Ain Shams Univeristy

Computer & Systems Engineering; Spring 2025

CSE352s: Parallel and Distributed Systems

AMAZONAWY E-COMMERCE WEB APP
Final Project Report

2001118	Karim Ibrahim Saad Abd-Elrazek
2001436	Mahmoud Abdelraouf Mahmoud Abdelaal
2001307	Yassa Sfen Ayed Helmy
2002081	Omar Salah Mansour Abdelkader
2000252	Eslam Mohamed Marzouk Abdelaziz
2001184	Mohamed Adham Mohamed Nagib
2100427	Sarah Tarek Mohamed Bishry Abdelhakeem
2001273	Mariam Ahmed Abd Elgalil
2001378	Adham Mohamed Mohamed Hassan
2002011	Waleed Khalid Mohamed Waleed
2000427	Sarah Sherif Mohamed Abd Elhady
2002139	Abdelrahman Elsayed Ahmed Mohamed

Under Supervision of :

Prof. Gamal Ibrahim

Eng. Yasmeen Mostafa

Contents

List of Figures

List of Tables

AMAZONAWY

I Project Links

https://github.com/mohamedAdhamc/store

VIDEO Link

II Introduction

AMAZONAWY is a dynamic e-commerce platform designed as a specialized online marketplace where users can efficiently purchase and sell a diverse array of technology products. To facilitate smooth transactions, it offers a variety of secure online payment options. The platform's robust backend infrastructure is constructed using the **`Node.js`** framework, ensuring reliability and scalability.

- AMAZONAWY provides a comprehensive suite of features designed to enhance the user experience for both buyers and sellers:
  1. User Management:
     - Secure account creation and login functionalities.
     - Support for distinct user roles (e.g., buyers, sellers, administrators).
  2. Product & Data Management:
     - Tools for users to add, edit, and remove shared data, primarily focusing on product listings.
     - Sophisticated inventory management capabilities for sellers.
  3. Transaction & Financials:
     - Integrated online payment system, allowing users to choose from multiple methods, including credit card and e-wallet options.

    - Personalized dashboards displaying account information such as current cash balance, lists of purchased items, sold items, and items pending sale.
  4. Search & Discovery:
     - Advanced multi-option search functionalities to enable users to easily find specific products.
  5. Reporting & Analytics:
     - Generation of various reports, including detailed summaries of transactions performed on the system and sales analytics.
  6. Technical Architecture:
     - Utilization of REST APIs to facilitate seamless communication between the frontend client and backend services.
     - System architecture designed to efficiently manage concurrent user sessions.

- In essence, AMAZONAWY serves as a modern hub where technology enthusiasts and sellers converge, benefiting from a streamlined shopping experience augmented by contemporary features.

- Welcome to AMAZONAWY, where technology meets convenience and robust functionality.

III Target Beneficiaries of the Project

- The AMAZONAWY project is designed to benefit a wide range of users. Key groups include:
  1. `Tech Enthusiasts`: Individuals who are passionate about technology and gadgets, and are looking for a convenient online platform to make purchases.
  2. `Technology Sellers`: This includes both established businesses and individual entrepreneurs who wish to sell technology products online. AMAZONAWY offers them a way to reach a larger customer base.
  3. `General Online Shoppers`: Anyone who enjoys the convenience of online shopping can use AMAZONAWY to easily browse, find, and buy technology-related items.
  4. `Small Technology Businesses`: Smaller companies in the tech sector can leverage AMAZONAWY to market and sell their

products without the significant investment required for building and maintaining their own dedicated e-commerce website.

5. **`Freelancers and Startup Ventures`**: Individuals or new companies creating or reselling tech items can use AMAZONAWY as an accessible marketplace to offer their products and generate revenue.
6. **`Technology Communities and Groups`**: Clubs, educational institutions, or online communities focused on technology can find AMAZONAWY to be a useful resource to recommend for sourcing tech products.
7. **`Students of Technology`**: Individuals learning about web development or software engineering may find the AMAZONAWY project itself a practical example to understand how e-commerce platforms are structured and function.

- In essence, AMAZONAWY strives to bring these diverse groups together, simplifying and enhancing the experience of buying and selling technology products for everyone involved.

IV Adopted Programming Language

1. **`Node JS`**
2. **`JavaScript`**

V System Architecture

- Our project uses a 3-Tier Client/Server Model, valued for its strengths in scalability, modularity, and maintainability.
- In this setup, the client tier handles what the user sees and interacts with. The server tier manages the application's core logic and processing, while the database tier is responsible for storing and accessing data.
- This clear division of tasks makes it easier to develop, deploy, and update the system later on.

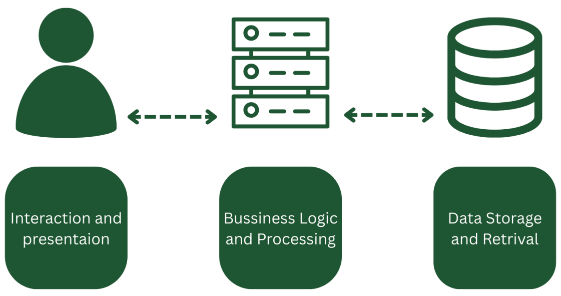

Interaction and presentaion
Bussiness Logic and Processing
Data Storage and Retrival

Figure 1: System Architecture Diagram

VI Application-Level Protocol "ALP"

