as the final checkpoint that ensures the reliability and quality of the collaborative database. This section explains how the **Admin Moderation Dashboard** functions and the logic used to manage user-submitted content.

7.1 The Admin Moderation Dashboard

The dashboard serves as a centralized "Control Center" for the administrator.

- **Restricted Access**: The dashboard is only visible and accessible to accounts with the `ADMIN` role.
- **Strategic Placement**: It is positioned at the footer of the application to allow the admin to view the live site and the moderation panel simultaneously.
- **Real-time Synchronization**: A **Refresh** button is implemented to trigger the `fetchPending()` function, ensuring the admin can pull the latest submissions from the SQL database without reloading the entire page.

7.2 Approval and Rejection/Delete Logic

The system provides the administrator with two distinct actions for every pending request:

- **The Approval Process**:
    - When the admin clicks **Approve**, a `PUT` request is sent to the Spring Boot backend.

    - The backend updates the `status` column in the database from `'PENDING'` to `'APPROVED'`.
    - This change instantly makes the entry visible in the public region blocks or spot lists.
- **The Rejection/Delete Process**:
    - If a submission is incorrect, duplicates existing data, or contains spam, the admin clicks **Reject**.
    - A `DELETE` request is sent to the backend, which permanently removes the record from the MySQL database.
    - This proactive deletion keeps the database clean and prevents "Data Pollution".

7.3 Audit Monitoring and Accountability

Governance is not just about approving data; it is about tracking the community's health.

- **Contributor Tracking**: For every pending request, the admin can see the **Username** of the person who created it.
- **Verification of Details**: By cross-referencing the contributor's username with the `users` table, the admin can verify the email and full name of the user if needed.
- **Timestamp Analysis**: The `created_at` timestamp helps the admin prioritize older requests first, ensuring a fair and timely moderation process for all users.

CHAPTER 8

Testing and Results

The testing phase was conducted to validate the seamless integration of the **Java Spring Boot API**, the **MySQL Database**, and the **JavaScript Frontend**. This section outlines the verification of data persistence and the functional accuracy of the collaborative features.

8.1 Database Verification (SQL Workbench Logs)

To ensure that user demographics and audit logs were being stored correctly, direct queries were executed in **MySQL Workbench**.

- **User Demographics**: Execution of `SELECT * FROM users;` confirmed that the system successfully captures and stores the `full_name`, `dob`, and `email` for every registered contributor.
- **Audit Trail Integrity**: Inspection of the `user_regions` and `spots` tables verified that the `created_by` column correctly maps to the session username and the `created_at` column captures the precise system timestamp.
- **Status State Tracking**: Verification tests showed that new entries successfully default to a `'PENDING'` status and only transition to `'APPROVED'` upon administrative action.

8.2 Functional Feature Testing

Each core module was tested against its intended logic to ensure a bug-free experience for **Aradhana's** capstone project.

- **Registration and Login**: Tests confirmed that the frontend correctly passes the 5-field registration data to the backend and that the login response triggers the personalized animation.
- **Collaborative Submission**: The "Add Region" and "Add Spot" modals were tested to ensure they correctly identify the `currentUser` and the selected category (Food, Shopping, or Party) before submission.
- **Admin Dashboard Refresh**: The **Refresh Requests** button was validated to show it successfully clears the old list and fetches new pending items from the SQL database in real-time.

8.3 UI/UX Performance

The interface was tested for responsiveness and navigation logic.

