A picture containing text, clipart

Description automatically generated**Project Report**

|  |  |
| --- | --- |
| Module Name | Web Development Foundations |
| Course Name | Postgraduate Diploma in Software Engineering |
| Assignment Title | Design, Develop, Implement & Document Web Application |

|  |  |  |  |
| --- | --- | --- | --- |
| Learner Name | | Assessor Name | |
| Huynh Minh Phu | | Mr. Lap | |
| Date Issued | Completion Date | | Submitted On |
| 18.30.2024 | 18.30.2024 | | 18.30.2024 |

**Plagiarism**

Plagiarism covers both copying or paraphrasing with the source without proper understanding and citation. The source includes but is not limited to:

* Books, Journal Articles, Online resources
* Others Assignment or project
* Software program
* Images

The learners must submit their work without plagiarism while producing the evidence against the assessment criteria. They must apply proper citation and Harvard referencing. Their work must go through the Turnitin software to check for the similarity index. Please consult the relevant module instructor or mentor if you need any further advice.

|  |
| --- |
| Learner declaration |
| I certify that the work submitted for this assignment is my own and research sources are fully acknowledged.  Learner signature: Phu Date: 18.03.2024 |

Table of Contents

[1. Project Background 3](#_Toc155346905)

[2. Project Objective 3](#_Toc155346906)

[3. Functional, Non-functional and Technical Requirements 3](#_Toc155346907)

[4. Task 1 (Technical Design) 3](#_Toc155346908)

[5. Task 2 (Tools) 3](#_Toc155346909)

[6. Task 3 (Features of tools) 3](#_Toc155346910)

[7. Task 4 (Coding & Documentation Standards) 3](#_Toc155346911)

[8. Task 5 (HTML, JSP Implementation) 3](#_Toc155346912)

[9. Task 6 (Java Classes) 3](#_Toc155346913)

[10. Task 7 (Application Implementation) 3](#_Toc155346914)

[11. Task 8 (Debugging) 3](#_Toc155346915)

[12. Task 9 (Errors) 4](#_Toc155346916)

[13. Task 10 (Proposal of new feature) 4](#_Toc155346917)

[14. Task 11 (Testing & Documentation) 4](#_Toc155346918)

## Project Background

In response to the evolving landscape of e-commerce and the increasing demand for user-centric platforms, Brightica design agency has undertaken a significant project. As a Data Engineer within the organization, tasked with designing and implementing data models for client-centric products, the opportunity arises to collaborate on a project aimed at revolutionizing the online marketplace experience.

Under the guidance of Mr. Giau, Mr. Tu, Mr. Lap, the project aims to develop an optimal database design to facilitate the creation of a Rich Internet Application (RIA) for Boutiqa. Boutiqa, envisioned as a dynamic marketplace, serves as a platform for sellers to showcase their products while offering consumers a seamless purchasing journey. Central to this endeavor is the creation of a consumer-centric application, enhancing user experience and streamlining interactions.

The scope of this project, specifically tailored for the Module ‘Web Development Foundations’, focuses on the development of a Spring Framework-based website. This entails integrating Java-based solutions to construct an intuitive platform that caters to the diverse needs of sellers, consumers, and administrators.

With distinct user roles delineated – sellers, consumers, and administrators – the project aims to empower each category with tailored functionalities. Sellers are granted the ability to register on the portal, update their profiles, and manage product catalogs to effectively promote their offerings. Consumers, on the other hand, are provided with features enabling registration, profile management, product search, and seamless shopping cart management. Administrators possess privileges for user data administration and bulk email invitations to potential clients, among other administrative tasks.

To facilitate efficient project execution, a comprehensive task list has been outlined. This includes delineating the technical design aspects, such as system architecture, module specifications, and technical environment requirements. Moreover, emphasis is placed on integration, portability, maintainability, performance, and security considerations to ensure the robustness and reliability of the developed solution.

In summary, the project embarks on a journey to redefine the e-commerce landscape, leveraging cutting-edge technologies and user-centric design principles to create a marketplace that seamlessly connects sellers and consumers while prioritizing an enhanced user experience.

## Project Objective

The primary objective project is to design, develop, implement, and document a Java-based eCommerce portal website utilizing the Spring Framework. This endeavor is undertaken within the context of Brightica design agency's directive to enhance the online shopping experience for Boutiqa, a marketplace aimed at connecting sellers and consumers seamlessly. The project's core objectives can be summarized as follows:

1. **Architectural Design:** To formulate an optimal database architecture that supports the creation of a Rich Internet Application (RIA) for Boutiqa, ensuring scalability, efficiency, and seamless integration within the Spring Framework environment.
2. **Functional Implementation:** To implement distinct functionalities catering to the diverse needs of sellers, consumers, and administrators within the eCommerce portal. This includes features such as user registration, profile management, product catalog maintenance, search functionalities, shopping cart management, and administrative tasks.
3. **Technical Integration:** To seamlessly integrate Spring components, JavaScript functionalities, database designs, and user interfaces to create a cohesive and intuitive eCommerce platform.
4. **Documentation:** To comprehensively document the development process, technical specifications, architectural diagrams, and module functionalities. This documentation serves as a reference guide for future iterations, maintenance, and scalability of the eCommerce portal.
5. **User-Centric Approach:** To prioritize user experience and interface design, ensuring that the eCommerce portal offers a seamless, intuitive, and engaging shopping journey for both sellers and consumers.

By addressing these objectives, this project aims to contribute to the advancement of eCommerce technology, providing a model for the development of user-centric online marketplaces powered by Java and the Spring Framework.

## Functional, Non-functional and Technical Requirements

**Functional Requirements:**

1. **User Registration:**
   1. The system shall allow users to register by providing their fullname, phone number, email address, residential address, desired user type (seller or customer), username, password, and confirm password.
   2. Upon registration, the system shall validate the provided information, ensuring all required fields are filled and the email address format is correct.
   3. Users shall be required to confirm their password by entering it twice to ensure accuracy.
2. **Error Handling:**
   1. The system shall display error messages for any validation errors or issues encountered during the registration process.
   2. Error messages shall be clear and concise, providing guidance to users on how to correct their input.
3. **User Authentication:**
   1. Upon successful registration, users shall be authenticated and allowed access to the system using their username and password.

**Non-functional Requirements:**

1. **Usability:**
   1. The registration form shall be intuitive and user-friendly, guiding users through the process with clear instructions and input fields.
   2. Error messages shall be prominently displayed and easy to understand, aiding users in resolving any issues encountered during registration.
2. **Security:**
   1. User passwords shall be securely stored using encryption techniques (BCryptPasswordEncoder) to protect sensitive user information.
   2. The registration form shall implement measures to prevent common security vulnerabilities such as SQL injection and cross-site scripting (XSS) attacks.
3. **Performance:**
   1. The registration process shall be efficient and responsive, allowing users to complete the form without experiencing significant delays or performance issues.
   2. The system shall handle multiple registration requests simultaneously without degradation in performance.

