E-Commerce System – Project Plan

Project Idea & Background

Project Idea:
We will develop an e-commerce system where users can browse products, add items to a shopping cart, and complete purchases. The system will also include an admin panel for managing products and orders.

Background & Purpose:

- E-commerce is one of the fastest-growing markets, with most businesses from big to small being able to set up an online store.
- Many small businesses need a user-friendly platform to sell products online.
- We aim to create a system that is easy to use, secure, and efficient for both customers and administrators.

Vision & Goals

Vision:
To create a modern E-commerce platform that provides a fluent and easy to use shopping experience and efficient order management.

Goals:

- Users should be able to create accounts, log in, and complete purchases.
- Administrators should be able to manage products, orders, and users.
- The system should be responsive and work on both mobile and desktop.

Target Audience and Stakeholder

Target Audience

The target audience includes businesses and individuals looking to sell products online. This could include:

- **Small to Medium-sized Businesses (SMBs):** Local stores, niche brands, and startups that need an affordable e-commerce solution.

- **Entrepreneurs & Independent Sellers:** Individuals who want to sell handmade products, digital goods, or customized items.
- **Retail Companies:** Established retailers wanting to expand their online presence.

Stakeholder	Role
Customer (End User)	Buys products, creates an account, places order
Administrator	Manages products, order, and users
System Developer	Builds and maintains the system
Finance Department	Handles payments and invoicing

Requirements Specification

Functional Requirements:

1. Users should be able to register and log in.
2. Users should be able to add products to a shopping cart.
3. Users should be able to complete purchases through a payment gateway.
4. Administrators should be able to add, edit, and delete products.
5. The system should send order confirmation emails.
6. Users should be able to view past orders in an order history.
7. The system should have a product search function.

Non-Functional Requirements:

1. **Security** – All user data must be encrypted, and the system should use secure authentication methods.
2. **Performance** – The system should be able to handle at least 1,000 concurrent users.
3. **Usability** – The design should be responsive and intuitive.
4. **Availability** – The system should have 99.9% uptime.
5. **Scalability** – The system should be easy to expand with additional features in the future.

Prioritization (MoSCoW Method):

- **MUST:** User authentication, product browsing, shopping cart, checkout, order management.
- **SHOULD:** Order history, email confirmations, search functionality.
- **COULD:** Product reviews, discount codes, additional payment options.

User Stories & Use Cases

User Stories:

- **As a customer, I want to create an account** so that I can save my orders.
- **As a customer, I want to add products to a shopping cart** so that I can purchase multiple items at once.
- **As an administrator, I want to add new products** so that I can update the store's inventory.

Use Cases:

Use Case 1: User Places an Order

Actors: Customer, System, Payment Gateway

Preconditions:

- The user is registered and logged in.
- The shopping cart contains at least one item.

Main Flow:

1. The user proceeds to checkout.
2. The system displays order summary and payment options.
3. The user enters payment details and confirms the purchase.
4. The system validates the payment with the payment gateway.
5. If payment is successful, the system confirms the order and updates stock.
6. The system sends an order confirmation email to the user.

Alternative Flows:

- **Invalid Payment:** If the payment fails, the system prompts the user to retry or choose another method.
- **Stock Issue:** If an item is out of stock during checkout, the system notifies the user and updates the cart.

Postconditions:

- The order is successfully placed and stored in the database.
- The payment is processed, and the user receives confirmation.

Use Case 2: User Adds a Product to the Shopping Cart

Actors: Customer, System

Preconditions:

- The user is browsing the product catalog.

Main Flow:

1. The user selects a product and clicks "Add to Cart."
2. The system verifies if the product is in stock.
3. The system adds the product to the user's shopping cart.
4. The system updates the cart total.

Alternative Flows:

- **Out of Stock:** If the product is unavailable, the system displays an error message.

Postconditions:

- The selected product is added to the cart.
- The cart total is updated.

Use Case 3: Administrator Manages Products

Actors: Administrator, System

Preconditions:

- The administrator is logged into the system.

Main Flow:

1. The administrator navigates to the product management section.
2. The system displays the list of available products.
3. The administrator selects an option: add, edit, or remove a product.
4. The system validates the input and updates the database.