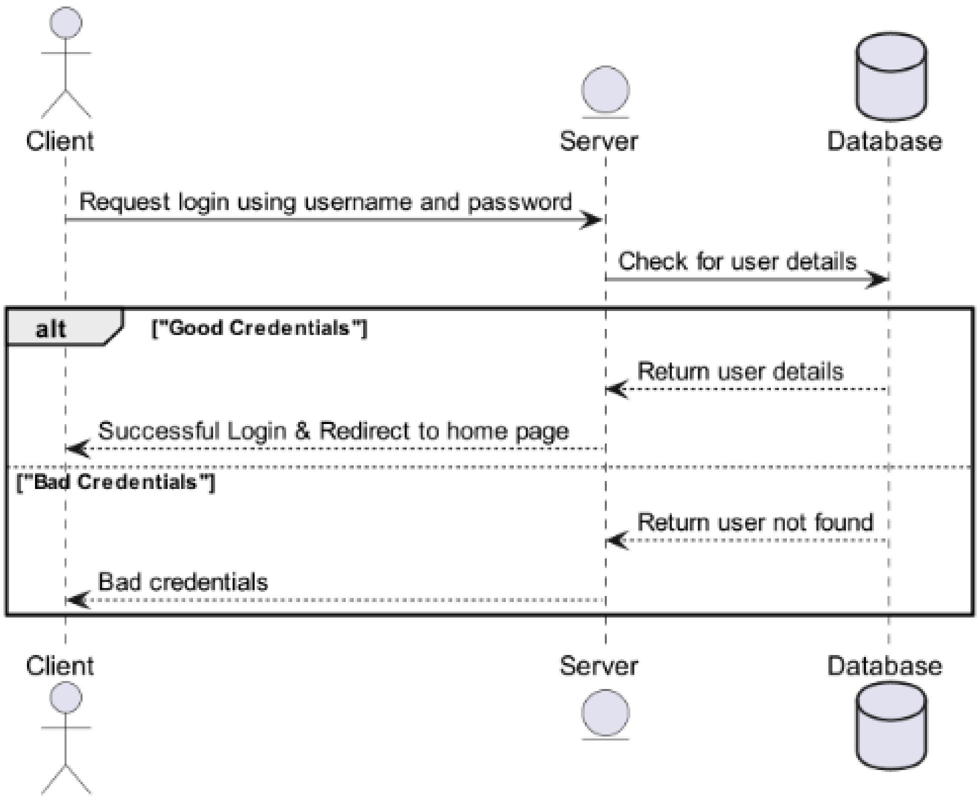


Figure 2: Login ALP Diagram

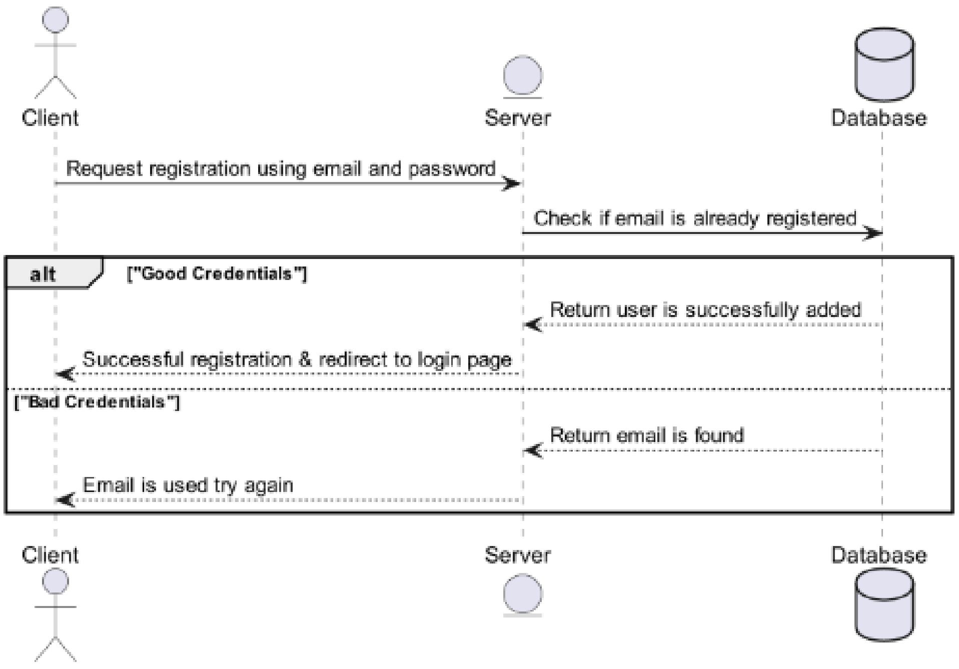

Client
Server
Database
Request registration using email and password
Check if email is already registered
alt
["Good Credentials"]
Return user is successfully added
Successful registration & redirect to login page
["Bad Credentials"]
Return email is found
Email is used try again
Client
Server
Database