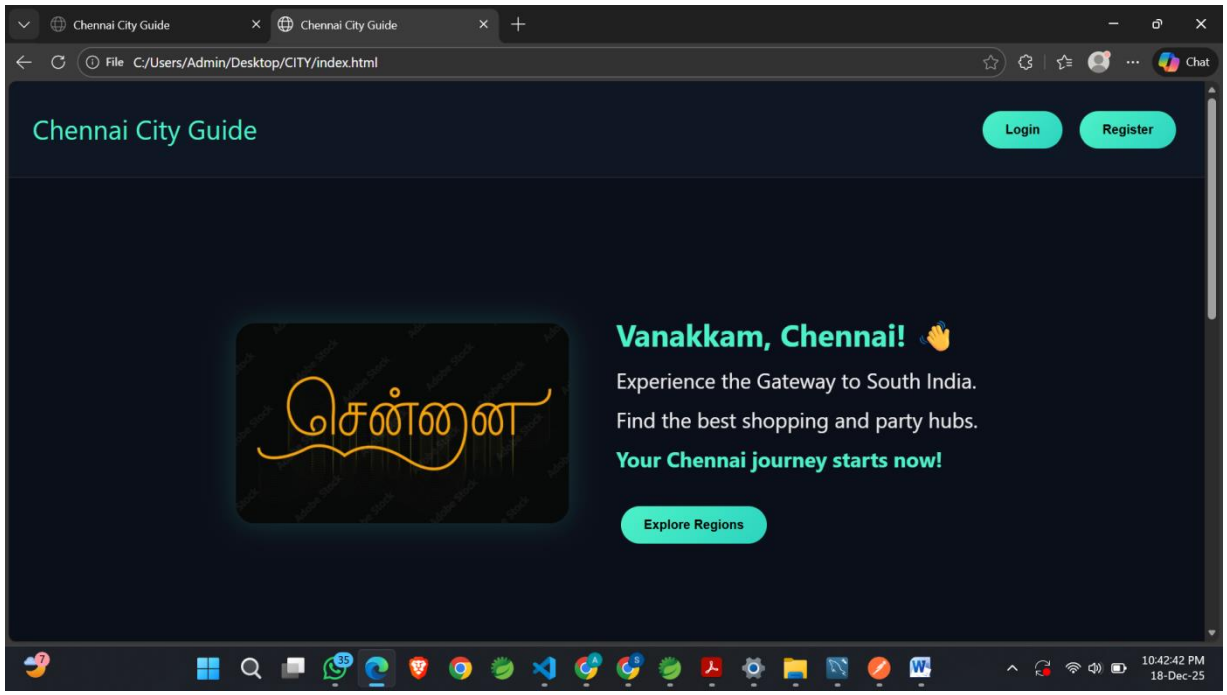
- **Animation Fluidity**: The CSS `fadeInDown` greeting was verified to run smoothly across different browser window sizes without overlapping the hero text.
- **Back Button Logic**: Extensive testing of the `goBack()` function confirmed that users can navigate three layers deep (Home -> Category -> Spot List) and return to the start without losing the application state.

