PROFESSIONAL TRAINING REPORT

Submitted in partial fulfillment of the requirements for the award of

Bachelor of Technology degree in

Information Technology

By

KIRAN M (Reg.No - 41120094)

DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A++" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600119

SEPTEMBER- 2023

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A++" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHISALAI,

CHENNAI-600119

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **KIRAN M (Reg.No - 41120094)** who carried out the Project entitled "**WEB DEVELOPMENT – WEATHER API AND TRAIN TICKET RESERVATION**" under my supervision from _____2023 to _____2023.

Internal Guide

Dr.C . GEETHA M.TECH., Ph.D.

Head of the Department

Dr. R. SUBHASHINI M.E., Ph.D.

Submitted for Viva voce Examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A++" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHISALAI,

CHENNAI-600119

DECLARATION

I, **KIRAN M (Reg.No - 41120094),** hereby declare that the Project Phase-2 Report entitled "**WEB DEVELOPMENT – WEATHER API AND TRAIN TICKET RESERVATION**" done by me under the guidance of **Dr. C. GEETHA M.TECH., Ph.D** is submitted in partial fulfillment of the requirements for the award Bachelor of Technology degree in Information Technology.

DATE:

PLACE: CHENNAI **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph. D**, **Dean**, School of Computing, **Dr. R. SUBHASHINI M.E., Ph.D**.,Head of the Department ,Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. C. GEETHA M.TECH., Ph.D** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-2 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project

CERTIFICATE OF COMPLETION

ABSTRACT

Weather API integration is a crucial component of modern web development, offering real-time and forecasted weather data to enhance user experiences and functionality. This abstract provides an overview of the significance and key considerations for integrating weather APIs into web applications.

Weather data is essential for a wide range of web applications, from travel and tourism websites to e-commerce platforms and fitness apps. Integrating a weather API enables developers to access accurate and up-to-date weather information, which can be used to deliver personalized content, improve user engagement, and enhance decision-making processes.

The Train Ticket Reservation Web Development project aims to revolutionize the way passengers plan and book train journeys by providing an efficient and user-friendly online platform. This web application is designed to streamline the process of reserving train tickets, offering passengers a convenient and reliable means to access real-time train information, select seats, make secure payments, and manage their bookings. The system's architecture encompasses both frontend and backend components, including a responsive user interface, robust backend logic for ticket booking and management, secure payment integration, and seamless interaction with external data sources for up-to-date train schedules and information. The project follows a structured development process that encompasses planning, design, development, testing, deployment, and ongoing maintenance to ensure a high-quality and hassle-free experience for travelers. By leveraging cutting-edge web technologies and adhering to stringent security standards, this Train Ticket Reservation Web Development project seeks to enhance the overall travel experience for passengers and contribute to the modernization of the transportation industry.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
API	-Application Programming Interface
HTML	-Hypertext Markup Language
CSS	-Cascading Style Sheet
JS	-JavaScript
ZIP	-Zone Improvement Plan
PWA	-Progresssive Web Apps
UX	-Hpertext Processor
UI	-User Interface
GPS	-Global Positioning System
CDN	-Content Delivery Networks
XSS	-Cross-Site Scripting
CSRF	- Cross-Site Request

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 WEATHER API:

Weather APIs (Application Programming Interfaces) are tools that allow developers to access and retrieve weather-related data and information from meteorological databases or services. These APIs enable developers to integrate real-time and forecasted weather data into their applications, websites, or services. Weather data obtained through these APIs can include information like temperature, humidity, wind speed and direction, precipitation, atmospheric pressure, and much more.

Weather APIs have become invaluable tools for developers and businesses seeking to provide accurate and timely weather information to their users. They enhance user experiences by helping people plan their activities, make informed decisions, and stay safe in various weather conditions. Whether you're building a weather app, integrating weather data into an e-commerce site, or optimizing logistics operations, Weather APIs can be a valuable resource in your development toolkit

1.2 TRAIN TICKET RESERVATION:

Train ticket reservation web development is a field of web development focused on creating online platforms that allow users to book and manage train tickets conveniently. It plays a crucial role in the travel and transportation industry by providing passengers with a user-friendly and efficient way to plan their train journeys. In this introduction, we will explore the key components, technologies, and steps involved in developing a train ticket reservation website.

Developing a train ticket reservation website requires a combination of frontend and backend technologies, security measures, and a deep understanding of the transportation industry's complexities. With proper planning and execution, such a website can provide travelers with a convenient and efficient way to book and manage their train journeys.

CHAPTER 2
PROBLEM STATEMENT

2.1 CHALLENGES AND OB JECTIVES OF WEATHER API:

1. **User-Friendly Interface:** Create an intuitive and visually appealing user interface that allows users to easily input their desired location and access weather data without any technical barriers.

2. **Real-Time Weather Data:** Integrate a reliable and up-to-date weather API to fetch current weather conditions, forecasts, and historical weather data for the selected location. Ensure that the data is accurate and regularly updated.

3. **Location Selection:** Implement features for users to search for locations by name, postal code, or GPS coordinates. Provide auto-suggestions for location input to enhance user experience.

4. **Weather Presentation:** Display weather information in a clear and organized manner, including current temperature, humidity, wind speed, precipitation, and UV index. Use graphics and icons to make the data easily understandable.

5. **Forecasting:** Offer both short-term and long-term weather forecasts, including hourly and daily predictions. Users should be able to switch between different time frames to plan their activities accordingly.

6. **Historical Data:** Allow users to access historical weather data for their selected location, enabling them to review past weather patterns and trends.

7. **Responsive Design:** Ensure the web application is responsive and accessible on various devices, including desktops, tablets, and smartphones.

8. **User Preferences:** Implement user preferences and settings, such as unit preferences (e.g., Celsius vs. Fahrenheit), language selection, and the option to save favorite locations for quick access.

9. **Location-Based Services:** Utilize geolocation to automatically detect the user's current location and provide weather information without requiring manual input.

10. **Error Handling:** Develop robust error handling mechanisms to gracefully manage situations when the API data is unavailable or when users input invalid locations.

11. **Security:** Protect user data and privacy by securely handling location information and user preferences. Implement HTTPS for secure data transmission.

2.2 CHALLENGES AND OB JECTIVES OF TRAIN TICKET RESERVATION SYSTEM:

By addressing these challenges and objectives, the Weather API web development project aims to create a valuable and user-friendly tool for accessing accurate weather information, helping users make informed decisions based on current and forecasted weather conditions.

1. **Complex Booking Process:** The current system lacks a straightforward and intuitive booking process. Users are often required to navigate through multiple pages, fill out extensive forms, and select from confusing options, making it challenging for both new and experienced travelers to quickly secure their tickets.

2. **Limited Accessibility:** Accessing the ticket reservation system can be challenging for passengers with disabilities or those lacking access to high-speed internet and modern devices. The absence of accessible features and support for multiple languages can also hinder inclusivity.

3. **Real-Time Information:** Passengers struggle to obtain accurate and real-time information regarding train schedules, availability, delays, and platform

changes. This lack of timely updates can result in missed trains and disrupted travel plans.

4. **Payment Security:** Concerns about the security of payment transactions persist, deterring passengers from making online bookings. The current system must address these concerns and instill confidence in users regarding the safety of their financial data.

5. **Inadequate User Support:** The system often lacks comprehensive user support, including FAQs, chat support, and helplines, leaving passengers with unresolved queries or issues.

6. **Capacity Management:** The system may not effectively manage seat availability, leading to overbooking, ticket cancellations, or overcrowded trains.

7. **Outdated Technology:** Many existing systems rely on outdated technologies, resulting in slow performance, frequent downtime, and a poor user experience.

In light of these challenges, there is a pressing need to develop a modern, user-centric, and technologically advanced Train Ticket Reservation System for web platforms. This system should prioritize ease of use, accessibility, real-time information, payment security, comprehensive user support, efficient capacity management, and robust technology infrastructure to address the needs and expectations of today's travelers. The development of such a system will significantly enhance the train booking experience, streamline operations, and contribute to the overall improvement of the transportation industry.

CHAPTER 3
LITERATURE SURVEY

1. **Introduction to Weather APIs and Web Development**:

- Explore introductory materials on web development technologies, including HTML, CSS, JavaScript, and frameworks like React, Angular, or Vue.js.
- Understand the basics of APIs, their usage, and how they can be integrated into web applications.
- Learn about the significance of weather data and its applications in various industries.

2. ****Weather APIs**:**

- Review available weather APIs such as OpenWeatherMap, AccuWeather, Weather.com, and Dark Sky, among others.
- Compare the features, pricing, and data accuracy of different weather APIs.
- Examine case studies or documentation of successful implementations using weather APIs in web applications.

3. ****Web Development Best Practices**:**

- Investigate best practices in web application design, including responsive web design principles, user interface (UI) and user experience (UX) design considerations.

4. ****Geolocation and Location Services**:**

- Understand the use of geolocation services to determine a user's location.
- Research how to implement location-based services in web applications, including privacy considerations.

5. ****Data Visualization**:**

- Study techniques for visualizing weather data, such as charts, graphs, and maps.
- Explore libraries and tools like D3.js, Chart.js, Leaflet, or Mapbox for creating interactive data visualizations.

6. **Security and Privacy in Web Development**:
   - Review security best practices, including data encryption, secure API usage, and protection against common web vulnerabilities like Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF).

7. **Performance Optimization**:
   - Investigate strategies for optimizing web application performance, including minimizing load times and optimizing resource delivery (e.g., using Content Delivery Networks – CDNs).

8. **User Experience (UX)**:
   - Examine UX design principles, accessibility standards (e.g., WCAG), and usability testing methods to create a user-friendly weather application.

9. **Cross-Browser Compatibility**:
   - Learn about cross-browser testing tools and methodologies to ensure compatibility with various web browsers.

10. **Case Studies and Examples**:
   - Explore real-world examples of weather-related web applications, including their architecture, design, and user interactions.
   - Analyze any challenges faced and innovative solutions implemented in these projects.

11. **Future Trends and Emerging Technologies**:
   - Look for emerging technologies and trends in web development, such as Progressive Web Apps (PWAs), serverless architecture, and the use of machine learning for weather predictions.

12. **Conclusion and Gaps in Existing Literature**:
   - Summarize key findings from the literature survey.
   - Identify any gaps or areas where your project can contribute to the field of weather API web development.

CHAPTER 4
TECHNOLOGICAL OVERVIEW

A technological overview for a Weather API web development project and train ticket reservation system project involves outlining the key technologies and tools that will be used to build the web application.

4.1 COMPONENTS:

1. ****Front-End Development**:**

- **HTML/CSS**: These fundamental technologies are used for structuring web content (HTML) and styling (CSS) the user interface.
- **JavaScript**: JavaScript is essential for creating interactive features, handling user input, and making API requests asynchronously.
- **Front-End Framework**: Consider using a front-end framework like React, Angular, or Vue.js to simplify the development process, enhance user experience, and manage the application state efficiently.

2. ****Back-End Development**:**

- **Server-Side Language**: Choose a server-side language such as Node.js (JavaScript), Python, Ruby, or Java to build the back-end logic of your application.
- **Web Framework**: Utilize a web framework (e.g., Express.js for Node.js) to handle routing, server setup, and API interactions.
- **Database**: If your project requires user accounts, location-based data storage, or historical weather data storage, integrate a suitable database system like MySQL, PostgreSQL, MongoDB, or Firebase Realtime Database.

3. ****Weather API Integration**:**

- **Weather API**: Select a weather API provider like OpenWeatherMap, AccuWeather, or Weather.com, and obtain API keys for accessing weather data.
- **HTTP Requests**: Use libraries like Axios or the built-in `fetch` API in JavaScript to make HTTP requests to the weather API endpoints.

4. **Geolocation and Location Services**:

- **Geolocation API**: Leverage the Geolocation API in browsers to detect the user's location automatically. This helps in providing weather information without manual input.
- **Location Search**: Implement location search functionality using the user's input, whether it's a place name, postal code, or GPS coordinates.

