**Chapter 1**

**INTRODUCTION**

**1.1 Preamble**

In today’s world, organizing events can be quite challenging. People often find it hard to locate venues such as hotels and halls that meet their specific needs in terms of availability, cost, and facilities. This issue is especially noticeable for residents of Belgaum, who lack a single platform to search for and book venues efficiently.

To solve this, our project proposes an online event management system. This platform aims to bring everything into one place, allowing users to easily find, compare, and book venues. By using modern technologies, this system will make event planning smoother, quicker, and less stressful, providing benefits to both users and venue managers.

**1.2 Problem Definition**

**Definition:**  
Belgaum residents face significant challenges in finding nearby hotels or event halls with accurate and reliable details about availability, pricing, and facilities. The lack of a centralized and efficient platform makes the event planning process time-consuming and cumbersome, often resulting in frustration for users and inefficiency for venue managers.

**Description:**  
This project aims to solve the problem by developing an online event management system that acts as a centralized platform for venue discovery and booking. The system provides users with real-time updates on venue availability, a streamlined interface for comparing options, and a shortest path finder to navigate to their selected locations easily. By bridging the gap between users and venue managers, the project simplifies event planning while improving operational efficiency and convenience for all parties involved.

**1.3 Objectives**

1. To create a centralized online platform for users to search and book event venues easily.
2. To implement a dynamic hotel availability calendar that provides real-time updates on venue bookings.
3. To automatically generate booking receipts with details such as event, venue, date for each booking.
4. To develop a shortest path finder module that guides users to their selected venues efficiently.
5. **To secure user details during data entry,** adhering to data privacy standards and ensuring safety.

**1.4 Motivation**

The motivation behind this project is to simplify the event planning process for Belgaum residents by creating a centralized online platform where users can easily search and book venues like hotels and event halls. Currently, people face challenges in finding accurate and up-to-date information about venue availability, pricing, and services, often relying on outdated sources or multiple platforms. This project aims to address these issues by providing real-time venue availability, an efficient route planner for navigation, and a secure platform to ensure a seamless and organized event planning experience for both users and venue managers.

**Chapter 2**

**SOFTWARE REQUIREMENT SPECIFICATIONS**

**2.1 Functional Requirements:**

1. Event and Hotel Listings with Social Media Integration  
   The system should provide a comprehensive list of events and hotels, including their services, team information, and a photo section linked to their social media accounts for easy access to additional details.
2. Event Details Navigation  
   When a user clicks on a specific event, the system should redirect them to a dedicated event page with detailed information about the event, venue, and available options.
3. Hotel Availability and Calendar Integration  
   The system should display a hotel availability calendar, allowing users to view and select free dates for their bookings.
4. Shortest Path Finder  
   The system should feature a route planning tool that calculates and displays the shortest path from the user's input location to the selected hotel.
5. Booking Interface for Events and Halls  
   The system should provide an interactive interface where users can book a specific hall for a particular event, complete with options for customization and confirmation.

**2.2 Non-Functional Requirements:**

1. Usability  
   The website's content and functionality should be designed to ensure that users can easily navigate and understand the platform without requiring extensive guidance.
2. Efficiency  
   The system should be optimized to process user inputs and load content quickly, ensuring a seamless experience even during high usage.
3. Scalability  
   The software should support the addition of more events, venue halls, and services without requiring major structural changes, allowing for business growth and expansion.
4. Safety and Security  
   The platform should authenticate users, protect user data, and adhere to data privacy standards, ensuring that no harm is caused to users in any form.
5. Reliability  
   The system should maintain stability and perform efficiently, even under heavy user loads, ensuring continuous availability and responsiveness.

**2.3 Hardware and software requirements**

**Hardware Requirements**

**1. Processor:** Intel i5/i7 or equivalent AMD processor (or higher)

**2. RAM:** Minimum 8GB (16GB recommended for smoother performance)

**3. Storage:**

**-** 500GB HDD or 256GB SSD (minimum)

- Additional storage for project files and dependencies (~10GB)

**4. Graphics:** Integrated GPU is sufficient, but a dedicated GPU (e.g., NVIDIA/AMD) is recommended for high-resolution designs or large datasets.

**5. Operating System:** Windows 10/11, macOS 10.15+ (Catalina or later), or a Linux distribution like Ubuntu 20.04+.

**Software Requirements**

**1. Operating System:**

- Windows 10/11, macOS, or Linux

**2. IDE/Code Editor:**

**-** [VS Code](https://code.visualstudio.com/) (recommended)

- Other options: Atom, WebStorm, or Sublime Text

**3. Node.js and npm:**

**-** Node.js (version 16+ recommended)

- npm (comes with Node.js) or yarn for package management

**4. Framework and Libraries:**

**-** React.js (latest version)

- React Router for navigation

- State management library (e.g., Redux or Context API)

- Tailwind CSS/Bootstrap/Material-UI for styling

**5. Backend Requirements:**

**-** Node.js with Express.js or other backend frameworks

- Database: MongoDB, MySQL, or PostgreSQL

**6. Development Tools:**

**-** Git for version control

- Browser: Chrome/Firefox with developer tools

- Postman or Insomnia for API testing

**7. Deployment Tools:**

**-** Hosting services: Vercel, Netlify, or AWS

- Docker (optional, for containerized deployment)

**2.4 Test plan and Test cases**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case Id** | **Description** | **Steps** | **Expected Result** |
| 01 | Valid Login with Correct Credentials | 1. Open the login page. 2. Enter valid email and password. 3. Submit the login form. | Successfully logged in and redirected to /home. |
| 02 | Valid Sign-Up Test | 1. Open the signup page. 2. Enter a username, email, password, and confirm the password. 3. Click the submit button. | Signup process should display a "Signup successful!" message or an appropriate error message if signup fails. |
| 03 | Valid Shortest Path Calculation for Venue | 1. Input the city "Pune". 2. Initiate the venue search. 3. Verify the nearest venue displayed. | The nearest venue displayed should be "Elegant Ballroom". |
| 04 | Available Dates Check on Calendar Popup | 1. Open the calendar popup by selecting "February". 2. Click the "Check Calendar" button. 3. Verify the displayed dates for February. 4. Click outside the popup to close it. | The calendar popup displays correct available dates for February and closes when clicked outside. |
| 05 | Booking Form and Receipt Functionality | 1. Open the booking form. 2. Fill in all required details. 3. Click the "Book Now" button manually. 4. Verify the booking receipt displayed after form submission. | The form accepts all details, and upon clicking "Book Now," displays a booking receipt with a booking ID. |