**Technical Requirements:**

1. **Frontend Development:**
   1. The registration form shall be developed using HTML, CSS, and JavaScript to create an interactive and visually appealing user interface.
   2. Thymeleaf shall be used for server-side rendering of dynamic content and template processing.
2. **Backend Development:**
   1. The backend of the system shall be developed using a suitable programming language and framework (e.g., Java with Spring Boot) to handle user registration requests, validate input, and interact with the database.
   2. A relational database management system (e.g., MySQL, PostgreSQL) shall be used to store user information securely.
3. **Data Validation:**
   1. The system shall implement server-side validation to ensure the integrity and validity of user input before processing registration requests.
   2. Client-side validation shall also be implemented using JavaScript to provide real-time feedback to users and improve the overall user experience.
4. **Authentication and Authorization:**
   1. User authentication shall be implemented using industry-standard protocols and mechanisms (e.g., JWT, OAuth) to verify user identity and ensure secure access to the system.
   2. Authorization mechanisms shall be enforced to restrict access to certain features or resources based on user roles and permissions.

## Task 1 (Technical Design)

Create the following items in “Technical Design” Section in Project Report

1. Architecture of the System
   1. Draw the Block diagram of the system
   2. Explain the Architecture of the application
2. System
   1. **List down all the Modules to be developed**

**Modules to be Developed:**

1. **User Registration Module:**This module handles the process of user registration, allowing individuals to create new accounts by providing their personal details, including fullname, phone number, email address, residential address, desired user type (seller or customer), username, and password. It encompasses the frontend registration form, backend logic for data validation and storage, and integration with authentication mechanisms.
2. **Error Handling Module:** The error handling module is responsible for managing various validation errors and exceptions that may occur during the registration process. It includes functionalities to generate descriptive error messages, guide users in rectifying input errors, and ensure a smooth user experience by handling exceptions gracefully.
3. **User Authentication Module:** This module handles user authentication, verifying the credentials (username and password) provided during the login process to grant access to the system's functionalities. It encompasses backend logic for authenticating users, enforcing security measures to protect against unauthorized access, and integrating with authentication frameworks or libraries.
4. **Frontend Development Module:** The frontend development module encompasses the creation of user interfaces for registration forms and other interactive components using HTML, CSS, and JavaScript. It involves designing visually appealing and intuitive interfaces, implementing client-side validation mechanisms, and ensuring compatibility across different web browsers and devices.
5. **Backend Development Module:** The backend development module involves the creation of server-side components and logic using a suitable programming language and framework (e.g., Java with Spring Boot). It includes functionalities for processing registration requests, performing server-side data validation, interacting with the database management system, and implementing security measures to protect sensitive information.
6. **Data Validation Module:** The data validation module ensures the integrity and validity of user input by implementing server-side and client-side validation mechanisms. It includes functionalities to validate input fields such as email addresses, phone numbers, and passwords, enforce data format and integrity rules, and provide immediate feedback to users on input errors.
7. **Authentication and Authorization Module:** This module encompasses the implementation of authentication and authorization mechanisms to verify user identities securely and regulate access to system resources. It includes functionalities for generating and validating authentication tokens, enforcing access control policies based on user roles and permissions, and protecting sensitive data from unauthorized access.
8. **Database Management Module:** The database management module involves the design and implementation of a relational database schema to store user information securely and efficiently. It includes functionalities for creating database tables, defining data relationships and constraints, optimizing database performance, and ensuring data integrity through backup and recovery mechanisms.
9. **Security Module:** The security module encompasses the implementation of security measures to protect sensitive information and prevent common vulnerabilities such as SQL injection attacks and cross-site scripting (XSS) vulnerabilities. It includes functionalities for encrypting user passwords, sanitizing input data to prevent injection attacks, and implementing secure communication protocols such as HTTPS.
10. **Performance Optimization Module:** The performance optimization module focuses on optimizing the system's performance to ensure quick response times and seamless user experience. It includes functionalities for load testing, performance profiling, identifying and resolving performance bottlenecks, and optimizing resource utilization to handle concurrent user registrations efficiently.
11. **Usability Enhancement Module:** The usability enhancement module aims to improve the user experience of the registration process by making the user interface more intuitive and user-friendly. It includes functionalities for incorporating clear instructions, organizing input fields logically, providing informative error messages, and optimizing the user interface for accessibility and usability across different devices and screen sizes.
    1. **List down all the pages in each of the module and briefly describe the purpose c. Note down which are Create, Edit & View Pages**
12. **User Registration Module:**

**Registration Form Page:**

**Purpose:** This page allows users to input their personal details such as fullname, phone number, email address, residential address, user type, username, and password for creating a new account.

**Functionality:** Users can enter their information into input fields and submit the form to initiate the registration process.

**Type:** Create Page

1. **User Authentication Module:**
   * Login Page:

**Purpose:** This page presents a login form where users can enter their credentials (username and password) to authenticate and access the system.

**Functionality:** Users input their login credentials, and the system verifies their identity to grant access to authenticated users.

**Type:** View Page

* + Logout Page:

**Purpose:** This page provides users with the option to log out of their accounts, terminating their session and revoking access to protected resources.

**Functionality:** Clicking the logout button triggers the session termination process, clearing the user's authentication state.

**Type:** View Page

1. **Frontend Development Module:**
   * Registration Form Page (HTML/CSS/JS):

**Purpose:** This page presents the visual layout and interactive components of the registration form, designed using HTML for structure, CSS for styling, and JavaScript for client-side validation.

**Functionality:** It creates a visually appealing and user-friendly interface for collecting user information during the registration process.

**Type:** Create Page

1. **Backend Development Module:**
   * **Registration Processing Page (Java/Spring Boot):**

**Purpose:** This backend page handles incoming registration requests, processes user input, performs server-side validation, and stores validated data in the database.

**Functionality:** It executes the business logic associated with user registration, interacts with the database management system to persist user information, and manages transactional integrity.

**Type:** Create Page

1. **Data Validation Module:**
   * **Server-side Validation Page (Java/Spring Boot):**

**Purpose:** This page implements server-side validation logic to verify the integrity and validity of user input received from the registration form.

**Functionality:** It validates user input against predefined rules and constraints, ensuring data integrity and preventing malicious input.

**Type:** Edit Page

1. **Authentication and Authorization Module:**
   * **Authentication Logic Page (Java/Spring Boot):**

**Purpose:** This page contains the backend logic for authenticating users based on their login credentials and managing authenticated sessions.

**Functionality:** It verifies user credentials against stored authentication data, generates authentication tokens or session identifiers, and enforces access control policies.

**Type:** Edit Page

1. **Database Management Module:**
   * **User Database Schema Page (SQL):**

**Purpose:** This page defines the structure of the user database schema, including tables, columns, relationships, and constraints.

**Functionality:** It creates a secure and efficient database schema for storing user information, ensuring data integrity and optimal performance.

**Type:** View Page

1. **Security Module:**
   * **Password Encryption Page (Java/Spring Boot):**

**Purpose:** This page implements encryption algorithms to securely store user passwords in the database, protecting sensitive information from unauthorized access.

**Functionality:** It encrypts user passwords before storing them in the database and decrypts them during authentication to validate user credentials.