Figure 3: Registration ALP Diagram

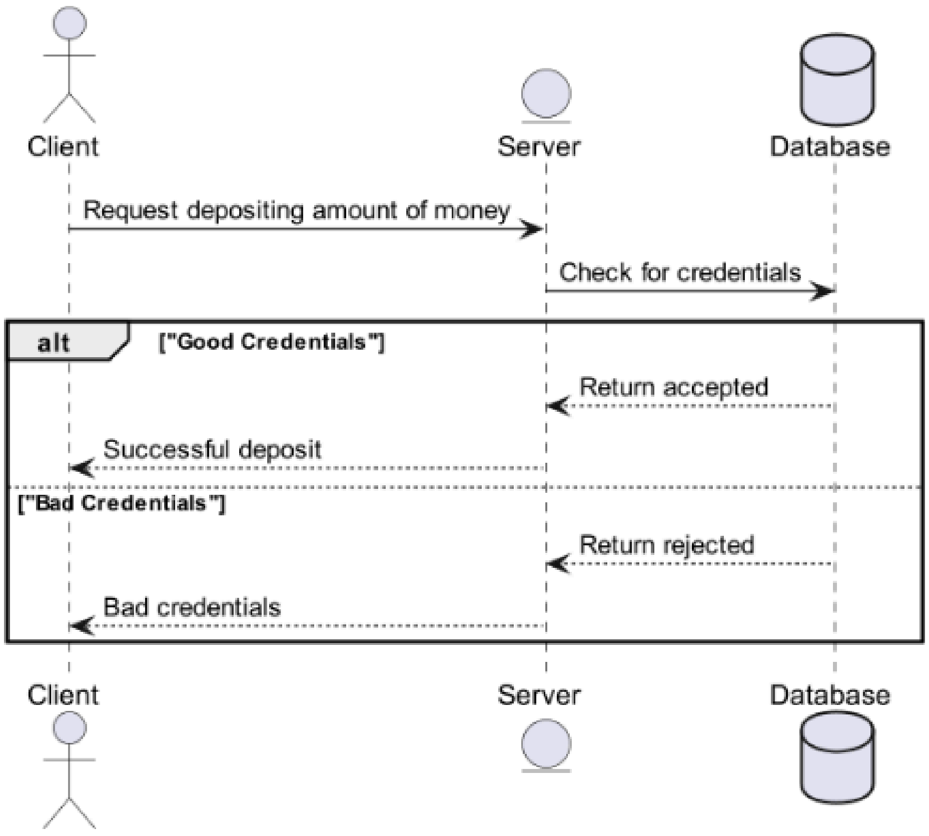


Figure 4: Deposit ALP Diagram

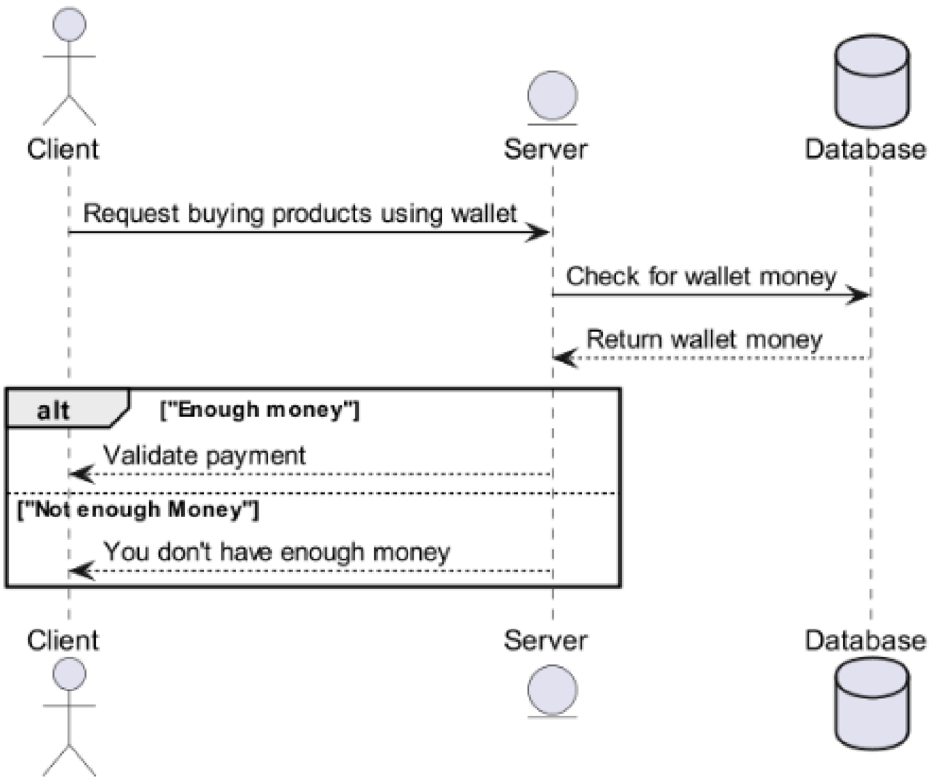


Figure 5: Payment by Wallet ALP Diagram

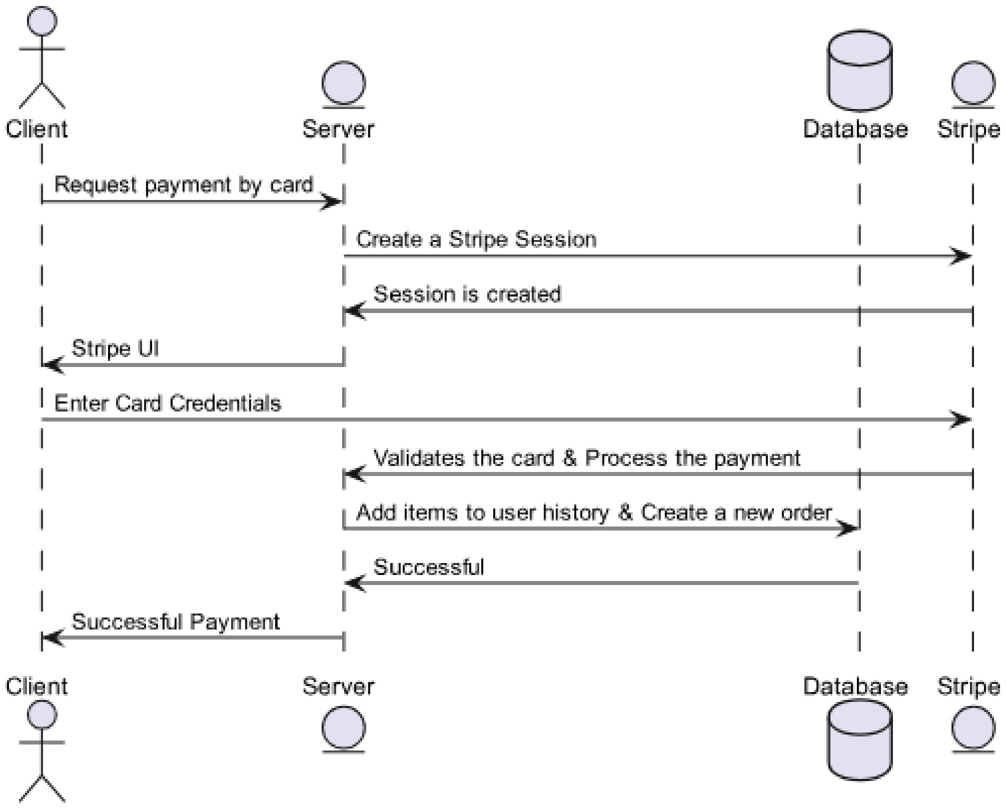


Figure 6: Payment by Card ALP Diagram

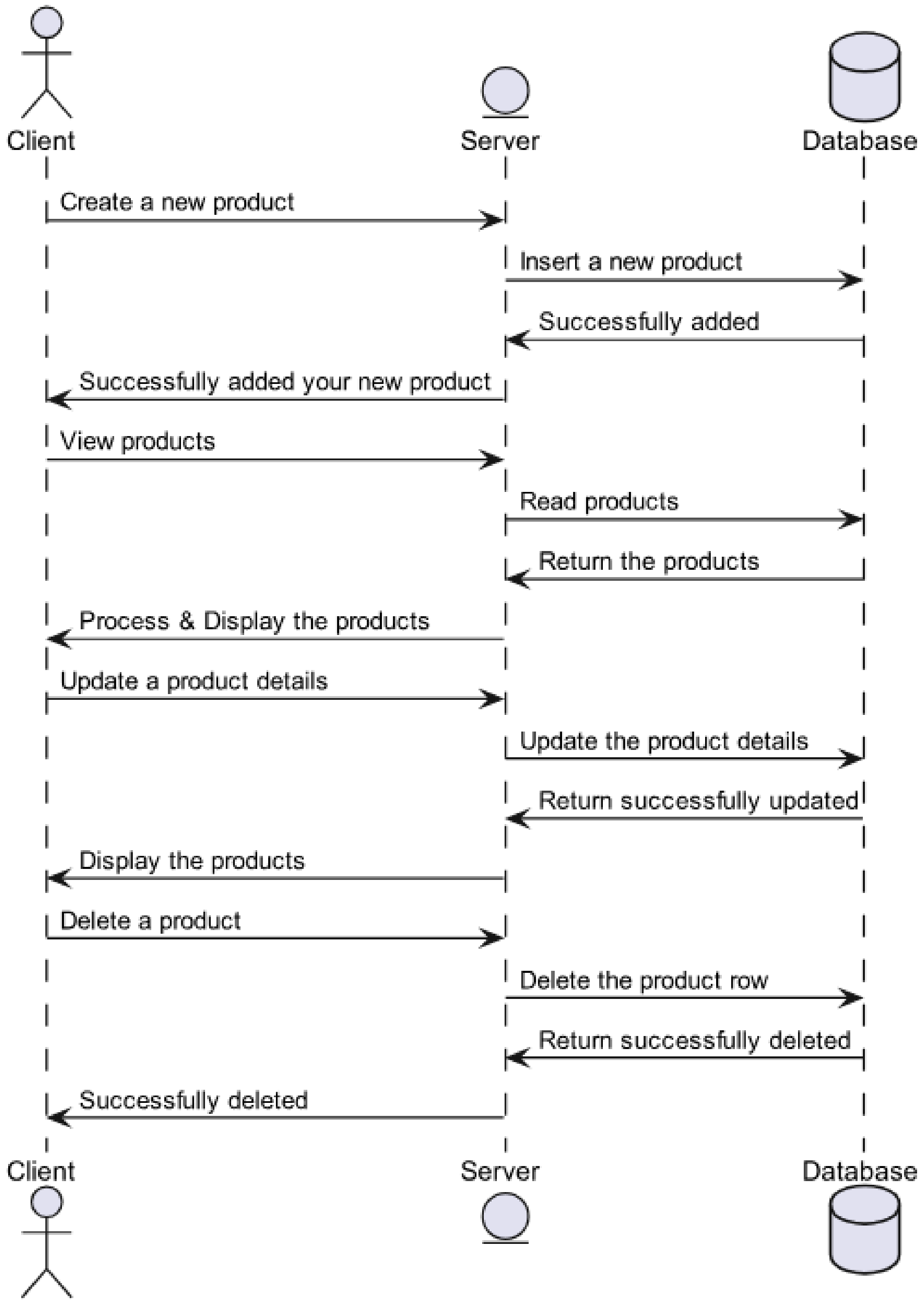


Figure 7: CRUD Operations ALP Diagram

VII Distributed Database Design

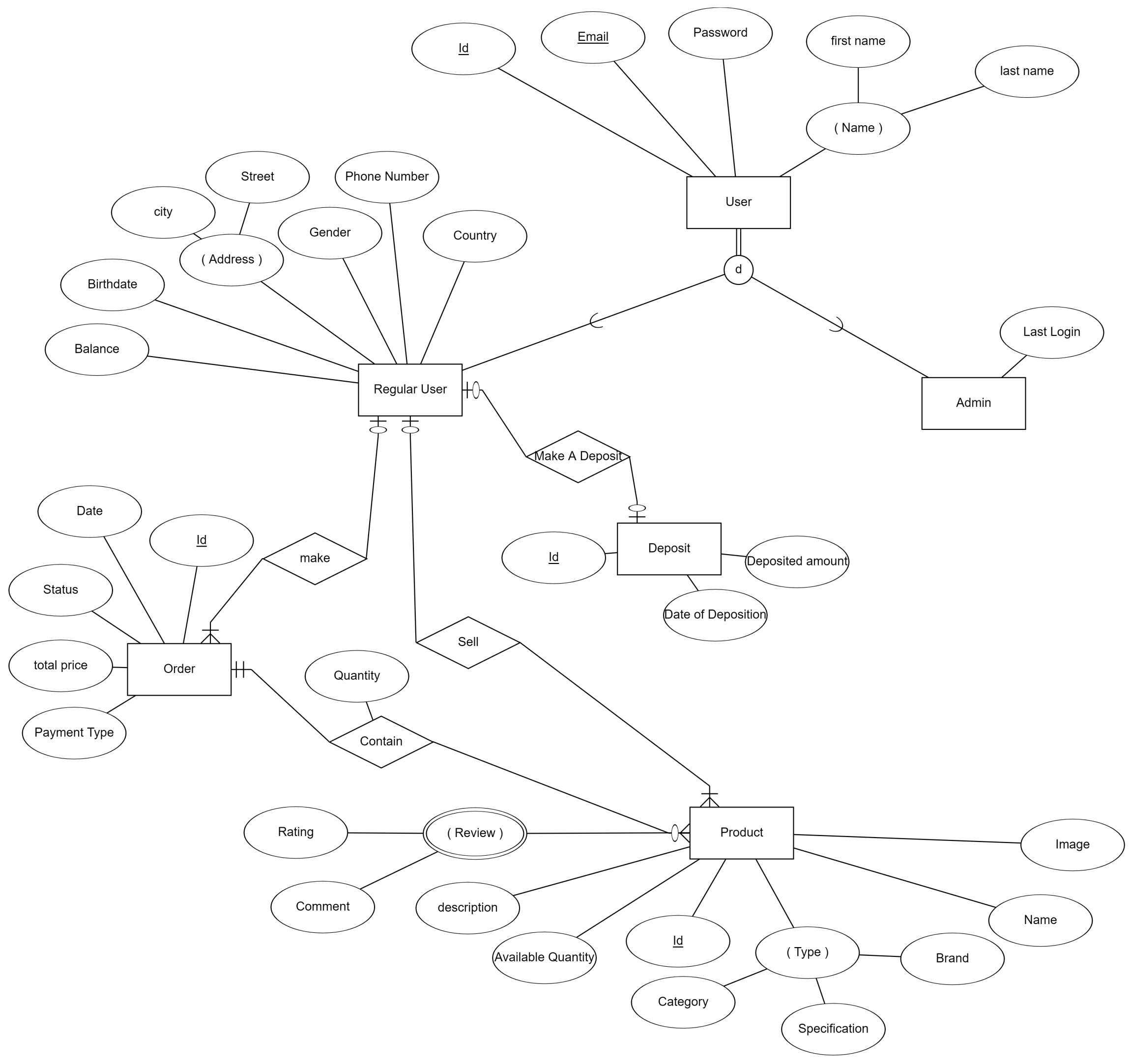


Figure 8: EER Diagram

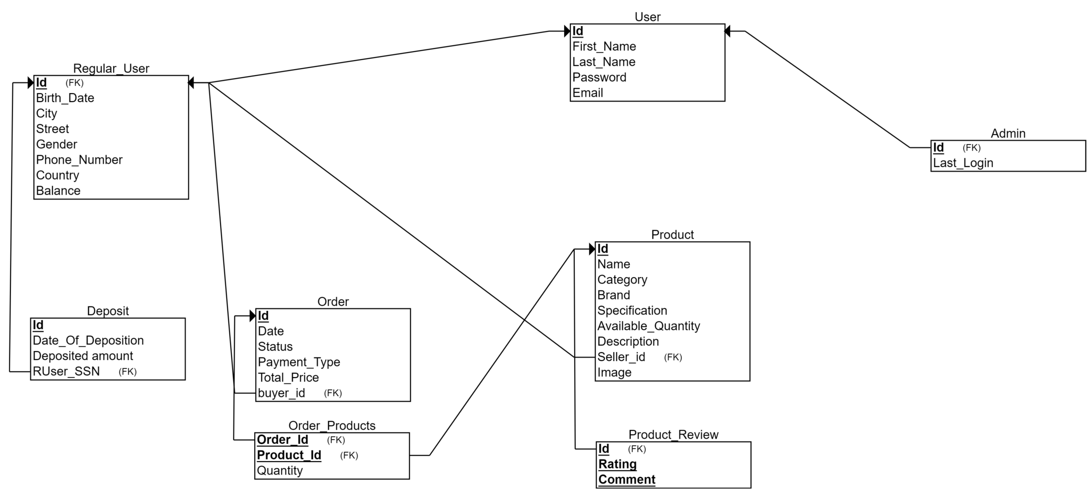


Figure 9: Relational Design

VIII Time Plan

Task	Duration	Start Date	End Date
Requirement Analysis & Design	3 days	01/04/2025	03/04/2025
Frontend Development	15 days	04/04/2025	24/04/2025
Database Design & Setup	15 days	04/04/2025	24/04/2025
Backend Development	7 days	25/04/2025	05/05/2025
Testing	3 days	06/05/2025	08/05/2025
Deployment	1 day	09/05/2025	09/05/2025

Table 1: Time Plan Table

IX System Testing

IX.a General System Testing Table

Test Case ID(s)	Area Covered
T1 → T10	Checking the user login process.
T11 → T12	Moving between the login and sign-up pages.
T13 → T18	Using the main menu and navigating to different pages from it.
T19 → T26	Purchasing an in-stock product and verifying the payment history on the user's profile page.
T27	Attempting to buy a product that is sold out (not available).
T28 → T31	Searching for products using the search feature.
T32 → T35	Adding a new product to the online store.