**Test case1**: Login Test

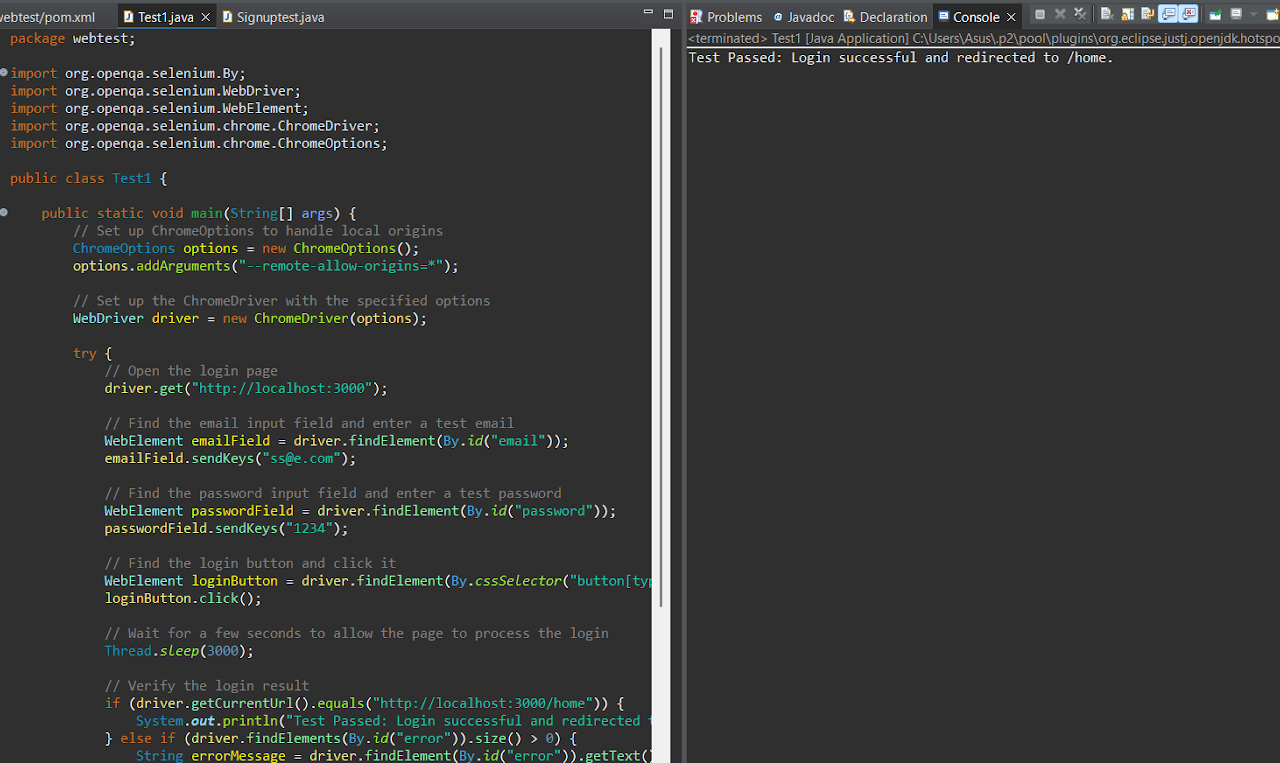


FIG.1

**Test Case**

The test case verifies the login functionality of a locally hosted website (http://localhost:3000). It inputs valid credentials (ss@e.com and 1234), submits the login form, and checks if the user is redirected to the home page (/home).

**Expected Output**

The user should be redirected to http://localhost:3000/home upon successful login with valid credentials. No error messages should appear.

**Actual Output**

The test passed as expected; the user was successfully redirected to /home. No error messages were encountered during the test.

**Importance**

This test ensures that the login functionality works correctly with valid credentials. It confirms that the authentication process and redirection mechanisms are implemented correctly, improving the reliability of the website.

**Test case2**: SignUp Test

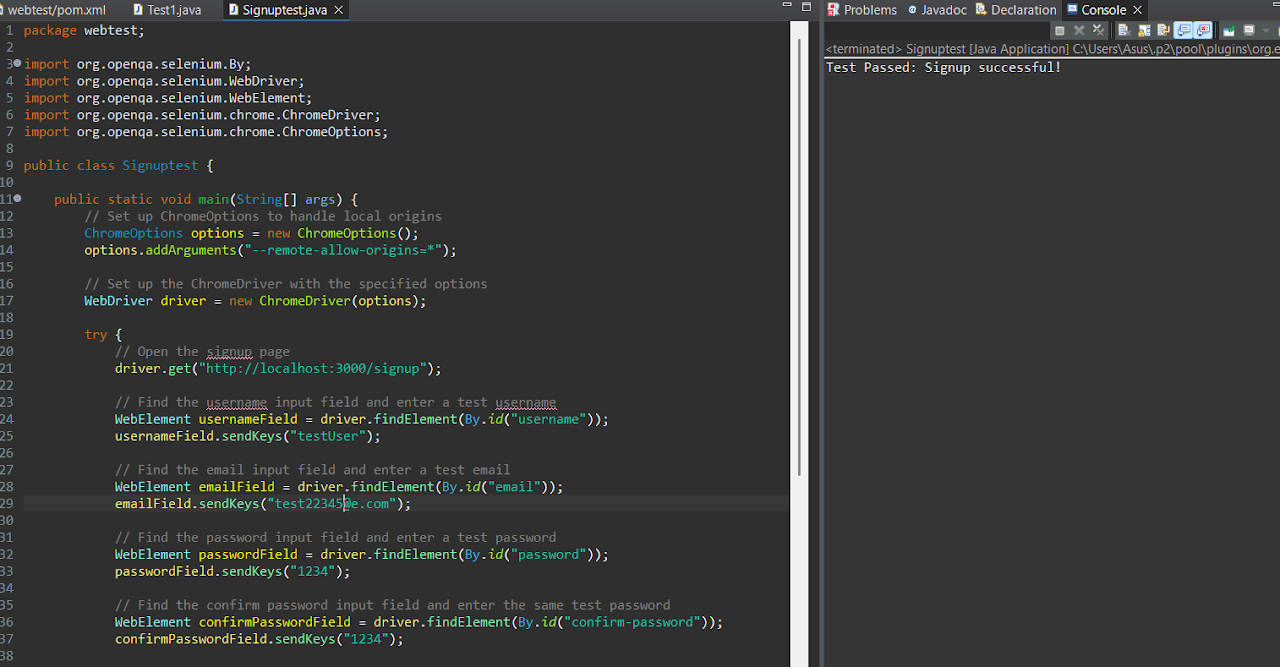


FIG.2

**Description**:  
This test case checks the signup functionality by entering a username, email, password, and confirming the password. After clicking the submit button, it verifies if the signup process is successful or if there are any error messages displayed.

**Expected Output**:  
The expected output is either a "Signup successful!" message or an error message if the signup fails. If the signup is successful, the test passes.

**Actual Output**:  
The actual output will either be a "Signup successful!" message or an error message, as seen in the test result. If the success message is found, the test prints "Test Passed," and if the error message is displayed, the test prints "Test Failed."

**Importance**:  
This test case is crucial for validating that users can successfully register by providing valid input. It ensures that the signup page properly handles both successful and failed signup attempts, providing feedback to users.

**Test Case3**: Shortest Path

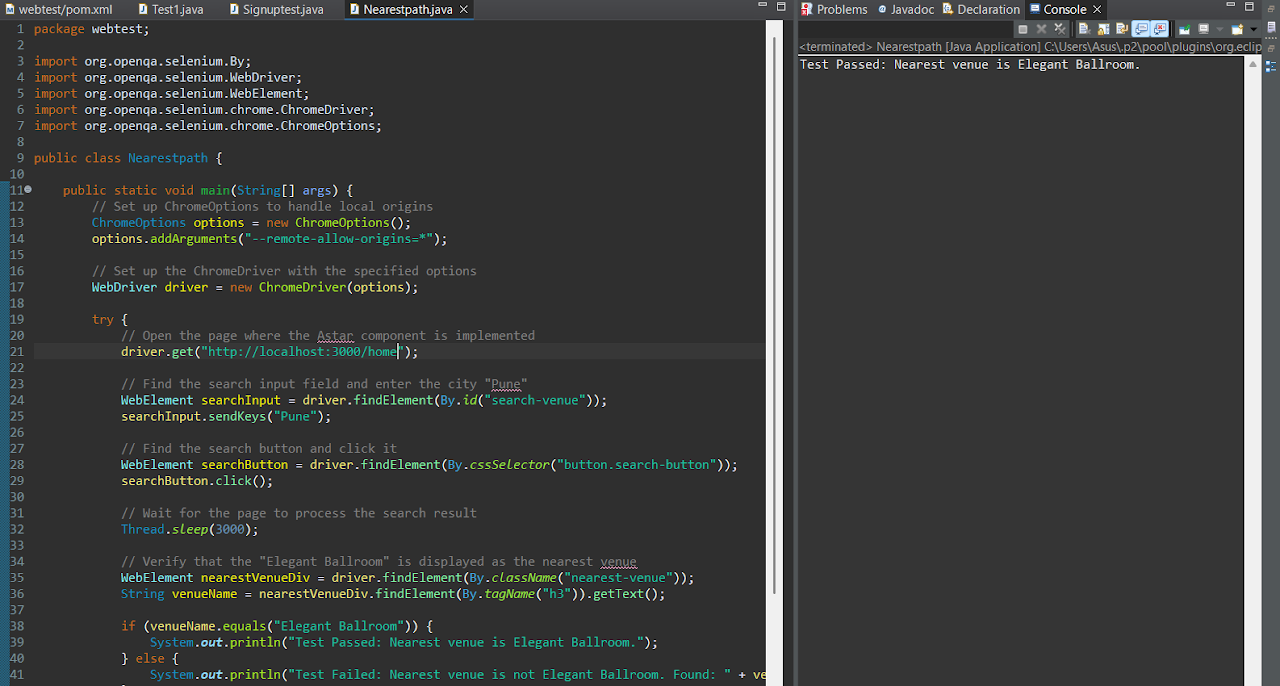


FIG.3

**Description of Test Case**:  
The test case checks whether the nearest venue displayed for the city "Pune" is "Elegant Ballroom" based on the A\* algorithm and Haversine distance calculations.