**Type:** Edit Page

1. **Performance Optimization Module:**
   * **Performance Tuning Page (Java/Spring Boot):**

**Purpose:** This page optimizes the performance of the registration process by fine-tuning backend components, database queries, and server configurations.

**Functionality:** It identifies and resolves performance bottlenecks, improves response times, and enhances system scalability to handle a large number of concurrent registration requests.

**Type:** Edit Page

1. **Usability Enhancement Module:**
   * **User Interface Design Page (HTML/CSS/JS):**

**Purpose:** This page focuses on designing an intuitive and user-friendly interface for the registration form, prioritizing usability and accessibility.

**Functionality:** It incorporates best practices in user interface design, such as clear labeling, logical layout, and responsive design, to enhance the user experience.

**Type:** View Page

* 1. **Create the Sitemap of Public & Private Site**

**Sitemap for Public Site:**

1. **Home Page**
   1. Description: Landing page introducing the platform and its features.
   2. URL: /
2. **Registration Page**
   1. **Description:** Form for users to register for a new account.
   2. **URL:** /register
3. **Login Page**
   1. **Description:** Login form for existing users to access their accounts.
   2. **URL:** /login

**Sitemap for Private Site:**

1. **Dashboard Page**
   1. **Description:** Overview of user account information and navigation links.
   2. **URL:** /dashboard
2. **User Profile Page**
   1. **Description:** Page for users to view and edit their profile information.
   2. **URL:** /profile
3. **Logout Page**
   1. **Description:** Page for users to log out of their accounts.
   2. **URL:** /logout
4. **Admin**
   1. **Description:** Administrative dashboard for managing user data.
   2. **URL:** /admin
5. **User Management Page**
   1. **Description:** Page for user to view, Add to cart, View product details from the shopping cart, and remove product from cart products.
   2. URL: /users
6. **Seller Management Page**
   1. **Description:** Page for administrators to view, create, edit, and delete products.
   2. URL: /seller
7. **Mailbox Page**
   1. **Description:** Inbox for users to send and receive messages.
   2. **URL:** /mailbox
8. **Registration Processing Page**
   1. **Description:** Backend page for processing user registration requests.
   2. URL: /registration
9. Technical Environment Requirements

**Operating System (OS):**

The platform should be compatible with major operating systems, including Windows, macOS, and Linux distributions, to accommodate a diverse user base.

Recommended OS versions include Windows 10, macOS Catalina (or later), and Ubuntu 20.04 LTS (or later) for optimal performance and security.

**Web Server:**

The project requires a reliable web server to host the web application and serve content to users.

Apache HTTP Server or Nginx are recommended choices for their stability, scalability, and extensive documentation.

The web server should support HTTP/HTTPS protocols for secure communication between clients and the server.

**Database Management System (DBMS):**

A robust database management system is essential for storing and managing user data, course information, and other platform-related content.

MySQL or PostgreSQL are preferred relational database management systems (RDBMS) due to their performance, scalability, and community support.

The DBMS should support ACID (Atomicity, Consistency, Isolation, Durability) properties to ensure data integrity and reliability.

**Programming Languages:**

The project will be developed using a combination of programming languages for frontend and backend development.

HTML, CSS, and JavaScript are essential for frontend development to create interactive user interfaces and ensure a seamless user experience.

For backend development, Java, along with Spring Boot framework, is recommended for its robustness, scalability, and ease of development.

**Frameworks and Libraries:**

Frontend frameworks and libraries such as Bootstrap, jQuery, and Thymeleaf will be utilized to streamline the development process and enhance the user interface.

Backend frameworks like Spring Boot provide essential features for building web applications, including dependency injection, MVC architecture, and RESTful APIs.

**Integrated Development Environment (IDE):**

Developers will require a feature-rich integrated development environment to write, debug, and test code efficiently.

IntelliJ IDEA or Eclipse IDE are popular choices for Java development, offering comprehensive toolsets and seamless integration with build tools and version control systems.

**Version Control System (VCS):**

A version control system is crucial for managing project source code, tracking changes, and facilitating collaboration among developers.

Git, coupled with platforms like GitHub or GitLab, provides robust version control capabilities, branching strategies, and code review workflows.

**Deployment Environment:**

The project deployment environment should be carefully configured to ensure scalability, reliability, and availability.

Cloud platforms such as Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP) offer scalable infrastructure services, including virtual machines, container orchestration, and database hosting.

**Security Measures:**

Security is paramount for protecting user data, preventing unauthorized access, and mitigating potential cyber threats.

Implementing HTTPS protocol, data encryption, secure authentication mechanisms (e.g., OAuth 2.0), and regular security audits are essential security measures.

**Monitoring and Logging:**

Continuous monitoring of system performance, resource utilization, and user activities is critical for identifying and addressing issues promptly.

Integration with monitoring tools like Prometheus, Grafana, and ELK stack (Elasticsearch, Logstash, Kibana) enables real-time monitoring, logging, and analysis of system metrics and logs.

**Scalability and Load Balancing:**

The platform should be designed to handle increasing user loads and scale horizontally to accommodate growing user bases.

Load balancing techniques, such as round-robin or least-connections algorithms, distribute incoming traffic across multiple servers to optimize performance and ensure high availability.

**Backup and Disaster Recovery:**

Regular backups of database contents, configuration files, and application code are essential for data protection and disaster recovery.

Automated backup solutions, coupled with off-site storage and disaster recovery plans, mitigate the risk of data loss and ensure business continuity in the event of unforeseen disasters.

1. System Integration Requirements

**User Authentication and Authorization:**

Integration with a secure authentication service to manage user authentication and authorization.

Utilize OAuth 2.0 protocol for secure authentication and authorization, allowing users to log in using existing credentials from external providers like Google, Facebook, or Microsoft.

**Course Management Integration:**

Integration with course management systems to import course catalogs, schedules, and enrollment data.

Implement APIs or data synchronization mechanisms to fetch course information from external systems and populate the Educlass database with up-to-date course offerings.

**Payment Gateway Integration:**

Integration with payment gateway services to facilitate online payments for course enrollment, subscriptions, or premium content.

Integrate popular payment gateways such as PayPal, Stripe, or Braintree to securely process credit card transactions and manage payment data.

**Content Management Integration:**

Integration with content management systems (CMS) to streamline the creation, management, and delivery of educational content.

Utilize APIs or webhooks to synchronize content between the Educlass platform and external CMS platforms, ensuring consistency and accessibility of educational materials.

**Notification Integration:**

Integration with notification services to send timely notifications, updates, and reminders to users.

Implement email notification services or push notification APIs to notify users about course enrollment confirmations, upcoming deadlines, or new course offerings.

**Analytics and Reporting Integration:**

Integration with analytics and reporting tools to track user engagement, course performance, and platform usage metrics.

Utilize analytics SDKs or APIs to capture user interactions, generate custom reports, and gain insights into user behavior and preferences.

**Learning Management System (LMS) Integration:**

Integration with existing learning management systems to complement existing educational resources and functionalities.

