



JEPPIAAR

ENGINEERING COLLEGE

**JEPPIAAR EDUCATIONAL TRUST
JEPPIAAR NAGAR, OLD MAHABALIPURAM ROAD,
SEMMENCHERI, CHENNAI- 600 119.**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROJECT REPORT

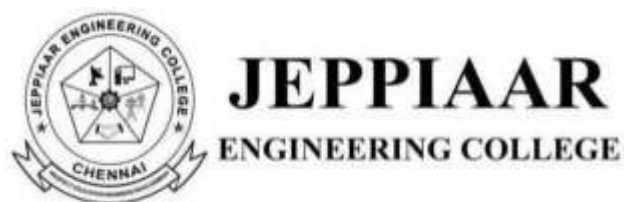
YEAR: IV

SEMESTER: VII

SUBJECT CODE: NM1117

SUBJECT NAME: FULL STACK WITH JAVA

EXAMINATION: NOVEMBER/ DECEMBER 2025



JEPPIAAR EDUCATIONAL TRUST
JEPPIAAR NAGAR, OLD MAHABALIPURAM ROAD,
SEMMENCHERI, CHENNAI- 600 119.

Bonafide Certificate

This is a certified Bonafide record work of _____, Register
Number _____ submitted for the Anna University Practical Examination held on
_____ in **NM1117- Full Stack with Java** Laboratory during the academic year
2025-2026.

Signature of the HOD

*Signature of the
Faculty-Incharge*

Submitted for the University Practical on _____

Internal Examiner

External Examiner

Infosys



NM1117 – FULL STACK WITH JAVA

Project Created by:

- 1.Sherley.G
- 2.Shama Arzu.H
- 3.Taruna M K

Project Created on: 29-10-2025

Project Title: SUBSCRIPTION

MANAGEMENT SYSTEM

Table of Contents:

Project Objective.....	4
Scope.....	4
Methodology.....	5
Artifacts used.....	6
Technical Coverage.....	8
Results.....	10
Challenges and Resolutions.....	11
Conclusion.....	13
.....	13 Certification
.....	13

Project Objective:

The *Subscription Management System* helps businesses manage their subscription services easily. It allows users to buy, renew, or cancel their subscriptions, while admins can monitor users, plans, and payments in one place.

It includes:

- User sign-up, login, and authentication
- Creating and managing different subscription plans
- Online payment using Razorpay or PayPal
- Automatic renewal and expiry reminders
- Admin dashboard with reports and analytics

This project mainly focuses on automating the process, improving accuracy, and saving time for both users and administrators. It doesn't include offline payments or mobile app integration.

Scope:

The scope of this project includes the following key features:

- **User Management:** Secure user registration, authentication, and profile management for both subscribers and administrators.
- **Plan Management:** Creation, modification, and deletion of various subscription plans with flexible durations and pricing options.
- **Payment Integration:** Integration with secure online payment gateways such as Razorpay or PayPal to handle real-time transactions.
- **Subscription Tracking:** Monitoring of active, expired, and renewed subscriptions with automated reminders and alerts through email or SMS.
- **Admin Dashboard:** Comprehensive dashboard for administrators to track user activities, revenue reports, and overall system analytics.
- **Reports and Analytics:** Generation of data-driven reports for performance monitoring and business decision-making.

Methodology

The *Subscription Management System* was developed using a structured and systematic approach following the **Software Development Life Cycle (SDLC)** model. The process includes several key stages that ensure efficiency, quality, and accuracy in the final outcome.

1. Requirement Analysis

- Collected system requirements from both admin and user perspectives.
- Identified main functionalities like subscription creation, renewal, and payment tracking.
- Defined performance, security, and usability requirements.
- Prepared the *Software Requirement Specification (SRS)* document.
- Finalized scope, features, and technology stack.

2. System Design

- Created system architecture and database schema using **MySQL**.
- Designed **ER diagrams**, **DFDs**, and **architecture diagrams** for data flow visualization.
- Prepared UI wireframes for user and admin dashboards.
- Defined navigation flow and backend service connections.
- Focused on scalability, modularity, and secure design.

3. Development

- Implemented backend logic using **Java Spring Boot**.
- Developed REST APIs for subscription, user, and payment modules.
- Built frontend using **HTML**, **CSS**, **JavaScript**, and **Thymeleaf**.
- Integrated **Razorpay/PayPal** for payment processing.
- Added automated email and SMS notifications for renewals and expiries.

