E AUTHENTICATION USING OTP

SOFTWARE ENGINEERING AND CONCEPTS

DOCUMENTATION

Submitted by

DISHANTH T	**220701067**
GEETHA R	**220701073**
HAREESH S	**220701079**
HEMA S	**220701091**
JANANI J	**220701097**

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM

CHENNAI-602105

2023 - 24

TABLE OF CONTENTS

CHAPTER NO. **TITLE**

CHAPTER 1

PROJECT OVERVIEW

In the digital age, ensuring secure and reliable user authentication for online services is crucial. Many websites face challenges related to account security, user verification, and secure online transactions. Traditional username and password systems are increasingly vulnerable to attacks such as phishing, brute force, and credential stuffing. Furthermore, securing online transactions is essential to protect users from unauthorized access and fraud.

This project aims to develop an e-authentication system using One-Time Password (OTP) verification to enhance security during user registration, login, and performs online transactions with two factor authentication.

DATA PRESPECTIVES

1. **User Data:**
    - **Registration Information:** Users provide personal details such as name, email address, and password during registration.
    - **Email Verification Data:** An OTP is generated and sent to the user's email address to verify their identity.
    - **Login Information:** Users enter their username, password, and captcha response to log in.
    - **Transaction Data:** Details of online purchases including product details, transaction amount, and user details.
2. **System Data:**
    - **OTP Generation and Validation:** The system generates OTPs and verifies them during registration and transactions.
    - **Captcha System:** Implements captcha generation and validation to prevent automated login attempts.
    - **Transaction Logs:** Records of all transactions for monitoring and auditing purposes.

3. **Security Data:**
    - **Encryption and Storage:** Secure storage of user passwords and transaction details using encryption techniques.
    - **Activity Monitoring:** Logs and monitors user activities to detect suspicious behavior and potential security breaches.

USER BENIFITS

1. **Enhanced Security:**
    - **Two-Factor Authentication:** The use of OTP for both registration and transaction verification provides an additional layer of security, making it harder for unauthorized users to gain access.
    - **Captcha Protection:** Captcha implementation during login prevents automated attacks and ensures that only genuine users can access the system.
2. **User Convenience:**
    - **Email Verification:** Users can easily verify their identity via email, which is a familiar and straightforward process.
    - **Secure Transactions:** OTP-based transaction verification ensures that users can confidently make online purchases without the fear of unauthorized access.
3. **Improved Trust:**
    - **Data Protection:** With secure storage and encryption of sensitive data, users can trust that their personal information is safe.
    - **Audit Trails:** Comprehensive transaction logs provide transparency and can be used to resolve disputes or investigate suspicious activities.

CHAPTER 2

REQUIREMENTS AS USER STORIES

Sign Up :

As a new user,

I want to sign up with my details

so that I can access the system securely.

Acceptance Criteria:

- User can input their details including username, email, and password.

- Password should meet minimum security requirements.

- Upon successful registration, user data is stored securely in the database.

Poker Estimation: 3

Receive Confirmation Email :

As a new user,

I want to receive a confirmation email after registration

so that I can verify my email address.

Acceptance Criteria:

- After registration, a confirmation email is sent to the user's provided email address.

- The email contains a link or OTP for email verification.

- User's email address is marked as unverified until they confirm.

Poker Estimation: 2

Secure Login with Captcha:

As a registered user,

I want to log in securely with my username, password, and captcha

So that it is more secure.

Acceptance Criteria:

- User can log in with their username, password, and captcha.

- Captcha is generated securely and refreshed after a certain time or after each use.

- Failed login attempts are limited to prevent brute force attacks.

Poker Estimation: 5

Select Product:

As a user,

I want to select a product

so that I can proceed with the transaction.

Acceptance Criteria:

- User can browse through the list of available products.

- User can select a product they wish to purchase.