Alternative Flows:

- **Invalid Input:** If required fields are missing or incorrect, the system prompts for correction.

Postconditions:

- The product list is updated accordingly.

Project Plan & Structure

Timeline (Sprints – 4 Weeks)

Week	Focus Area
1	Requirement gathering, UML diagram design
2	Back-end: Database model and API design
3	Front-end: UI/UX and integration
4	Testing, documentation, final review

Work Process:

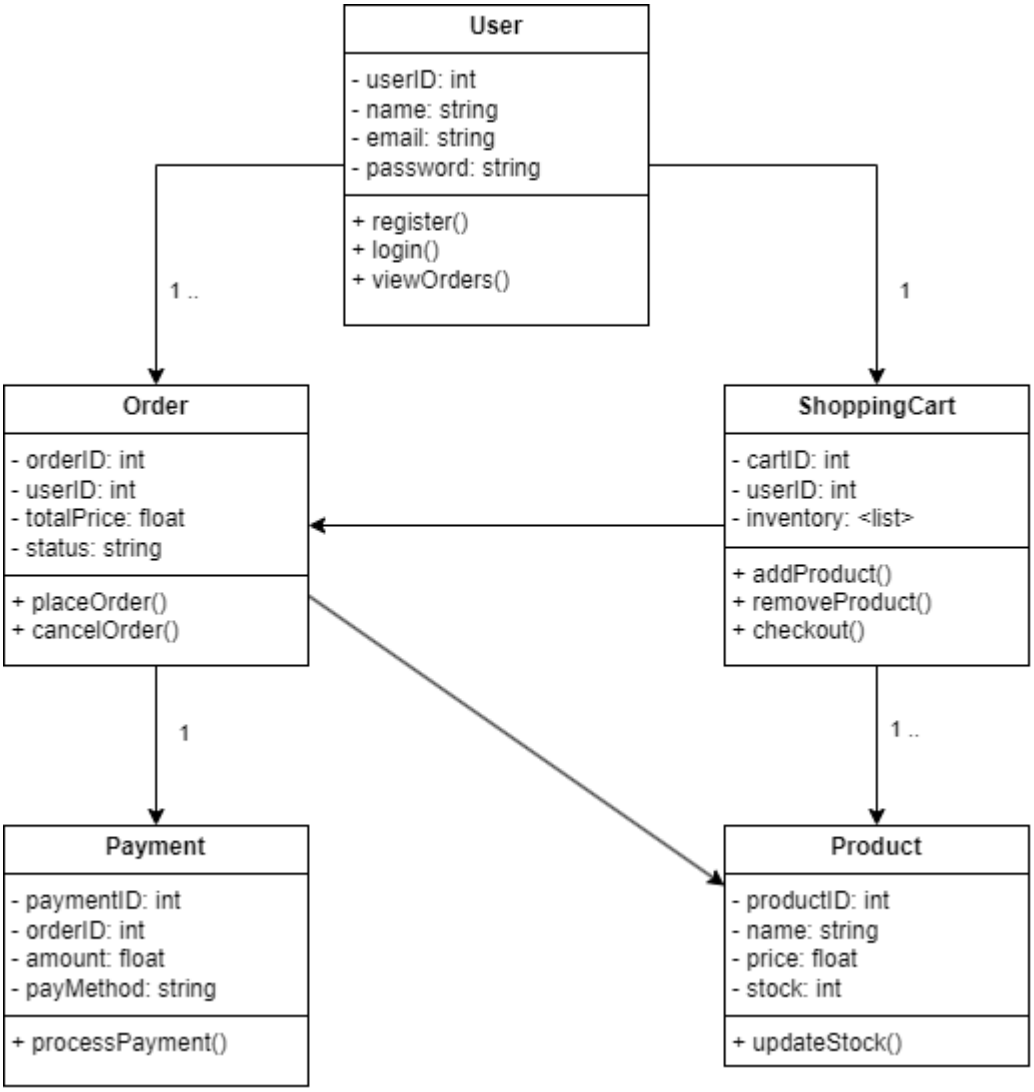
- Scrum-based approach with daily stand-up meetings.
- Tasks managed using **Jira or GitHub Projects**.

Change Management:

- Backlog for new requirements.
- Change requests handled through GitHub Issues.
- Regular meetings to prioritize updates.