4. Testing

- Conducted **unit testing** for all core modules.
- Performed **integration testing** to validate complete workflow.
- Carried out **user acceptance testing (UAT)** with sample data.
- Tested security, input validation, and error handling.
- Documented bugs and ensured resolution before deployment.

5. Deployment

- Deployed application on **AWS/Heroku** for real-time access.
- Configured database connectivity and environment variables.
- Used **GitHub** for version control and project management.
- Implemented **SSL encryption** for secure data transmission.
- Verified performance under live conditions.

6. Maintenance and Future Enhancements

- Regularly monitored application logs and server performance.
- Collected user feedback for UI and functionality improvements.
- Released minor updates and patch fixes as required.
- Planned integration of advanced analytics for admin insights.
- Proposed mobile app version for broader accessibility.

7. Security Implementation

- Implemented **JWT (JSON Web Token)** for secure login and user authentication.
- Applied **role-based access control** to separate admin and user privileges.
- Enabled **HTTPS** and **SSL certificates** for encrypted communication.
- Used **Spring Security** features to protect APIs and sensitive data.
- Sanitized user inputs and validated requests to prevent SQL injection and cross-site scripting (XSS).
- Regular security audits and log monitoring were performed to ensure data protection.

8. Documentation and User Training

- Created detailed **API documentation** for developers and maintainers.
- Prepared **user manuals** explaining registration, plan management, and renewal processes.
- Developed **technical documentation** for deployment and database configuration.
- Conducted short user training sessions for admin users.
- Maintained version history and change logs for future reference.
- Ensured all documentation is regularly updated as new features are added.

References:

1. www.google.com
2. www.chatgpt.com

Artifacts used

During the development of the *Subscription Management System*, several important artifacts were created to support the design, implementation, and testing process. These documents ensured clarity, accuracy, and smooth progress throughout the project.

1. Software Requirement Specification (SRS)

Defined all functional and non-functional requirements, user roles, and system features used during design and development.

2. System Design Documents

Included architecture diagrams, DFDs, and ER diagrams showing the relationship between modules, database, and user operations.

3. Database Schema

Designed using **MySQL** to store user data, subscription plans, payments, and transactions securely and efficiently.

4. UI Wireframes

Created simple layouts for the admin dashboard and user pages to visualize structure and navigation flow before coding.

5. Source Code Repository

All project files and configurations were maintained in **GitHub** for version control and easy collaboration.

6. API Documentation

Detailed REST API references describing endpoints, inputs, and outputs for payment, subscription, and user modules.

7. Testing Reports

Included unit and integration test cases with results to ensure the stability and correctness of each module.

8. Deployment Scripts

Scripts for deploying the system on **AWS/Heroku** and connecting application with

database.

9. User Manuals

Simple user and admin guides explaining system usage, subscription handling, and payment operations.

10. Maintenance Log

Recorded updates, bug fixes, and version history for future reference and improvements

11. Project Plan Document

Outlined timelines, milestones, and assigned responsibilities throughout the project lifecycle.

12. Meeting and Review Notes

Included summaries of discussions, design changes, and decisions taken during team meetings.

13. Training Materials

Contained brief tutorials and instructions used for demonstrating system features and usage to users.

Technical coverage:

The *Subscription Management System* integrates several modern technologies to ensure high performance, scalability, and security. The project combines backend, frontend, and database components with API-based communication for smooth functionality.

1. Backend Technology

- Developed using **Java Spring Boot** framework for faster and modular application development.
- Supports RESTful API creation and dependency management through **Spring MVC** and **Maven**.
- Handles business logic, authentication, and communication between database and frontend.

2. Frontend Technology

- Designed using **HTML**, **CSS**, and **JavaScript** for user-friendly interfaces.
- **Thymeleaf** templates used for connecting UI with backend data dynamically.
- Provides responsive and interactive dashboards for both users and admins.

3. Database Management

- Implemented using **MySQL** for secure and efficient data storage.
- Stores information related to users, plans, subscriptions, and transactions.
- Includes indexing and foreign key relationships to maintain data integrity.

4. Payment Integration

- Integrated **Razorpay** or **PayPal API** for online payments.
- Supports real-time transaction tracking and automated renewal processing.
- Uses secure **HTTPS** protocols for safe payment operations.