5. **Data Visualization**:

- **Charting Libraries**: Integrate data visualization libraries like D3.js, Chart.js, or Highcharts to display weather data in charts and graphs.
- **Mapping Libraries**: If displaying weather data on maps is required, consider using Leaflet, Mapbox, or Google Maps JavaScript API.

6. **User Interface (UI) Design**:

- **UI Framework**: Utilize UI frameworks or component libraries like Bootstrap, Material-UI, or Bulma to ensure a responsive and visually appealing design.
- **CSS Preprocessors**: Use CSS preprocessors like SASS or LESS for more maintainable and organized CSS code.

7. **Security and Authentication**:

- **HTTPS**: Implement HTTPS to secure data transmission between the user's browser and your server.
- **User Authentication**: If the application requires user accounts or personalized features, integrate user authentication using technologies like OAuth, JWT, or traditional username/password authentication.

8. **Performance Optimization**:

- **Content Delivery Network (CDN)**: Utilize CDNs for serving static assets like images and scripts to improve load times.
- **Caching**: Implement server-side and client-side caching to reduce the load on the API and improve response times.

9. **Cross-Browser Compatibility**:

- Use browser testing tools like BrowserStack or CrossBrowserTesting to ensure that the application functions correctly on various web browsers.

10. **Deployment and Hosting**:

- Choose a hosting platform such as AWS, Heroku, Netlify, or Vercel to deploy your web application.
- Set up a domain and configure DNS settings if necessary.

11. **Documentation and Version Control**:

- Maintain clear and up-to-date documentation for developers and users.
- Use version control systems like Git for collaborative development and code management.

12. **Monitoring and Analytics**:

- Implement monitoring and analytics tools like Google Analytics, New Relic, or Sentry to track user interactions and application performance.

By selecting the appropriate technologies and tools for each of these components, you can build a robust, efficient, and user-friendly Weather API web application that meets your project's objectives and requirements.

CHAPTER 5

METHODOLOGY AND ARCHITECTURE

5.1 Building a GUI Application:

We have used python and its various libraries to build a Weather Forecasting GUI Application toshow the weather data of a specific city in a User-friendly way. The flowchart for designing the GUI for this research is shown in Figure 1. To start, enter "guide" in the Tkinter command window to bring up the GUIDE Quick Start window. The user has four alternatives while building a new GUI. The user is given a blank GUI and a list of available items on the left side of the window. The finished design GUI is then represented utilising the elements that were given. After that, Tkinter's editor window automatically creates the coding for the stored GUI.

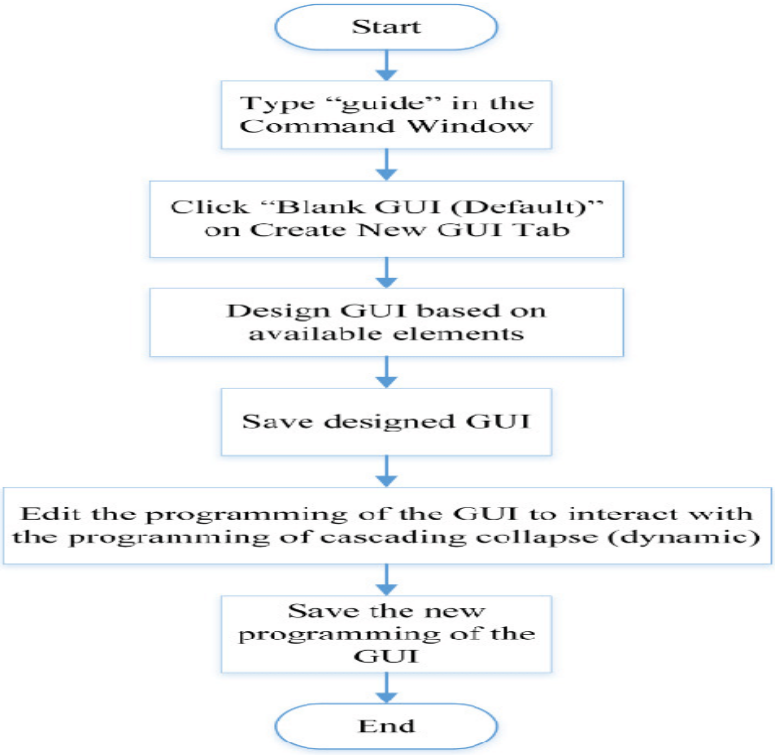


Fig: 5.1 Building of Application.

5.2 Making a API Connection:

API connection is the essential and core component of our GUI Application. We have made the connection of the GUI with Open Weather API which provides us the Real Time weather data of a specific city. The Search Bar of the application accepts a city name of any part of the world and provides or displays the accurate weather details of that specific location.

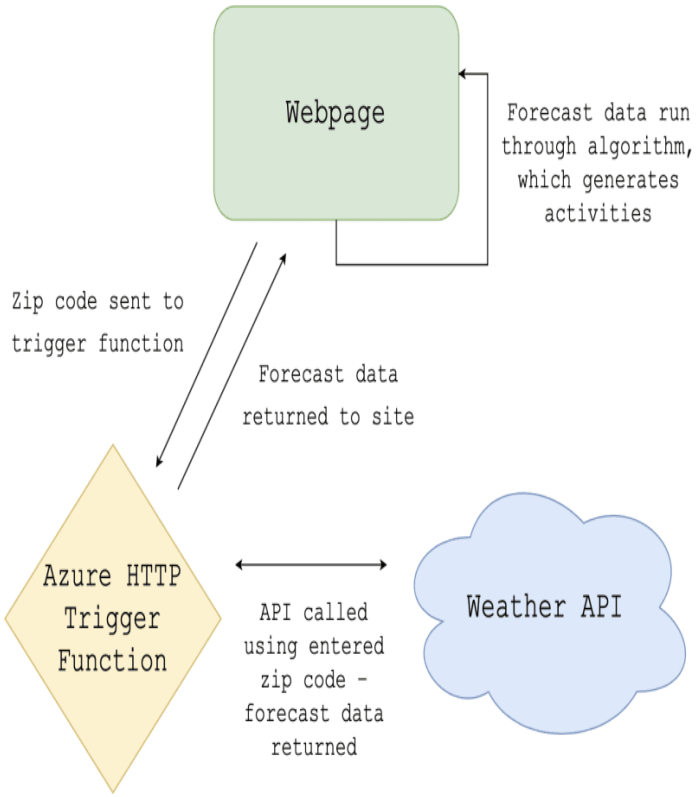


Fig: 5.2 Connecting to the API.

5.3 Retrieving the Real Time Weather Data:

As we enter the city name in the search bar option of GUI application, our API retrieves or fetches the corresponding weather data of that specific location which is available and Displays iton the GUI.

5.4 ARCHITECTURE OF TRAIN TICKET RESERVATION SYSTEM:

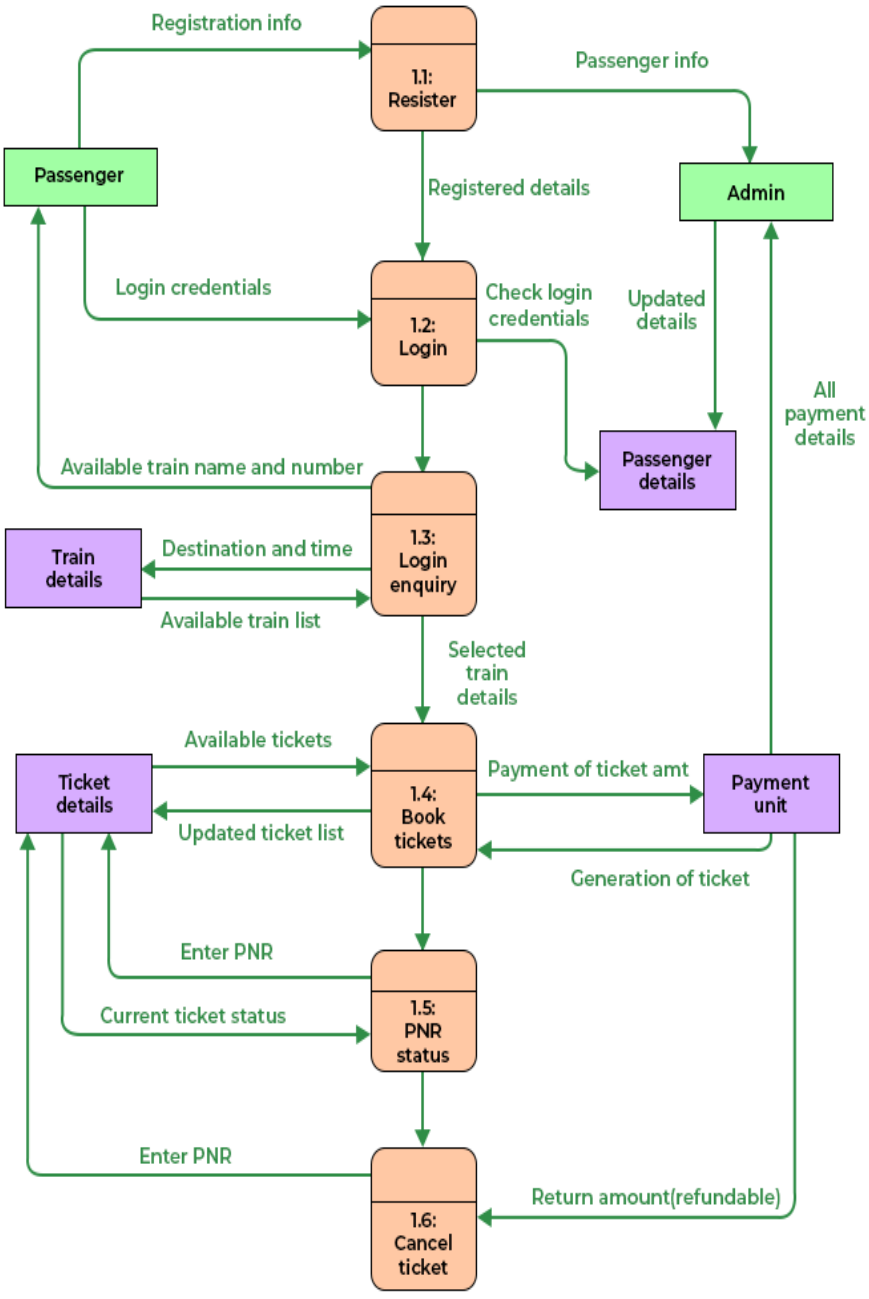
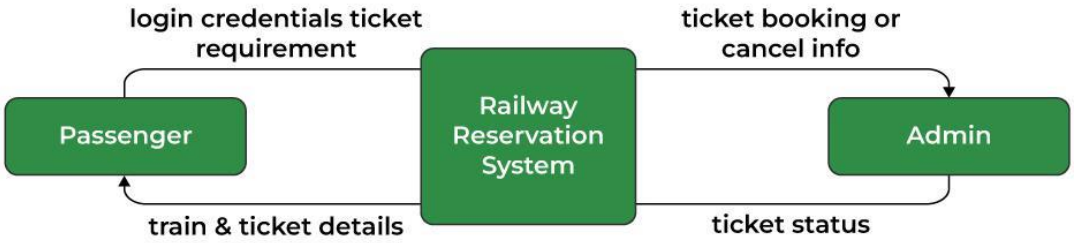


Fig: 5.3 Architecture of Train ticket reservation system

The existing train ticket reservation system faces numerous challenges that hinder its efficiency and user-friendliness. Passengers often encounter difficulties when attempting to book train tickets, leading to frustration and inconvenience. There is a pressing need to develop a modern, user-centric, and technologically advanced Train Ticket Reservation System for web platforms. This system should prioritize ease of use, accessibility, real-time information, payment security, comprehensive user support, efficient capacity management, and robust technology infrastructure to address the needs and expectations of today's travelers. The development of such a system will significantly enhance the train booking experience, streamline operations, and contribute to the overall improvement of the transportation industry.



