Style Guidelines for Final Year Project ReportsDroPilot: AI Assisted Dropshipping Automation

Final Year Project – Mid Report

Session 2021-2025

A 4th Year Student

A project submitted in partial fulfilment of the

COMSATS University Degree

of

BSc. (Hons.)BS in Computer Science / Software Engineering (CUI)



Department of Computer Science

COMSATS University Islamabad, Lahore Campus

04 December 2024

# Project Detail

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type (Nature of project) | | | [✔] **Development**  [ ] **Research** & **Development** | | |
| Area of specialization | | |  | | |
| **Project Group Members** | | | | | |
| Sr.# | Reg. # | Student Name | | Email ID | \*Signature |
| (i) | FA21-BSE-040 | Abdul Hanan | | fa21-bse-040@cuilahore.edu.pk |  |
| (ii) | FA21-BSE-044 | Yahya Irfan | | fa21-bse-044@cuilahore.edu.pk |  |
| (iii) | FA21-BSE-153 | Bushra Hasan | | fa21-bse-153@cuilahore.edu.pk |  |

\*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to the work of others

# Plagiarism Free Certificate

This is to certify that, I am \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ S/D/o \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, group leader of FYP under registration no CIIT/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/LHR at Computer Science Department, COMSATS University Islamabad, Lahore Campus. I declare that my FYP report is checked by my supervisor and the similarity index is \_\_\_\_\_\_\_\_% that is less than 20%, an acceptable limit by HEC. Report is attached herewith as Appendix A.

Date: \_\_\_\_\_\_\_\_\_\_\_\_ Name of Group Leader: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_

Name of Supervisor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Co-Supervisor (if any):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Designation: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Designation: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Abstract**

Write abstract here

**Acknowledgement**

It is optional

Table of Contents

[1 Introduction 8](#_Toc436367060)

[2 Styles in Word 8](#_Toc436367061)

[3 Page Layout & Size 10](#_Toc436367062)

[4 Headings 10](#_Toc436367063)

[4.1 Second Level Headings 11](#_Toc436367064)

[4.1.1 Third Level Headings 11](#_Toc436367065)

[4.2 A Word on Numbering 11](#_Toc436367066)

[5 Figures and Charts 11](#_Toc436367067)

[6 Program Code 12](#_Toc436367068)

[7 Table of Contents 12](#_Toc436367069)

[8 Final Year Project (FYP) Report Outline 13](#_Toc436367070)

[9 References 14](#_Toc436367071)

**List of Tables**

Table 1 Literature review sources 1

**List of Figures**

Figure 1 Selecting the Heading 1 style from the Word style menu.

# Chapter 1: Introduction

A project introduction is a document that provides an overview of a project and its purpose, objectives, and scope. It is typically used to introduce the project to stakeholders, such as project sponsors, team members, and customers, and to provide a common understanding of the project's goals and deliverables.

## Introduction

In today's fast-paced e-commerce industry, managing and scaling a dropshipping business is a complex and time-consuming process. With the rise of global competition and increasing customer demands, dropshippers face challenges such as selecting trending products, automating order fulfillment, handling customer inquiries, and improving their marketing strategies.

**DroPilot** aims to revolutionize dropshipping by providing an AI-powered automation system that simplifies and streamlines every aspect of the dropshipping process. Our platform offers advanced tools such as web scraping to identify trending products, automated order processing, AI-driven customer support chatbots, and intelligent analytics for decision-making.

**Key Features of DroPilot include:**

1. Trending Product Recommendations
2. Web Scraping for Market Research
3. Order Automation
4. AI-Powered Customer Support Chatbot
5. Advanced Analytics and Reporting
6. Product Comparison
7. Copywriting Assistance
8. Image Background Enhancement for Product Photos
9. Marketing Tools Integration

This platform will be accessible via both web and mobile applications, ensuring ease of use for dropshippers at any time and on any device.