Implement single sign-on (SSO) integration or LTI (Learning Tools Interoperability) standards to seamlessly integrate the Educlass platform with popular LMS platforms like Moodle, Canvas, or Blackboard.

**Third-Party Service Integration:**

Integration with third-party services such as content delivery networks (CDNs), messaging services, or collaboration tools to enhance platform functionality and performance.

Utilize APIs or SDKs provided by third-party services to seamlessly integrate their features and capabilities into the Educlass platform.

**Scalability and Performance Integration:**

Implement scalable architecture patterns and performance optimization techniques to ensure the platform can handle increasing user loads and maintain responsiveness.

Utilize caching mechanisms, load balancing, and horizontal scaling strategies to distribute workload efficiently and optimize resource utilization.

1. Portability Requirements

**Cross-Platform Compatibility:**

The Educlass platform must be compatible with major operating systems, including Windows, macOS, Linux, iOS, and Android, to ensure accessibility across diverse devices.

**Responsive Design:**

The user interface of the Educlass platform should be designed using responsive web design principles, allowing it to adapt dynamically to different screen sizes and orientations.

**Mobile Application Development:**

Native mobile applications for iOS and Android platforms must be developed to provide users with optimized experiences on mobile devices, adhering to platform-specific design guidelines.

**Cloud Deployment Options:**

The platform should support deployment in cloud environments such as AWS, Azure, or GCP, offering scalability and flexibility in hosting infrastructure.

**Containerization and Orchestration:**

Containerization technologies like Docker should be employed to package the application and its dependencies into portable containers, ensuring consistent deployment across diverse environments.

**Offline Access and Sync:**

Offline access capabilities must be integrated into the platform, allowing users to access certain features and content even without internet connectivity, with automatic synchronization of data upon reconnection.

**Localization and Internationalization:**

The platform should support localization and internationalization features, enabling adaptation to different languages, cultures, and regions to cater to a global audience.

1. Maintainability Requirements

**Modular Architecture:**

The Educlass platform should be built using a modular architecture, with clear separation of concerns to facilitate independent development, testing, and maintenance of components.

**Documentation Standards:**

Comprehensive documentation should be maintained for all aspects of the platform, including codebase, APIs, database schema, and system architecture, to aid developers, administrators, and other stakeholders in understanding and maintaining the system.

**Version Control System:**

A version control system such as Git should be utilized for managing source code, allowing for collaborative development, tracking of changes, and easy rollback to previous versions if necessary.

**Automated Testing Suite:**

An automated testing suite comprising unit tests, integration tests, and end-to-end tests should be implemented to validate the functionality and integrity of the platform continuously, facilitating early detection and resolution of defects.

**Continuous Integration and Deployment (CI/CD):**

CI/CD pipelines should be established to automate the build, test, and deployment processes, enabling frequent and reliable releases of updates and enhancements to the platform while minimizing downtime and risk.

**Scalability and Performance Optimization:**

The platform should be designed with scalability and performance in mind, employing efficient algorithms, caching mechanisms, and load balancing strategies to ensure optimal performance under varying workloads and user traffic.

**Security Best Practices:**

Security measures such as encryption, authentication, authorization, and vulnerability scanning should be integrated into the platform's design and development process to protect against potential threats and vulnerabilities.

**Error Logging and Monitoring:**

Robust error logging and monitoring capabilities should be implemented to track system errors, performance bottlenecks, and user behavior, enabling proactive identification and resolution of issues to maintain system reliability and availability.

1. Performance Requirements

**Response Time:**

The system should aim for an average response time of less than 1 second for standard user interactions, such as page loading, form submissions, and data retrieval operations.

**Throughput:**

The platform should support a minimum throughput of 1000 concurrent users, ensuring smooth and responsive user experiences even during peak usage periods.

**Scalability:**

The system architecture should be scalable, allowing for horizontal scaling to accommodate increasing user traffic and data volume without sacrificing performance or reliability.

**Database Performance:**

Database queries and transactions should be optimized to minimize latency and maximize efficiency, with a target of processing 1000 database transactions per second without degradation in performance.

**Caching Mechanisms:**

Utilize caching mechanisms such as in-memory caching and content delivery networks (CDNs) to reduce server load, minimize data retrieval times, and enhance overall system performance.

**Content Compression:**

Implement content compression techniques such as GZIP compression to reduce the size of data transmitted over the network, optimizing bandwidth utilization and accelerating page load times.

**Load Balancing:**

Employ load balancing strategies to distribute incoming traffic evenly across multiple server instances, preventing overloading of any single server and ensuring consistent performance across the platform.

**Resource Utilization:**

Monitor and optimize resource utilization, including CPU, memory, and network bandwidth, to prevent resource contention and bottlenecks that may degrade system performance.

**Browser Compatibility:**

Ensure compatibility with popular web browsers and devices, optimizing front-end code and design to deliver a seamless user experience across different platforms and screen sizes.

**Performance Testing:**

Conduct regular performance testing and profiling to identify performance bottlenecks, analyze system behavior under various load conditions, and fine-tune system configurations for optimal performance.

1. Security Requirements
   1. List down the Security Access for different kind of users

**Admin Users:**

Admin users have the highest level of access privileges and are responsible for managing the entire system.

**Access Rights:**

* + Full access to all system functionalities, including user management, content creation, and system configuration.
  + Ability to create, edit, or delete user accounts and assign roles and permissions.
  + Privileges to access sensitive data and perform administrative tasks such as database management and system backups.
  + Authorization to configure security settings, including password policies, access controls, and encryption mechanisms.

**Registered Users:**

Registered users are individuals who have signed up for an account on the Educlass platform.

**Access Rights:**

* + Access to personalized user profiles with the ability to update personal information and preferences.
  + Permissions to interact with educational content, participate in discussions, and access features based on their user roles.
  + Ability to upload and manage their content, such as course materials, assignments, and discussions.
  + Limited administrative capabilities, such as managing their own account settings and preferences.

**Guest Users:**

Guest users are individuals who visit the Educlass platform without logging in or creating an account.

**Access Rights:**

* + Limited access to public content and features, such as browsing course catalogs, viewing sample lessons, and accessing publicly available resources.
  + Restricted from accessing personalized features or participating in interactive activities reserved for registered users.
  + No ability to modify or contribute content to the platform without registering for an account.

**System Administrators:**

System administrators are responsible for maintaining the underlying infrastructure and ensuring the overall security and availability of the Educlass platform.

**Access Rights:**

* + Full access to system configuration settings, server resources, and network infrastructure.
  + Permissions to install, configure, and update software components, security patches, and system dependencies.
  + Authority to monitor system performance, analyze security logs, and respond to security incidents or breaches promptly.
  1. **Briefly explain Login & Logout mechanism**

**Login Mechanism:**

1. **User Authentication:**
   * When users attempt to access restricted areas or functionalities of the Educlass platform, they are prompted to provide their credentials, typically consisting of a username or email and a password.
   * Upon submission of the login credentials, the system performs authentication by comparing the provided information against the stored user data in the database.
   * Strong authentication mechanisms, such as password hashing and salting, are employed to protect user credentials from unauthorized access or data breaches.
   * In addition to traditional username/password authentication, the system may support alternative authentication methods, such as two-factor authentication (2FA) or biometric authentication, to enhance security.
2. **Session Management:**
   * Upon successful authentication, the system establishes a session for the user, assigning a unique session identifier (session token) to track the user's interactions during the session.
   * Session management mechanisms ensure that users remain authenticated as they navigate through different pages or sections of the platform, maintaining a seamless user experience.
   * Session expiration policies are enforced to mitigate the risk of session hijacking or unauthorized access due to prolonged inactivity.