T36 → T37	Navigating between different market or category pages.

Table 2: General System Testing Table

IX.b Detailed System Testing Report

- `Test Case T1`
  - **Description:**
    - Login with empty email and empty password fields.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: (empty)
    - Password: (empty)
    - User clicks ‘Login’.
  - **Expected Output:**
    - Login fails.
    - Message: “Please enter valid credentials” appears.
  - **Actual Output:**
    - Login failed.
    - Message: “Please enter valid credentials” appeared.
  - **Test Result:** Pass

- `Test Case T2`
  - **Description:**
    - Login with a non-existent email and empty password.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: `user@example.com` (wrong)
    - Password: (empty)
    - User clicks ‘Login’.
  - **Expected Output:**
    - Login fails.
    - Message: “Please enter valid credentials” appears.
  - **Actual Output:**
    - Login failed.
    - Message: “Please enter valid credentials” appeared.
  - **Test Result:** Pass

- **`Test Case T3`**
  - **Description:**
    - Login with a wrong password and an empty email field.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: (empty)
    - Password: `pa55word` (wrong)
    - User clicks 'Login'.
  - **Expected Output:**
    - Login fails.
    - Message: "Please enter valid credentials" appears.
  - **Actual Output:**
    - Login failed.
    - Message: "Please enter valid credentials" appeared.
  - **Test Result:** Pass