**Expected Output**:  
The nearest venue displayed should be "Elegant Ballroom."

**Actual Output**:  
The nearest venue displayed is "Elegant Ballroom," as expected.

**Importance of Test Case**:  
This test ensures that the venue search logic is functioning correctly by verifying that the correct venue ("Elegant Ballroom") is returned when the city "Pune" is entered, ensuring accurate venue recommendations based on the user's location.

**Test Case4**: Available Dates Check

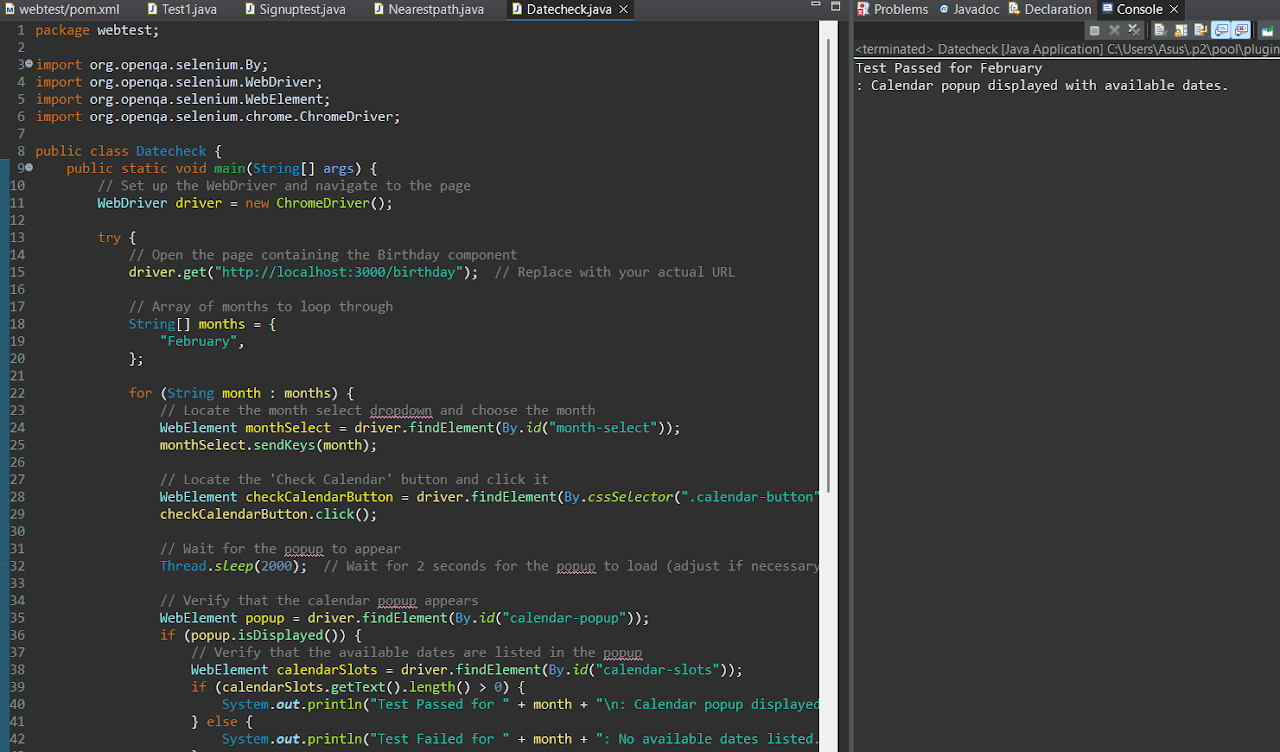


FIG.4

**Description:**

This test case checks the functionality of the calendar popup for the "February" selection. The goal is to ensure that the popup appears with the correct dates and closes when clicking outside. It verifies both the popup display and interaction behavior.

**Expected Result:**

The calendar popup should appear when the user selects "February" and clicks the "Check Calendar" button. The popup should display available dates for the selected month. After verifying the dates, the popup should close when the user clicks outside it.

**Actual Result:**

The calendar popup successfully appears after selecting "February" and clicking the "Check Calendar" button. The popup shows available dates for February. Clicking outside the popup closes it as expected.

**Importance:**

This test ensures that the calendar popup displays the correct available dates for the selected month. It validates that the user interface works as intended and provides an intuitive experience. Ensuring the popup closes when clicked outside is crucial for proper user interaction.

**Test Case5**: Booking Form and Receipt functionality

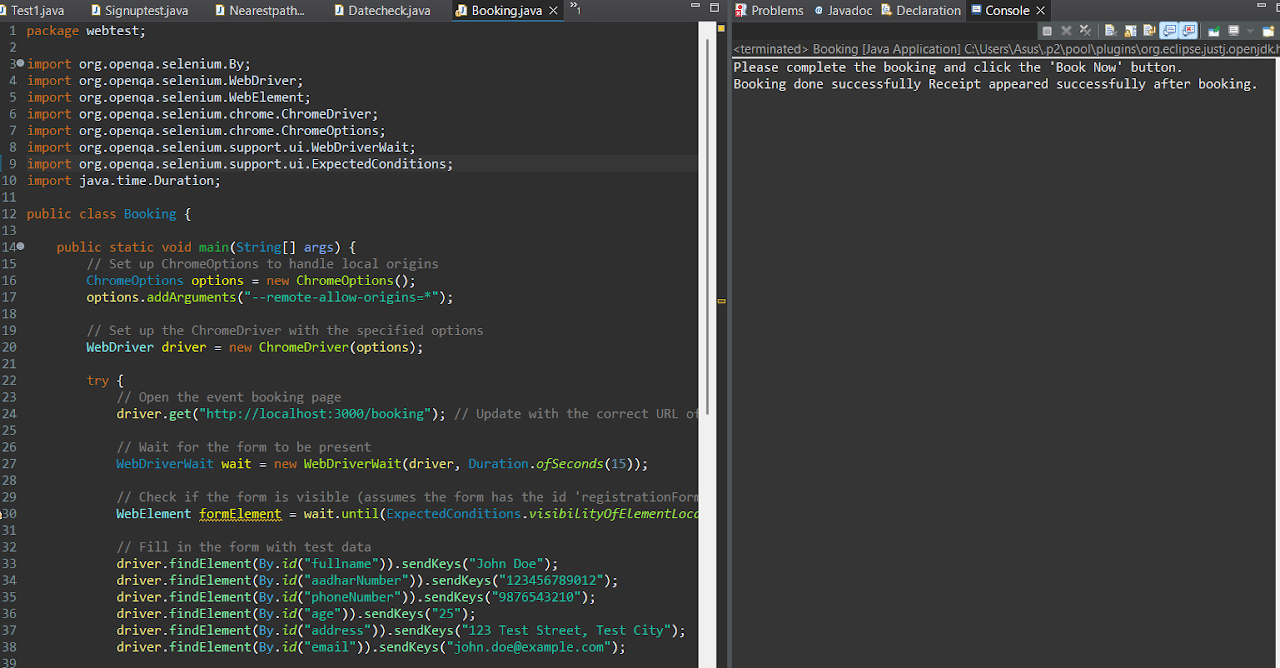


FIG.5

**Description:** This test case is designed to verify that all the details on the booking form can be filled in properly, and that after entering the necessary information, the form will allow the user to manually click the "Book Now" button. The system waits for the user to click the button, and once clicked, it checks if the receipt appears, confirming the successful booking. The test case ensures that the form behaves as expected and interacts properly with user actions.