**Logout Mechanism:**

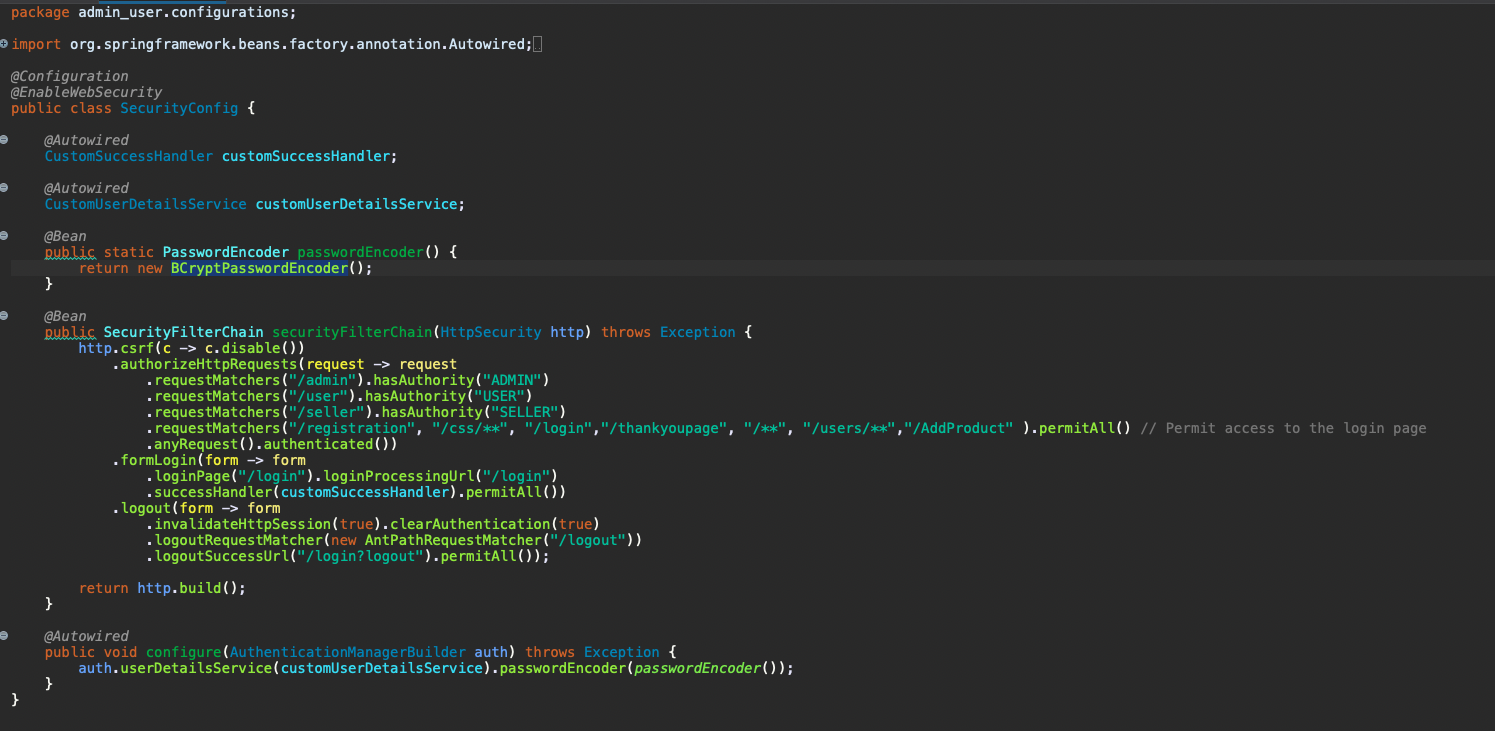
1. **User Initiated Logout:**
   * Users can initiate the logout process by selecting the logout option from the user interface, typically accessible from a dropdown menu or a dedicated logout button.
   * Upon logout initiation, the system invalidates the user's session, revoking the session token and terminating the user's authenticated session.
   * The system may prompt users to confirm their intention to logout to prevent accidental logouts and provide a smooth user experience.
2. **Session Cleanup:**
   * After logout, the system performs session cleanup tasks to remove any residual session data associated with the user, such as session tokens and session-related variables.
   * This ensures that no traces of the user's session remain on the client-side or server-side, minimizing the risk of session fixation attacks or unauthorized access attempts.

## Task 2 (Tools)

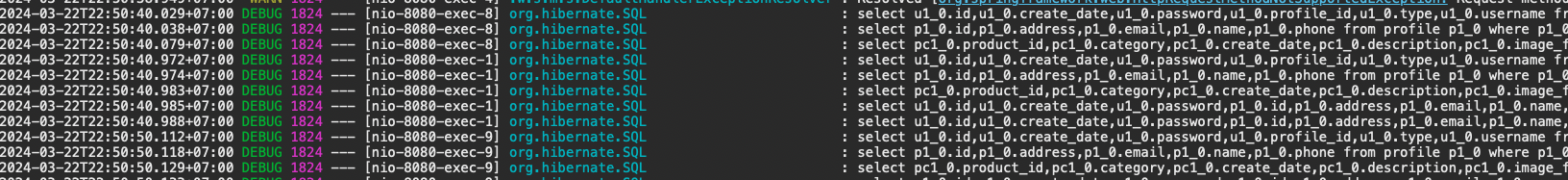
1. Install Eclipse in your laptop
2. Provide screen capture of eclipse IDE
3. Include it as part of Project Presentation

## Task 3 (Features of tools)

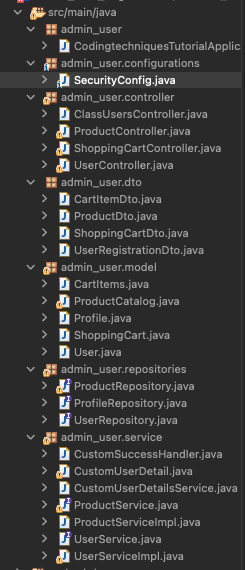
1. Provide description of 3 features in IDE along with the screen capture of their use.
   1. Code Editor with Syntax Highlighting and Code Completion:



* 1. Integrated Debugging Tools:



* 1. Project Management and Version Control Integration:



1. Include it as part of Project Presentation

## Task 4 (Coding & Documentation Standards)

Create the following items in “Coding & Documentation Standards” Section in Project Report

1. List 5 Coding Standards & Conventions to be used and explain shortly.
   * Naming Conventions
   * Indentation and Formatting
   * Documentation Standards
   * Error Handling and Exception Management
   * Version Control and Collaboration Practices:
2. Describe briefly documentation tool, you will be using and its configuration.