- **`Test Case T4`**
  - **Description:**
    - Login with a correct format email and empty password.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: `test`
    - Password: (empty)
    - User clicks 'Login'.
  - **Expected Output:**
    - Login fails.
    - Message: "Please enter valid credentials" appears.
  - **Actual Output:**
    - Login failed.
    - Message: "Please enter valid credentials" appeared.
  - **Test Result:** Pass

- **Test Case T5**
  - **Description:**
    - Login with a correct password and empty email.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: (empty)
    - Password: `1230`
    - User clicks ‘Login’.
  - **Expected Output:**
    - Login fails.
    - Message: “Please enter valid credentials” appears.
  - **Actual Output:**
    - Login failed.
    - Message: “Please enter valid credentials” appeared.
  - **Test Result:** Pass

- **Test Case T6**
  - **Description:**
    - Login with a non-existent email and correct password.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: `email@example.com` (wrong)
    - Password: `1230`
    - User clicks ‘Login’.
  - **Expected Output:**
    - Login fails.
  - **Actual Output:**
    - Login failed.
  - **Test Result:** Pass

- **`Test Case T7`**
  - **Description:**
    - Login with a correct email and wrong password.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: `test@example.com` (correct)
    - Password: `123456788` (wrong)
    - User clicks ‘Login’.
  - **Expected Output:**
    - Login fails.
  - **Actual Output:**
    - Login failed.
  - **Test Result:** Pass

- **`Test Case T8`**
  - **Description:**
    - Login with a non-existent email and wrong password.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: `user@example.com` (wrong)
    - Password: `8e0k*8` (wrong)
    - User clicks ‘Login’.
  - **Expected Output:**
    - Login fails.
  - **Actual Output:**
    - Login failed.
  - **Test Result:** Pass

- **`Test Case T9`**
  - **Description:**
    - Login with correct email and correct password (standard user).
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Email: `test@example.com` (correct)
    - Password: `12345678` (correct)
    - User clicks 'Login'.
  - **Expected Output:**
    - Login successful.
    - User is redirected to the market page.
  - **Actual Output:**
    - Login successful.
    - User was redirected to the market page.
  - **Test Result:** Pass

- **`Test Case T10`**
  - **Description:**
    - Login with admin email and admin password.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - Enter admin's email.
    - Enter admin's password.
    - User clicks 'Login'.
  - **Expected Output:**
    - Login successful.
    - User is redirected to the admin page.
  - **Actual Output:**
    - Login successful.
    - User was redirected to the admin page.
  - **Test Result:** Pass

- **Test Case T11**
  - **Description:**
    - Navigate to sign-up page from login page.
  - **Pre-Conditions:**
    - User is on the login page.
  - **Test Input / Steps:**
    - User clicks the ‘Sign Up’ button/link.
  - **Expected Output:**
    - User is redirected to the sign-up page.
  - **Actual Output:**
    - User was redirected to the sign-up page.
  - **Test Result:** Pass

- **Test Case T12**
  - **Description:**
    - Navigate to login page from sign-up page.
  - **Pre-Conditions:**
    - User is on the sign-up page.
  - **Test Input / Steps:**
    - User clicks the ‘Login’ button/link.
  - **Expected Output:**
    - User is redirected to the login page.
  - **Actual Output:**
    - User was redirected to the login page.
  - **Test Result:** Pass

- **`Test Case T13`**
  - **Description:**
    - Open the navigation menu.
  - **Pre-Conditions:**
    - User is logged in and on any page.
  - **Test Input / Steps:**
    - User clicks the menu icon.
  - **Expected Output:**
    - The navigation menu opens.
  - **Actual Output:**
    - The navigation menu opened.
  - **Test Result:** Pass

- **`Test Case T14`**
  - **Description:**
    - Navigate to Profile page from menu.
  - **Pre-Conditions:**
    - User is logged in.
    - Navigation menu is open.
    - User is not on the profile page.
  - **Test Input / Steps:**
    - User clicks 'Profile' in the menu.
  - **Expected Output:**
    - User is redirected to the profile page.
  - **Actual Output:**
    - User was redirected to the profile page.
  - **Test Result:** Pass

- **`Test Case T15`**
  - **Description:**
    - Navigate to Market page from menu.
  - **Pre-Conditions:**
    - User is logged in.
    - Navigation menu is open.
    - User is not on the market page.
  - **Test Input / Steps:**
    - User clicks ‘Market’ in the menu.
  - **Expected Output:**
    - User is redirected to the market page.
  - **Actual Output:**
    - User was redirected to the market page.
  - **Test Result:** Pass

- **`Test Case T16`**
  - **Description:**
    - Navigate to Wallet page from menu.
  - **Pre-Conditions:**
    - User is logged in.
    - Navigation menu is open.
    - User is not on the wallet page.
  - **Test Input / Steps:**
    - User clicks ‘Wallet’ in the menu.
  - **Expected Output:**
    - User is redirected to the wallet page.
  - **Actual Output:**
    - User was redirected to the wallet page.
  - **Test Result:** Pass

- **`Test Case T17`**
  - **Description:**
    - Sign out from the application.
  - **Pre-Conditions:**
    - User is logged in.
    - Navigation menu is open.
  - **Test Input / Steps:**
    - User clicks 'Sign Out' in the menu.
  - **Expected Output:**
    - User is signed out.
    - User is redirected to the login page.
  - **Actual Output:**
    - User was signed out.
    - User was redirected to the login page.
  - **Test Result:** Pass

- **`Test Case T18`**
  - **Description:**
    - Close the navigation menu.
  - **Pre-Conditions:**
    - User is logged in.
    - Navigation menu is open.
    - User is on the market page.
  - **Test Input / Steps:**
    - User clicks the 'Close' (X) icon in the menu.
  - **Expected Output:**
    - Menu closes.
    - User remains on the market page.
  - **Actual Output:**
    - Menu closed.
    - User remained on the market page.
  - **Test Result:** Pass

- **`Test Case T19`**
  - **Description:**
    - Open an available product's details/purchase card.
  - **Pre-Conditions:**
    - User is logged in and on the market page.
  - **Test Input / Steps:**
    - User clicks the "Buy!" button for an available product.
  - **Expected Output:**
    - The product's details/purchase card appears.
  - **Actual Output:**
    - The product's details/purchase card appeared.
  - **Test Result:** Pass

- **`Test Case T20`**
  - **Description:**
    - Increase and decrease product quantity on product card.
  - **Pre-Conditions:**
    - User is logged in.
    - Product card is open.
  - **Test Input / Steps:**
    - User clicks the "+" button.
    - User clicks the "-" button.
  - **Expected Output:**
    - Quantity displayed increases with "+" press.
    - Quantity displayed decreases with "-" press (not below minimum).
  - **Actual Output:**
    - Quantity displayed increased with "+" press.
    - Quantity displayed decreased with "-" press.
  - **Test Result:** Pass

- **Test Case T21**
  - **Description:**
    - Adding an available product to the cart.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page.
    - The product's details card is open.
  - **Test Input / Steps:**
    - Set Product Quantity = 4.
    - User presses the "Add to cart!" button.
  - **Expected Output:**
    - The product's card closes.
    - The number of products bought (4) appears on the cart icon at the top right.
  - **Actual Output:**
    - The product's card closed.
    - The number of products bought (4) appeared on the cart icon at the top right.
  - **Test Result:** Pass