Poker Estimation: 1

View Product List with Costs:

As a user,

I want to a list of available products with their respective costs

so that it is easy to purchase.

Acceptance Criteria:

- User can view a list of available products with their respective costs.

- Product list is displayed clearly and comprehensively.

- Prices are accurate and updated regularly.

Poker Estimation: 2

Proceed to Secure Payment:

As a user ready to make a purchase,

I want to proceed to pay securely for the selected product

so that I can avoid any data breach.

Acceptance Criteria:

- User can proceed to pay securely for the selected product.
- Payment process is encrypted and complies with industry standards.
- Payment gateway redirects users securely to complete the transaction.

Poker Estimation: 4

Password Authentication:

As a user,

I want to enter my password for authentication before completing the transaction

so that it is more secure.

Acceptance Criteria:

- User must enter their password before completing the transaction.
- Password is securely hashed and validated.
- Password strength requirements are enforced.

Poker Estimation: 3

Receive OTP for Verification:

As a user,

I want to receive an OTP for verification during online transactions

so that I can have dual security.

Acceptance Criteria:

- User receives an OTP for verification during online transactions.
- OTP is sent to the user's registered email or mobile number.
- OTP is valid for a limited time and single-use only.

Poker Estimation: 2

Verify OTP:

As a user,

I want to verify the OTP

so that I can ensure secure completion of the transaction.

Acceptance Criteria:

- User can enter the OTP received and verify it.

- OTP verification process is secure and reliable.

- Upon successful verification, the transaction can proceed.

Poker Estimation: 2

Redirect to Google Pay:

As a user,

I want to be redirected to an online payment application like Google Pay

so that I can complete my transaction.

Acceptance Criteria:

- User is redirected to Google Pay for secure transaction processing.

- Google Pay integration is seamless and user-friendly.

- Transaction confirmation is displayed upon completion.

Poker Estimation: 3

Non-Functional Requirements:

1. **Security:**
    - The system must use strong encryption methods (e.g., AES-256) for storing passwords and sensitive data.
    - All OTPs must be randomly generated and have a short expiry time (e.g., 5 minutes) to enhance security.