## Objectives

The primary goal of **DroPilot** is to provide a comprehensive, user-friendly automation platform that addresses the core challenges of running a dropshipping business. By leveraging the power of AI, we aim to:

* **Enhance Efficiency:** Automate time-consuming tasks such as product sourcing, order fulfillment, and customer support.
* **Boost Profitability:** Offer insights and recommendations for trending products and competitive pricing.
* **Simplify Operations:** Provide a centralized system for managing all dropshipping operations.
* **Improve User Experience:** Ensure seamless navigation through intuitive web and mobile interfaces.

## Problem statement

Managing a dropshipping business involves numerous repetitive and complex tasks, including finding products, fulfilling orders, and handling customer queries. These processes can become overwhelming for individuals and businesses alike, resulting in inefficiencies and lost opportunities.

Manual efforts often lead to errors, delayed order processing, and suboptimal customer service, which can negatively impact brand reputation. Dropshippers also struggle to stay updated with market trends and analyze their business performance effectively.

**DroPilot** resolves these pain points by offering a one-stop solution that automates critical operations and provides data-driven insights to support growth.

## Assumptions & constraints

* The platform will be updated with real-time data using secure APIs and machine learning algorithms.
* Product and market data will be updated daily using web scraping and AI tools.
* The system will rely on reliable internet connectivity to provide seamless functionality.
* Data security and privacy protocols will be strictly implemented to protect user and business information.

## Project scope (what and what not to consider)

The **DroPilot** platform will deliver an end-to-end solution for dropshipping businesses. The system will include the following features:

1. **Market Research Tools:** Web scraping functionality to identify trending products and analyze competitor pricing.
2. **AI-Driven Insights:** Recommendations for products, pricing strategies, and marketing campaigns based on market data.
3. **Order Fulfillment Automation:** Tools to streamline order processing and inventory management.
4. **Customer Support Integration:** AI chatbot capable of handling common customer queries and issues.
5. **Analytics Dashboard:** Comprehensive reporting on sales performance, trends, and customer behavior.
6. **Customization Options:** Features like image editing and copywriting assistance to enhance product listings.
7. **Seamless Communication:** Integration with supplier APIs and direct messaging for smooth coordination.
8. **Multi-Platform Access:** Availability of web and mobile applications for 24/7 access.

# Chapter 2: Requirements Analysis

Requirement analysis is the process of gathering and documenting the requirements for a system or product. It involves identifying the needs and goals of the stakeholders, such as users, customers, and project sponsors, and determining how the system or product will meet those needs.

The goal of requirement analysis is to ensure that the system or product is designed and developed to meet the needs of the stakeholders. It is a crucial step in the software development process as it helps to define the scope of the project and establish a common understanding of the requirements among all team members.

## Literature review / Existing system study:

### Dropy.ai [1]:

Dropy.AI aims to simplify dropshipping by automating core tasks with AI. It helps users find products, keeps stores updated with new items, and offers an AI-driven customer support system. While the brand spy tool is useful for checking competitors’ strategies, it's not revolutionary and might not always provide a clear advantage.

### AutoDS [2]:

AutoDS is known for being easy to use, especially for beginners, but it's fairly basic. It automates routine tasks like processing orders and adjusting prices when suppliers change them. You can quickly import products from platforms like AliExpress, though it can feel limited if you're looking for more advanced customization or control.

### Glitching AI [3]:

Glitching AI offers tools to discover trending products and manage orders, but its standout feature is access to a network of suppliers. However, the platform's analytics are decent but not as in-depth as some users might want, and while the automated marketing tools are convenient, they can feel generic without much personalization.

### Sell The Trend [4]:

Sell The Trend gives access to over 7 million products and lets you fulfill orders with one click. Its NEXUS tool helps track competitors’ trends, but some may find it overwhelming due to the sheer volume of data. The built-in marketing tools are functional but won’t blow you away if you're expecting cutting-edge features.