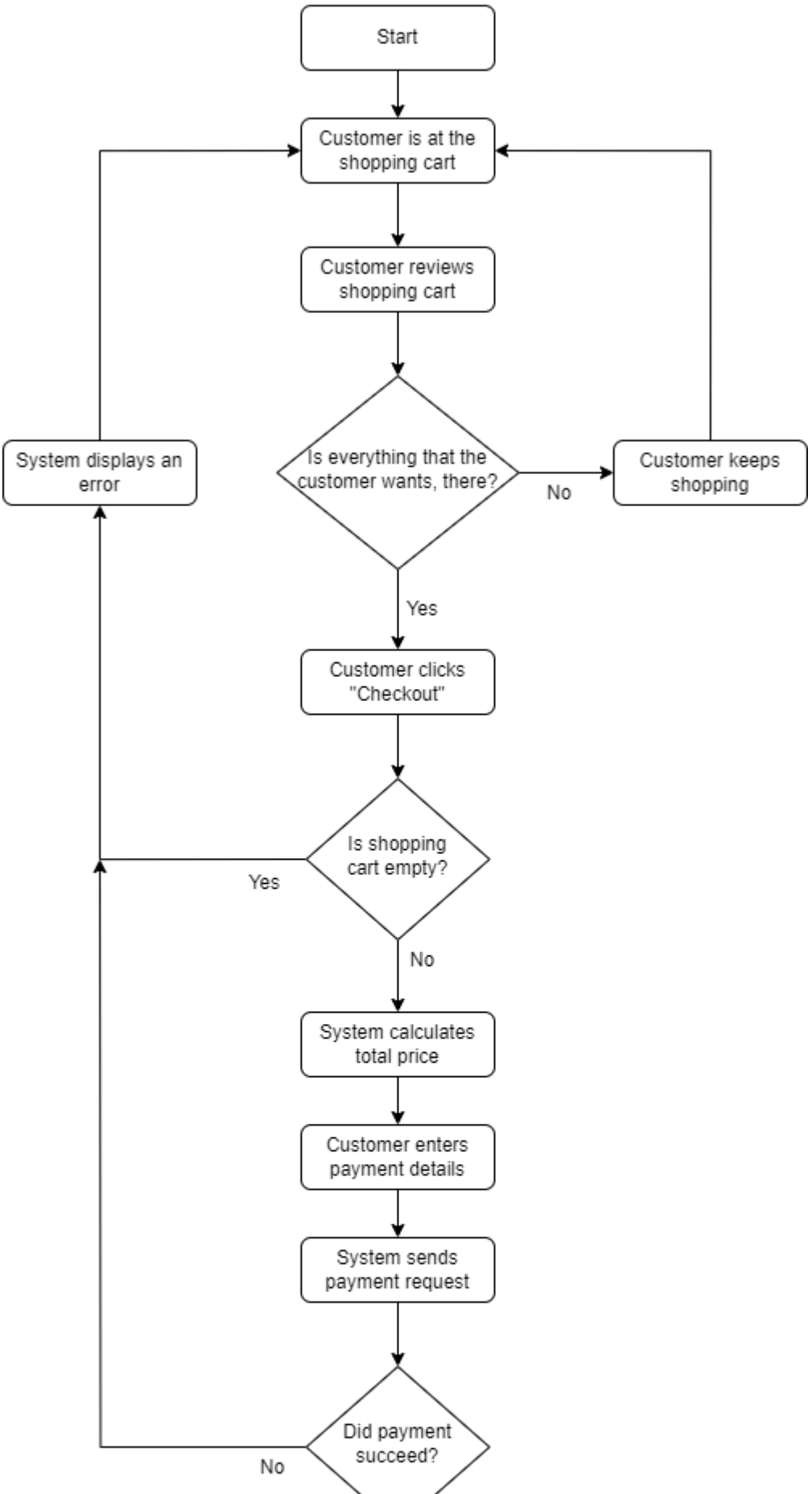
Diagrams

UML Diagram

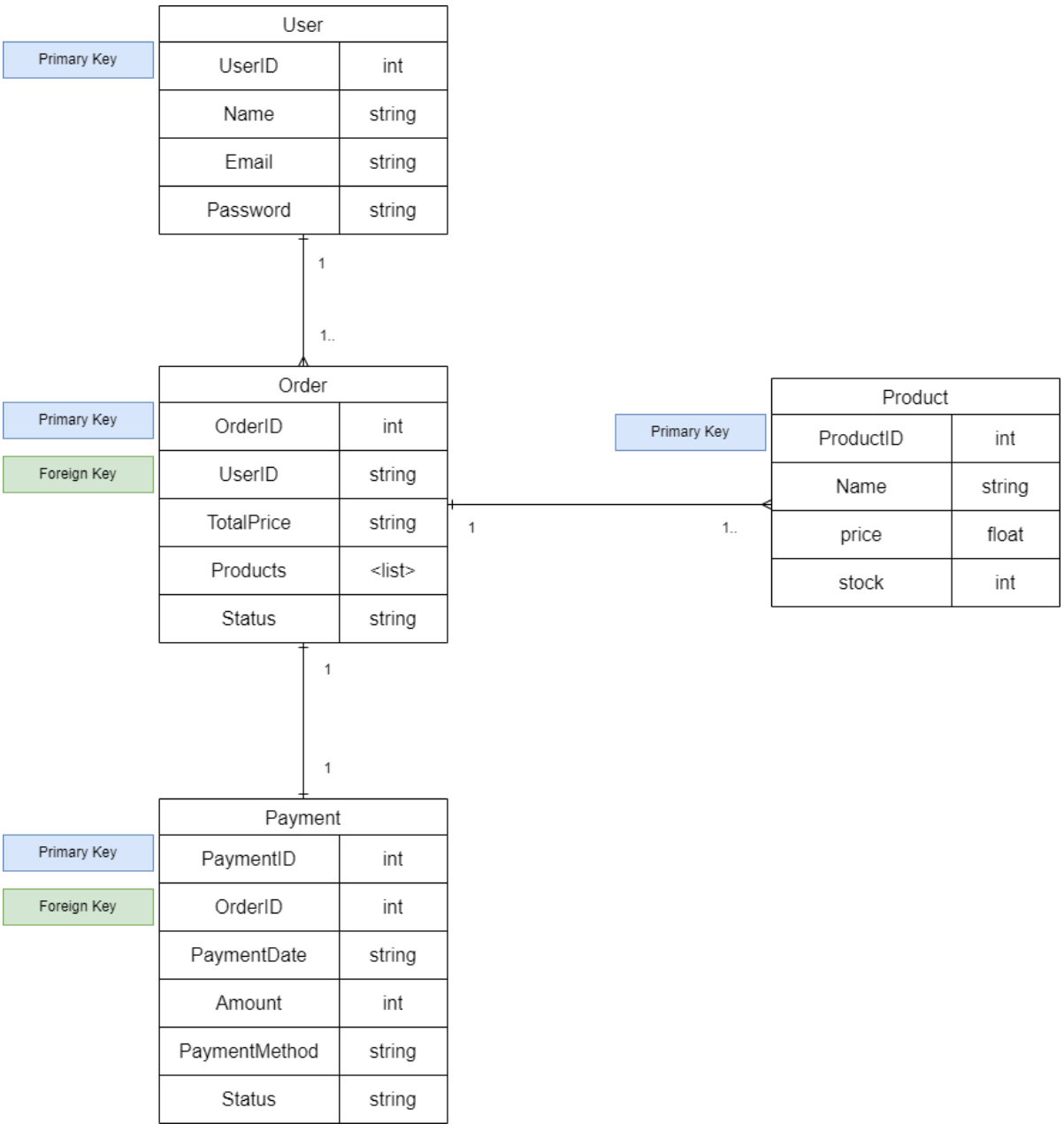


Activity Diagram

User creates an order
Activity Diagram



ER Diagram


User
Primary Key
UserID
int
Name
string
Email
string
Password
string
1
1..
Order
Primary Key
OrderID
int
Foreign Key
UserID
string
TotalPrice
string
Products
<list>
Status
string
1
1..
Product
Primary Key
ProductID
int
Name
string
price
float
stock
int
1
1
Payment
Primary Key
PaymentID
int
Foreign Key
OrderID
int
PaymentDate
string
Amount
int
PaymentMethod
string
Status
string