5. Security Implementation

- Applied **JWT (JSON Web Token)** authentication for secure user login.
- Used **Spring Security** for role-based access (admin and subscriber).
- Implemented **SSL encryption** to protect user and transaction data.

6. Tools and IDEs

- **IntelliJ IDEA** and **VS Code** used for coding and debugging.
- **GitHub** for version control and project collaboration.
- **Postman** used for API testing and verification.

7. Deployment and Hosting Application deployed on **AWS** or

- **Heroku** cloud environment. Ensures scalability, 24/7 availability, and easy remote access. Supports database connection and API hosting securely.

8. Testing Tools

- Used **JUnit** for unit testing and validation.
- Tested API responses, input handling, and database queries.
- Performed manual testing for user interface and payment modules.

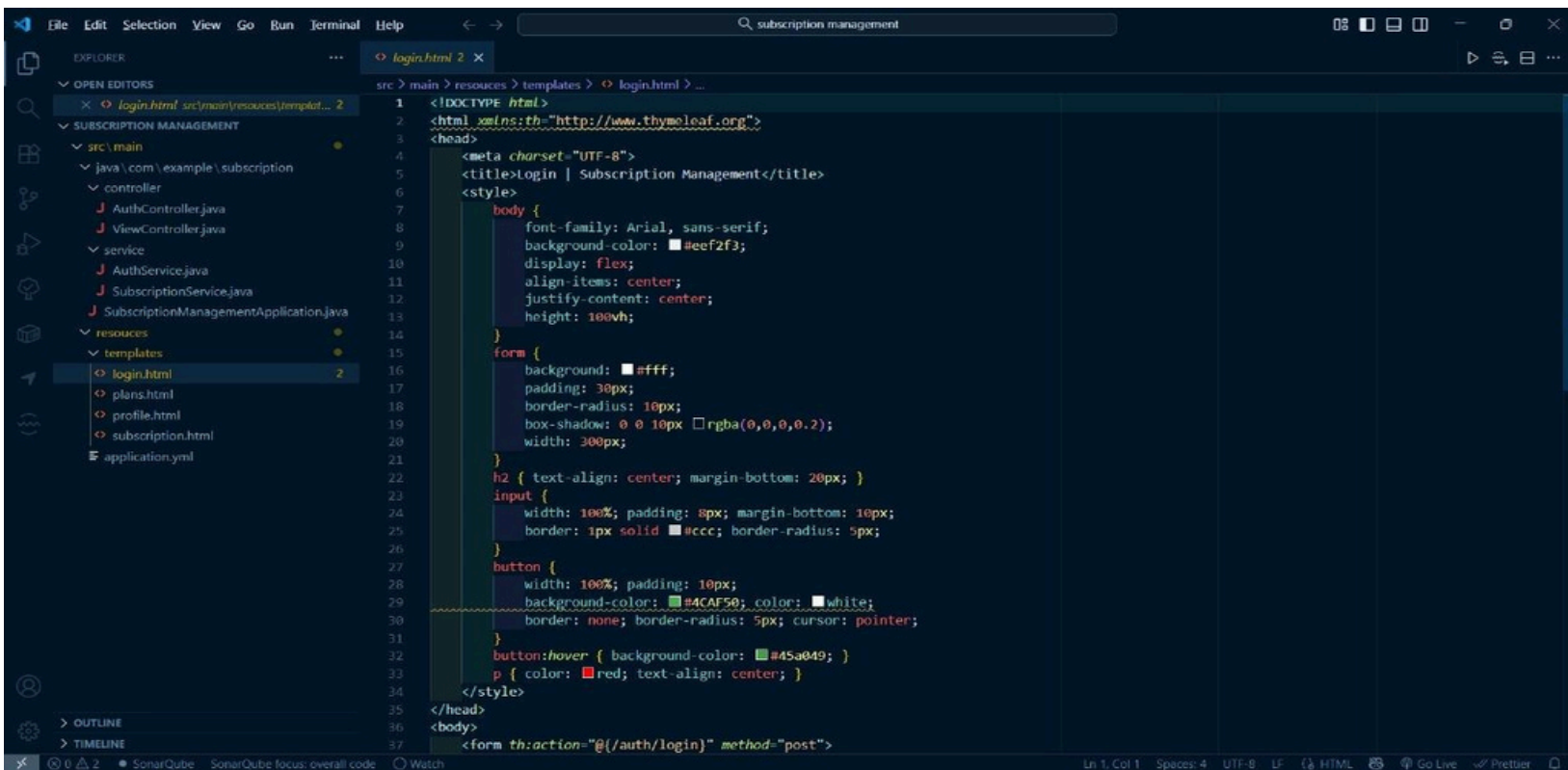
9. Logging and Monitoring

- Integrated logging using **Spring Boot Actuator** and server logs.
- Tracks API requests, responses, and user activities.
- Helps identify errors and performance issues during maintenance.