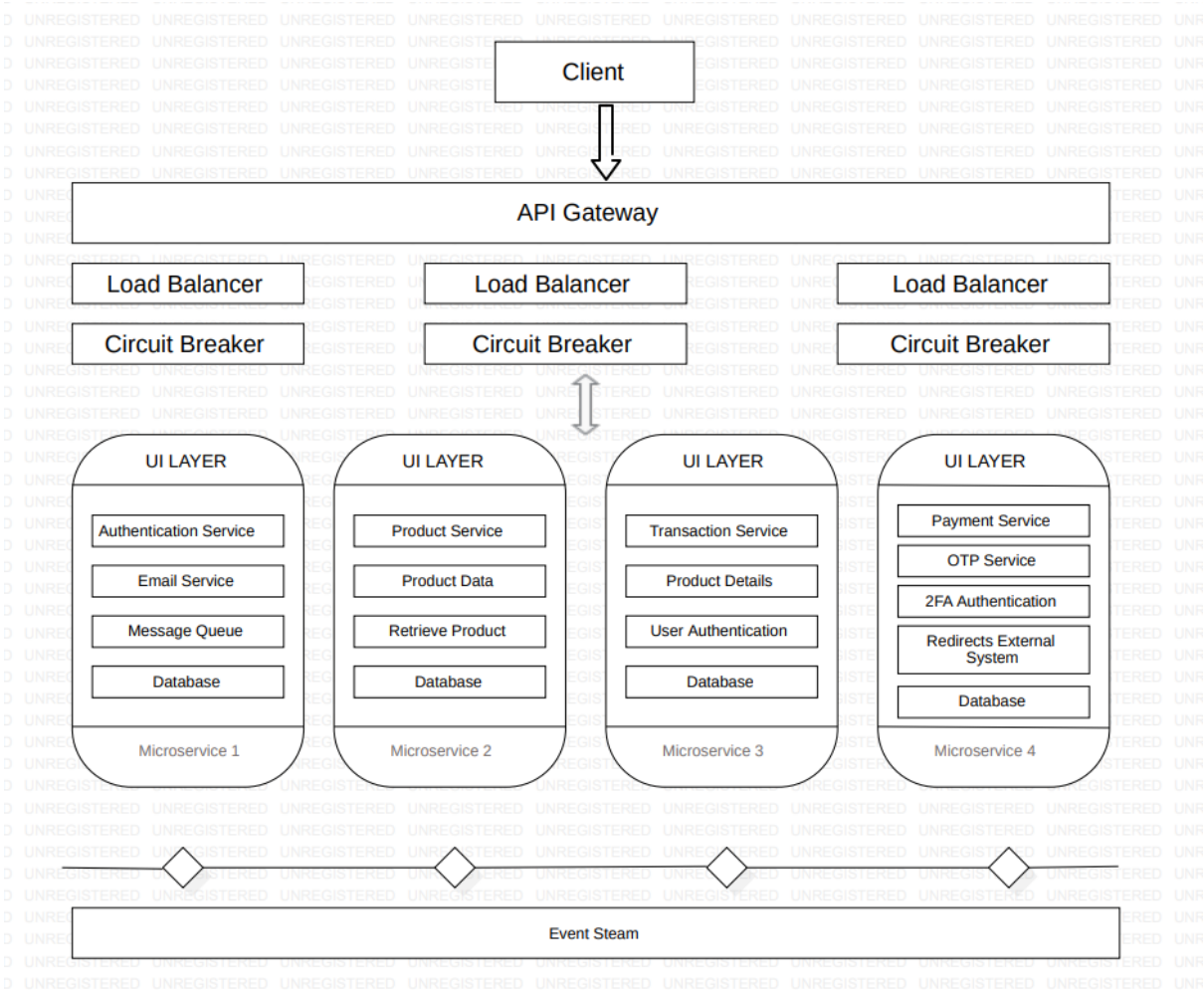
2. **Performance:**
    - The system should send the OTP email within 10 seconds of a registration or transaction request.
    - The website must handle up to 500 concurrent users without significant performance degradation.
3. **Usability:**
    - The user interface should be intuitive and user-friendly, requiring minimal user training or instruction.
    - The captcha should be easy to solve for humans but difficult for automated bots, with a success rate of at least 90% for human users.

CHAPTER 3

ARCHITECTURE DIAGRAM

MICROSERVICE ARCHITECTURE

Microservice architecture is a design pattern that structures an application as a collection of small, autonomous services modelled around a business domain. Each service is self-contained and implements a single business capability. These services can be developed, deployed, and scaled independently, allowing for greater flexibility and agility in software development.



API Gateway

- Role: Serves as a single entry point for all client requests.
- Responsibilities:

- Routes requests from the UI layers to the appropriate microservices.

Microservices

1. **Authentication Service**:
    - **Role**: Manages user authentication and authorization.
    - **Responsibilities**:
        - Handles user registration and login processes.
        - Verifies CAPTCHA to prevent automated abuse.
        - Authenticates users and generates authentication tokens.
        - Interacts with the UserDB to store and retrieve user credentials and details.
        - Sends confirmation emails and OTPs via the Email Service.
2. **Email Service**:
    - **Role**: Manages sending emails for various purposes.
    - **Responsibilities**:
        - Sends confirmation emails after user registration.
        - Sends OTP emails for additional security during transactions.
        - Utilizes a message queue for asynchronous email processing.
3. **Product Service**:
    - **Role**: Manages the product catalog.
    - **Responsibilities**:
        - Lists available products and their details.
        - Retrieves product information from the ProductDB.
        - Ensures users have up-to-date product data for their purchasing decisions.
4. **Transaction Service**:
    - **Role**: Manages user transactions and order processing.
    - **Responsibilities**:

- Creates and updates transactions as users make purchases.
- Verifies user authentication before processing transactions.
- Interacts with the Product Service to get product details.
- Stores transaction data in the TransactionDB.
- Uses a message queue to process transaction messages asynchronously.