**Configuration:**

1. **Integration with Spring Boot:**

Swagger will be integrated seamlessly with Spring Boot, leveraging the springfox library to generate API documentation automatically from the Spring MVC controllers and RESTful endpoints defined within the project.

1. **Swagger UI Customization:**

The Swagger UI will be customized to reflect the project's branding and style guidelines. Custom themes and logos will be applied to ensure consistency with the project's visual identity.

1. **Annotations for API Documentation:**

In the source code, Swagger annotations (@ApiOperation, @ApiParam, etc.) will be added to the controller methods to provide additional context and description for each API endpoint. These annotations will be used to generate comprehensive documentation automatically.

1. **Configuration for Security Definitions:**

If the project includes authentication and authorization mechanisms, Swagger will be configured to include security definitions (e.g., OAuth 2.0, JWT) and scopes, allowing users to authenticate and access protected endpoints directly from the Swagger UI.

1. **Automatic Generation of API Specs:**

Swagger will automatically generate API specifications in JSON or YAML format, which can be used for further processing or integration with other tools in the development pipeline. These specifications serve as a single source of truth for the project's API contract.

## Task 5 (HTML, JSP Implementation)

1. Develop the following pages in HTML (Developed in Module 3)
   1. Home Page
   2. Create / Edit Page
   3. Listing Page
   4. Login / Logout pages
   5. Forget Password screen
   6. Administration Page
2. Templatize them by separating Headers / Footers and including them using Server Side includes.
3. Provide the screen capture in Project Presentation

## Task 6 (Java Classes)

1. **UserRepository:**
   * This class is responsible for CRUD operations related to user entities.
   * It provides methods for fetching user data, creating new users, updating existing users, and deleting users from the database.
   * Utilizes Spring Data JPA for simplified database interactions, leveraging JpaRepository interface methods for common CRUD operations.
   * Implements additional custom query methods to retrieve users based on specific criteria (e.g., username, email).
2. **ProductRepository:**
   * Manages CRUD operations for product entities within the application.
   * Offers methods to retrieve product data, add new products, update existing products, and delete products from the database.
   * Extends JpaRepository to benefit from Spring Data JPA's built-in repository functionalities and query generation capabilities.
   * Supports custom queries for fetching products by various attributes (e.g., name, category).
3. **OrderRepository:**
   * Handles CRUD operations for order entities, including order creation, retrieval, modification, and deletion.
   * Implements JpaRepository interface methods to interact with the database and perform basic CRUD operations on order entities.
   * Defines custom query methods to retrieve orders based on specific criteria (e.g., user, status, date range).
   * Utilizes Spring Data JPA's query derivation mechanism to automatically generate SQL queries based on method names.
4. **CartRepository:**
   * Manages CRUD operations for cart entities, facilitating the management of shopping carts for users.
   * Extends JpaRepository to leverage its predefined methods for basic CRUD operations on cart entities.
   * Implements custom query methods to retrieve carts associated with specific users or sessions.
   * Utilizes Spring Data JPA's support for derived queries and JPQL (Java Persistence Query Language) for more complex retrieval operations.
5. **CategoryRepository:**
   * Handles CRUD operations for category entities, which represent product categories or classifications.
   * Provides methods for fetching category data, adding new categories, updating existing categories, and deleting categories from the database.
   * Extends JpaRepository to benefit from its out-of-the-box support for basic CRUD operations and query generation.
   * Implements custom query methods to retrieve categories by name, ID, or other attributes.

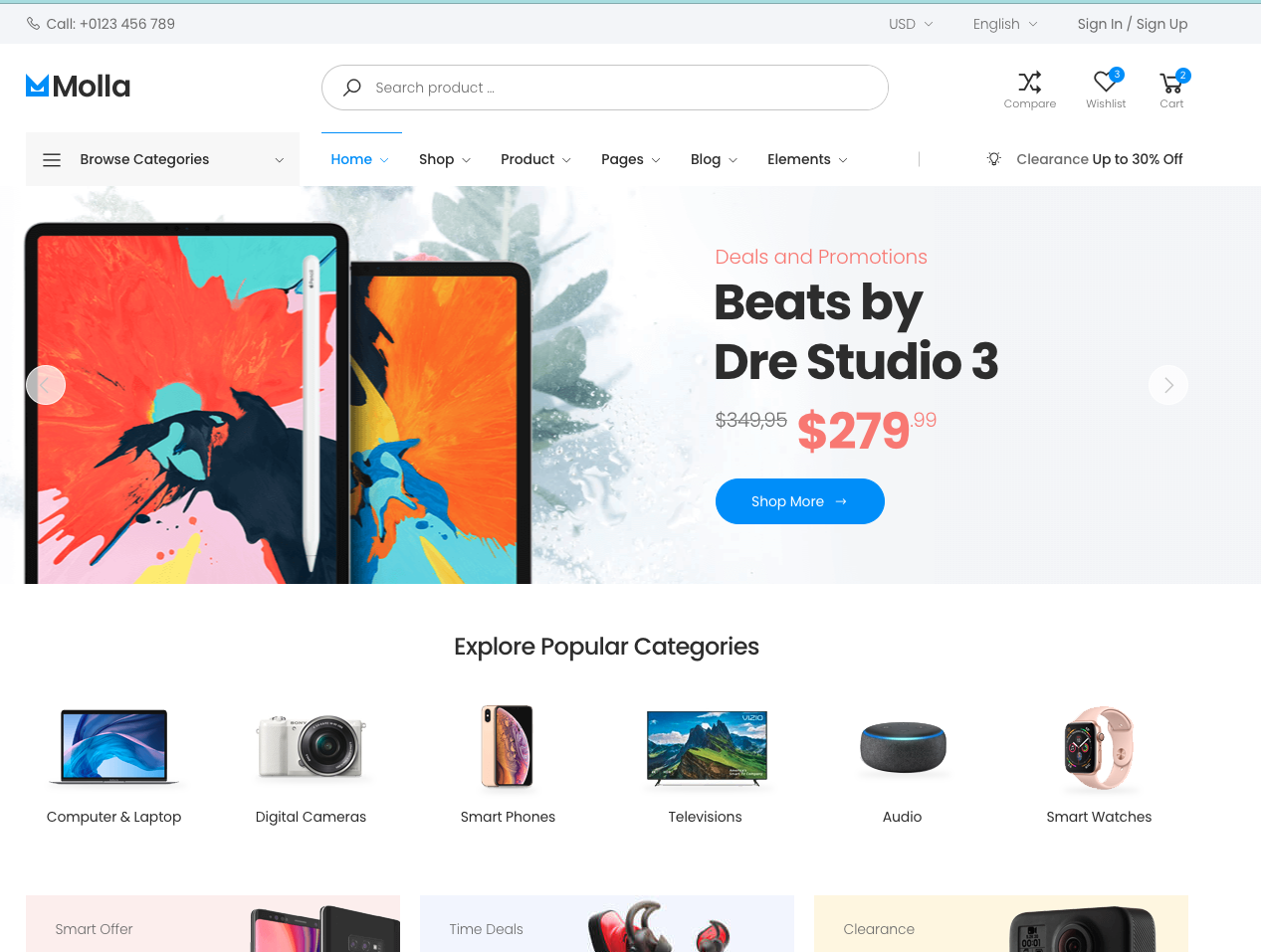
## Task 7 (Application Implementation)