CHAPTER 6

PROGRAM DEVELOPMENT

USED PROGRAMMING LANGUAGES:

- HTML
- CSS
- JAVASCRIPT
- PHP

6.1 HTML CODES:

6.1.1 LOGIN PAGE FOR WEATHER API

```
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" href="styles/login.css">
    <title>Login</title>
</head>
<body>
    <form action="login_process.php" method="POST">
        <h1 style="color:dodgerblue">Login</h1>
        <input type="text" placeholder="username" name="username"><br>
        <input type="password" placeholder="password" name="password"><br>
        <input type="submit" value="Login"><br>
        <a href="register.php">Register</a>
    </form>
</body>
</html>
```

6.1.2 REGISTRATION PAGE FOR WEATHER API:

```
<!DOCTYPE html>
<html>
<head>
    <title>Registration</title>
    <link rel="stylesheet" href="styles/register.css">
</head>
<body>
    <form action="register_process.php" method="POST">
    <h1 style="color:dodgerblue">Registration</h1>
        <input type="text" placeholder="username" name="username"><br>
        <input type="password" placeholder="password" name="password"><br>
        <input type="email" placeholder="email"name="email"><br>
        <input type="submit" value="Register">
    </form>
</body>
</html>
```

6.1.3 MAIN PAGE FOR WEATHER API:

```
<!DOCTYPE html>
<html lang="en">
 <head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">

   <title>Weather App Using OpenWeatherMap API</title>

   <link rel="icon" type="image/x-icon" href="icons/favicon.ico">

   <link rel="stylesheet" href="styles/style.css">
   <link rel="prefetch" href="media/day1.jpg">
   <link rel="prefetch" href="media/day2.jpg">
   <link rel="prefetch" href="media/day3.jpg">
   <link rel="prefetch" href="media/day4.jpg">
   <link rel="prefetch" href="media/day5.jpg">
   <link rel="prefetch" href="media/night1.jpg">
   <link rel="prefetch" href="media/night2.jpg">
   <link rel="prefetch" href="media/night3.jpg">
   <link rel="prefetch" href="media/night4.jpg">
   <link rel="prefetch" href="media/night5.jpg">
   <link rel="prefetch" href="media/rainy1.jpg">
   <link rel="prefetch" href="media/rainy2.jpg">
   <link rel="prefetch" href="media/rainy3.jpg">
   <link rel="prefetch" href="media/rainy4.jpg">
   <link rel="prefetch" href="media/rainy5.jpg">
   <link rel="prefetch" href="media/cloudy1.jpg">
   <link rel="prefetch" href="media/cloudy2.jpg">
   <link rel="prefetch" href="media/cloudy3.jpg">
   <link rel="prefetch" href="media/cloudy4.jpg">
   <link rel="prefetch" href="media/cloudy5.jpg">
   <link rel="prefetch" href="icons/loader.gif">

   <link rel="preconnect" href="https://fonts.googleapis.com">
   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
   <link
href="https://fonts.googleapis.com/css2?family=Montserrat:wght@500&display=swap"
rel="stylesheet">

   <style>
     body
     {
       /* background-image: linear-gradient(rgba(0, 0, 0, 0.5),rgba(0, 0, 0, 0.5)) ,
url('media/day1.jpg'); */
       -webkit-background-size: cover;
       -moz-background-size: cover;
       -o-background-size: cover;
```

```
      /* Added background-repeat and set it to no-repeat to help prevent the images from
duplicating. I added !important to
       * background-repeat and background-size in order to place importance on the
background images not repeating and
       * to cover the entire webpage. Phillip Andrews
       */
      background-repeat: no-repeat !important;
      background-size: cover !important;
      background-attachment: fixed;

      height: 100vh;
    }
    @media (max-width: 600px)
    {
      body
      {
        height: 100%;
      }
    }
  </style>
 </head>
 <body>
  <ul class="navbar">
    <li style="float: left; margin-left: -10px;" class="forecast"><a class="repo-link"
href="https://github.com/Kiran301103">Forecast</a></li>
    <li><a href="index.html">⟳</a></li>
    <li><a href="login.html">☆</a></li>
    <li><a href="about.html"><font size="5">about</font></a></li>
    <li><input spellcheck="false" class="search-city" id="searchCity" type="text"
autocomplete="off" placeholder="⍰ Search City"></li>
  </ul>

  <ul class="mobile-navbar">
    <li><input spellcheck="false" class="mobile-search-city" id="mobileSearchCity"
type="text" autocomplete="off" placeholder="Search City..."></li>
  </ul>

  <h1 class="location-name" id="locationName">Search City...</h1>

  <div class="temperature-container">
    <div class="temperature-icon"><img class="imgs-as-icons"
src="icons/sunny.png"></div>
    <div class="temperature-value" id="temperatureValue"></div>
  </div>

  <h2 class="weather-type" id="weatherType"></h2>

  <div class="additionals-first-row">
    <span class="air-quality-additional">
      <span class="air-quality-additional-label">Air Quality Index </span>
```

```
      <span class="air-quality-index-additional-value">Moderate</span>
    </span>

    <span class="real-feel-additional">
      <span class="real-feel-additional-label">Real Feel </span>
      <span class="real-feel-additional-value" id="realFeelAdditionalValue">-</span>
    </span>

    <span class="humidity-additional">
      <span class="humidity-additional-label">Humidity </span>
      <span class="humidity-additional-value" id="humidityAdditionalValue">-</span>
    </span>

    <span class="max-temperature-additional">
      <span class="max-temperature-additional-label">Highest Temperature </span>
      <span class="max-temperature-additional-value"
id="maxTemperatureAdditionalValue">-</span>
    </span>

    <span class="min-temperature-additional">
      <span class="min-temperature-additional-label">Lowest Temperature </span>
      <span class="min-temperature-additional-value" id="minTemperatureAdditionalValue">-
</span>
    </span>
  </div>

  <div class="additionals-second-row">
    <span class="wind-speed-additional">
      <span class="wind-speed-additional-label">Wind Speed </span>
      <span class="wind-speed-additional-value" id="windSpeedAdditionalValue">-</span>
    </span>

    <span class="wind-direction-additional">
      <span class="wind-direction-additional-label">Wind Direction </span>
      <span class="wind-direction-additional-value" id="windDirectionAdditionalValue">-
</span>
    </span>

    <span class="visibility-additional">
      <span class="visibility-additional-label">Visibility </span>
      <span class="visibility-additional-value" id="visibilityAdditionalValue">-</span>
    </span>

    <span class="pressure-additional">
      <span class="pressure-additional-label">Pressure </span>
      <span class="pressure-additional-value" id="pressureAdditionalValue">-</span>
    </span>

    <span class="sunrise-additional">
      <span class="sunrise-additional-label">Sunrise </span>
      <span class="sunrise-additional-value" id="sunriseAdditionalValue">-</span>
```

```
    </span>

    <span class="sunset-additional">
      <span class="sunset-additional-label">Sunset </span>
      <span class="sunset-additional-value" id="sunsetAdditionalValue">-</span>
    </span>
  </div>

      <script src="scripts/script.js"></script>
  <script src="scripts/mobile.js"></script>
  <br><br><br><br><br>
 </body>
</html>
```

6.1.4 ABOUT PAGE:

```
<!DOCTYPE html>
<html lang="en">

    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width,
initial-scale=1.0">
        <title>About Us page</title>
        <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.3.0/css/all.min.css" />
        <style>
            @import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;50
0;600&display=swap');
            *{
                padding: 0;
                margin: 0;
                box-sizing: border-box;
                font-family: poppins;
            }
            #about-section {
                width: 100%;
                height: auto;
                display: flex;
                justify-content: space-between;
                align-items: center;
```

```
        padding: 80px 12%;
    }
    .about-left img{
        width: 420px;
        height: auto;
        transform: translateY(50px);
    }
    .about-right {
        width: 54%;
    }

    .about-right ul li {
        display: flex;
        align-items: center;
    }

    .about-right h1 {
        color: #e74d06;
        font-size: 37px;
        margin-bottom: 5px;
    }

    .about-right p {
        color: #444;
        line-height: 26px;
        font-size: 15px;
    }

    .about-right .address {
        margin: 25px 0;
    }

    .about-right .address ul li {
        margin-bottom: 5px;
    }

    .address .address-logo {
        margin-right: 15px;
        color: #e74d06;
    }

    .address .saprater {
        margin: 0 35px;
    }
```

```
            .about-right .expertise ul {
                width: 80%;
                display: flex;
                align-items: center;
                justify-content: space-between;
            }

            .expertise h3 {
                margin-bottom: 10px;
            }

            .expertise .expertise-logo {
                font-size: 19px;
                margin-right: 10px;
                color: #e74d06;
            }
        </style>
    </head>

    <body>
        <section id="about-section">
            <!-- about left  -->
            <div class="about-left">
                <img src="kiran.jpeg">
            </div>

            <!-- about right  -->
            <div class="about-right">
                <h1>About Me</h1>
        <h1>I'm Kiran</h1>
                <p>I am a student currently pursuing B.Tech IT at
Sathyabama Institute of science and technology.
                </p>
                <div class="address">
                    <ul>
                        <li>
                            <span class="address-logo">
                                <i class="fas fa-paper-
plane"></i>
                            </span>
                            <p>Address</p>
                            <span class="saprater">:</span>
                            <p>Chennai, Tamilnadu, India</p>
```

```
                    </li>
                    <li>
                        <span class="address-logo">
                            <i class="fas fa-phone-alt"></i>
                        </span>
                        <p>Phone No</p>
                        <span class="saprater">:</span>
                        <p>+91 8870292116</p>
                    </li>
                    <li>
                        <span class="address-logo">
                            <i class="far fa-envelope"></i>
                        </span>
                        <p>Email ID</p>
                        <span class="saprater">:</span>
                        <p>kiran301103@gmail.com</p>
                    </li>
                </ul>
            </div>
            <div class="expertise">
                <h3>My Expertise</h3>
                <ul>
                    <li>
                        <span class="expertise-logo">
                            <i class="fab fa-html5"></i>
                        </span>
                        <p>HTML</p>
                    </li>
                    <li>
                        <span class="expertise-logo">
                            <i class="fab fa-css3-alt"></i>
                        </span>
                        <p>CSS</p>
                    </li>
                    <li>
                        <span class="expertise-logo">
                            <i class="fab fa-node-js"></i>
                        </span>
                        <p>Java Script</p>
                    </li>
                    <li>
                        <span class="expertise-logo">
                            <i class="fab fa-react"></i>
                        </span>
```

```
                        <p>Python</p>
                    </li>
                </ul>
            </div>
        </div>
    </section>
</body>

</html>
```