- **Test Case T22**
  - **Description:**
    - Navigating to the cart page.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page.
  - **Test Input / Steps:**
    - User presses the cart icon at the top right.
  - **Expected Output:**
    - User is redirected to the cart page.
  - **Actual Output:**
    - User was redirected to the cart page.
  - **Test Result:** Pass

- **`Test Case T23`**
  - **Description:**
    - Completing checkout using the wallet with sufficient funds.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the cart page.
    - Products have been added to the cart.
    - User has enough money in their wallet for the purchase.
  - **Test Input / Steps:**
    - User chooses the "wallet" payment option.
    - User presses the "Complete checkout" button.
  - **Expected Output:**
    - User is redirected to the market page.
    - The cart icon at the top right no longer displays a number.
  - **Actual Output:**
    - User was redirected to the market page.
    - The cart icon at the top right no longer displayed a number.
  - **Test Result:** Pass

- **`Test Case T24`**
  - **Description:**
    - Attempting to complete checkout using the wallet with insufficient funds.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the cart page.
    - Products have been added to the cart.
    - User does **not** have enough money in their wallet for the purchase.
  - **Test Input / Steps:**
    - User chooses the "wallet" payment option.
    - User presses the "Complete checkout" button.
  - **Expected Output:**
    - User is redirected to the market page. (Note: Original text implies checkout still clears cart items even with insufficient funds for wallet. This might be a system design choice or test case flaw.)
    - The cart icon at the top right no longer displays a number.

- **Actual Output:**
  - User was redirected to the market page.
  - The cart icon at the top right no longer displayed a number.
- **Test Result:** Pass

- **Test Case T25**
  - **Description:**
    - Attempting to complete checkout using a credit card.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the cart page.
    - Products have been added to the cart.
  - **Test Input / Steps:**
    - User chooses the "credit card" payment option.
    - User presses the "Complete checkout" button.
  - **Expected Output:**
    - Checkout process does not complete (precise behavior may vary, but no successful purchase).
  - **Actual Output:**
    - Checkout did not complete.
    - User remained on the cart page.
  - **Test Result:** Pass (but revise)

- **Test Case T26**
  - **Description:**
    - Checking the product in the payment history.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the profile page.
  - **Test Input / Steps:**
    - User presses the history icon (located beside the cart icon).
  - **Expected Output:**
    - Purchased products appear.
    - Payment details for these products are displayed.
  - **Actual Output:**
    - Purchased products appeared.
    - Payment details for these products were displayed.
  - **Test Result:** Pass

- **Test Case T27**
  - **Description:**
    - Attempting to open a non-available (sold out) product's card to buy it.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page.
  - **Test Input / Steps:**
    - User presses the "Buy!" button for a sold-out product.
  - **Expected Output:**
    - No action occurs; nothing happens.
  - **Actual Output:**
    - No action occurred; nothing happened.
  - **Test Result:** Pass

- **Test Case T28**
  - **Description:**
    - Searching for a product by name when the product exists in the store.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page.
  - **Test Input / Steps:**
    - User selects "Name" as the search criteria (from top right options).
    - User types the product name into the search field.
    - User presses the search icon.
  - **Expected Output:**
    - Only products matching the entered name appear in the results.
  - **Actual Output:**
    - Only products matching the entered name appeared in the results.
  - **Test Result:** Pass

- **Test Case T29**
  - **Description:**
    - Searching for a product by name when the product does not exist in the store.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page.
  - **Test Input / Steps:**
    - User selects "Name" as the search criteria.
    - User types a non-existent product name into the search field.
    - User presses the search icon.
  - **Expected Output:**
    - No products are displayed; no change in displayed items or an empty result message.
  - **Actual Output:**
    - No products were displayed; nothing happened.
  - **Test Result:** Pass

- **Test Case T30**
  - **Description:**
    - Searching for a product by price when products at or below that price exist.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page.
  - **Test Input / Steps:**
    - User selects "Price" as the search criteria.
    - User types the desired price into the search field.
    - User presses the search icon.
  - **Expected Output:**
    - Only products with the same price or a lower price appear.
  - **Actual Output:**
    - Only products with the same price or a lower price appeared.
  - **Test Result:** Pass

- **Test Case T31**
  - **Description:**
    - Searching for a product by price when no products exist at or below that price.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page.
  - **Test Input / Steps:**
    - User selects "Price" as the search criteria.
    - User types a price into the search field for which no matching or lower-priced products exist.
    - User presses the search icon.
  - **Expected Output:**
    - No products are displayed; no change in displayed items or an empty result message.
  - **Actual Output:**
    - No products were displayed; nothing happened.
  - **Test Result:** Pass

- **Test Case T32**
  - **Description:**
    - Adding a new product to the market.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on their profile page.
  - **Test Input / Steps:**
    - User accesses the "Add Product" form (e.g., by pressing a "cart" or "sell" icon on the profile page).
    - User fills in all required product information.
    - User presses the "Add" button.
  - **Expected Output:**
    - The product appears on the user's profile page (e.g., in "My Products").
    - The product also appears on the main market page.
  - **Actual Output:**
    - The product appeared on the user's profile page.
    - The product also appeared on the main market page.
  - **Test Result:** Pass

- **Test Case T33**
  - **Description:**
    - Editing an existing product's information.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on their profile page.
    - The user has already added at least one product.
  - **Test Input / Steps:**
    - User accesses their product listings (e.g., by pressing a "cart" or "sell" icon on the profile page).
    - User presses the "Edit!" button beside the product they want to modify.
    - User changes the product information.
    - User presses the "Save" button.
  - **Expected Output:**
    - The product's information is updated on the user's profile page.
    - The product's information is also updated on the main market page.

  - **Actual Output:**
    - The product’s information was updated on the user’s profile page.
    - The product’s information was also updated on the main market page.
  - **Test Result:** Pass

- **`Test Case T34`**
  - **Description:**
    - Attempting to add a new product with an empty required field.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on their profile page.
  - **Test Input / Steps:**
    - User accesses the “Add Product” form.
    - User fills in some product information but leaves at least one required field empty.
    - User presses the “Add” button.
  - **Expected Output:**
    - The product is not added.
    - An error message may appear, or the form fields may be cleared/ reset.
    - (Original specified: “all the info should be deleted”)
  - **Actual Output:**
    - The product was not added.
    - All information entered in the form was cleared.
  - **Test Result:** Pass

- **`Test Case T35`**
  - **Description:**
    - Removing a user's product from the market.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on their profile page.
    - The user has already added at least one product.
  - **Test Input / Steps:**
    - User accesses their product listings.
    - User presses the remove icon (on the far right) next to the product.
  - **Expected Output:**
    - The product disappears from the user's profile page listings.
    - The product also disappears from the main market page.
  - **Actual Output:**
    - The product disappeared from the user's profile page listings.
    - The product also disappeared from the main market page.
  - **Test Result:** Pass

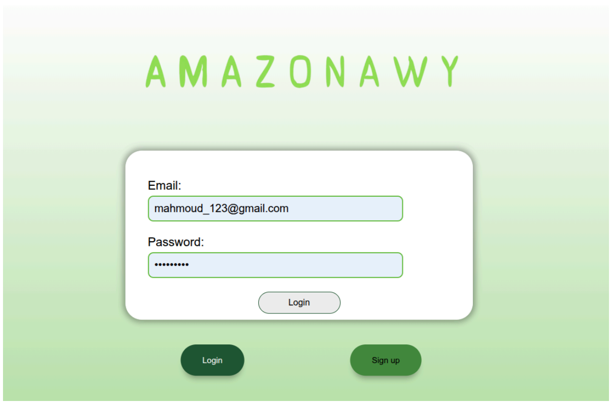
- **`Test Case T36`**
  - **Description:**
    - Navigating to the next page in the market listings.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page.
    - There are enough products for a "next page" to exist.
  - **Test Input / Steps:**
    - User presses the "Next" button at the bottom of the page.
  - **Expected Output:**
    - The displayed page number increases by one.
    - The products corresponding to the new (next) page are displayed.
  - **Actual Output:**
    - The displayed page number increased by one.
    - The products corresponding to the new (next) page were displayed.
  - **Test Result:** Pass