5. **Payment Service**:
    - **Role**: Handles the payment process securely.
    - **Responsibilities**:
        - Processes payments and ensures secure handling of payment information.
        - Generates and verifies OTPs via the OTP Service for added transaction security.
        - Redirects users to external payment gateways like Google Pay to complete transactions.
        - Ensures secure payment data transmission and compliance with payment standards.
6. **OTP Service**:
    - **Role**: Provides OTP generation and verification.
    - **Responsibilities**:
        - Generates OTPs for user transactions to ensure dual authentication.
        - Verifies OTPs provided by users during transactions to enhance security.

External Systems

- **Google Pay**:
    - **Role**: Acts as an external payment application.
    - **Responsibilities**:

- Completes transactions initiated by users within the system.
- Ensures secure handling and processing of payment data.

Databases

1. **UserDB (User Database)**:
    - **Role**: Stores user information and credentials.
    - **Responsibilities**:
        - Saves user registration data and authentication credentials.
        - Provides secure storage and retrieval of user information for the Authentication Service.
2. **ProductDB (Product Database)**:
    - **Role**: Stores product information.
    - **Responsibilities**:
        - Contains details about products available for purchase.
        - Allows the Product Service to retrieve and manage product data.
3. **TransactionDB (Transaction Database)**:
    - **Role**: Stores transaction information.
    - **Responsibilities**:
        - Saves data related to user transactions and purchases.
        - Allows the Transaction Service to store and retrieve transaction details.

Message Queue

- **Role**: Manages asynchronous communication between services.
- **Responsibilities**:
    - Handles the processing of email messages sent by the Email Service.
    - Processes transaction-related messages sent by the Transaction Service.
    - Ensures decoupling of services and smooth handling of background tasks.

ARCHITECTURE PATTERN USED :

Microservices Architecture: An architectural style that structures an application as a collection of loosely coupled, independently deployable services, each serving a specific business function.

1. **Modular Design:**
    - Breaks down the application into distinct services (e.g., Authentication, Product, Transaction, Payment) that match the project requirements, enhancing clarity and focus.
2. **Independent Development:**
    - Each microservice can be developed and maintained independently, allowing different teams to work on different services simultaneously without interfering with each other.
3. **Enhanced Security:**
    - Isolates sensitive operations (e.g., Payment Service, OTP Service) into separate services, reducing the risk of a security breach affecting the entire system.
4. **Improved Fault Isolation:**
    - If one microservice fails (e.g., Email Service), it does not bring down the entire application, ensuring better overall system reliability and availability.