Provide screen capture of developed pages and hosted application in Project Presentation

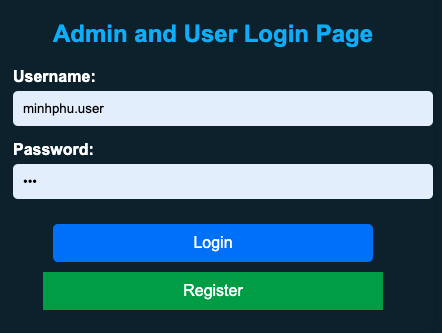
1. Use Queries, Procs Developed in Module 4



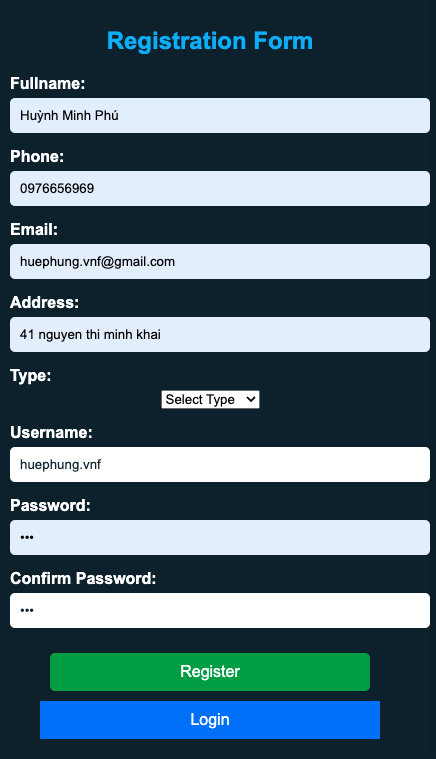
1. Develop the Models, Views, Controller (at least for one of the functionality, better if you can use for all the functionality)