- **Test Case T37**
  - **Description:**
    - Navigating to the previous page in the market listings.
  - **Pre-Conditions:**
    - User is logged in.
    - User is on the market page (and not on the first page).
    - There is a “previous page” to navigate to.
  - **Test Input / Steps:**
    - User presses the “Previous” button at the bottom of the page.
  - **Expected Output:**
    - The displayed page number decreases by one.
    - The products corresponding to the new (previous) page are displayed.
  - **Actual Output:**
    - The displayed page number decreased by one.
    - The products corresponding to the new (previous) page were displayed.
  - **Test Result:** Pass

X End-User Guide

X.a Introduction

- Fisrt of all go to the website of our store through this link AMAZONAWY
- If you already have an account on the website, simply log in using your credentials.

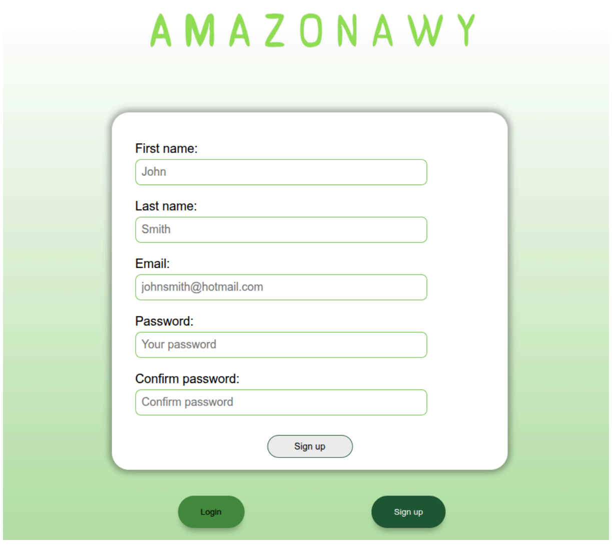


Email: mahmoud_123@gmail.com

Password: •••••••••

Figure 10: Login Page

- If not, please click on the 'Sign Up' button to proceed to the registration page and create a new account.
- Please provide your first name, last name, desired account username, and a strong password.
- Once you've filled in the required information, click the 'Sign Up' button to complete the registration process.



First name: John

Last name: Smith

Email: johnsmith@hotmail.com

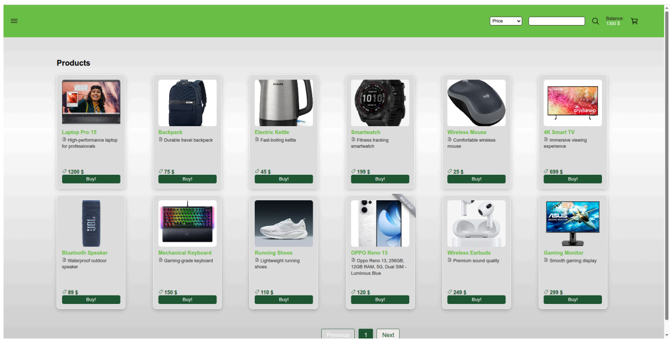
Password: Your password

Confirm password: Confirm password

Figure 11: Sign up Page

X.b Market Page

- Welcome to the main page displaying the available products for purchase.



Products

Laptop Pro 15
High-performance laptop for professionals
1200 $

Backpack
Durable travel backpack
75 $

Electric Kettle
Fast-boiling kettle
45 $

Smartwatch
Fitness tracking smartwatch
199 $

Wireless Mouse
Comfortable wireless mouse
25 $

4K Smart TV
Immersive viewing experience
699 $

Bluetooth Speaker
Waterproof outdoor speaker
89 $

Mechanical Keyboard
Gaming-grade keyboard
150 $

Running Shoes
Lightweight running shoes
110 $

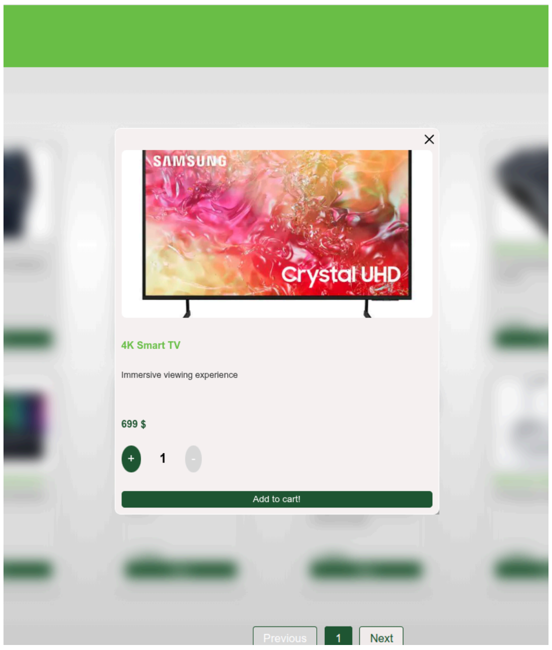
OPPO Reno 13
Oppo Reno 13, 256GB, 12GB RAM, 5G, Dual SIM - Luminous Blue
120 $

Wireless Earbuds
Premium sound quality
249 $

Gaming Monitor
Smooth gaming display
299 $

Figure 12: Market Overview

- If you're interested in purchasing any item, simply click on the 'Buy' button next to the product.



4K Smart TV

Immersive viewing experience

699 $

Figure 13: Buying an item

- Click Add to cart

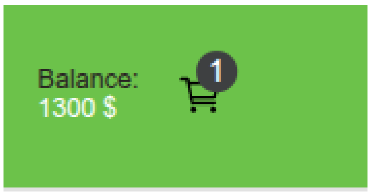


Figure 14: Added to Cart

X.c Payment

- You'll notice that the cart button displays a notification indicating that the item has been added successfully.
- Now, to proceed with the payment for your selected product, simply click on the cart button.