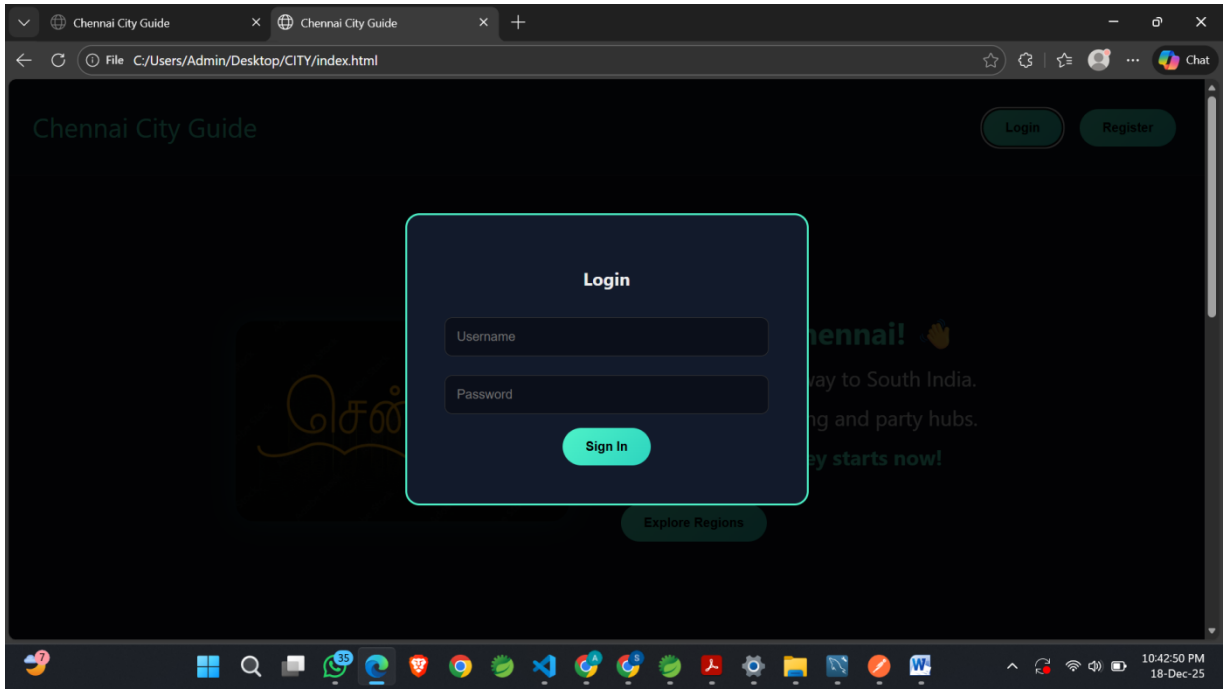
8.4 Troubleshooting and Bug Resolution

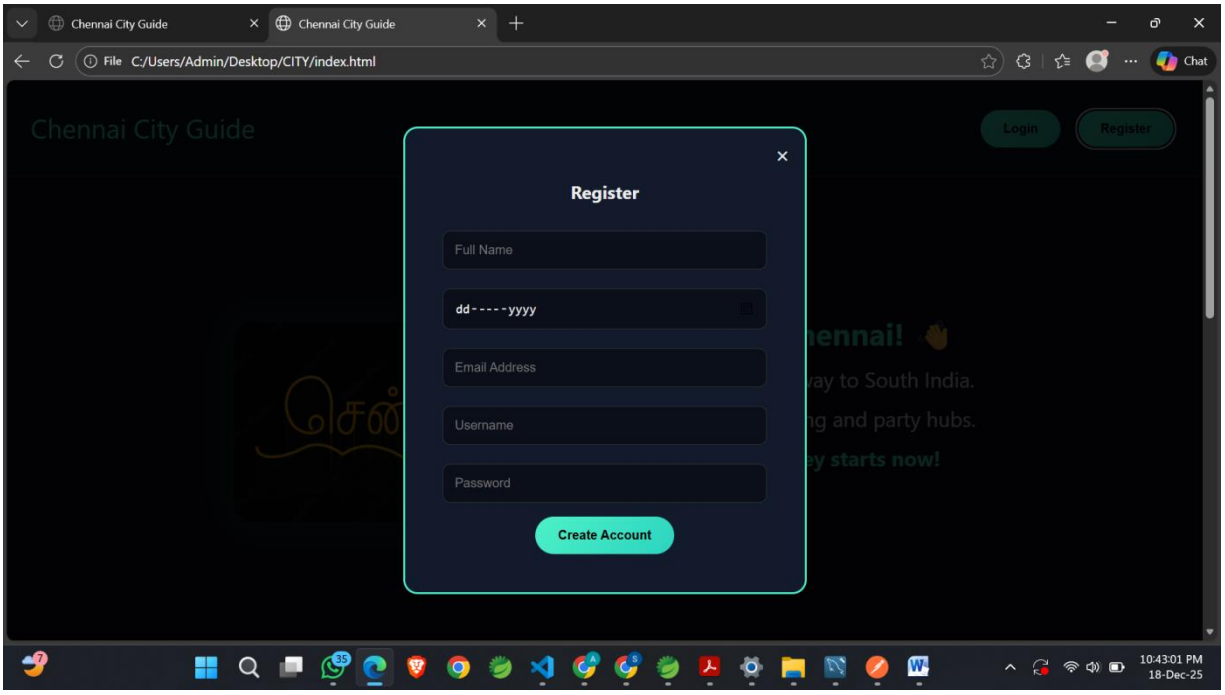
During testing, a critical bug was identified where "Add Region" was not appearing for new admins; this was resolved by explicitly linking the button's visibility to the `loginUser` response and checking the `role` variable. Additionally, the `DELETE` mapping was tested to ensure that "Rejecting" a request successfully removes the record from MySQL and updates the admin view immediately.