**Expected Output:** The form should allow for all fields to be populated correctly, and once the user clicks the "Book Now" button, the system should display a booking receipt. The booking ID should appear on the receipt, indicating that the booking has been successfully processed.

**Actual Output:** The test case passed successfully, confirming that the user can input all required details into the form and that the system responds correctly after manually clicking the "Book Now" button, displaying the receipt with the booking ID.

**Importance:** This test case ensures that the booking form is functioning correctly by allowing users to enter all required details and manually submit the form, confirming that the application properly processes the booking. It is crucial for ensuring the user experience is smooth and that the booking flow works as intended.

**Chapter 3**

**SYSTEM DESIGN & IMPLEMENTATION**

#### ****3.1 MVC Architecture****

The **Model-View-Controller (MVC)** architecture is used to organize the components of the online event management system, ensuring separation of concerns and simplifying the management of the codebase. It divides the system into three interconnected components:

**Model:** Represents the data and business logic. It is responsible for directly managing the data, logic, and rules of the application. For this system, the model will include the hotel availability calendar, event booking details, user data, and more, interacting with the database (MongoDB).

**View:** Represents the user interface (UI). It is the part of the system that the user interacts with directly. The view will display event and hotel details, user information, and allow users to perform actions like booking or checking availability.

**Controller:** Acts as an intermediary between the Model and the View. It listens to user input (from the view), processes it (via the model), and updates the view accordingly. For example, if a user selects a date for booking, the controller will ensure the availability data is checked and updates the view.

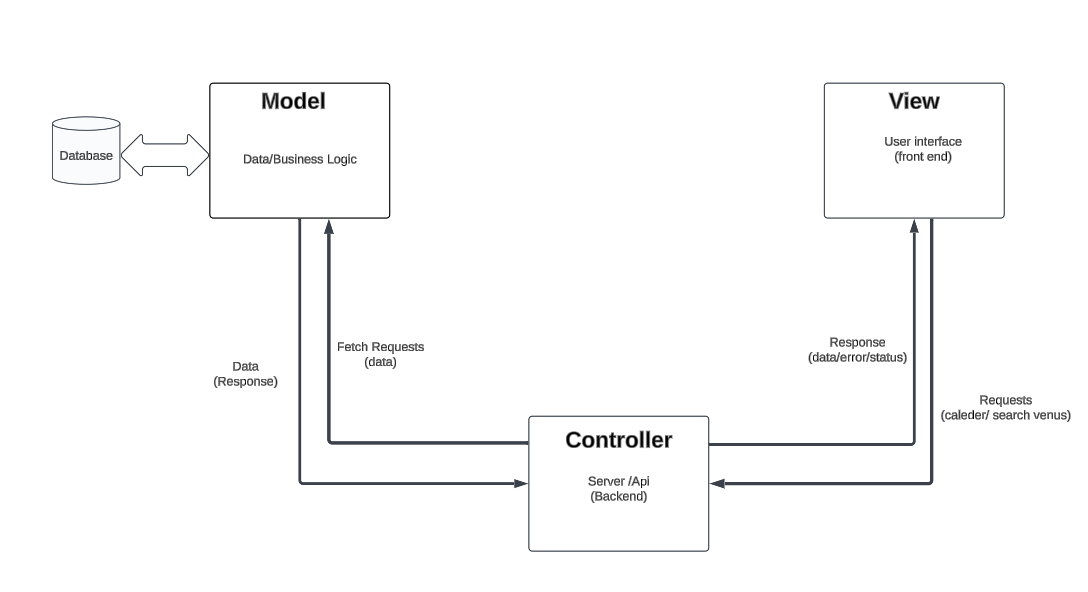


FIG.6

**3.2 MEAN/MERN framework**

#### ****MEAN Framework****

The **MEAN** stack comprises the following components:

1. MongoDB:

* A NoSQL database that stores data in a JSON-like format, offering flexibility and scalability.
* Supports dynamic schema design, enabling easy adaptation to evolving project requirements.

1. Express.js:

* A lightweight and fast web application framework for Node.js.
* Simplifies server-side logic and provides robust middleware for routing and handling HTTP requests.

1. Angular:

* A frontend framework maintained by Google, designed for building dynamic Single Page Applications (SPAs).
* Features two-way data binding and dependency injection for modular and maintainable development.

1. Node js:

* A runtime environment that enables JavaScript to run on the server.
* Built on an event-driven, non-blocking I/O model, ensuring efficient handling of concurrent requests.

#### ****MERN Framework****

The **MERN** stack includes:

1. **MongoDB:**

Acts as the database layer, providing document-based storage for application data.

1. **Express.js:**

Manages server-side logic and API routing for efficient backend operations.

1. **React:**

· A JavaScript library developed by Facebook for creating interactive user interfaces.

· Ensures high performance by using a virtual DOM and component-based architecture.

1. **Node.js:**

Facilitates server-side execution of JavaScript, making it a key part of this stack.

**3.3 Detailed Database Description(Mongo DB/ MySQL)**

#### ****Database Design****

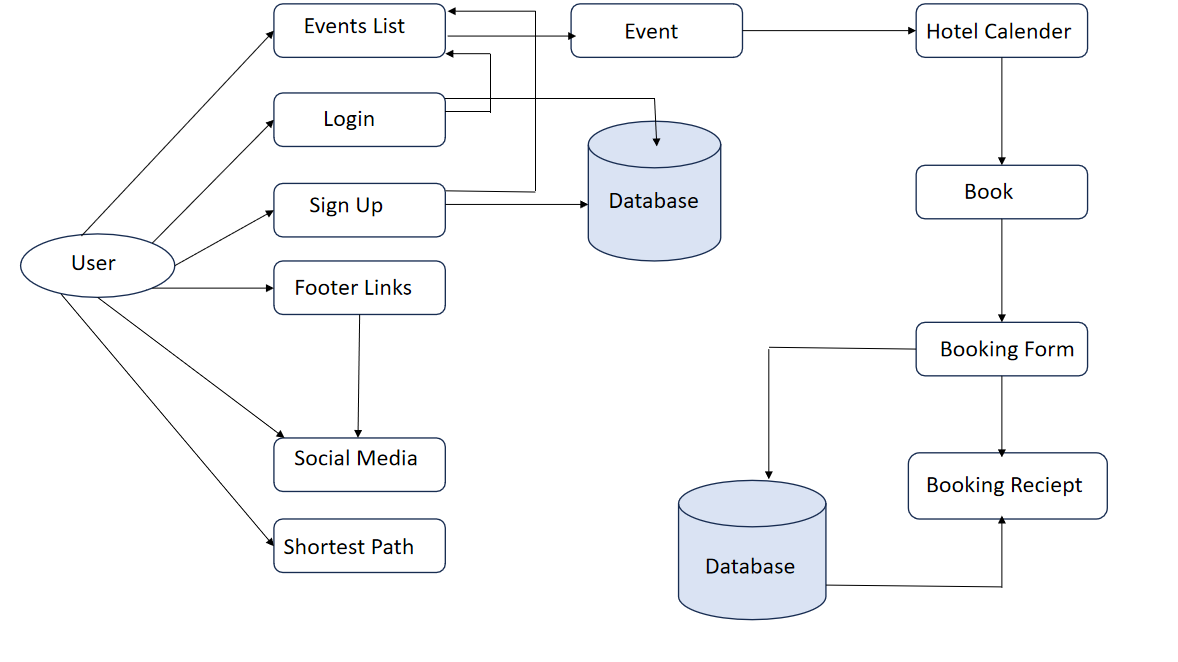


FIG.7

* **MongoDB for Data Storage:**

**All user data, venue details, event details, and bookings are stored in a well-structured MongoDB database.**

**The use of Mongoose provides a streamlined way to interact with the database through schemas and models.**

**3.4 Modules Description**

**1. HTML, CSS, and JavaScript:**

HTML and CSS are used to build a visually appealing and user-friendly interface.