10. Overall System Features

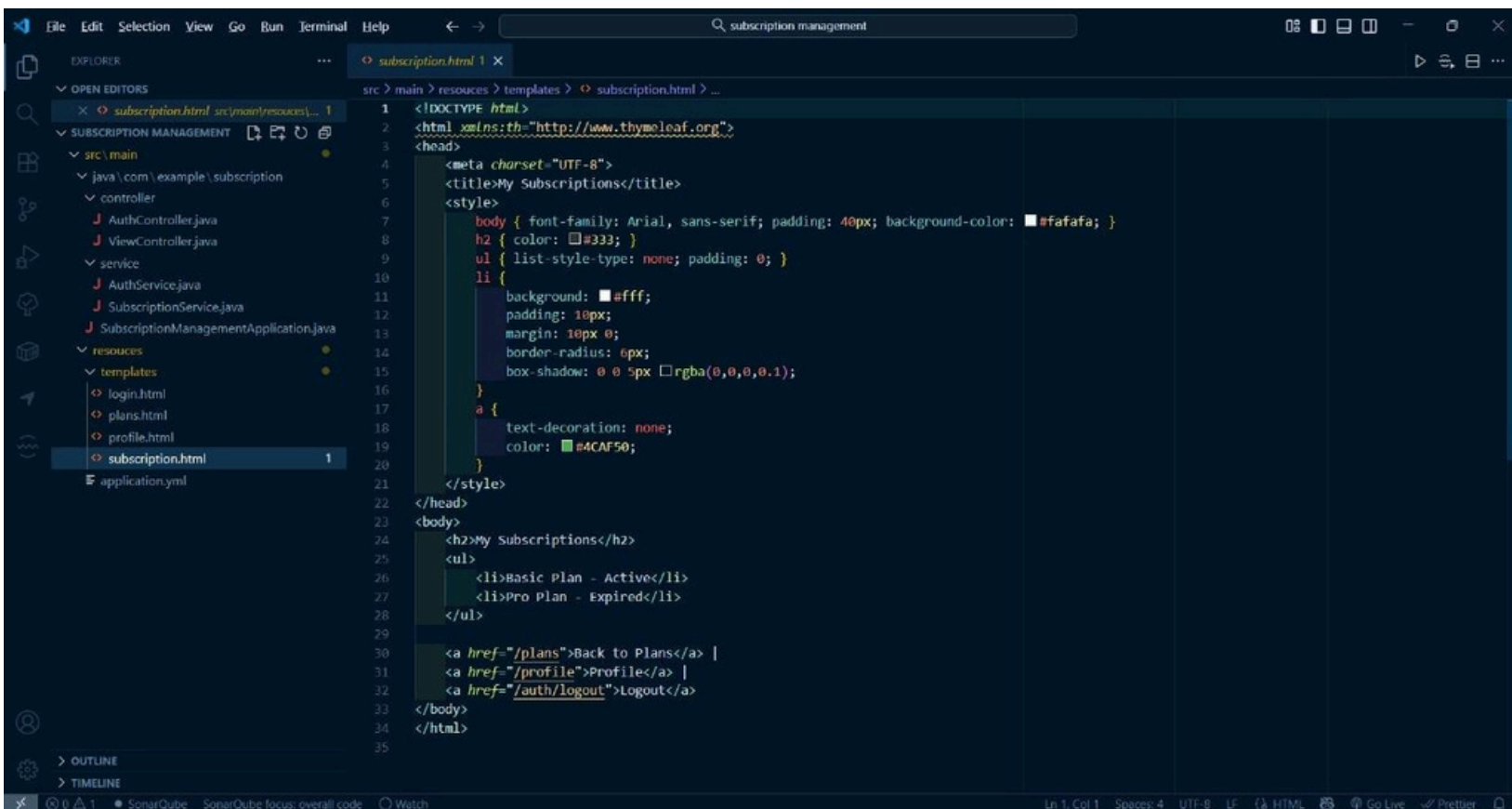
- Cloud-based, secure, and scalable architecture.
- Modular structure allows easy integration of new features.
- Real-time analytics and performance monitoring for admins.