### AppyPie [5]:

Appy Pie is a no-code development platform that enables users to create websites, and other digital solutions without needing extensive programming knowledge. The platform's primary aim is to democratize web development by providing a user-friendly interface and a variety of templates that cater to different industries and functionalities.

### Mobirise [6]:

Mobirise is a user-friendly website builder that allows individuals and businesses to create responsive websites without needing any coding skills. The platform is designed for simplicity, featuring a drag-and-drop interface that facilitates the construction of websites through customizable templates and pre-built blocks.

## Stakeholders list (Actors)

A stakeholders list is a document that identifies the individuals, groups, or organizations that are stakeholders in a project or business. Stakeholders are individuals or groups that have a vested interest in the success or failure of the project or business.

### **Admin**

The admin has complete control over the application and is responsible for managing the platform. Their duties include:

* Adding, removing, or updating product listings, customer data, and analytics.
* Monitoring system performance and resolving any operational issues.
* Ensuring compliance with policies and regulations related to dropshipping and user data.

### **User**

The user represents the dropshipper or the customer accessing the application. Their capabilities include:

* Searching for trending products and recommended suppliers.
* Comparing product prices and details.
* Uploading or managing their product catalogs for automation.
* Accessing analytics and performance insights for their business.

### **Supplier**

Suppliers are stakeholders who provide the products to be dropshipped. Their involvement includes:

* Integrating product catalogs with the platform through APIs.
* Updating real-time inventory and price details.
* Facilitating order fulfillment through seamless communication with the system.

### **AI System**

The AI component serves as a virtual assistant to users, enabling:

* Automation of order placements and tracking.
* Customer support through chatbots and FAQs.
* Generating reports and insights about market trends and user preferences.

## Requirements elicitation

Requirement elicitation is the process of gathering the requirements for a system or product. It involves identifying the needs and goals of the stakeholders and determining how the system or product will meet those needs.

### Functional requirements

**FR01: Authentication & Onboarding**

**FR01-01: User Registration (Sign Up)**

|  |  |
| --- | --- |
| **FR01-01-01** | The system shall allow users to register by providing their email, password, and basic account information. |
| **i** | The system shall validate the user information before creating an account to ensure a valid email format and a strong password policy. |
| **ii** | The system shall send an email verification link to the user after successful registration to confirm the email address |
| **iii** | The registration process shall ensure secure data handling by encrypting passwords using industry-standard encryption methods. |

**FR01-02: User Login (Sign In)**

|  |  |
| --- | --- |
| **FR01-02-01** | The system shall allow registered users to log in using their email and password. |
| **i** | The system shall validate user credentials to ensure that only registered users can access their accounts. |
| **ii** | The system shall provide options for password recovery through email, which includes sending a reset link to the registered email address. |
| **iii** | The login system shall protect against brute force attacks by implementing a lockout mechanism after multiple failed attempts. |

**FR01-03: User Onboarding**

|  |  |
| --- | --- |
| **FR01-03-02** | The system shall guide the user through the onboarding experience to create their first store. |
| **i** | The onboarding process shall include interactive tutorials and prompts to assist users in setting up their store. |
| **ii** | The system shall offer templates and quick-start configurations to streamline the onboarding experience. |

**FR02: Payment Integration**

|  |  |
| --- | --- |
| **FR02-01** | The system shall integrate with popular payment systems to facilitate transactions. |
| **i** | The system shall support integrations with PayPal, Stripe, and major credit card providers. |
| **ii** | The payment integration shall ensure secure processing by complying with PCI DSS (Payment Card Industry Data Security Standard). |
| **iii** | Users shall be able to configure their preferred payment gateways through an admin dashboard. |

**FR03: Stock Management**

**FR03-01: Real-time Stock Updates**

|  |  |
| --- | --- |
| **FR03-01-01** | The system shall automatically update product stock levels from drop-shipping suppliers. |
| **i** | The system shall connect with supplier APIs to receive real-time inventory data. |
| **ii** | Stock updates shall reflect in the store’s inventory dashboard within seconds of a change from the supplier. |
| **iii** | The system shall handle stock discrepancies and notify the user of any synchronization issues. |