6.2 CSS CODES:

6.2.1 STYLESHEET OF MAIN PAGE:

```
*, *::before, *::after
{
  box-sizing: border-box;
  outline: none;
  -webkit-tap-highlight-color: transparent;
  opacity: 100%;
  /* transition: 0.25s; */
}

.mobile-navbar
{
  display: none;
}

body
{
  margin: 0;
  font-family: "Montserrat", sans-serif;
  font-weight: 500;
}

#loader1, #loader2, #loader3
{
  width: 37.5px;
  height: 37.5px;
}

.imgs-as-icons
{
  width: 50px;
  height: 50px;
}

.navbar
```

```
{
  /* background: #212a51; */
  background: rgba(0, 0, 0, 0.375);
  text-align: right;
  opacity: 100%;
  transition: 0.3s;
}

/* START OF MOBILE STYLES */

.mobile-navbar
{
  list-style-type: none;
  /* background: rgba(0, 0, 0, 1.0); */
  opacity: 100%;
  transition: 0.3s;
}

.mobile-navbar li
{
  margin: 7.5px auto;
  width: 87.5%;
}

.mobile-navbar li input
{
  height: 50px;
  width: 100%;
  background: #1d1d1f;
  color: #fff;
  text-align: center;
  border: none;
  border-radius: 50px;
  outline: none;
  font-size: 20px;
}

.mobile-navbar input::placeholder
{
  color: lightgrey;
}

/* END OF MOBILE STYLES */

ul
{
  margin: 0;
  padding: 5px;
  transition: 0.3s;
}
```

```css
.navbar li
{
  margin: 0 0;
  padding: 5px;
  display: inline-block;
  transition: 0.3s;
}

.navbar li a
{
  font-size: 25px;
  color: #fff;
  padding: 5px 10px;
  text-decoration: none;
  transition: 0.3s;
  border-radius: 5px;
}

.navbar .search-city
{
  width: 25vw;
  height: 5.75vh;
  font-size: 18.75px;
}

.search-city
{
  letter-spacing: 0.50px;
  padding-left: 20px;
  background: rgba(0, 0, 0, 0.50);
  border: none;
  color: #fff;
  border-radius: 50px;
  outline: none;
}

.navbar a:hover:not(.repo-link)
{
  background: #000;
  transition: 0.3s;
  border-radius: 5px;
}

.location-name
{
  font-weight: 400;
  text-align: center;
  color: #fff;
}

.temperature-container
```

```
{
  text-align: center;
  margin: auto;
  font-size: 50px;
}

.temperature-icon
{
  margin-bottom: -15px;
  padding: 0;
}

.temperature-value
{
  margin-top: -15px;
  padding: 0;
  color: #fff;
}

.weather-type
{
  font-weight: 400;
  text-align: center;
  color: #fff;
}

.additionals-first-row, .additionals-second-row
{
  font-size: 12.5px;
  width: 100%;
  text-align: center;
}

.additionals-first-row
{
  margin-bottom: 12.5px;
}

.additionals-second-row
{
  margin-top: 12.5px;
}

.air-quality-additional, .real-feel-additional, .humidity-additional, .max-temperature-additional,
.min-temperature-additional
{
  margin-left: 5px;
  margin-right: 5px;
  background: #fff;
  border-radius: 50px;
  padding: 2.5px 10px;
```

```
}

.wind-speed-additional, .wind-direction-additional, .visibility-additional, .pressure-additional,
.sunrise-additional, .sunset-additional
{
  margin-left: 5px;
  margin-right: 5px;
  background: #fff;
  border-radius: 50px;
  padding: 2.5px 10px;
}

.separator
{
  width: 25%;
  margin: auto;
  height: 7.5px;
  background: lightgrey;
  border-radius: 50px;
}

.daily-forecast
{
  padding: 25px;
  margin-top: -25px;
  text-align: center;
}

.daily-forecast .daily-forecast-label
{
  color: #fff;
}


.daily-forecast-card
{
  display: inline-block;
  border-radius: 10px;
  background: rgba(0, 0, 0, 0);
  width: 125px;
  height: 162.5px;
  padding: 2.5px 10px;
  margin-left: 2.5px;
  margin-right: 2.5px;
  color: #fff;
}

.daily-forecast-card .daily-forecast-logo
{
  font-size: 37.5px;
  margin: -10px -10px;
```

```
}

.daily-forecast-card .daily-forecast-date
{
  font-size: 17.5px;
  margin-top: 12.5px;
  font-weight: 400;
}

.daily-forecast-card .daily-forecast-logo
{
  margin-top: -15px;
  margin-bottom: 2.5px;
}

.daily-forecast-card .weather-type-daily-forecast
{
  font-weight: normal;
  margin-top: 2.5px;
}

.max-daily-forecast
{
  font-weight: bold;
  font-size: 25px;
}

@media (max-width: 600px)
{
  .navbar
  {
    display: none;
  }

  .mobile-navbar
  {
    display: block;

  }

  .additionals-first-row, .additionals-second-row
  {
    line-height: 27.5px;
    margin: 0;
  }

  .additionals-first-row span, .additionals-second-row span
  {
    display: none;
  }
```

```
 .daily-forecast-card
 {
   margin-top: 10px;
   margin-bottom: 10px;
 }
 .daily-forecast .daily-forecast-label
 {
   background: lightgrey;
   color: #252525;
   padding: 10px;
   width: 75%;
   border-radius: 50px;
   margin: auto;
 }
 #weatherType
 {
   margin-top: 0;
 }
}
```

6.2.2 STYLESHEET OF LOGIN PAGE:

```
Body
{
    font-size:35px;
    padding:0;
    margin:0;
    background:url('https://wallpaperaccess.com/full/4893694.jpg');
    background-size:cover;

}
form
{
    top:30%;
    left:40%;
    position: absolute;
    padding:20px;
}
form>h1
{
    text-align:center;
}
#name
{
    width:250px;
}
```

6.2.3 STYLESHEET OF REGISTRATION PAGE:

```
Body
{
    font-size:35px;
    padding:0;
    margin:0;
    background:url('https://wallpaperaccess.com/full/4893694.jpg');
    background-size:cover;

}
form
{
    top:30%;
    left:37%;
    position: absolute;
    padding:20px;
}
form>h1
{
    text-align:center;
}
#name
{
    width:250px;
}
```

6.3 JAVASCRIPT CODE:

6.3.1 DESKTOP JAVASCRPT:

```
var cityInput = document.getElementById("searchCity");

var backgroundsList = [
 "day1.jpg",
 "day2.jpg",
 "day3.jpg",
 "day4.jpg",
 "day5.jpg",
 // "night1.jpg",
 // "night2.jpg",
 // "night3.jpg",
 // "night4.jpg",
 // "night5.jpg",
 "cloudy1.jpg",
 "cloudy2.jpg",
 "cloudy3.jpg",
 "cloudy4.jpg",
 "cloudy5.jpg",
```

```
  // "rainy1.jpg",
  // "rainy2.jpg",
  // "rainy3.jpg",
  // "rainy4.jpg",
  // "rainy5.jpg",
];

var randomBackground = backgroundsList[Math.floor(Math.random() *
backgroundsList.length)];

document.body.style.background = "linear-gradient(rgba(0, 0, 0, 0.5),rgba(0, 0, 0, 0.5)) ,
url('media/" + randomBackground + "')";

cityInput.addEventListener("keyup", function(event)
{
  if(event.key === "Enter")
  {
    loader();
    function loader()
    {

      document.getElementById("locationName").innerHTML = "";
      document.getElementById("temperatureValue").innerHTML = "";
      document.getElementById("weatherType").innerHTML = "";

      const img1 = document.createElement("img");
      const img2 = document.createElement("img");
      const img3 = document.createElement("img");

      img1.id = "loader1";
      img2.id = "loader2";
      img3.id = "loader3";

      img1.src = "icons/loader.gif";
      img2.src = "icons/loader.gif";
      img3.src = "icons/loader.gif";

      const parentElement1 = document.getElementById("locationName");
      const parentElement2 = document.getElementById("temperatureValue");
      const parentElement3 = document.getElementById("weatherType");

      parentElement1.appendChild(img1);
      parentElement2.appendChild(img2);
      parentElement3.appendChild(img3);

      // document.getElementById("loader1").src = "icons/loader.gif";
      // document.getElementById("loader2").src = "icons/loader.gif";
      // document.getElementById("loader3").src = "icons/loader.gif";
    }

    var cityInputValue = cityInput.value;
```

```
    var apiKey = "b1fd6e14799699504191b6bdbcadfc35"; // Default
    var unit = "metric";
    var apiUrl =
`https://api.openweathermap.org/data/2.5/weather?q=${cityInputValue}&appid=${apiKey}&un
its=${unit}`;

    if(cityInputValue != "")
    {
      async function getWeather()
      {
        var response = await fetch(apiUrl);
        var data = await response.json();

        if(data.message != "city not found" && data.cod != "404")
        {
          var location = data.name;
          var temperature = data.main.temp;
          var weatherType = data.weather[0].description;
          var realFeel = data.main.feels_like;
          var windSpeed = data.wind.speed;
          var windDirection = data.wind.deg;
          var visibility = data.visibility / 1000;
          var pressure = data.main.pressure;
          var maxTemperature = data.main.temp_max;
          var minTemperature = data.main.temp_min;
          var humidity = data.main.humidity;
          var sunrise = data.sys.sunrise;
          var sunset = data.sys.sunset;

          document.getElementById("locationName").innerHTML = location;
          document.getElementById("temperatureValue").innerHTML = temperature +
"<sup>o</sup>C";
          document.getElementById("weatherType").innerHTML = weatherType;
          document.getElementById("realFeelAdditionalValue").innerHTML = realFeel +
"<sup>o</sup>C";
          document.getElementById("windSpeedAdditionalValue").innerHTML = windSpeed + "
km/h";
          document.getElementById("windDirectionAdditionalValue").innerHTML =
windDirection;
          document.getElementById("visibilityAdditionalValue").innerHTML = visibility + " km";
          document.getElementById("pressureAdditionalValue").innerHTML = pressure;
          document.getElementById("maxTemperatureAdditionalValue").innerHTML =
maxTemperature + "<sup>o</sup>C";
          document.getElementById("minTemperatureAdditionalValue").innerHTML =
minTemperature + "<sup>o</sup>C";
          document.getElementById("humidityAdditionalValue").innerHTML = humidity;
          document.getElementById("sunriseAdditionalValue").innerHTML = sunrise;
          document.getElementById("sunsetAdditionalValue").innerHTML = sunset;
        }
        else
```

```
      {
        document.getElementById("locationName").innerHTML = "City Not Found";
        document.getElementById("temperatureValue").innerHTML = "";
        document.getElementById("weatherType").innerHTML = "";
      }
    }

    getWeather();
  }

  else document.getElementById("locationName").innerHTML = "Enter a city name...";
 }
});
```

6.3.2 MOBILE JAVASCRIPT:

```
var cityInputMobile = document.getElementById("mobileSearchCity");

cityInputMobile.addEventListener("keyup", function(event)
{
 if(event.key === "Enter")
 {
  loader();
  function loader()
  {

    document.getElementById("locationName").innerHTML = "";
    document.getElementById("temperatureValue").innerHTML = "";
    document.getElementById("weatherType").innerHTML = "";

    const img1 = document.createElement("img");
    const img2 = document.createElement("img");
    const img3 = document.createElement("img");

    img1.id = "loader1";
    img2.id = "loader2";
    img3.id = "loader3";

    img1.src = "icons/loader.gif";
    img2.src = "icons/loader.gif";
    img3.src = "icons/loader.gif";

    const parentElement1 = document.getElementById("locationName");
    const parentElement2 = document.getElementById("temperatureValue");
    const parentElement3 = document.getElementById("weatherType");

    parentElement1.appendChild(img1);
    parentElement2.appendChild(img2);
    parentElement3.appendChild(img3);
```

```
    // document.getElementById("loader1").src = "icons/loader.gif";
    // document.getElementById("loader2").src = "icons/loader.gif";
    // document.getElementById("loader3").src = "icons/loader.gif";
  }

  var cityInputValue = cityInputMobile.value;

  var apiKey = "b1fd6e14799699504191b6bdbcadfc35"; // Default
  var unit = "metric";
  var apiUrl =
`https://api.openweathermap.org/data/2.5/weather?q=${cityInputValue}&appid=${apiKey}&un
its=${unit}`;

  if(cityInputValue != "")
  {
    async function getWeather()
    {
      var response = await fetch(apiUrl);
      var data = await response.json();

      if(data.message != "city not found" && data.cod != "404")
      {
        var location = data.name;
        var temperature = data.main.temp;
        var weatherType = data.weather[0].description;
        var realFeel = data.main.feels_like;
        var windSpeed = data.wind.speed;
        var windDirection = data.wind.deg;
        var visibility = data.visibility / 1000;
        var pressure = data.main.pressure;
        var maxTemperature = data.main.temp_max;
        var minTemperature = data.main.temp_min;
        var humidity = data.main.humidity;
        var sunrise = data.sys.sunrise;
        var sunset = data.sys.sunset;

        document.getElementById("locationName").innerHTML = location;
        document.getElementById("temperatureValue").innerHTML = temperature +
"<sup>o</sup>C";
        document.getElementById("weatherType").innerHTML = weatherType;
        document.getElementById("realFeelAdditionalValue").innerHTML = realFeel +
"<sup>o</sup>C";
        document.getElementById("windSpeedAdditionalValue").innerHTML = windSpeed + "
km/h";
        document.getElementById("windDirectionAdditionalValue").innerHTML =
windDirection;
        document.getElementById("visibilityAdditionalValue").innerHTML = visibility + " km";
        document.getElementById("pressureAdditionalValue").innerHTML = pressure;
        document.getElementById("maxTemperatureAdditionalValue").innerHTML =
maxTemperature + "<sup>o</sup>C";
```

```
        document.getElementById("minTemperatureAdditionalValue").innerHTML =
minTemperature + "<sup>o</sup>C";
        document.getElementById("humidityAdditionalValue").innerHTML = humidity;
        document.getElementById("sunriseAdditionalValue").innerHTML = sunrise;
        document.getElementById("sunsetAdditionalValue").innerHTML = sunset;
      }
      else
      {
        document.getElementById("locationName").innerHTML = "City Not Found";
        document.getElementById("temperatureValue").innerHTML = "";
        document.getElementById("weatherType").innerHTML = "";
      }
    }
    getWeather();
  }
  else document.getElementById("locationName").innerHTML = "Enter a city name...";
 }
});
```