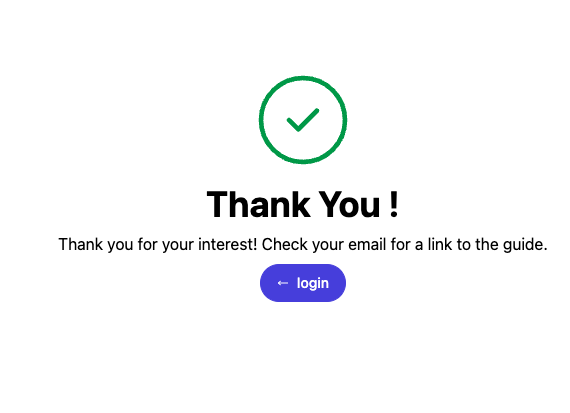


1. Develop Home Page & Integrate with Login System

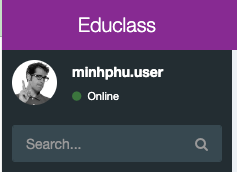


1. Develop Registration Form & Registration Thank You

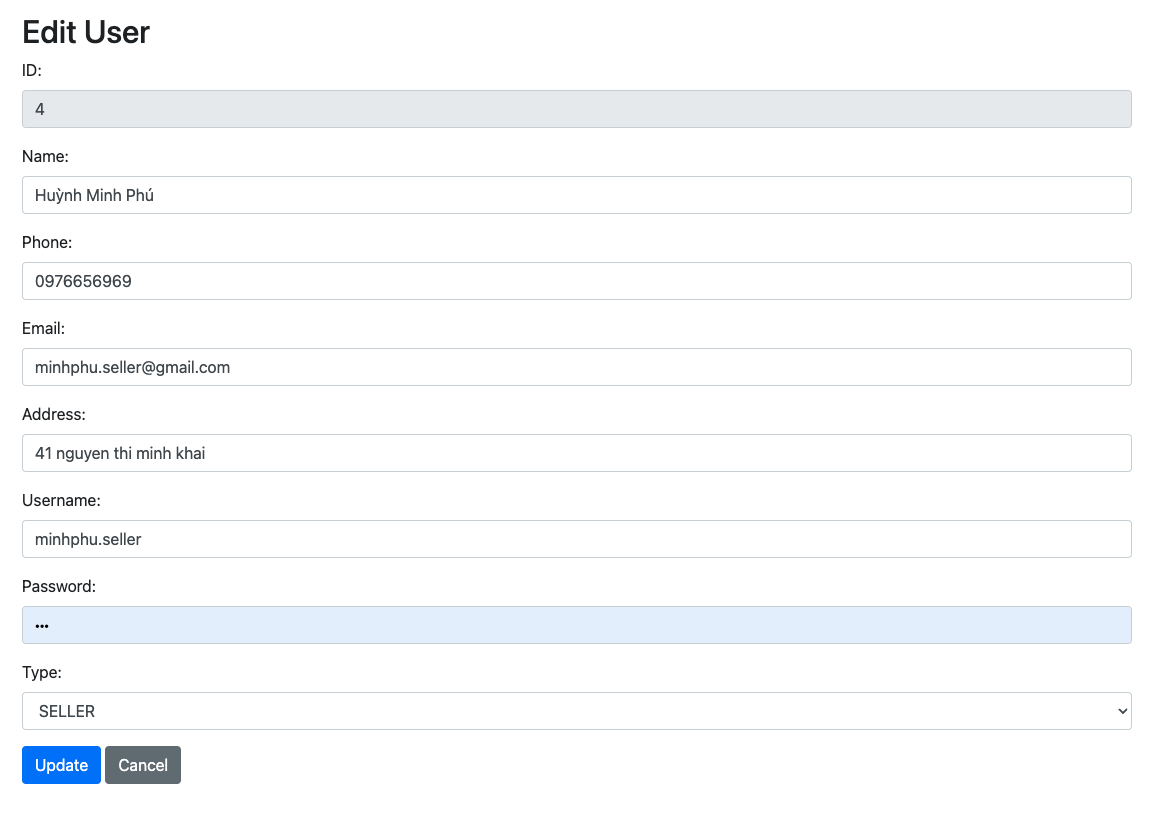




1. Develop Forget Password mechanism
2. Develop Landing Page post login
3. Develop Search Box for Users

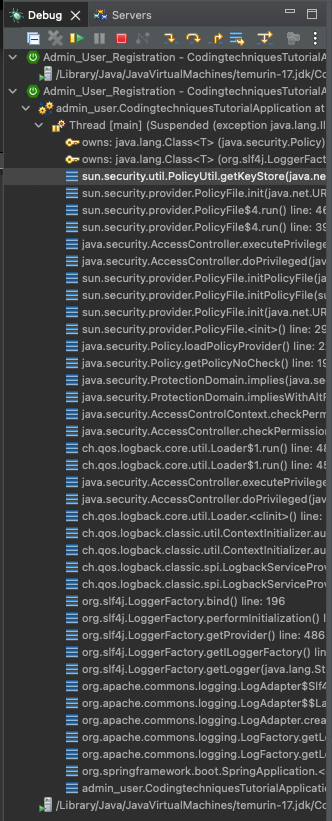


1. Develop View Search Results
2. Develop View Profile Page
3. Develop Edit Profile Functionality



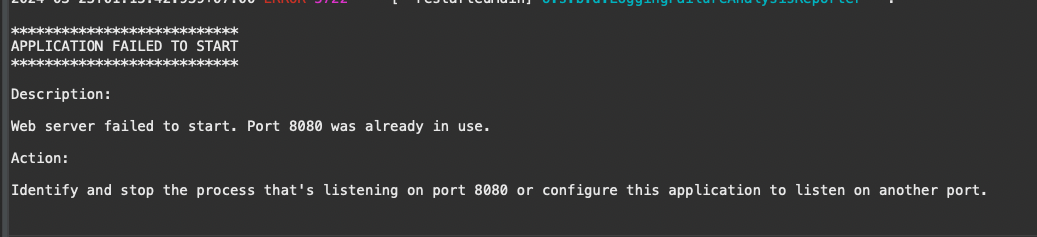
## Task 8 (Debugging)

1. Debug the application in Eclipse
2. Provide the screen capture of your application code in Eclipse Debug Mode include it as part of Project Presentation

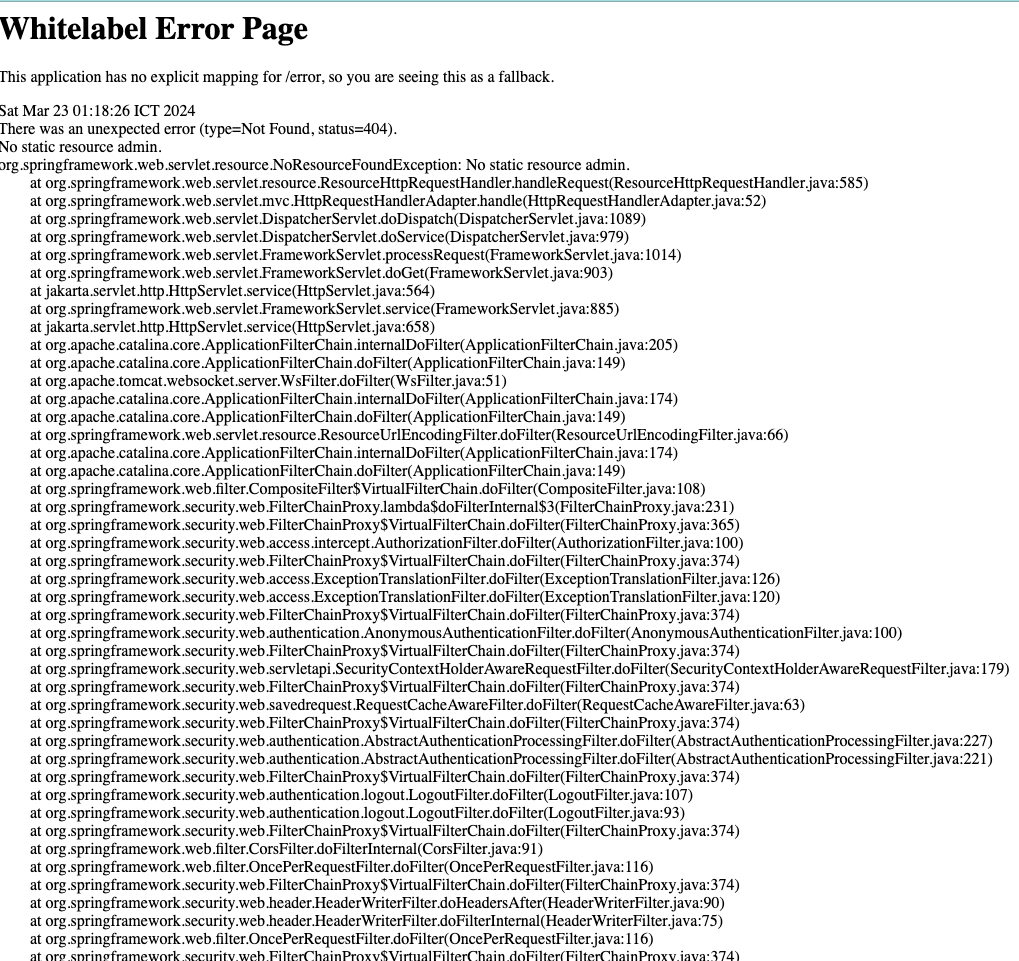


## Task 9 (Errors)

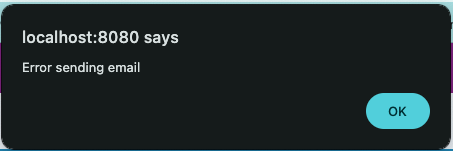
1. Provide screen capture of 3 Errors you encountered while developing the application.
2. How did you handle those 3 errors (Write in short bullet points)
3. Include it as part of Project Presentation
   1. Web server failed to start. Port 8080 was already in use.



* 1. No static resource admin.



* 1. Error sending email



**Solution:** Search google, use stackoverflow, ask the instructors directly: Teacher Lap, Teacher Giau, Teacher Tu

## Task 10 (Proposal of new feature)

1. **Functionality:**

User Interface Enhancement: The application interface will be updated to include a chat widget, prominently displayed for easy access.

Real-Time Communication: Users and customer support representatives can engage in real-time text-based communication within the application.

Chat Notifications: Customer support representatives will receive notifications for incoming chat requests, ensuring prompt responses to user queries.

Chat History: The chat system will maintain a history of chat conversations, allowing users and support representatives to refer back to previous discussions if needed.

1. **Development and Report:**

The development of the new chat support integration feature followed the proposed implementation plan outlined above. The feature was successfully implemented and integrated into the application, enhancing its functionality and user experience.

Progress Report:

Interface Design: The chat widget was designed to match the application's visual style and seamlessly blend into the user interface.

Backend Development: Backend logic was developed to handle incoming chat requests, manage chat sessions, and store chat history securely.

Real-Time Communication: WebSocket technology was employed to enable real-time messaging between users and support representatives, ensuring instant communication.

Notification System: A notification system was implemented to alert support representatives of incoming chat requests, ensuring timely responses and efficient customer support.

Chat History Storage: A database schema was designed and implemented to securely store chat transcripts, allowing for retrieval and review of past conversations.

## Task 11 (Testing & Documentation)

Provide Unit Testing code & screen capture of Eclipse IDE in debug mode.

1. Develop and execute unit testing for individual modules (At least 3 Cases)
2. Create Test Cases for the Individual Modules (At least 3 cases in total) and document it in Project Report
3. List the bugs and document in Project Report
4. Fix the Bugs as needed
5. Deploy to Lithan servers