**FR03-02: Low Stock Alerts**

|  |  |
| --- | --- |
| **FR03-02-01** | The system shall notify store owners when stock for certain products is running low. |
| **i** | The system shall allow users to set a threshold for low stock alerts |
| **ii** | Notifications shall be sent via email and displayed in the store dashboard. |
| **iii** | The system shall prioritize low stock alerts for high-demand products. |

**FR04: Order Management**

**FR04-01: Order Processing**

|  |  |
| --- | --- |
| **FR04-01-01** | The system shall automatically forward order details to the drop-shipping supplier for fulfillment. |
| **I** | The system shall send order details to the supplier via secure API integration. |
| **Ii** | Order processing shall include customer details, shipping preferences, and product information. |
| **Iii** | The system shall confirm successful transmission of order data to the supplier. |

**FR04-02: Order Status Tracking**

|  |  |
| --- | --- |
| **FR04-02-01** | The system shall track the status of orders within the dashboard. |
| **i** | Users shall be able to view order statuses, including pending, shipped, and delivered. |
| **ii** | The system shall provide real-time status updates by syncing with supplier systems. |
| **iii** | Users shall receive notifications for status changes, such as shipping updates. |

**FR04-02: Automatic Invoice Generation**

|  |  |
| --- | --- |
| **FR04-02-01** | The system shall generate and email invoices to customers automatically after order confirmation. |
| **i** | Invoices shall include product details, pricing, and customer information. |
| **ii** | The system shall format invoices to comply with standard accounting practices. |
| **iii** | Invoices shall be stored in the user’s account for future reference |

**FR05: Product Management**

**FR05-01: Product Sourcing**

|  |  |
| --- | --- |
| **FR05-01-01** | The system shall automatically fetch product data from drop-shipping suppliers. |
| **I** | The system shall support integration with suppliers like AliExpress, Oberlo, Sadadropshipping, and hrdropshipping. |
| **Ii** | Product data shall include descriptions, images, and pricing. |
| **Iii** | Users shall have the option to review and modify fetched product data before listing it in their store. |

**FR05-02: AI-Enhanced Product Descriptions**

|  |  |
| --- | --- |
| **FR05-02-01** | The system shall use AI to generate SEO-optimized product titles, descriptions, and tags. |
| **i** | AI-generated content shall be customizable to fit different marketing strategies. |
| **ii** | The system shall evaluate product keywords and trends to improve search engine visibility. |
| **iii** | Users shall have the option to edit AI-suggested descriptions before publishing. |

**FR05-03: Bulk Products Upload**

|  |  |
| --- | --- |
| **FR05-03-01** | The system shall allow users to upload multiple products at once. |
| **i** | Users shall be able to use CSV or Excel file formats for bulk uploads. |
| **ii** | The system shall validate the data in the upload file and notify users of any errors. |
| **iii** | Bulk uploads shall be processed efficiently, with progress tracking available. |

**FR06: Store Creation & Setup**

**FR06-01: AI-Assisted Store Setup**

|  |  |
| --- | --- |
| **FR06-01-01** | The system shall use AI to assist users in creating a fully functional e-commerce store. |
| **I** | AI shall generate product listings, descriptions, and images based on user preferences. |
| **Ii** | The setup wizard shall include prompts for branding elements, such as logos and themes. |
| **Iii** | Users shall receive recommendations for optimizing their store layout and content. |

**FR06-02: Template Selection**

|  |  |
| --- | --- |
| **FR06-02-01** | The system shall provide users with a range of pre-designed store templates. |
| **I** | Templates shall be categorized by industry and design style. |
| **Ii** | Users shall preview templates before selecting them for their store. |
| **Iii** | Templates shall be customizable after selection. |