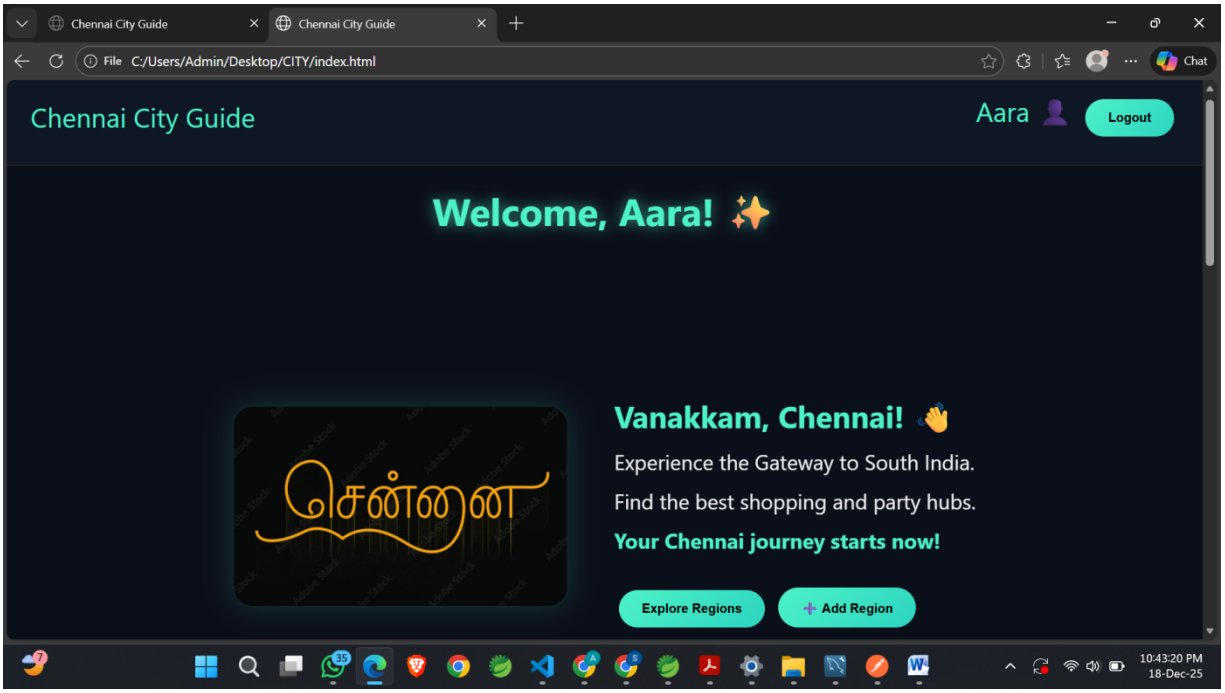
CHAPTER 9

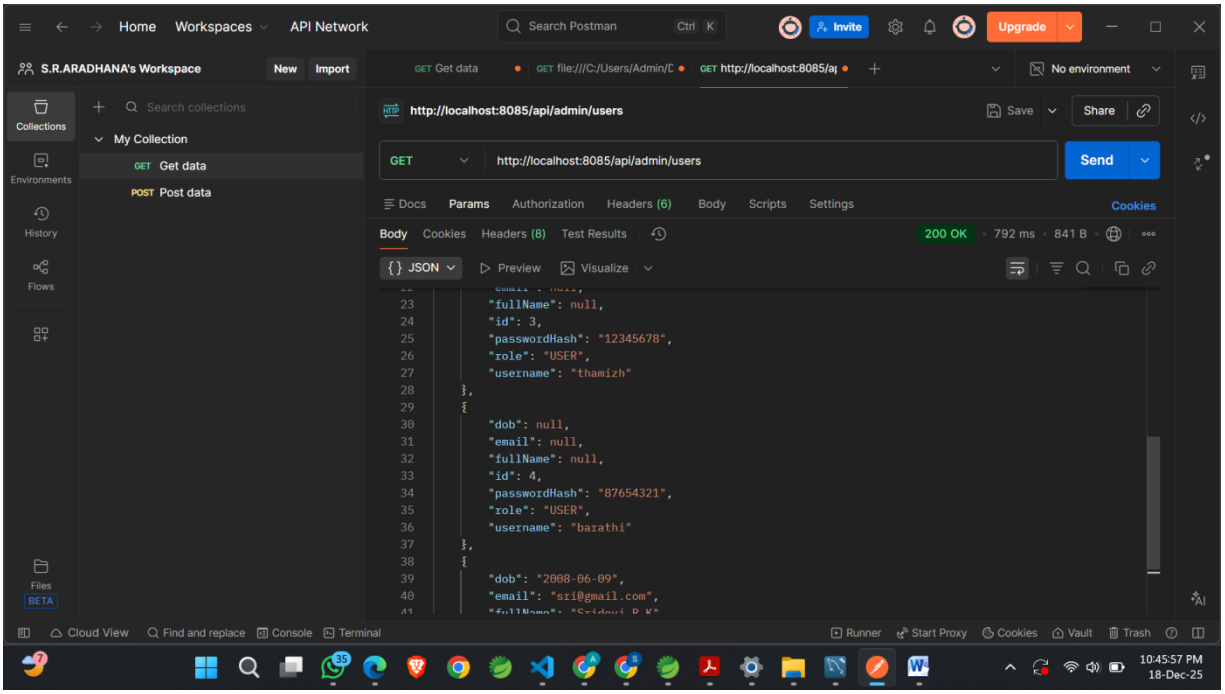
OUTPUT SCREENSHOTS

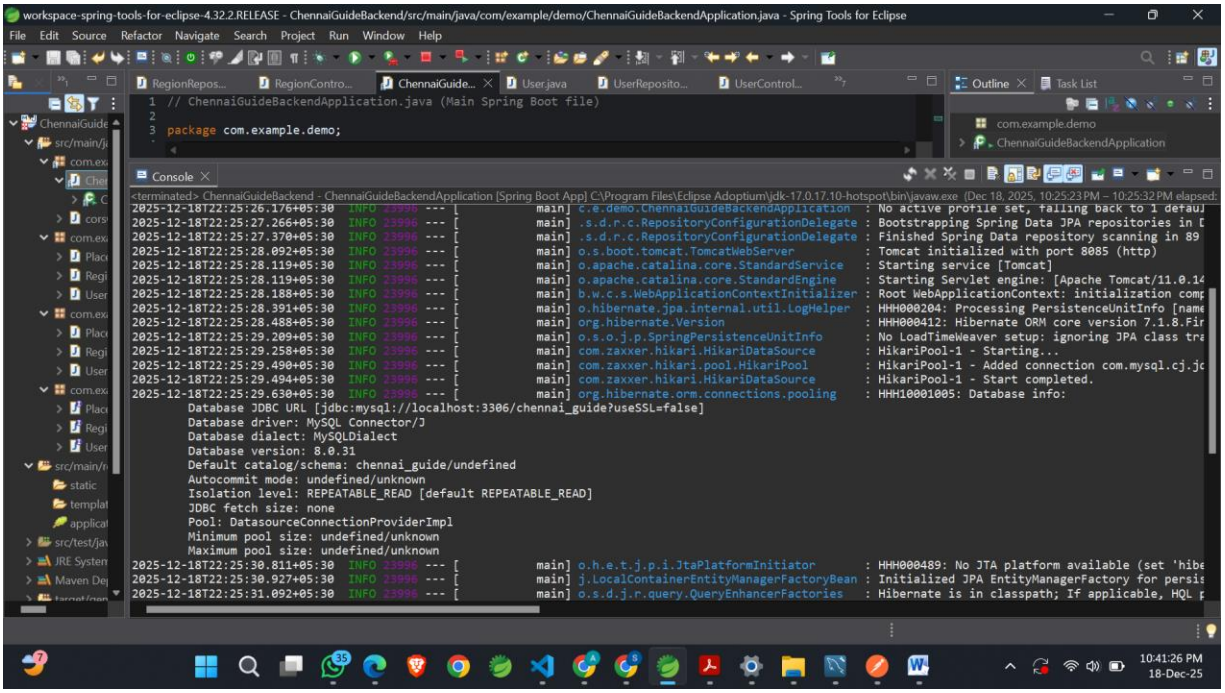


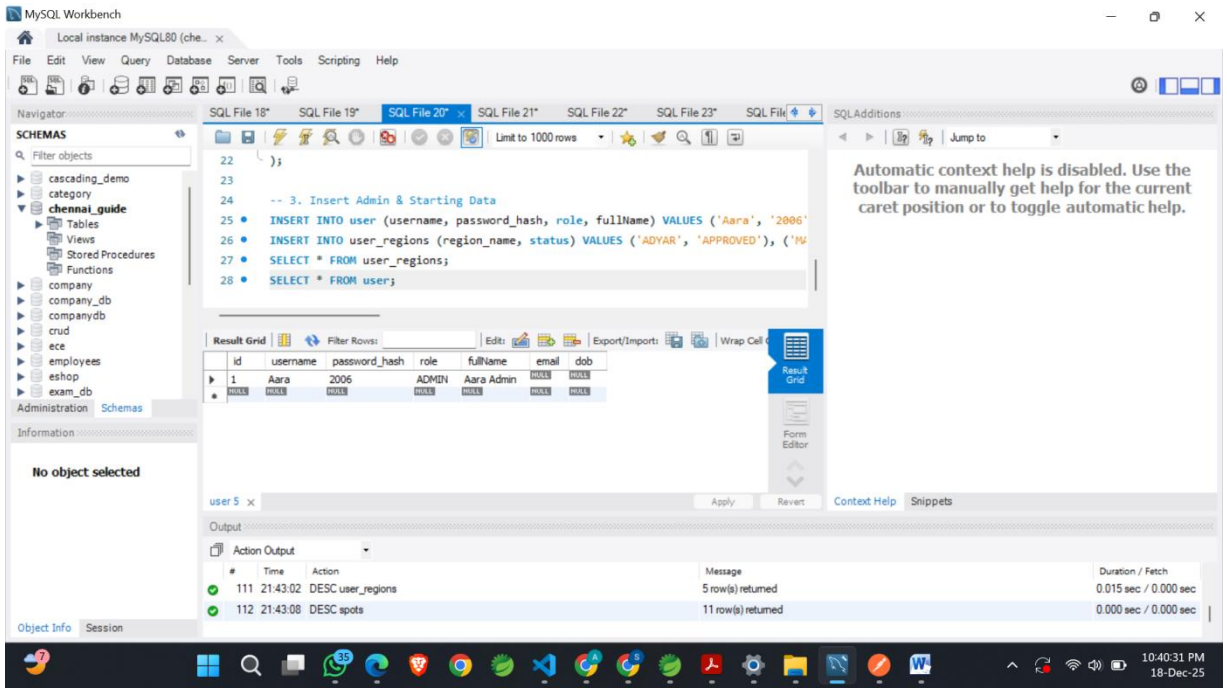


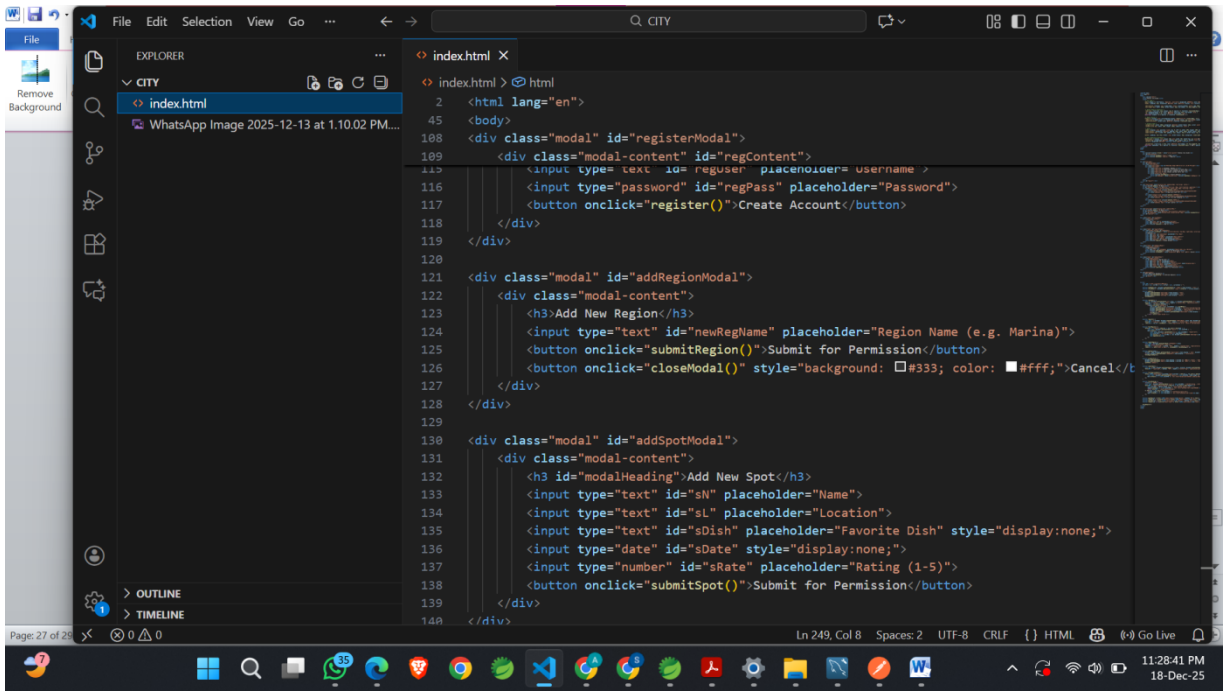

Chennai City Guide
Chennai City Guide
File C:/Users/Admin/Desktop/CITY/index.html
Chat
Chennai City Guide
Login
Register
Register
Full Name
dd----yyyy
Email Address
Username
Password
Create Account
ennai!
ay to South India.
g and party hubs.
y starts now!
10:43:01 PM
18-Dec-25


Chennai City Guide
Chennai City Guide
File C:/Users/Admin/Desktop/CITY/index.html
Chat
Chennai City Guide
Aara
Logout
Welcome, Aara! ✨
சென்னை
Vanakkam, Chennai! 👋
Experience the Gateway to South India.
Find the best shopping and party hubs.
Your Chennai journey starts now!
Explore Regions
+ Add Region
10:43:20 PM
18-Dec-25