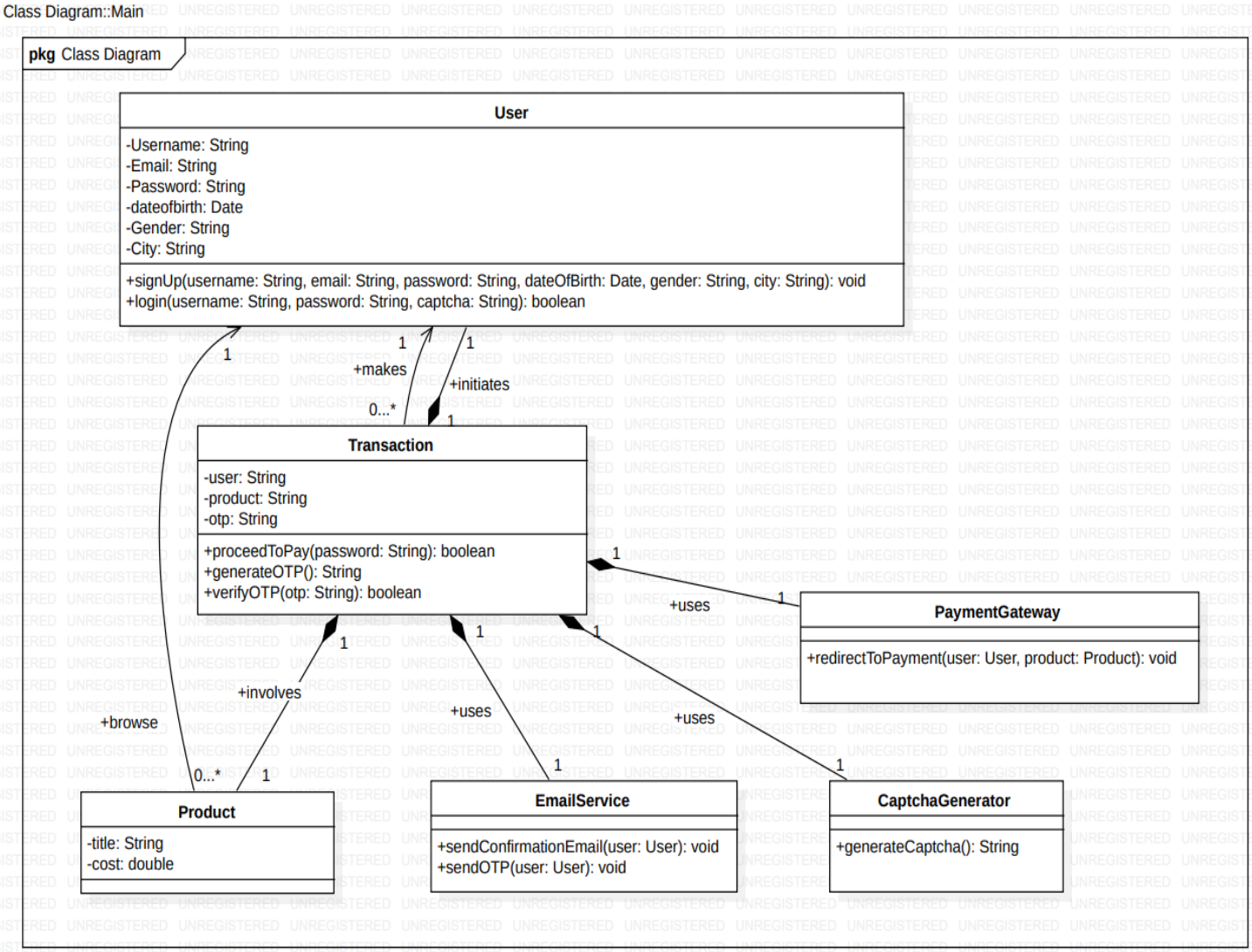
DESIGN PATTERN

The design pattern used here is Service-Oriented Architecture (SOA):