**FR06-03: Customization Options**

|  |  |
| --- | --- |
| **FR06-03-01** | The system shall allow users to modify the layout, colors, fonts, and overall design. |
| **i** | Customization tools shall include a user-friendly interface for real-time changes |
| **ii** | Users shall save changes and revert to previous versions if needed. |
| **iii** | The system shall support custom CSS and HTML for advanced users. |

**FR06-04: Domain Integration**

|  |  |
| --- | --- |
| **FR06-04-01** | The system shall offer an option for users to use a custom domain or a subdomain provided by DroPilot. |
| **i** | Domain setup shall include step-by-step instructions for connecting external domains. |
| **ii** | The system shall provide SSL certificates for secure browsing. |
| **iii** | Users shall receive domain management support. |

**FR07: Analytics & Reporting**

**FR07-01: Sales Report**

|  |  |
| --- | --- |
| **FR06-01-01** | The system shall generate reports on daily, weekly, or monthly sales performance. |
| **I** | Sales data shall include metrics such as total revenue, best-selling products, and average order value. |
| **Ii** | Users shall have the option to export reports in standard file formats (e.g., PDF, Excel). |
| **Iii** | Reports shall include visual representations like charts and graphs for better comprehension. |

**FR07-02: Product Profitability Analysis**

|  |  |
| --- | --- |
| **FR06-02-01** | The system shall offer detailed insights into product profitability, including costs and margins to help users understand financial performance. |
| **I** | Profit analysis shall include metrics such as gross and net profit calculations. |
| **ii** | The system shall automatically highlight the most profitable products and suggest potential areas for reducing costs or optimizing product margins. |
| **iii** | Users shall be able to access a detailed cost breakdown that includes product prices, shipping fees, supplier fees, and overall revenue to help them make informed decisions. |

### Non-functional requirements

NFR01: Performance

|  |  |
| --- | --- |
| NFR01-01 | Home page must load within 10 seconds. |
| NFR01-02 | System must respond within 10 seconds on user request. |

NFR02: User Friendly

|  |  |
| --- | --- |
| NFR02-01 | Display of our application will be very user friendly. |

NFR03: Portability

|  |  |
| --- | --- |
| NFR03-01 | Our web application can be operating on any platform. (Mac, Windows, Linux) |
| NFR03-02 | Our mobile application can be operating on Android platform. |

NFR04: Accuracy

|  |  |
| --- | --- |
| NFR04-01 | Our application will generate results that will be reasonably accurate. |

NFR05: Disaster Recovery

|  |  |
| --- | --- |
| NFR05-01 | If system crashes, all data should be recoverable within 30 minutes of the crash. |

NFR06: Defect Maintenance

|  |  |
| --- | --- |
| NFR06-01 | Post release bug fixing should not take more than 4 hours. |

NFR07: Security

|  |  |
| --- | --- |
| NFR07-01 | No user can access database except the Admin. |
| NFR07-02 | System must provide access only through login module to authorized users. |

## Requirement’s traceability matric

### Use case descriptions

Use case descriptions are an important part of the software development process as they help to define the requirements for a system or product and ensure that it meets the needs of the end users. They are typically used to create user stories or acceptance criteria, which are used to guide the development and testing of the system or product.

**User Registration**

|  |  |
| --- | --- |
| Use Case ID | 1 |
| Name | User Registration |
| Actors | User, System |
| Description | User registers an account by providing required information. |
| Pre-Condition | User is not registered on the system. |
| Post-Condition | User account is created successfully. |
| Normal flow of Events | * User provides required information for registration. * System validates the information * System sends an email verification |
| Alternative Flow | Invalid user input leads to an error message. |
| Exception | System failure prevents account creation. |

**Login User**

|  |  |
| --- | --- |
| Use Case ID | 2 |
| Name | Login User |
| Actors | User, System |
| Description | User logs into their account using valid credentials. |
| Pre-Condition | User account must exist and be verified. |
| Post-Condition | User is logged into the system successfully. |
| Normal flow of Events | * User enters login credentials. * System validates credentials * If incorrect credentials are provided repeatedly, a lockout mechanism is implemented * If forgotten password, provide password recovery |
| Alternative Flow | Incorrect login credentials prompt an error. |
| Exception | System fails to authenticate due to server error. |