6.4 PHP CODES:

6.4.1 LOGIN PAGE FOR WEATHER API:

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST["username"];
    $password = $_POST["password"];
    // Check the user's credentials in the database
    $conn = new mysqli("localhost", "root", "", "kiran");
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }

    $sql = "SELECT id, username, password FROM users WHERE username =
'$username'";
    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        if (password_verify($password, $row["password"])) {
            header("Location: index.html");
        } else {
            echo "Incorrect password.";
        }
    } else {
        echo "User not found.";
    }

    $conn->close();
}
?>
```

6.4.2 REGISTRATION PAGEWEATHER API:

```php
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST["username"];
    $password = password_hash($_POST["password"], PASSWORD_BCRYPT);
    $email = $_POST["email"];

    // Insert data into the "users" table in the database
    $conn = new mysqli("localhost", "root", "", "kiran");

    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }

    $sql = "INSERT INTO users (username, password, email) VALUES
('$username', '$password', '$email')";

    if ($conn->query($sql) === TRUE) {
        echo "Registration successful!";
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }

    $conn->close();
}
?>
```

6.5 CODES FOR TRAIN TICKET RESERVATION:

```php
<?php
session_start();
require('firstimport.php');
if(isset($_SESSION['name'])){}
    else{
        header("location:login1.php");
    }
$tbl_name="booking";
mysqli_select_db($conn,"$db_name") or die("cannot select db");
$uname=$_SESSION['name'];
$num=$_GET['tno'];
$seat= $_GET['selct'];
$name=$_GET['name1'];
$age=$_GET['age1'];
$sex=$_GET['sex1'];
$fromstn=$_GET[ 'fromstn'];
$tostn=$_GET['tostn'];
$doj=$_GET['doj'];
$dob=$_GET['dob'];
echo "..".$num."..".$name."..".$age."..".$sex."..".$seat."..";
```

```
//echo "</br>".$value."</br>".$seat."</br>";
if($value>0){
    $status="Confirmed";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
$sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
else{
    $status="Waiting";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
    $sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
    $name=$_GET['name2'];
    $age=$_GET['age2'];
    $sex=$_GET['sex2'];

if($value>0){
    $status="Confirmed";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
```

```
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
$sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
else{
    $status="Waiting";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
    $sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
    $name=$_GET['name3'];
    $age=$_GET['age3'];
    $sex=$_GET['sex3'];
if($value>0){
    $status="Confirmed";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
```

```
$sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
else{
    $status="Waiting";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
    $sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
    $name=$_GET['name4'];
    $age=$_GET['age4'];
    $sex=$_GET['sex4'];

if($value>0){
    $status="Confirmed";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
$sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
```

```
else{
    $status="Waiting";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
    $sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}

   $name=$_GET['name5'];

   $age=$_GET['age5'];

    $sex=$_GET['sex5'];

if($value>0){
    $status="Confirmed";
    if(!empty($name) || !empty($age) )
    {
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
$sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
else{
    $status="Waiting";
    if(!empty($name) || !empty($age) )
    {
```

```
    $sql="INSERT INTO
$tbl_name(uname,Tnumber,class,doj,DOB,fromstn,tostn,Name,Age,sex,Status)
    VALUES
('$uname','$num','$seat','$doj','$dob','$fromstn','$tostn','$name','$age',
'$sex','$status')";
    $result=$conn->query($sql);
    echo "$sql</br>";
    if(!$result) die ($conn->error);
    $value-=1;
    $sql2="UPDATE seats_availability SET ".$seat."=".$value." WHERE
doj='".$doj."' and Train_No=".$num."";
    $result2=$conn->query($sql2);
    echo "</br>".$sql2."</br>";
    if(!$result) die ($conn->error);
    }
}
    echo("file succesfully inserted");

header("location:display.php?tno='$num'&& doj='$doj'&& seat='$seat'");
?>
```

```php
<?php

session_start();
$pass=$_GET['pass'];
$name=$_SESSION['name'];

require('firstimport.php');

$db_name="railres";
$tbl_name="users"; // Table name

mysqli_select_db($conn, "$db_name")or die("cannot select DB");

$sql="UPDATE users SET password=$pass WHERE f_name='$name'";


$result=mysqli_query($conn,$sql);

$_SESSION['error']=6;

//echo "name : ".$name."  Pass : ".$pass;
header('location:profile.php');
?>
```

```php
<?php
session_start();


require('firstimport.php');
if(isset($_SESSION['name'])){}
    else{
        header("location:login1.php");

    }
$tbl_name="users"; // Table name

mysqli_select_db($conn,"$db_name")or die("cannot select DB");


if(!isset($_SESSION["name"]))
header("location:login1.php");

$name=$_SESSION['name'];
$lname=$_POST['ln'];
$mail=$_POST['mail1'];
$gender=$_POST['gnd1'];
$marital=$_POST['mrt1'];
$dob=$_POST['dob1'];
$mobile=$_POST['mon1'];
$ques=$_POST['que1'];
$ans=$_POST['ans1'];



$sql="UPDATE $tbl_name SET
l_name='$lname',email='$mail',gender='$gender',marital='$marital'
,dob='$dob',mobile='$mobile',ques='$ques',ans='$ans' WHERE
f_name='$name'";
$result=mysqli_query($conn,$sql);


$_SESSION['error']==4;

header('location:profile.php');

?>
```

```php
<?php
$host="localhost"; // Host name
$username="root"; // Mysql username
$password=""; // Mysql password
$db_name="railres"; // Database name

$conn=mysqli_connect("$host", "$username", "$password")or
die("cannot connect");

?>
```

```php
<?php
session_start();
?>

<!DOCTYPE html>
<html>
<head>
    <title> Indian Railways </title>
    <link rel="shortcut icon" href="images/favicon.png"></link>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta name="description" content="">
    <meta name="author" content="">
    <link href="css/bootstrap.min.css" rel="stylesheet" ></link>
    <link href="css/bootstrap.css" rel="stylesheet" ></link>
    <link href="css/Default.css" rel="stylesheet" > </link>
    <script type="text/javascript" src="js/jquery.js"></script>
    <script>
        $(document).ready(function()
        {
            var x=(($(window).width())-1024)/2;
            $('.wrap').css("left",x+"px");
        });
    </script>

    <script type="text/javascript"
src="js/bootstrap.min.js"></script>
    <script type="text/javascript"
src="js/bootstrap.js"></script>
    <script type="text/javascript" src="js/man.js"></script>
```

```
</head>
<body>

    <div class="wrap">
        <!-- Header -->
        <div class="header">
            <div style="float:left;width:150px;">
                <img src="images/logo.jpg"/>
            </div>
            <div>
            <div style="float:right; font-size:20px;margin-
top:20px;">
            <?php
             if(isset($_SESSION['name']))
             {
             echo
"Welcome,".$_SESSION['name']."   <a
href=\"logout.php\" class=\"btn btn-info\">Logout</a>";
             }
             else
             {
             ?>
                <a href="login1.php" class="btn btn-
info">Login</a>    
                <a href="signup.php?value=0" class="btn btn-
info">Signup</a>
            <?php } ?>


            </div>
            <div id="heading">
                <a href="index.php">Indian Railways</a>
            </div>
            </div>
        </div>

        <!-- Navigation bar -->
        <div class="navbar navbar-inverse">
            <div class="navbar-inner">
                <div class="container" >
                <a class="brand" href="index.php" >HOME</a>
                <a class="brand" href="train.php" >FIND TRAIN</a>
                <a class="brand"
href="reservation.php">RESERVATION</a>
```

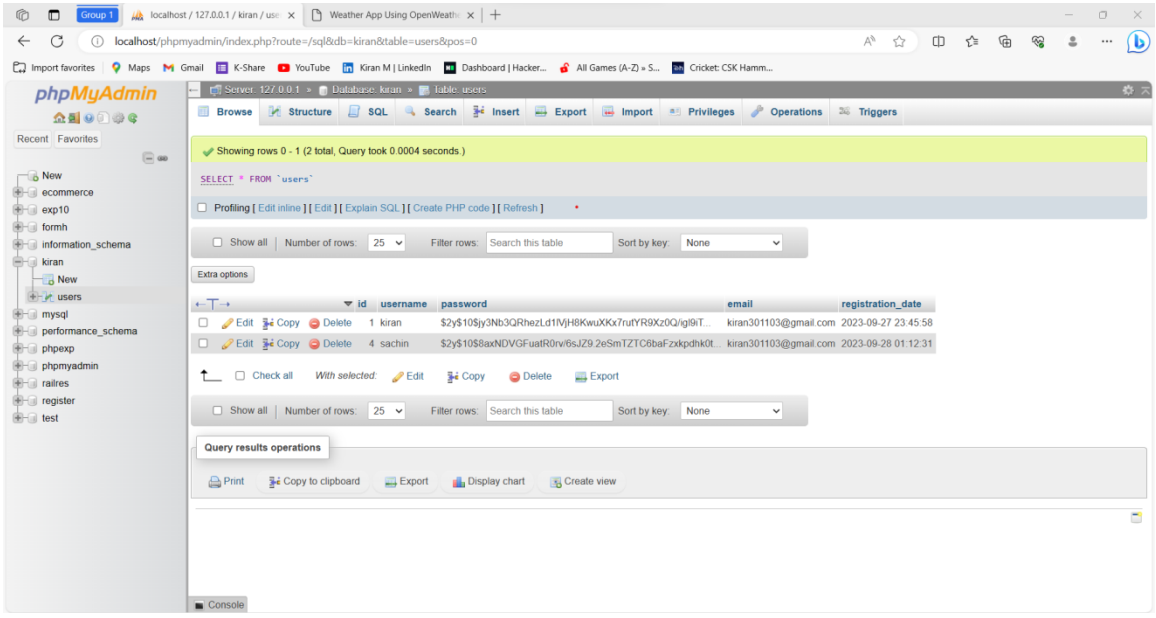
```
                <a class="brand" href="profile.php">PROFILE</a>
                <a class="brand" href="booking.php">BOOKING
HISTORY</a>
                </div>
            </div>
        </div>
        <div class="span12 well">
            <!-- Photos slider -->
            <div id="myCarousel" class="carousel slide"
style="width:600px; float:left;margin-bottom:3px;">
                <div class="carousel-inner">
                <div class="active item"><img src="images/6.jpg"
style="width:600px;height:350px;"/></div>
                <div class="item"><img src="images/7.jpg"
style="width:600px;height:350px;"/> </div>
                <div class="item"><img src="images/8.jpg"
style="width:600px;height:350px;"/></div>
                <div class="item"><img src="images/9.jpg"
style="width:600px;height:350px;"/></div>
                <div class="item"><img src="images/10.jpg"
style="width:600px;height:350px;"/> </div>
                <div class="item"><img
src="images/11.png"style="width:600px;height:350px;"/></div>