JavaScript handles form validation, dynamic interactions, and the implementation of the A\* search algorithm for finding the nearest venue**.**

**2. React for Dynamic Frontend:**

React is used to build a modular and interactive frontend.

React's state management enables smooth transitions between components, such as venues, events, and booking forms.

Private Routing: React's private routing ensures that users cannot access restricted pages without logging in, enhancing security and flow.

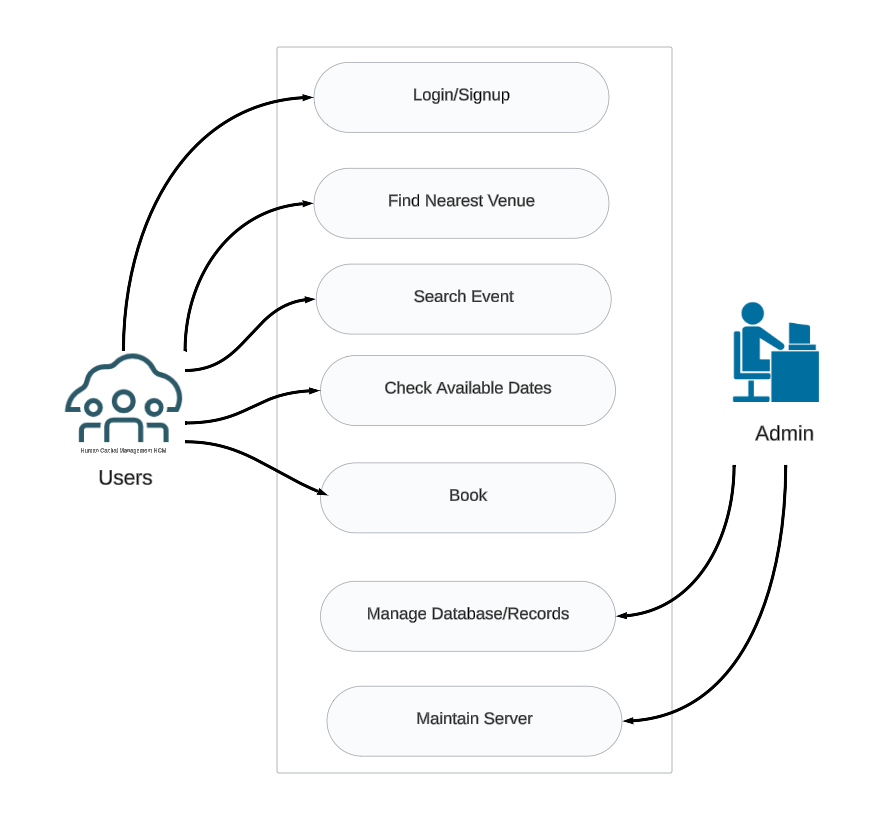


FIG.8

**Use Case Diagram**

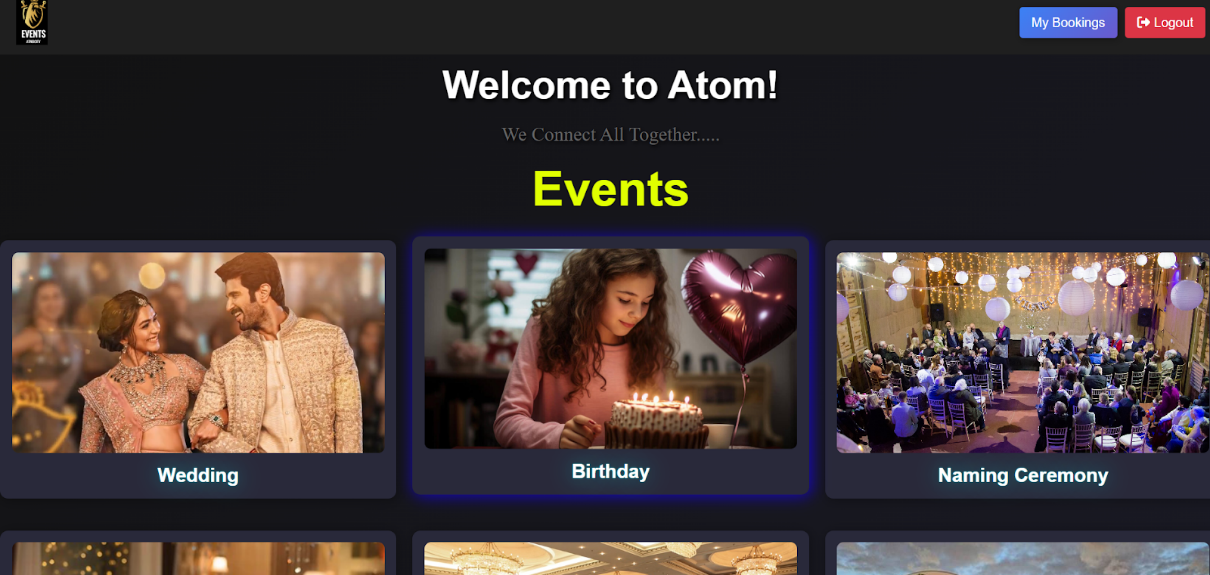
The database design includes multiple tables to store the necessary information, ensuring efficient data retrieval and management. The main tables are:

* **Users**: Stores user information like name, email, contact details, and profile settings.
* **Venues**: Stores information about available venues (hotel or event hall), including name, location, pricing, services, and availability.
* **Bookings**: Tracks user bookings, linking users to specific venues and events, including booking dates, payment details, and status.
* **Events**: Contains event details like event type, date, and description, which are linked to the venue bookings.
* **Navigation:**Stores map data and venue geolocations to enable shortest path calculations and facilitates retrieval of location coordinates for efficient route navigation and planning.

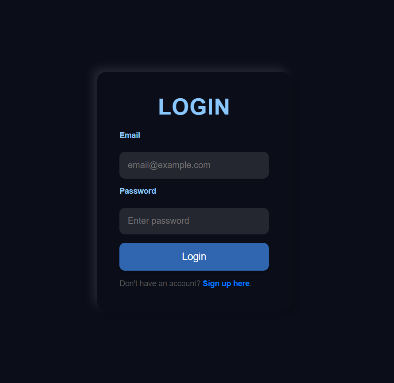
#### ****Front-End Design****

The front-end of the platform is designed to be user-friendly and responsive. The key components of the front-end are:

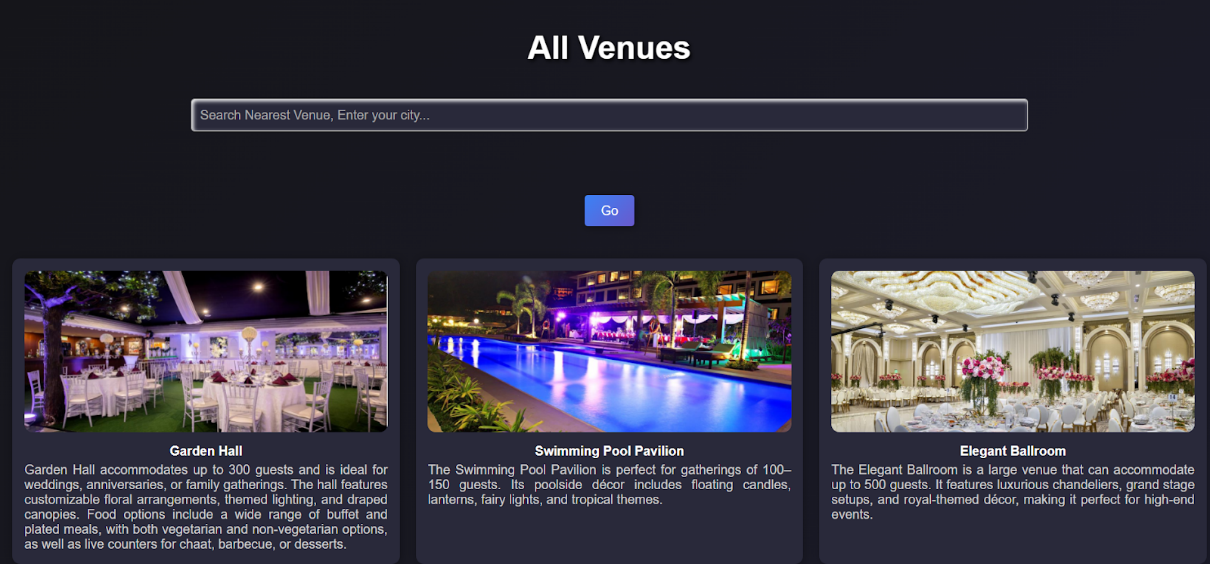
* **Home Page**: Displays featured events, available venues, and a search bar for users for each event type, and login, signup buttons.
* **Event Details Page**: Displays detailed information about a specific event, its venue, pricing, and available options for customization.
* **Booking Interface**: Provides an interactive form for users to select event dates, venue preferences, and additional services.



***FIG.9 Home Page***

***FIG.10 Login page FIG.11 Signup page***

** FIG.12 *Nearest Venue Using A\* Algorithm***

|  |  |
| --- | --- |
|  |  |

***FIG.13 Checking Available Dates***

|  |  |  |
| --- | --- | --- |
| ***FIG.14 Checking past bookings*** | ***FIG.15 Booking*** | ***FIG.16***  ***Receipt download*** |

**Chapter 4**

**4.1 RESULTS AND DISCUSSIONS**

**System Testing and Validation:**

1. **Functional Testing**: Each module (user registration, event search, booking interface, etc.) was tested to verify its functionality. This testing confirmed that users could search for events, view available venues, book halls, and make payments seamlessly.
2. **Integration Testing**: After integrating the individual modules, the system was tested as a whole to ensure that all components work together without issues. This included checking the interaction between the front-end and back-end, as well as the seamless operation of the booking process and payment system.
3. **Usability Testing**: The platform was tested by a small group of users to gauge ease of use. Feedback was collected regarding the user interface, navigation, and overall experience. Based on the feedback, minor adjustments were made to enhance user interaction and the clarity of certain features.

**Key features and outcomes:**

1. **Event and Venue Listings**: Users can search and filter events and venues based on criteria such as location, availability, and services. The listings are dynamically updated, ensuring that users always have access to the latest information.

**Outcome**: This feature was tested for accuracy and speed, ensuring that the results are relevant to the user’s search. The user interface allows easy browsing of event venues and event types, improving the event planning experience.

1. **Hotel Availability Calendar**: The system includes an integrated calendar to show the availability of event halls or hotels. Admins can update the calendar dynamically, while users can check the availability of venues before making a booking.

**Outcome**: The calendar functionality was tested by simulating different booking scenarios, and it successfully updated in real-time without any delays, providing users with accurate venue availability.

1. **Shortest Path Finder**: The route planning tool calculates the shortest path from the user's input location to the selected venue using Dijkstra’s algorithm.

**Outcome**: This feature was tested with different starting points and venues, and the route calculations were correct and provided users with the most efficient paths.

1. **Booking Interface**: The booking module allows users to choose a specific venue, date, and customize their booking by selecting services and additional options. After confirming the booking, users receive a detailed receipt.

**Outcome**: Testing confirmed that the booking process was smooth, with users being able to easily customize their bookings. The booking receipt was generated correctly, including all relevant details like venue, event type.

**Discussions:**

The event management system developed for Belgaum residents has successfully addressed the common challenges in event planning by providing a centralized platform for searching, comparing, and booking venues. Users now have easy access to up-to-date information about available venues, their pricing, facilities, and services, along with real-time updates on availability through the integrated calendar. This significantly simplifies the event planning process, saving users time and effort. Furthermore, the booking interface allows for customization, making it flexible for users to select and book venues according to their specific event requirements. The integration of a route planning tool, utilizing algorithms like Dijkstra’s, also ensures that users can easily find the shortest path to their selected venue, adding convenience for attendees unfamiliar with the area.

For venue managers, the system offers a streamlined way to manage bookings and availability. The dynamic hotel calendar allows venue admins to update their availability in real time, preventing double bookings and improving operational efficiency. The system's scalability also ensures that as more venues and events are added, the platform can handle increased user load without compromising performance. Although there were challenges during the integration of the shortest path feature, the system successfully provides a user-friendly solution to event planning, and with future improvements.

**4.2 Testing Report**

**Five Test cases with results:**

Login, SignUp, Shortest path, Available Dates Check, Booking Form and Receipt functionality.