CODING:



The screenshot shows the VS Code editor with the 'login.html' file open. The Explorer panel on the left shows the project structure: 'src/main/resources/templates' containing 'login.html', 'plans.html', 'profile.html', and 'subscription.html'. The main editor area displays the HTML code for 'login.html'.

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymoleaf.org">
3 <head>
4   <meta charset="UTF-8">
5   <title>login | Subscription Management</title>
6   <style>
7     body {
8       font-family: Arial, sans-serif;
9       background-color: #eef2f3;
10      display: flex;
11      align-items: center;
12      justify-content: center;
13      height: 100vh;
14    }
15    form {
16      background: #fff;
17      padding: 30px;
18      border-radius: 10px;
19      box-shadow: 0 0 10px rgba(0,0,0,0.2);
20      width: 300px;
21    }
22    h2 { text-align: center; margin-bottom: 20px; }
23    input {
24      width: 100%; padding: 8px; margin-bottom: 10px;
25      border: 1px solid #ccc; border-radius: 5px;
26    }
27    button {
28      width: 100%; padding: 10px;
29      background-color: #4CAF50; color: white;
30      border: none; border-radius: 5px; cursor: pointer;
31    }
32    button:hover { background-color: #45a049; }
33    p { color: red; text-align: center; }
34  </style>
35 </head>
36 <body>
37   <form th:action="@{/auth/login}" method="post">
```



The screenshot shows the VS Code editor with the 'subscription.html' file open. The Explorer panel on the left shows the project structure: 'src/main/resources/templates' containing 'login.html', 'plans.html', 'profile.html', and 'subscription.html'. The main editor area displays the HTML code for 'subscription.html'.

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymoleaf.org">
3 <head>
4   <meta charset="UTF-8">
5   <title>My Subscriptions</title>
6   <style>
7     body { font-family: Arial, sans-serif; padding: 40px; background-color: #fafafa; }
8     h2 { color: #333; }
9     ul { list-style-type: none; padding: 0; }
10    li {
11      background: #fff;
12      padding: 10px;
13      margin: 10px 0;
14      border-radius: 6px;
15      box-shadow: 0 0 5px rgba(0,0,0,0.1);
16    }
17    a {
18      text-decoration: none;
19      color: #4CAF50;
20    }
21  </style>
22 </head>
23 <body>
24   <h2>My Subscriptions</h2>
25   <ul>
26     <li>Basic Plan - Active</li>
27     <li>Pro Plan - Expired</li>
28   </ul>
29   <a href="/plans">Back to Plans</a> |
30   <a href="/profile">Profile</a> |
31   <a href="/auth/logout">Logout</a>
32 </body>
33 </html>
```

Result:

The *Subscription Management System* was successfully developed to automate the complete process of managing user subscriptions, renewals, and payments. The system simplifies operations for both administrators and users by providing a secure and user-friendly platform. Online payments through **Razorpay** and **PayPal** were integrated for real-time transaction handling, while automated email and SMS alerts ensured timely renewals and notifications.

The application performed efficiently during testing, showing fast response time and accurate data processing. The use of **Spring Boot**, **MySQL**, and **Thymeleaf** provided strong backend performance and smooth frontend interaction. The admin dashboard helped monitor user activities, active plans, and revenue reports effectively. Security measures such as **JWT authentication** and **Spring Security** ensured safe access and data protection.

Overall, the project achieved its objectives by delivering a reliable, secure, and easy-to-use subscription management platform. It improved operational efficiency, reduced manual work, and provided accurate insights for better decision-making.

Key Outcomes:

Here are the key outcomes of the combined project:

The *Subscription Management System* project has successfully achieved its primary goal of automating and simplifying the subscription process for businesses that rely on recurring payments and customer memberships. By integrating modern technologies such as **Java Spring Boot**, **MySQL**, and **Razorpay/PayPal APIs**, the system provided an end-to-end solution for managing user registrations, plan creation, renewals, and payments in a single platform. This integration has resulted in improved efficiency, reduced manual effort, and enhanced reliability across all operations.