                </div>
                <a class="carousel-control left"
href="#myCarousel" data-slide="prev">&lsaquo;</a>
                <a class="carousel-control right"
href="#myCarousel" data-slide="next">&rsaquo;</a>
            </div>
            <!-- News and Alert-->
            <div class="news" Style="float:right;">
            <marquee behavior="scroll" id="marq"  scrollamount=3
direction="up" height="294px" onmouseover="nestop()"
onmouseout="nestrt()">
                <div>
                <p><b>There is no proposal to extend to
mail/express and superfast trains the flexi-fares currently
applicable only to Rajdhani, Shatabdi and Duronto trains,  said
Railways Minister Suresh Prabhu.</b></p>
                </br>
                <p><b>The Railway ministry has posted the rate
list on its Twitter account while asking people to lodge a
complaint if they are overcharged.</b></p></br>
```

```
                <p><b>The Comptroller and Auditor General (CAG)
has asked the railways to revise passenger fares and curtail
concessional passes to recover its operating cost in a phased
manner.</b></p></br>
                <p><b>Railway issues new catering policy for
better food.</b></p></br>
                <p><b>Passengers will now be involved in judging
cleanliness level of popular trains including Rajdhani, Shatabdi
and Duronto as well as major stations across the
country.</b><p></br>


                </div>
            </marquee>
            </div>
        </div>

        <!-- Copyright -->
<footer >

            <div style="float:right;">
      <p class="text-right text-info">    Desinged By:KIRAN</p>
            </div>
        </div>
        </footer>   </div>