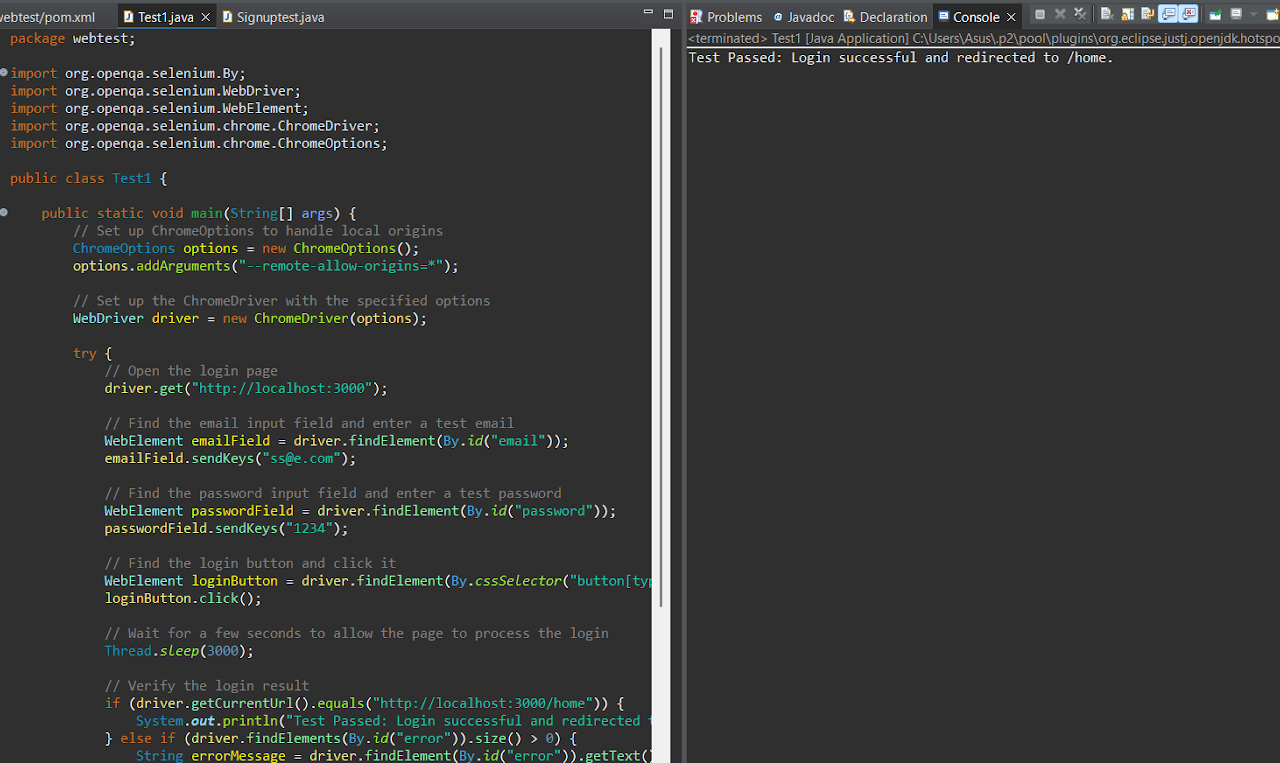


FIG.17

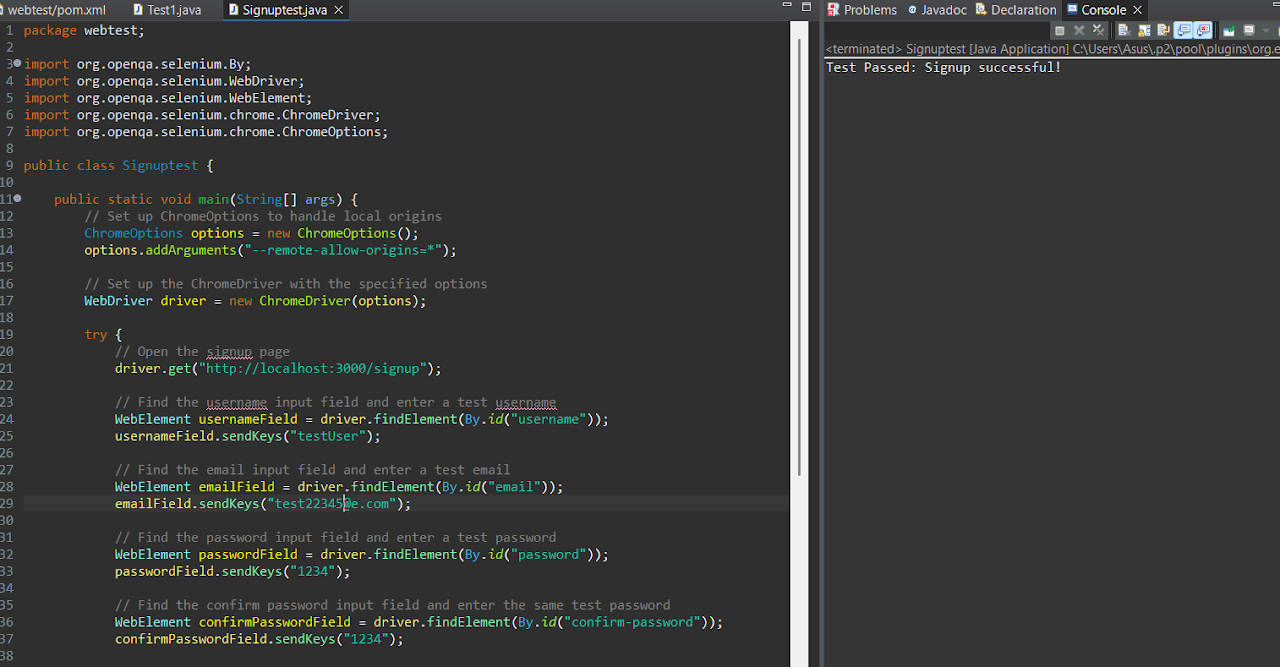


FIG.18

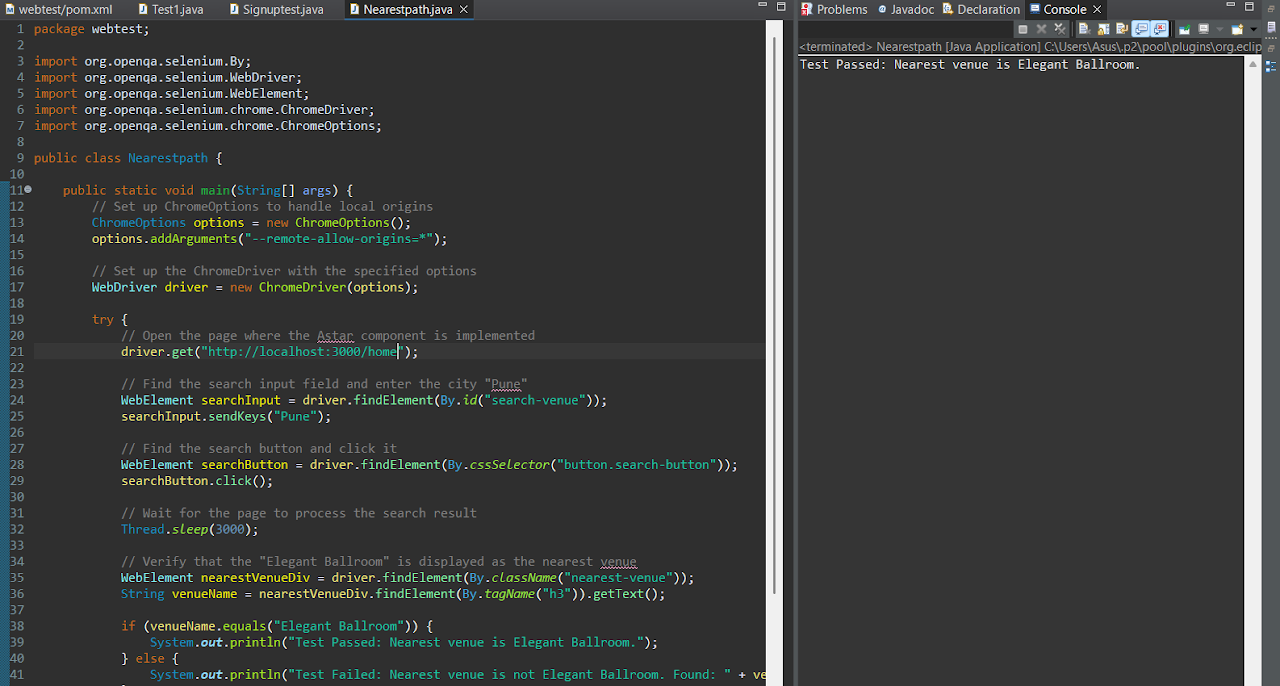


FIG. 19

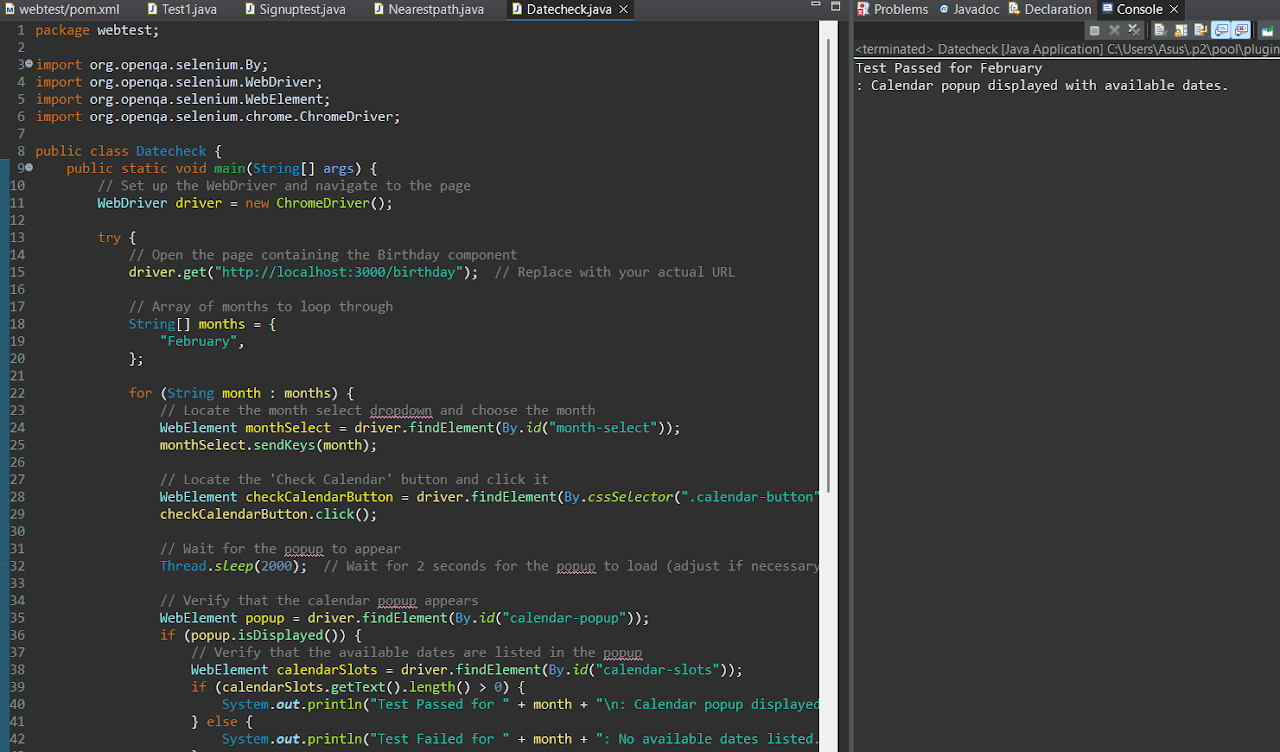


FIG. 20

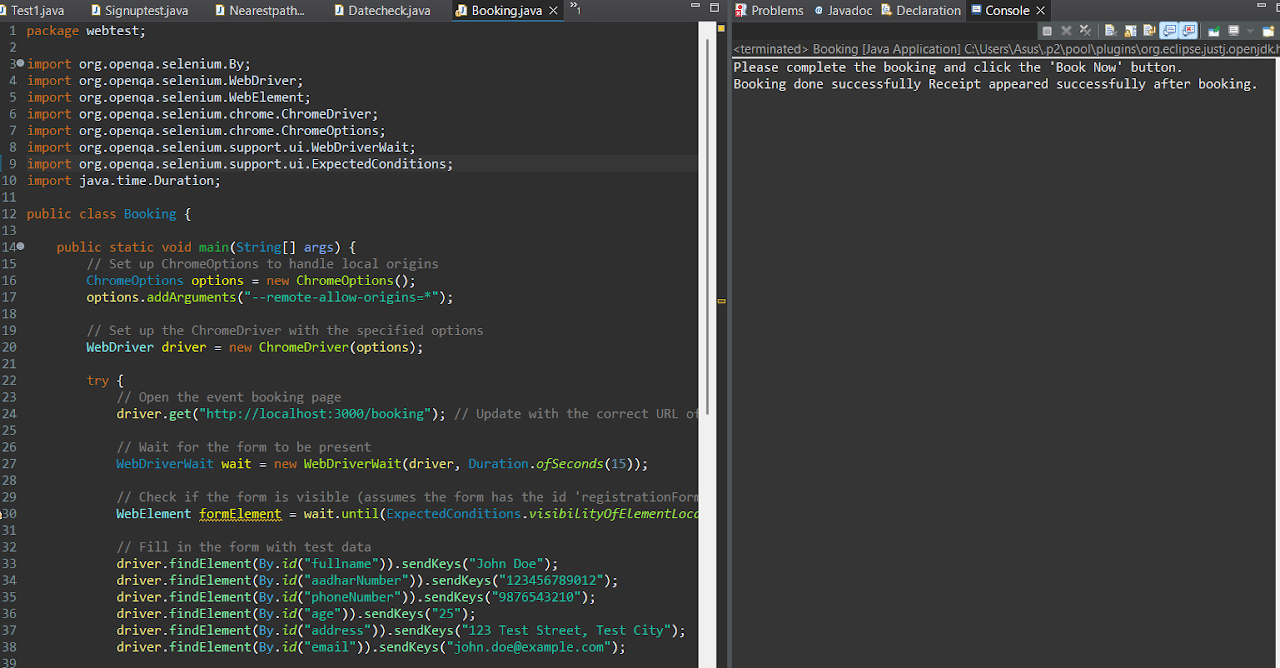


FIG. 21

**4.3 Testing Tools**

**1. Selenium**

Selenium is a popular open-source framework used for automating web browsers. It supports multiple programming languages like Java, Python, and C#, making it a versatile choice for web application testing.

Key Features:

Enables functional, regression, and UI testing of web applications.

Provides tools like Selenium WebDriver to interact with web elements programmatically.

Supports cross-browser testing (e.g., Chrome, Firefox, Edge).

Integrates with various testing frameworks like TestNG and JUnit.

Why you used it: To write and execute automated test scripts to validate the functionality of your website.

1. **Java**

Java is a high-level, object-oriented programming language widely used in automation testing for writing Selenium test scripts.

Key Features:

Robust and platform-independent.

Provides extensive libraries and frameworks (e.g., TestNG) for organizing test cases.

Excellent integration with Selenium for writing modular and reusable code.

Why you used it: To write the automation scripts in a structured, maintainable, and efficient manner.

**3. Chrome WebDriver**

Chrome WebDriver is a tool that bridges the Selenium framework and the Google Chrome browser. It allows Selenium to interact directly with the Chrome browser.

Key Features:

Executes Selenium commands to simulate user actions in the Chrome browser.

Supports the latest versions of Chrome for compatibility.

Provides options for headless testing (running tests without opening the browser GUI).

Why you used it: To execute your Selenium automation scripts specifically in the Chrome browser.

**4. Eclipse**

Eclipse is an Integrated Development Environment (IDE) for writing, debugging, and managing Java projects.

Key Features:

Provides tools for writing, compiling, and running Java programs.

Supports plugins for Selenium, Maven, and other tools to enhance functionality.

Offers debugging features and easy project organization.

Why you used it: To develop and manage your Selenium test scripts efficiently.