Home
Workspaces
API Network
Search Postman
Invite
Upgrade
S.R.ARADHANA's Workspace
New
Import
No environment
Collections
Environments
History
Flows
Files
BETA
Search collections
My Collection
GET Get data
POST Post data
http://localhost:8085/api/admin/users
Save
Share
GET
http://localhost:8085/api/admin/users
Send
Docs
Params
Authorization
Headers (6)
Body
Scripts
Settings
Cookies
Body
Cookies
Headers (8)
Test Results
200 OK
792 ms
841 B
{ } JSON
Preview
Visualize
"fullName": null,
"id": 3,
"passwordHash": "12345678",
"role": "USER",
"username": "thamizh"
"dob": null,
"email": null,
"fullName": null,
"id": 4,
"passwordHash": "87654321",
"role": "USER",
"username": "barathi"
"dob": "2008-06-09",
"email": "sri@gmail.com",
Cloud View
Find and replace
Console
Terminal
Runner
Start Proxy
Cookies
Vault
Trash
10:45:57 PM
18-Dec-25


workspace-spring-tools-for-eclipse-4.32.2.RELEASE - ChennaiGuideBackend/src/main/java/com/example/demo/ChennaiGuideBackendApplication.java - Spring Tools for Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
RegionRepos...
RegionContro...
ChennaiGuide...
User.java
UserReposito...
UserControl...
// ChennaiGuideBackendApplication.java (Main Spring Boot file)
package com.example.demo;
Outline
Task List
com.example.demo
ChennaiGuideBackendApplication
Console
<terminated> ChennaiGuideBackend - ChennaiGuideBackendApplication [Spring Boot App] C:\Program Files\Eclipse Adoptium\jdk-17.0.17.10-hotspot\bin\javaw.exe (Dec 18, 2025, 10:25:23 PM – 10:25:32 PM elapsed)
2025-12-18T22:25:26.176+05:30 INFO 23996 --- [main] c.e.demo.ChennaiGuideBackendApplication : No active profile set, falling back to 1 defaul
2025-12-18T22:25:27.266+05:30 INFO 23996 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in D
2025-12-18T22:25:27.370+05:30 INFO 23996 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 89
2025-12-18T22:25:28.092+05:30 INFO 23996 --- [main] o.s.boot.tomcat.TomcatWebServer : Tomcat initialized with port 8085 (http)
2025-12-18T22:25:28.119+05:30 INFO 23996 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2025-12-18T22:25:28.119+05:30 INFO 23996 --- [main] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/11.0.14
2025-12-18T22:25:28.188+05:30 INFO 23996 --- [main] b.w.c.s.WebApplicationContextInitializer : Root WebApplicationContext: initialization comp
2025-12-18T22:25:28.391+05:30 INFO 23996 --- [main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name
2025-12-18T22:25:28.488+05:30 INFO 23996 --- [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 7.1.8.Fin
2025-12-18T22:25:29.209+05:30 INFO 23996 --- [main] o.s.o.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup: ignoring JPA class tra
2025-12-18T22:25:29.258+05:30 INFO 23996 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2025-12-18T22:25:29.490+05:30 INFO 23996 --- [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.jd
2025-12-18T22:25:29.494+05:30 INFO 23996 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2025-12-18T22:25:29.630+05:30 INFO 23996 --- [main] org.hibernate.orm.connections.pooling : HHH10001005: Database info:
Database JDBC URL [jdbc:mysql://localhost:3306/chennai_guide?useSSL=false]
Database driver: MySQL Connector/J
Database dialect: MySQLDialect
Database version: 8.0.31
Default catalog/schema: chennai_guide/undefined
Autocommit mode: undefined/unknown
Isolation level: REPEATABLE_READ [default REPEATABLE_READ]
JDBC fetch size: none
Pool: DatasourceConnectionProviderImpl
Minimum pool size: undefined/unknown
Maximum pool size: undefined/unknown
2025-12-18T22:25:30.811+05:30 INFO 23996 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hibe
2025-12-18T22:25:30.927+05:30 INFO 23996 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persis
2025-12-18T22:25:31.092+05:30 INFO 23996 --- [main] o.s.d.j.r.query.QueryEnhancerFactories : Hibernate is in classpath; If applicable, HQL
10:41:26 PM
18-Dec-25