One of the major outcomes of the project is the **complete automation of subscription handling**, which eliminates the need for manual tracking or billing. The system allows users to subscribe, renew, or cancel their plans easily while ensuring that all payments are processed securely in real-time. Automated notifications via email and SMS ensure

that users are informed about payment confirmations, plan expirations, and upcoming renewals, which increases user engagement and satisfaction.

The **administrative dashboard** has proven to be another significant achievement, offering real-time insights into user activity, revenue trends, and plan performance. It enables the administrator to generate analytical reports that support better decision-making and business planning. The dashboard also provides a clear overview of all active and expired subscriptions, giving full transparency and control over the system's operation.

Security and data protection were key focus areas, and the project achieved a high standard in this regard. The use of **Spring Security** and **JWT (JSON Web Token)** authentication mechanisms ensured that all user and admin interactions are secure and restricted based on user roles. The implementation of **HTTPS encryption** further strengthened data privacy, ensuring safe handling of sensitive payment and personal information.

In addition to technical success, the project demonstrated strong **system performance and scalability**. During testing, the system handled multiple concurrent user operations efficiently without data loss or processing delays. The database design was optimized to ensure faster query responses, and the code structure was modular, allowing for easy maintenance and future expansion.

Overall, the combined outcomes of the project include enhanced **automation, data accuracy, transaction reliability**, and **system transparency**. The application successfully bridges the gap between business operations and customer convenience by offering a secure, scalable, and user-friendly environment. It not only fulfills the objectives outlined at the start but also lays a strong foundation for future improvements such as mobile app integration, advanced analytics, and AI-driven customer insights. The *Subscription Management System* thus stands as a complete and efficient solution for managing recurring service-based operations in real-world business applications.

Challenges and Resolutions:

1. Payment Gateway Integration

Challenge: Integrating Razorpay and PayPal for real-time transactions was complex due to API handling and response validation.

Resolution: Implemented webhook services and tested API responses using Postman to ensure accurate transaction confirmation and error handling.

2. Data Consistency and Accuracy

Challenge: Managing multiple subscriptions simultaneously led to data duplication and inconsistencies.

Resolution: Designed a normalized MySQL database with foreign keys and indexing, and used transactional control in Spring Boot to maintain data integrity.

3. System Security

Challenge: Protecting sensitive user and payment data from unauthorized access.

Resolution: Implemented **JWT authentication**, **Spring Security**, and **HTTPS encryption** to ensure secure data transmission and restricted access.

4. Performance Optimization

Challenge: The system initially faced slow data retrieval when handling large datasets.

Resolution: Optimized SQL queries, added caching, and used pagination to improve overall system speed and performance.

5. API Communication Errors

Challenge: Ensuring smooth communication between frontend, backend, and payment modules.

Resolution: Used **RESTful APIs** with clear documentation and performed regular integration testing to ensure stable connections.

6. Deployment and Configuration

Challenge: Hosting the Spring Boot application and linking it with the cloud database was initially difficult.

Resolution: Created deployment scripts and environment variables for seamless setup on **AWS/Heroku**, ensuring stable operation and scalability.

Conclusion:

The *Subscription Management System* efficiently automates subscription, renewal, and payment processes. It provides a secure and user-friendly interface for both users and administrators. Integration of Spring Boot, MySQL, and Razorpay/PayPal ensures smooth performance. Security through JWT authentication and Spring Security protects user data. The project reduces manual work while improving speed and accuracy. Overall, it delivers a reliable and scalable solution for managing digital subscriptions.

Key Achievements:

The project successfully automated the complete subscription and payment management process. It implemented secure **JWT-based authentication** and **Spring Security** for data protection. Integrated **Razorpay/PayPal** APIs for real-time and reliable online transactions. Developed an interactive **admin dashboard** for monitoring users and revenue reports. Achieved high system performance, scalability, and accuracy in operations. Delivered a user-friendly and efficient platform suitable for real-world applications.

Future Prospects:

1. The system can be expanded to support **mobile applications** for better user accessibility.
2. Integration of **AI-based analytics** can help predict customer behavior and renewal trends.
3. Multi-currency and **multi-language support** can make the platform more globally adaptable.
4. Adding **voice-based assistants** can enhance user interaction and convenience.
5. Blockchain technology may be used for **secure transaction tracking** in the future.
6. Continuous updates and feature enhancements will keep the system efficient and modern.