**Chapter 5**

**CONCLUSION & FUTURE SCOPE**

**Conclusion:**

In conclusion, the online event management system developed for Belgaum residents effectively simplifies the event planning process. By providing a centralized platform for searching, comparing, and booking venues, the system enhances the efficiency for both event organizers and venue managers. Key features such as real-time venue availability, a booking interface, and a shortest path finder tool ensure a seamless, user-friendly experience. Overall, the system successfully meets the objectives of the project by improving the convenience and effectiveness of event planning for users while streamlining venue management.

**Future Scope:**

The event management system has significant potential for expansion and enhancement in the future. One of the primary areas for improvement is the addition of more advanced features, such as AI-based event recommendations, where the system could suggest venues based on user preferences and past bookings. The system could also be expanded to include other cities or regions, enabling it to reach a wider audience. Additionally, further optimization of the shortest path feature, such as incorporating real-time traffic data for more accurate route calculations, would improve user experience. As the platform grows, more venues, event types, and services can be added, ensuring the system remains scalable and adaptable to future needs.

**Chapter 6**

INFOSYS SPRINGBOARD CERTIFICATION COURSE ON DEVOPS

The Infosys Springboard Certification Course on DevOps provides an in-depth introduction to the principles and practices that enable efficient collaboration between development and operations teams, aiming to improve the software development lifecycle. The course covers key topics such as version control, continuous integration (CI), continuous delivery (CD), and infrastructure automation, with hands-on experience using tools like Git, Jenkins, Terraform, Ansible, Docker, and Kubernetes. Additionally, it emphasizes the importance of monitoring, logging, and the use of communication tools like Slack and JIRA, all of which play a crucial role in streamlining workflows and enhancing team collaboration.

The skills acquired in this course are highly valuable for implementing DevOps practices in real-world projects, enabling faster and more reliable software delivery. By understanding the DevOps lifecycle and learning to use essential tools, professionals can automate processes, manage infrastructure efficiently, and ensure the scalability and quality of applications. These capabilities are essential for improving development cycles, reducing errors, and fostering better collaboration between teams, making DevOps a key practice in modern software development.