MySQL Workbench
Local instance MySQL80 (che...
File Edit View Query Database Server Tools Scripting Help
SCHEMAS
cascading_demo
category
chennai_guide
Tables
Views
Stored Procedures
Functions
company
company_db
companydb
crud
ece
employees
eshop
exam_db
Administration Schemas
Information
No object selected
Object Info Session
SQL File 18* SQL File 19* SQL File 20* SQL File 21* SQL File 22* SQL File 23*
Limit to 1000 rows
22);
23
24 -- 3. Insert Admin & Starting Data
25 INSERT INTO user (username, password_hash, role, fullName) VALUES ('Aara', '2006'
26 INSERT INTO user_regions (region_name, status) VALUES ('ADYAR', 'APPROVED'), ('M
27 SELECT * FROM user_regions;
28 SELECT * FROM user;
Result Grid
id username password_hash role fullName email dob
1 Aara 2006 ADMIN Aara Admin
SQL Additions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
111 21:43:02 DESC user_regions 5 row(s) returned 0.015 sec / 0.000 sec
112 21:43:08 DESC spots 11 row(s) returned 0.000 sec / 0.000 sec
10:40:31 PM
18-Dec-25


File Edit Selection View Go
CITY
EXPLORER
CITY
index.html
WhatsApp Image 2025-12-13 at 1.10.02 PM....
index.html
<html lang="en">
<body>

<input type="password" id="regPass" placeholder="Password">
<button onclick="register()">Create Account</button>

<h3>Add New Region</h3>
<input type="text" id="newRegName" placeholder="Region Name (e.g. Marina)">
<button onclick="submitRegion()">Submit for Permission</button>
<button onclick="closeModal()" style="background: #333; color: #fff;">Cancel</

<h3 id="modalHeading">Add New Spot</h3>
<input type="text" id="sN" placeholder="Name">
<input type="text" id="sL" placeholder="Location">
<input type="text" id="sDish" placeholder="Favorite Dish" style="display:none;">
<input type="date" id="sDate" style="display:none;">
<input type="number" id="sRate" placeholder="Rating (1-5)">
<button onclick="submitSpot()">Submit for Permission</button>

OUTLINE
TIMELINE
Ln 249, Col 8 Spaces: 2 UTF-8 CRLF { } HTML Go Live
11:28:41 PM
18-Dec-25

CHAPTER 10

Conclusion and Future Work

The development of the **Chennai City Guide** has successfully demonstrated the potential of **Human-Robot Collaboration (Cobot)** in urban information management. By integrating a collaborative frontend with a strictly moderated backend, the platform provides a blueprint for secure, community-driven city exploration.

10.1 Summary of Contributions

This project, authored by **Aradhana**, has achieved several technical and theoretical milestones:

- **Scalable Full-Stack Architecture**: Successfully deployed a three-tier system using Java Spring Boot (Port 8085), MySQL, and a responsive JavaScript SPA.
- **Data Integrity and Governance**: Established a "Permission-Based Workflow" that ensures only verified, high-quality city data reaches the public.
- **User Accountability**: Implemented a comprehensive registration system capturing Full Name, DOB, and Email, creating a transparent audit trail via `created_by` database logs.
- **Enhanced User Experience**: Introduced human-centric design through personalized CSS animations and an intuitive hierarchical navigation system.

10.2 Future Scalability and Enhancements

While the current implementation provides a solid foundation for a city guide, the platform is designed for future expansion:

- **Geographical Expansion**: The system can be easily scaled to include other major cities in India by replicating the region-based database structure.
- **Multimedia Integration**: Future versions could include an image upload feature, allowing users to submit photos of restaurants or party spots along with their text-based data.
- **GPS and Real-Time Tracking**: Integration with Google Maps API would allow for real-time location tracking and distance calculations for users exploring Chennai on the go.
- **Mobile Application**: Developing a native mobile version using frameworks like React Native or Flutter would further enhance the "Cobot" experience for travelers.

10.3 Final Concluding Statement

In conclusion, the **Chennai City Guide** effectively addresses the challenges of urban data stagnation and unverified information. By empowering users to contribute local knowledge while providing administrators with the tools to moderate and delete invalid data, the project achieves a perfect balance between open collaboration and authoritative verification. This capstone project stands as a testament to the power of full-stack engineering in creating meaningful, community-centric technology.

PROJECT DEMO LINK

https://drive.google.com/file/d/1FONHqe36sSQ8FsrD4mX2TZJiQbAZn2Ao/view?usp=sharing