</body>
</html>

<?php

if(isset($_SESSION['error']))
{
session_destroy();
}

?>
```

CHAPTER 7

CONCLUSION

In conclusion, the weather API web development project has successfully delivered a user-friendly and reliable platform for accessing weather information. It has met its objectives and can be a valuable resource for individuals and businesses alike. As we move forward, we remain comited to improving and expanding the project to serve our users better and to keep pace with advancements in technology and meteorology. Continuous improvement, security, and user feedback will remain essential elements of the project as we move forward. This project has demonstrated the importance of leveraging technology to enhance our understanding and interaction with the world around us, and it holds great potential for future enhancements and expansions.



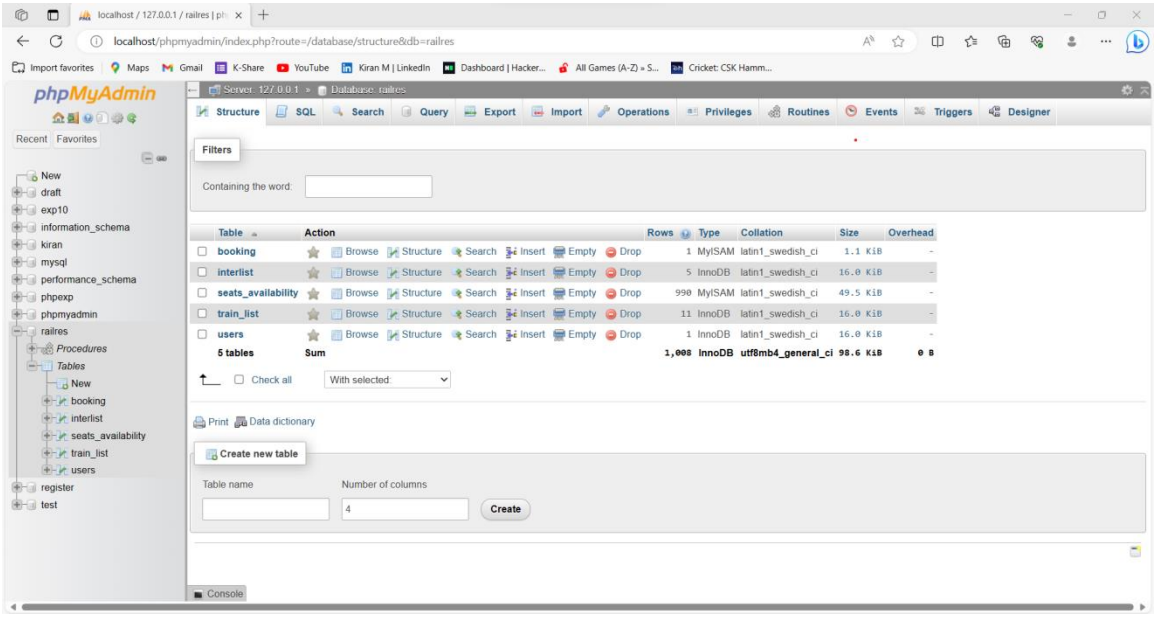


FIG: 7.1 CREATION OF TABLE AND DATABASE

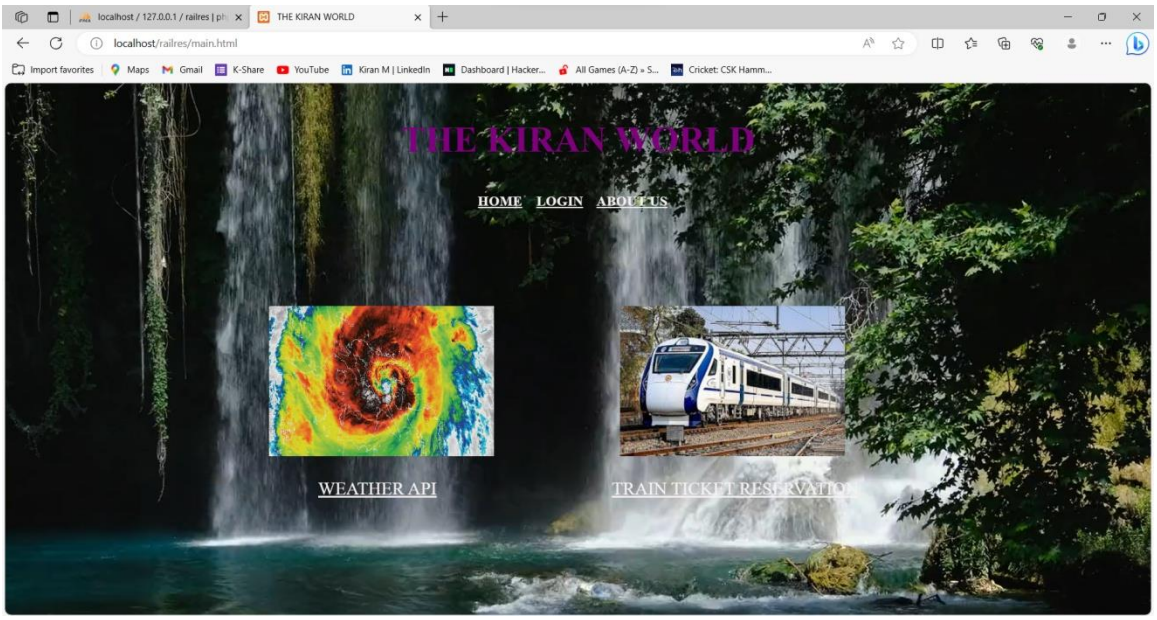


FIG: 7.2 HOME PAGE

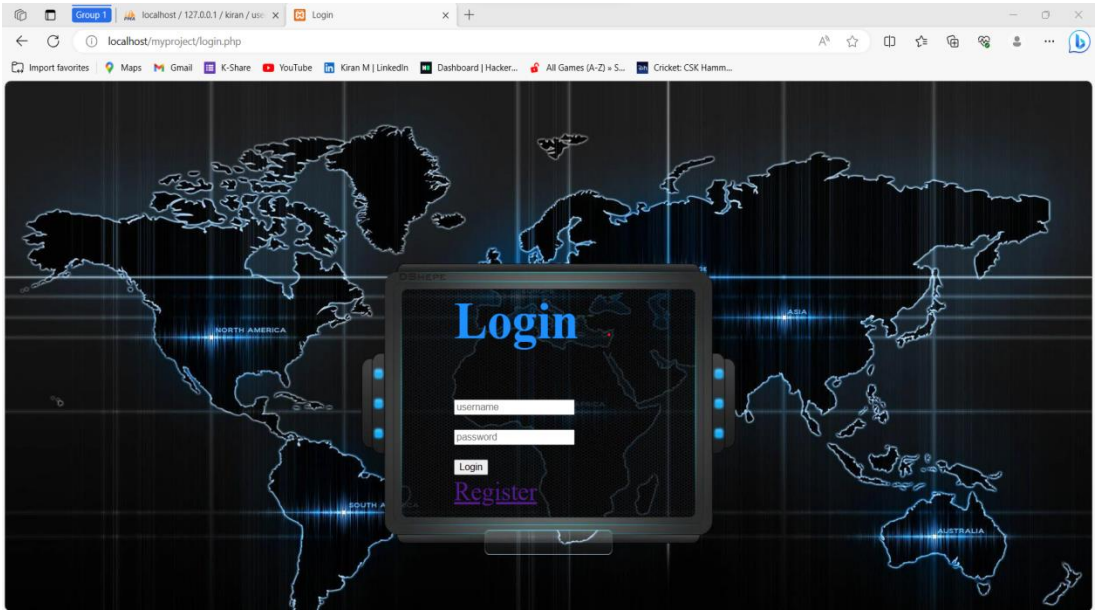


FIG: 7.3 LOGIN PAGE

FIG: 7.4 REGISTRATION PAGE

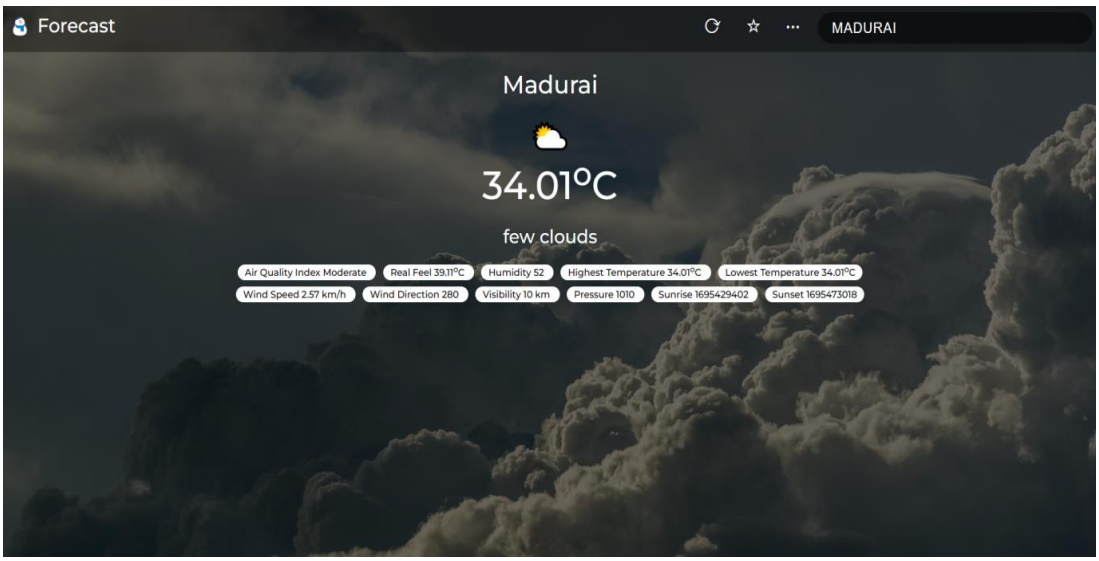


FIG: 7.5 OUTPUT1

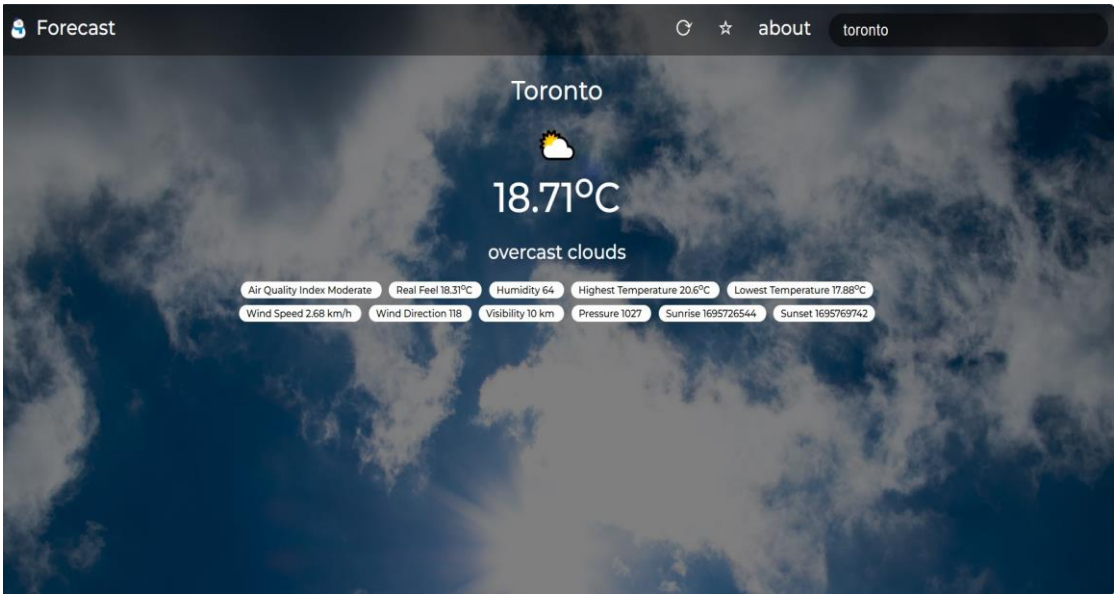


FIG: 7.6 OUTPUT2

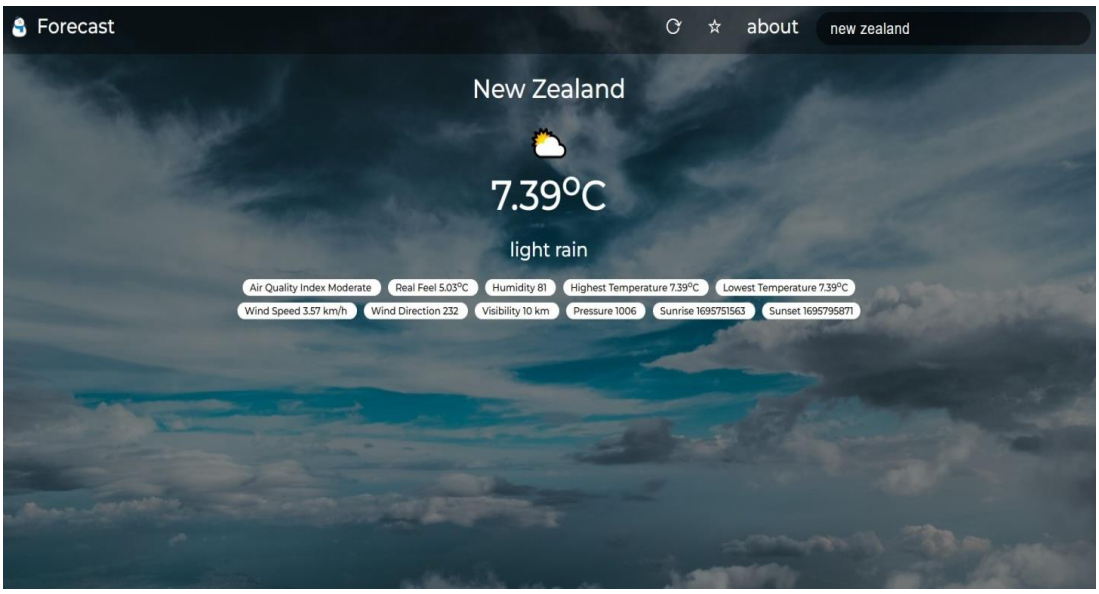


FIG: 7.7 OUTPUT3

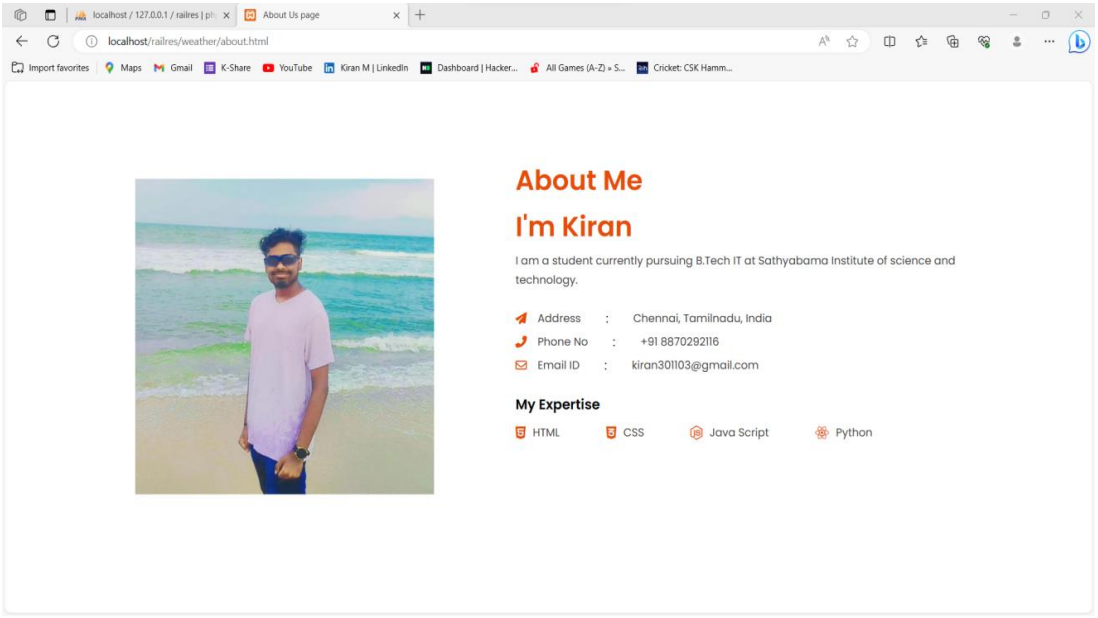
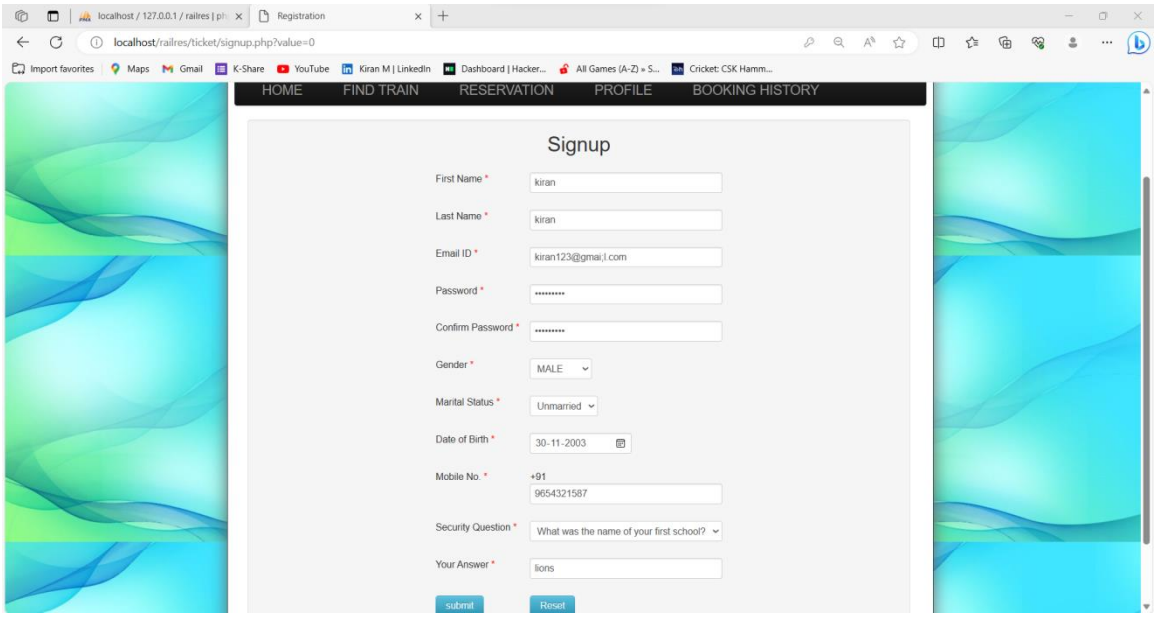
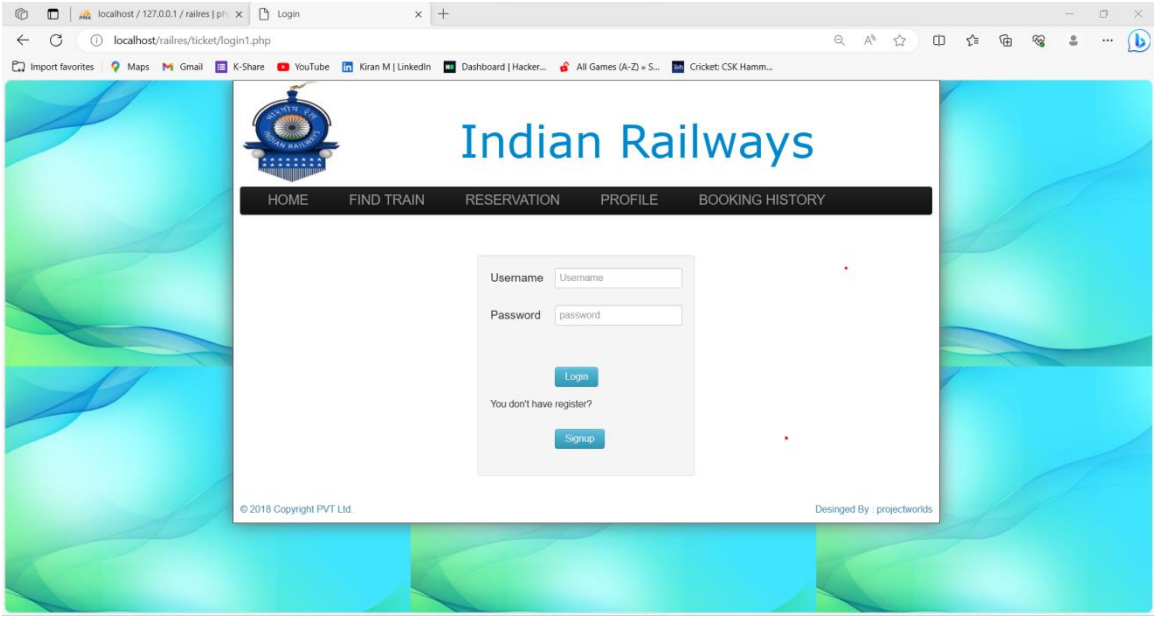
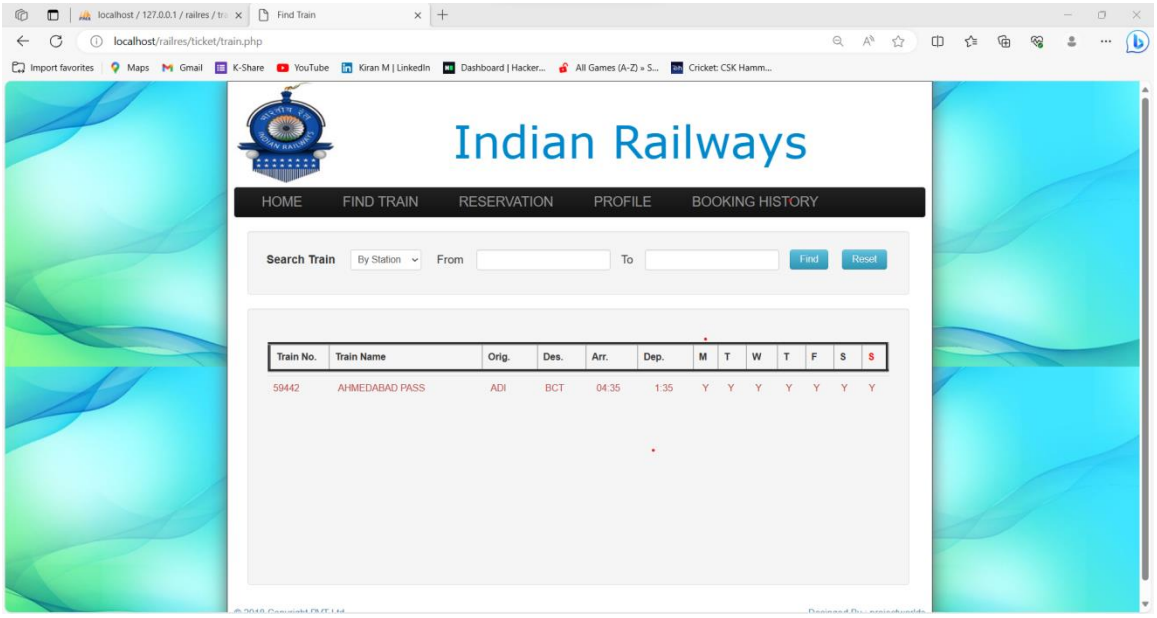


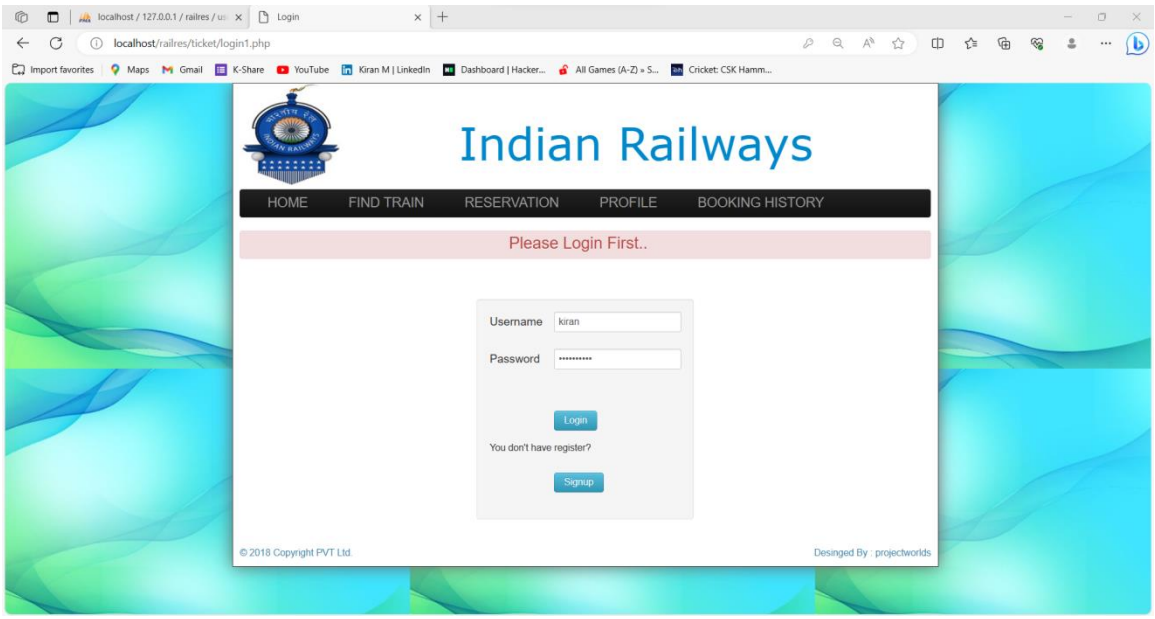
FIG: 7.7 ABOUT PAGE

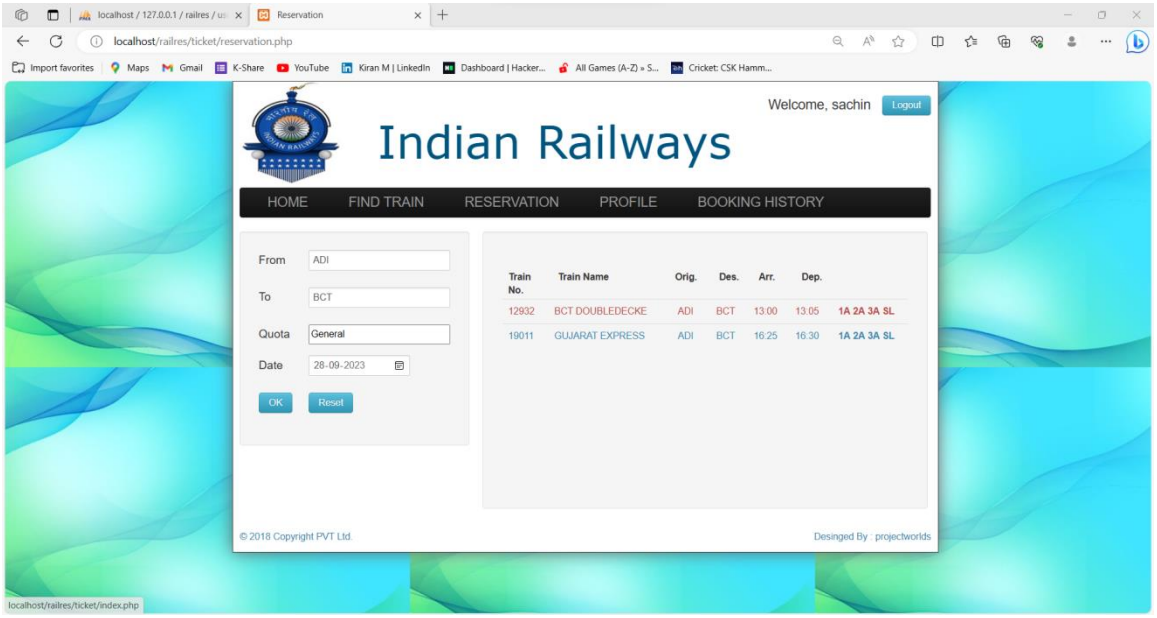
7.1 IMAGES OF THE TRAIN TICKET RESERVATION WEBPAGE:

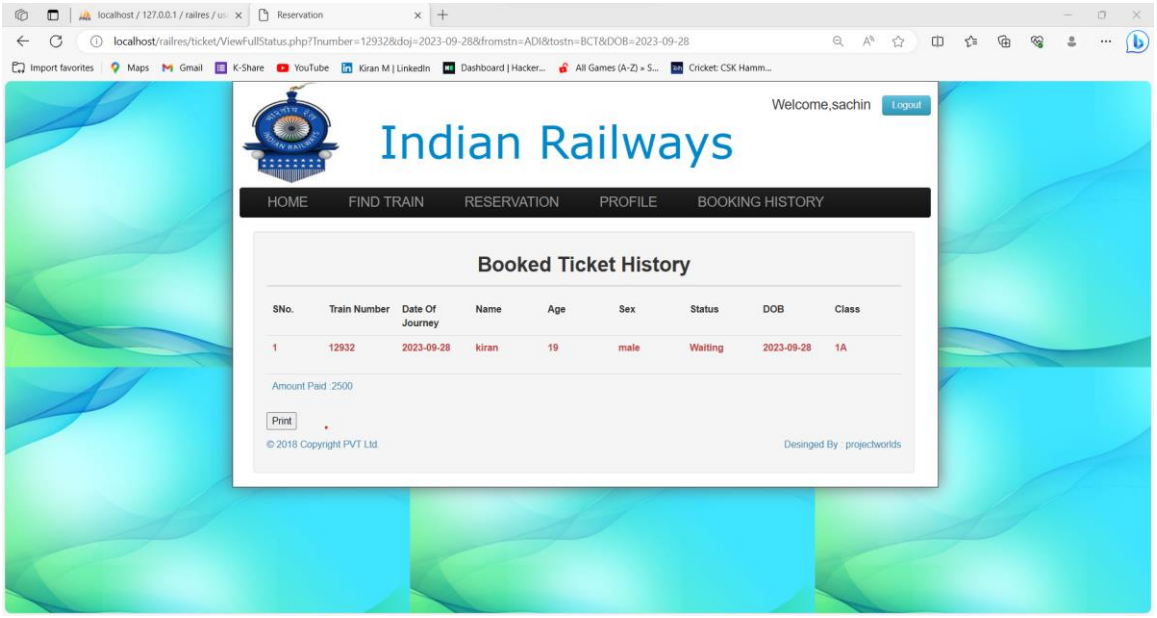





Find Train
localhost/railres/ticket/train.php
Import favorites
Maps
Gmail
K-Share
YouTube
Kiran M | LinkedIn
Dashboard | Hacker...
All Games (A-Z) » S...
Cricket: CSK Hamm...