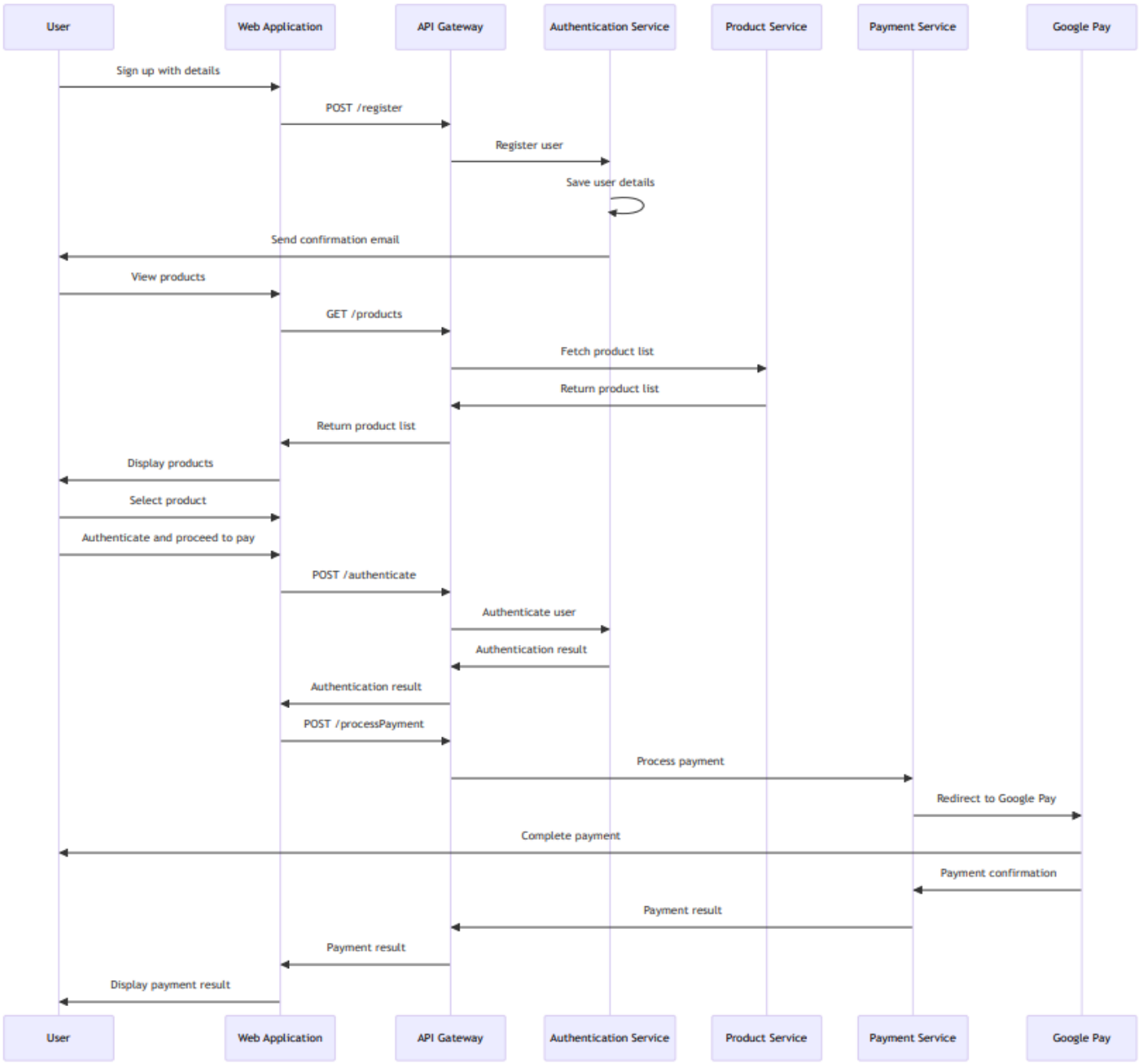
- SOA divides the application into smaller, loosely coupled services that communicate through well-defined interfaces (API Gateway).

- Each service (e.g., Authentication, ProductService) performs a specific business function and can be developed, deployed, and scaled independently, promoting modularity and flexibility.

CLASS DIAGRAM

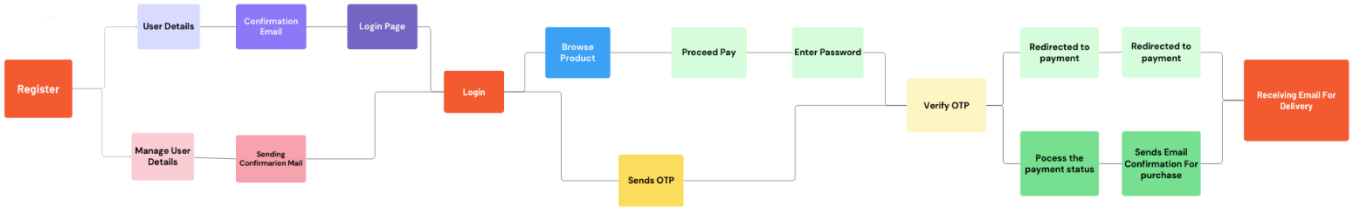


SEQUENCE DIAGRAM



CHAPTER 4

BUSINESS ARCHITECTURE DIAGRAM



Objective:

- Develop safe login and secure payment systems to ensure user trust and regulatory compliance.