**User Onboarding**

|  |  |
| --- | --- |
| Use Case ID | 3 |
| Name | User Onboarding |
| Actors | User, System |
| Description | Guide the user through the onboarding process after successful registration or login. |
| Pre-Condition | User is logged in. |
| Post-Condition | User is onboarded successfully with helpful resources. |
| Normal flow of Events | * System provides an onboarding guide. * System shows interactive tutorials * System offers templates or quick-start configurations |
| Alternative Flow | User skips the onboarding process. |
| Exception | Tutorials or templates fail to load. |

**Payment Integration**

|  |  |
| --- | --- |
| Use Case ID | 4 |
| Name | Payment Integration |
| Actors | User, System, Payment Provider |
| Description | User integrates payment systems such as PayPal or Stripe for their store. |
| Pre-Condition | User is logged into the admin dashboard. |
| Post-Condition | Payment system is successfully integrated. |
| Normal flow of Events | * User selects a payment system to integrate. * System connects to the selected payment provider. * System ensures secure payment processing * System complies with PCI DSS standards |
| Alternative Flow | Payment provider rejects integration request. |
| Exception | Security compliance issues cause integration failure. |

**Payment Gateways Configuration**

|  |  |
| --- | --- |
| Use Case ID | 5 |
| Name | Configure Payment Gateways |
| Actors | User (Store Owner), System |
| Description | User configures preferred payment gateway through the admin dashboard. |
| Pre-Condition | User is logged into the admin dashboard. |
| Post-Condition | Payment gateway is configured successfully. |
| Normal flow of Events | * User accesses the admin dashboard. * User selects and configures the preferred payment gateway. |
| Alternative Flow | Invalid configuration settings. |
| Exception | System fails to save configuration changes. |

**Real-Time Stock Updates**

|  |  |
| --- | --- |
| Use Case ID | 6 |
| Name | Real-Time Stock Updates |
| Actors | User, System, Supplier |
| Description | Automatically update product stock levels using supplier APIs. |
| Pre-Condition | Supplier API must be accessible. |
| Post-Condition | Stock levels are updated automatically in the system. |
| Normal flow of Events | * System connects with supplier APIs * System updates product stock levels. * System handles stock discrepancies if any * System notifies user of synchronization issues |
| Alternative Flow | Supplier API fails to respond. |
| Exception | Network issues prevent API connection. |

**Low Stock Alerts**

|  |  |
| --- | --- |
| Use Case ID | 7 |
| Name | Low Stock Alerts |
| Actors | User (Store Owner), System |
| Description | Notify the user when stock levels fall below a specified threshold. |
| Pre-Condition | Low stock threshold is set in the system. |
| Post-Condition | User is notified of low stock levels. |
| Normal flow of Events | * User sets a low stock threshold. * System monitors stock levels. * System sends notifications when stock levels fall below the threshold * System prioritizes alerts for high-demand products |
| Alternative Flow | User disables low stock alerts. |
| Exception | System fails to send notifications. |

**Order Processing**

|  |  |
| --- | --- |
| Use Case ID | 8 |
| Name | Order Processing |
| Actors | User, System, Supplier |
| Description | Forward order details to the supplier for fulfillment. |
| Pre-Condition | Order must be created and confirmed. |
| Post-Condition | Order details are securely transmitted to the supplier. |
| Normal flow of Events | * System compiles customer and order details. * System securely sends order details via API * System includes customer details, shipping preferences, and product information * System confirms successful transmission of order data. |
| Alternative Flow | Supplier API fails to accept order details. |
| Exception | Network issues or server errors prevent data transmission. |