Indian Railways
HOME
FIND TRAIN
RESERVATION
PROFILE
BOOKING HISTORY
Search Train
By Station
From
To
Find
Reset
Train No.
Train Name
Orig.
Des.
Arr.
Dep.
M
T
W
T
F
S
S
59442
AHMEDABAD PASS
ADI
BCT
04:35
1:35
Y
Y
Y
Y
Y
Y
Y


Login
localhost/railres/ticket/login1.php
Import favorites
Maps
Gmail
K-Share
YouTube
Kiran M | LinkedIn
Dashboard | Hacker...
All Games (A-Z) » S...
Cricket: CSK Hamm...
Indian Railways
HOME
FIND TRAIN
RESERVATION
PROFILE
BOOKING HISTORY
Please Login First..
Username
kiran
Password
Login
You don't have register?
Signup
© 2018 Copyright PVT Ltd.
Desinged By : projectworlds


localhost/railres/ticket/reservation.php
Welcome, sachin
Logout
Indian Railways
HOME
FIND TRAIN
RESERVATION
PROFILE
BOOKING HISTORY
From
ADI
To
BCT
Quota
General
Date
28-09-2023
OK
Reset
Train No.
Train Name
Orig.
Des.
Arr.
Dep.
12932
BCT DOUBLEDECKE
ADI
BCT
13:00
13:05
1A 2A 3A SL
19011
GUJARAT EXPRESS
ADI
BCT
16:25
16:30
1A 2A 3A SL
© 2018 Copyright PVT Ltd.
Desinged By : projectworlds
localhost/railres/ticket/index.php


localhost/railres/ticket/ViewFullStatus.php?tnumber=12932&doj=2023-09-28&fromstn=ADI&tostn=BCT&DOB=2023-09-28
Welcome,sachin
Logout
Indian Railways
HOME
FIND TRAIN
RESERVATION
PROFILE
BOOKING HISTORY
Booked Ticket History
SNo.
Train Number
Date Of Journey
Name
Age
Sex
Status
DOB
Class
1
12932
2023-09-28
kiran
19
male
Waiting
2023-09-28
1A
Amount Paid :2500
Print
© 2018 Copyright PVT Ltd.
Desinged By : projectworlds

CHAPTER 8

REFERENCES

- ***https://nap.nationalacademies.org/read/10637/chapter/5***
- ***https://en.wikipedia.org/wiki/OpenWeatherMap***
- ***https://www.windy.com/-Wind-gusts-gust?gust,12.985,80.041,8***
- ***https://www.indianrail.gov.in/enquiry/StaticPages/StaticEnquiry.jsp?StaticPage=index.html***
- ***https://en.wikipedia.org/wiki/Indian_Railways***