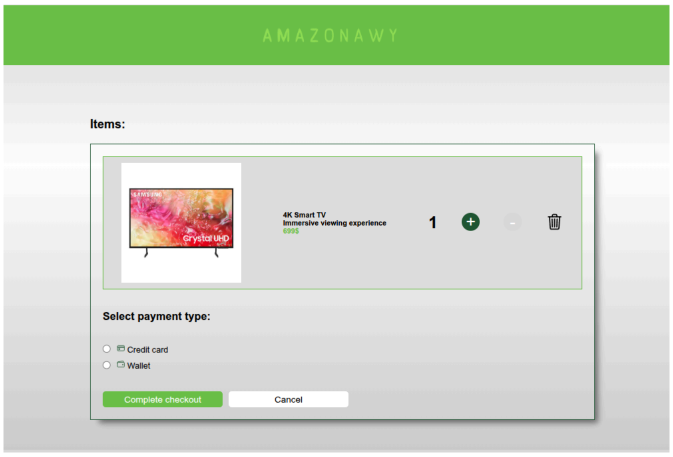
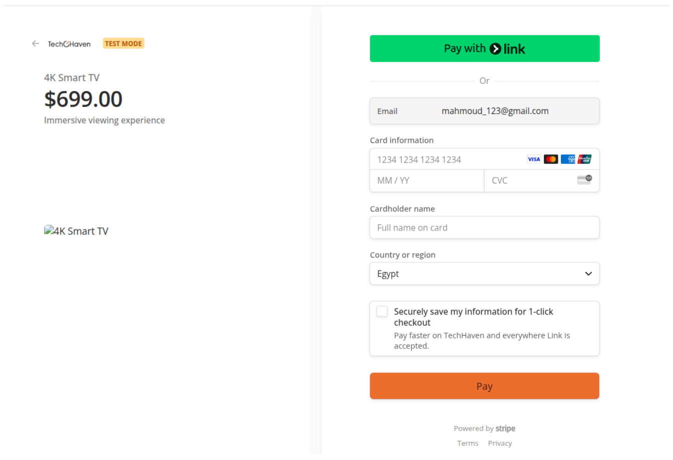


Figure 15: Payment Options Window

- Next, select your preferred payment method: either by Visa or by using your balance on the website.
- If you choose Visa as your payment method, please provide the required information, and then click the 'Pay' button to complete the transaction


TechHaven TEST MODE

4K Smart TV
$699.00
Immersive viewing experience

4K Smart TV

Pay with link

Or

Email mahmoud_123@gmail.com

Card information
1234 1234 1234 1234
MM / YY
CVC

Cardholder name
Full name on card

Country or region
Egypt

Securely save my information for 1-click checkout
Pay faster on TechHaven and everywhere Link is accepted.

Pay

Powered by stripe
Terms Privacy

Figure 16: Payment by Credit Card

- Congratulations! Your payment for the product has been successfully processed.

X.d Side Bar

- On the sidebar, you’ll find buttons for accessing your profile, returning to the home page, adding credit to your account, and logging out.

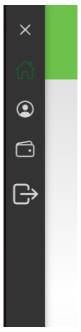


Figure 17: Side Bar

X.e Profile Page

- If you click the profile page, you will find this page.

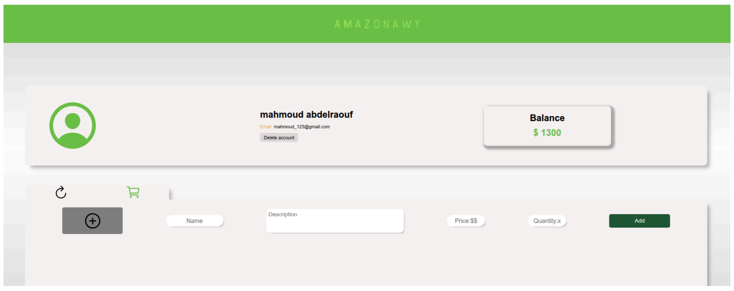


Figure 18: Profile Page

X.f Balance Page

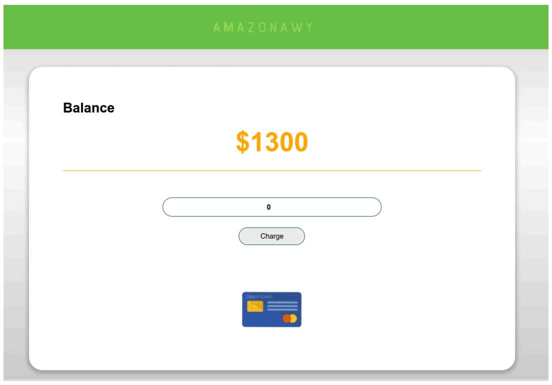


Figure 19: Balance Page

XI Dockerizing

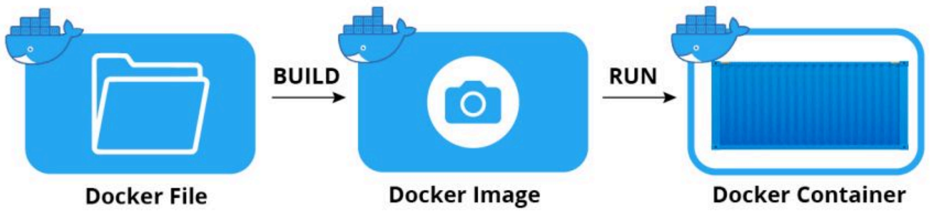


Figure 20: Docker Pipeline

XI.a DOCKERFILE

```
FROM node:22.11.0-alpine AS backend-dev

WORKDIR /app

# copy the root package.json and package-lock.json files
COPY package*.json ./

# install dependencies
RUN npm install --force

# copy the project files
COPY Backend/ ./Backend/
COPY Frontend/ ./Frontend/

COPY config.env .

EXPOSE 3000

# set the base command to run the chosen app
CMD ["sh", "-c", "npm run start"]
```

XII Resources Needed

- The AMAZONAWY project required a variety of resources, broadly categorized as follows:
  1. **`Hardware`**:
     - `Servers`: To host the website, manage databases, and handle customer transactions.
     - `Storage Devices`: For storing product images, descriptions, and other data.
     - `Networking Equipment`: Routers, switches, and firewalls to ensure reliable internet connectivity and security.
  2. **`Software`**:
     - `Version Control Systems`: **`Git`** for managing code versions and collaboration.
     - `Operating Systems`: Linux or Windows servers to host the website.
     - `Database Management Systems`: **`MySQL`** for storing product information and customer data.
  3. **`Human Resources`**:
     - `Developers`: Front-end, back-end, and full-stack developers to build and maintain the website.
     - `Designers`: UX/UI designers to create an intuitive and visually appealing user interface.
     - `Testers`: Quality Assurance (QA) engineers to test the website for bugs and ensure its functionality.
     - `Project Managers`: To oversee the project's progress, allocate resources, and coordinate tasks.
  4. **`Time`**:
     - Adequate time allocation for planning, development, testing, deployment, and ongoing maintenance of the website.
     - Established timelines and deadlines for completing different stages of the project.

XIII Roles and Responsibilities

Name	Role Covered
Karim Ibrahim	UI Design
Mahmoud Abdelraouf	Database Design
Yassa Sfen	Frontend Dev. & Documentation
Omar Salah	Backend Dev.
Eslam Mohamed	Backend Dev.
Mohamed Adham	Frontend Dev. & UI Design
Sarah Tarek	Database Design
Mariam Ahmed	Frontend Dev.
Adham Mohamed	Backend Dev.
Waleed Khalid	Frontend Dev.
Sarah Sherif	Backend Dev.
Abdelrahman Elsayed	Docker & Project Setup

Table 3: Roles and Responsibilities Table

XIV References

1. Brown, E. (2019). **Web Development with Node and Express: Leveraging the JavaScript Stack** (2nd ed.). O'Reilly Media.
2. Microsoft. (n.d.). **N-tier architecture style**. Microsoft Learn. Retrieved October 26, 2023, from https://learn.microsoft.com/en-us/azure/architecture/guide/architecture-styles/n-tier
3. Mozilla Developer Network (MDN). (n.d.). **MDN Web Docs**. Retrieved October 26, 2023, from https://developer.mozilla.org/
4. Node.js Foundation. (n.d.). **Node.js Documentation**. Retrieved October 26, 2023, from https://nodejs.org/en/docs/
5. Oracle Corporation. (n.d.). **MySQL Documentation**. Retrieved October 26, 2023, from https://dev.mysql.com/doc/