**Order Status Tracking**

|  |  |
| --- | --- |
| Use Case ID | 9 |
| Name | View Order Statuses |
| Actors | User (Store Owner), System, Supplier |
| Description | Track order statuses (Pending, Shipped, Delivered) and sync with the supplier's system. |
| Pre-Condition | Order must be placed and forwarded to the supplier. |
| Post-Condition | User can view real-time order statuses. |
| Normal flow of Events | * User views order status on the dashboard. * System syncs real-time status with supplier systems * System notifies users of status changes |
| Alternative Flow | Supplier system fails to sync order status. |
| Exception | System fails to retrieve updated order information. |

**Automatic Invoice Generation**

|  |  |
| --- | --- |
| Use Case ID | 10 |
| Name | Automatic Invoice Generation |
| Actors | System, Customer |
| Description | Automatically generate invoices upon order confirmation. |
| Pre-Condition | Order must be confirmed. |
| Post-Condition | Invoice is generated and stored in the user’s account. |
| Normal flow of Events | * System generates an invoice after order confirmation. * System includes product details, pricing, and customer information * System formats invoice according to accounting standards * Invoice is stored in the user’s account for future reference. |
| Alternative Flow | Invoice formatting issues arise. |
| Exception | System fails to generate or store the invoice. |

**Product Sourcing**

|  |  |
| --- | --- |
| Use Case ID | 11 |
| Name | Product Sourcing |
| Actors | User, System, Supplier |
| Description | Automatically fetch product data from integrated supplier systems. |
| Pre-Condition | Supplier APIs must be connected to the system. |
| Post-Condition | Product data is fetched and available for review. |
| Normal flow of Events | * System connects to supplier APIs. * System fetches product descriptions, images, and pricing * System supports integration with suppliers * User reviews and modifies fetched product data |
| Alternative Flow | Supplier system fails to send product data. |
| Exception | API connection issues prevent data fetching. |

**AI-Enhanced Product Descriptions**

|  |  |
| --- | --- |
| Use Case ID | 12 |
| Name | AI-Enhanced Product Descriptions |
| Actors | User, System |
| Description | AI generates SEO-friendly product descriptions for enhanced marketing. |
| Pre-Condition | Product details must be available in the system. |
| Post-Condition | SEO-optimized product descriptions are generated and editable by the user. |
| Normal flow of Events | * System generates SEO-optimized product descriptions using AI. * System customizes AI-generated content based on user input * System analyzes keywords and trends for improved SEO * User edits AI-suggested descriptions |
| Alternative Flow | User opts not to use AI-generated descriptions. |
| Exception | AI module fails to generate descriptions. |

**Bulk Products Upload**

|  |  |
| --- | --- |
| Use Case ID | 13 |
| Name | Bulk Products Upload |
| Actors | User (Store Owner), System |
| Description | Bulk upload multiple products using CSV or Excel files. |
| Pre-Condition | Bulk upload feature must be enabled in the system. |
| Post-Condition | Products are successfully uploaded and validated. |
| Normal flow of Events | * User uploads a CSV or Excel file containing product data. * System validates the uploaded data and notifies the user of errors * System supports CSV/Excel file format for uploads * System tracks upload progress efficiently |
| Alternative Flow | Incorrect file format or invalid data results in error messages. |
| Exception | System fails to process the uploaded file due to server errors. |

**Store Creation**

|  |  |
| --- | --- |
| Use Case ID | 14 |
| Name | AI-Assisted Store Setup |
| Actors | User, System |
| Description | AI helps the user set up an e-commerce store by generating essential elements like product listings and branding recommendations. |
| Pre-Condition | User must be logged into the system. |
| Post-Condition | AI-generated store setup recommendations and assets are available for review. |
| Normal flow of Events | * User initiates the store creation process. * System prompts the user to provide store preferences * System generates product listings, descriptions, and images based on user inputs * System suggests branding elements such as logos and themes * System provides recommendations for store layout and content optimization |
| Alternative Flow | User skips AI assistance and opts for manual store setup. |
| Exception | AI module fails to generate assets due to incomplete inputs or technical issues. |