Components:

- System includes front-end interfaces, back-end servers, databases, and payment gateways.
- Incorporates robust authentication and encryption mechanisms.

Risk Mitigation:

- Implement strategies to mitigate unauthorized access, data breaches, and fraud.
- Conduct regular security audits and employ fraud detection measures.

- **Scalability:**

  - Ensure scalability to accommodate growth and maintain performance under increased user traffic.
  - Design system with flexibility to adapt to evolving security standards and regulat

CHAPTER 5

TEST STRATEGY

1. User Story: Secure Login

Test Plan: Verify that registered users can log in securely using their username, password, and captcha.

Test Cases:

- **Happy Path:**
    1. **Test Case:** User enters valid username, password, and solves the captcha correctly.
        - **Expected Result:** User is successfully logged in.
- **Error Scenarios:**
    1. **Test Case:** User enters an invalid username.
        - **Expected Result:** System displays an error message indicating incorrect username.
    2. **Test Case:** User enters an incorrect password.
        - **Expected Result:** System displays an error message indicating incorrect password.
    3. **Test Case:** User fails to solve the captcha correctly.
        - **Expected Result:** System displays an error message indicating incorrect captcha.
    4. **Test Case:** User exceeds the maximum number of failed login attempts.
        - **Expected Result:** System locks the account and sends an email with instructions to unlock.

2. User Story: Select Product

Test Plan: Verify that users can select a product to proceed with the transaction.

Test Cases:

- **Happy Path:**
    1. **Test Case:** User browses the product catalog and selects a product.

        - **Expected Result:** Product details are displayed, and the user can proceed to the transaction.
- **Error Scenarios:**
    1. **Test Case:** User selects an out-of-stock product.
        - **Expected Result:** System displays a message indicating the product is out of stock.

3. User Story: Secure Payment Process

Test Plan: Verify that users can proceed to pay securely for the selected product.

Test Cases:

- **Happy Path:**
    1. **Test Case:** User selects a product and proceeds to the payment page.
        - **Expected Result:** User is redirected to a secure payment gateway.
- **Error Scenarios:**
    1. **Test Case:** User attempts to proceed to payment without selecting a product.
        - **Expected Result:** System prevents the action and displays an error message.
    2. **Test Case:** User encounters a network error during the payment process.
        - **Expected Result:** System displays a message indicating a network error and prompts the user to retry.

4. User Story: Authentication Before Completing Transaction

Test Plan: Verify that users are required to enter their password for authentication before completing the transaction.

Test Cases:

- **Happy Path:**
    1. **Test Case:** User proceeds to the payment page and is prompted to enter their password.
        - **Expected Result:** User enters the correct password and can proceed with the transaction.
- **Error Scenarios:**

1. **Test Case:** User enters an incorrect password.
    - **Expected Result:** System displays an error message indicating incorrect password.
2. **Test Case:** User attempts to proceed without entering a password.
    - **Expected Result:** System prevents the action and displays an error message.

5. User Story: OTP for Transaction Verification

Test Plan: Verify that users receive an OTP for verification during online transactions.

Test Cases:

- **Happy Path:**
    1. **Test Case:** User proceeds with the payment, and the system sends an OTP to the registered email.
        - **Expected Result:** User receives the OTP within the expected time frame.
- **Error Scenarios:**
    1. **Test Case:** User does not receive the OTP due to an email delivery failure.
        - **Expected Result:** System provides an option to resend the OTP after a specific time.
    2. **Test Case:** User receives the OTP after expiry time.
        - **Expected Result:** System indicates that the OTP has expired and prompts the user to request a new OTP.

6. User Story: OTP Verification

Test Plan: Verify that users can verify the OTP to ensure secure completion of the transaction.

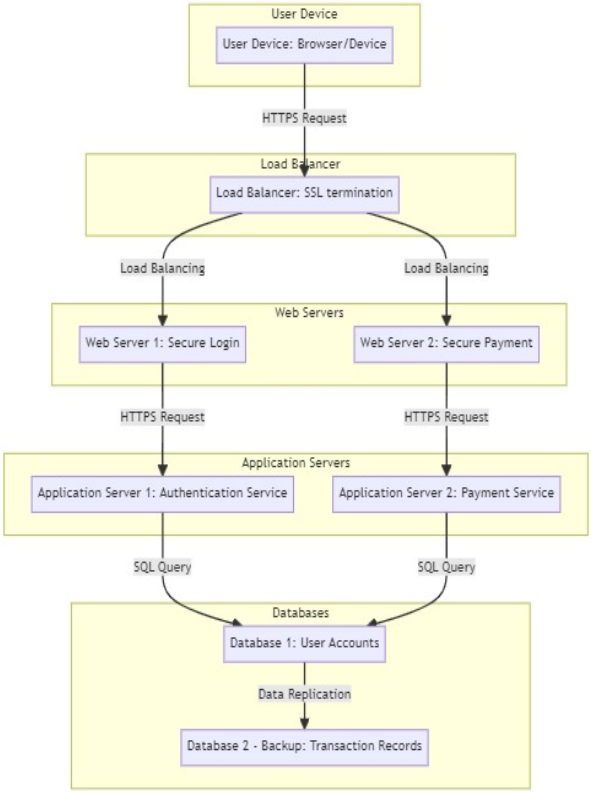
Test Cases:

- **Happy Path:**
    1. **Test Case:** User enters the correct OTP within the expiry time.

        - **Expected Result:** OTP is accepted, and the transaction is completed successfully.
- **Error Scenarios:**
    1. **Test Case:** User enters an incorrect OTP.
        - **Expected Result:** System displays an error message indicating incorrect OTP.
    2. **Test Case:** User enters the OTP after it has expired.
        - **Expected Result:** System displays an error message indicating OTP has expired and prompts the user to request a new OTP.
    3. **Test Case:** User makes multiple incorrect OTP attempts.
        - **Expected Result:** System temporarily locks the transaction process and sends an alert to the user.

CHAPTER 6

DEPLOYMENT DIAGRAM



Secure Communication: HTTPS ensures encrypted communication between user devices and the application.

Load Balancing: The load balancer evenly distributes traffic across multiple web servers for optimal performance.

Specialized Services: Application servers host authentication and payment processing services separately for efficiency.

Data Security: Databases store user accounts and transaction records securely, with a backup for redundancy.

Scalability and Reliability: The architecture scales easily and maintains service availability with redundant components and load balancing.

CHAPTER 7

CONCLUSION

The e-authentication system using OTP significantly enhances the security and user experience of online transactions. By integrating multi-factor authentication methods, including email-based OTP and captcha, the system effectively mitigates the risks associated with unauthorized access and fraud. This project successfully addresses the prevalent security challenges encountered by online platforms, ensuring that users can register, log in, and conduct transactions with a high level of security. Key accomplishments include improved security through robust protections against automated attacks and unauthorized access, user convenience via a straightforward and user-friendly process, robust data protection with encryption and secure data handling, and adherence to relevant data protection regulations, maintaining user trust and compliance. Overall, the implementation of this system demonstrates a significant step forward in providing a secure, reliable, and user-friendly online transaction experience.