**Template Selection**

|  |  |
| --- | --- |
| Use Case ID | 15 |
| Name | Template Selection |
| Actors | User, System |
| Description | Provide users with pre-designed templates categorized by industry and style, which they can preview and select. |
| Pre-Condition | System must have a repository of store templates. |
| Post-Condition | Selected template is applied to the user's store. |
| Normal flow of Events | * System displays available templates categorized by industry and design style * User previews templates * User selects a template, which is applied to the store. * User customizes the template post-selection |
| Alternative Flow | User chooses to start with a blank design. |
| Exception | System fails to load or display template options. |

**Customization Options**

|  |  |
| --- | --- |
| Use Case ID | 16 |
| Name | Customization Options |
| Actors | User, System |
| Description | Enable the user to customize the store's design through an intuitive interface. |
| Pre-Condition | User must have selected a store template. |
| Post-Condition | Customizations are saved and applied to the store. |
| Normal flow of Events | * User accesses customization options. * System provides a user-friendly interface for real-time changes * User modifies layout, colors, fonts, and other design elements. * User saves changes or reverts to the previous version * Advanced users add custom CSS/HTML |
| Alternative Flow | User keeps the default template without customization. |
| Exception | System fails to apply or save changes due to validation errors or technical issues. |

**Domain Integration**

|  |  |
| --- | --- |
| Use Case ID | 17 |
| Name | Domain Integration |
| Actors | User, System |
| Description | Allow the user to connect a custom domain or use a subdomain provided by the system. |
| Pre-Condition | Domain registration and SSL integration features must be active. |
| Post-Condition | Domain is integrated, and the store is accessible via the chosen domain. |
| Normal flow of Events | * User chooses between a custom domain or a DroPilot-provided subdomain. * System provides step-by-step instructions for connecting external domains * System integrates SSL certificates for secure browsing * System provides ongoing domain management support |
| Alternative Flow | User opts not to integrate a custom domain. |
| Exception | Domain setup fails due to incorrect configurations. |

**Sales Report**

|  |  |
| --- | --- |
| Use Case ID | 18 |
| Name | Sales Report |
| Actors | User, System |
| Description | Provide detailed sales reports with metrics like revenue, best-selling products, and order value. |
| Pre-Condition | Sales data must be recorded in the system. |
| Post-Condition | User can view and export sales reports. |
| Normal flow of Events | * User selects a reporting period (daily, weekly, monthly). * System generates the sales report, including key metrics like total revenue, best-selling products, and average order value * System provides visual representations like charts and * User exports the report in standard file formats like PDF or Excel |
| Alternative Flow | User customizes the reporting period or filters data. |
| Exception | Insufficient data prevents report generation. |

**Product Profitability Analysis**

|  |  |
| --- | --- |
| Use Case ID | 19 |
| Name | Product Profitability Analysis |
| Actors | User, System |
| Description | Analyze profitability metrics for products and suggest cost optimizations. |
| Pre-Condition | Product and sales data must be available in the system. |
| Post-Condition | Profitability insights are available for review. |
| Normal flow of Events | * System calculates gross and net profit for each product * System provides a detailed cost breakdown, including product prices, shipping fees, supplier fees, and revenue * System highlights the most profitable products and suggests areas for cost optimization |
| Alternative Flow | User filters profitability insights for specific categories or periods. |
| Exception | Missing or incomplete data affects the accuracy of profitability analysis. |

### Use case design

**User Registration**



**Sign In**



**User Onboarding**



**Payment Integration**



**Real-time Stock Updates**

**Low Stock Alerts**



**Order Processing**



**Order Status Tracking**



**Automatic Invoice Generation**



**Product Sourcing**



**AI-Enhanced Product Descriptions**



**Bulk Product Upload**



**AI-Assisted Store Setup**



**Template Selection**



**Customization Options**



**Domain Integration**



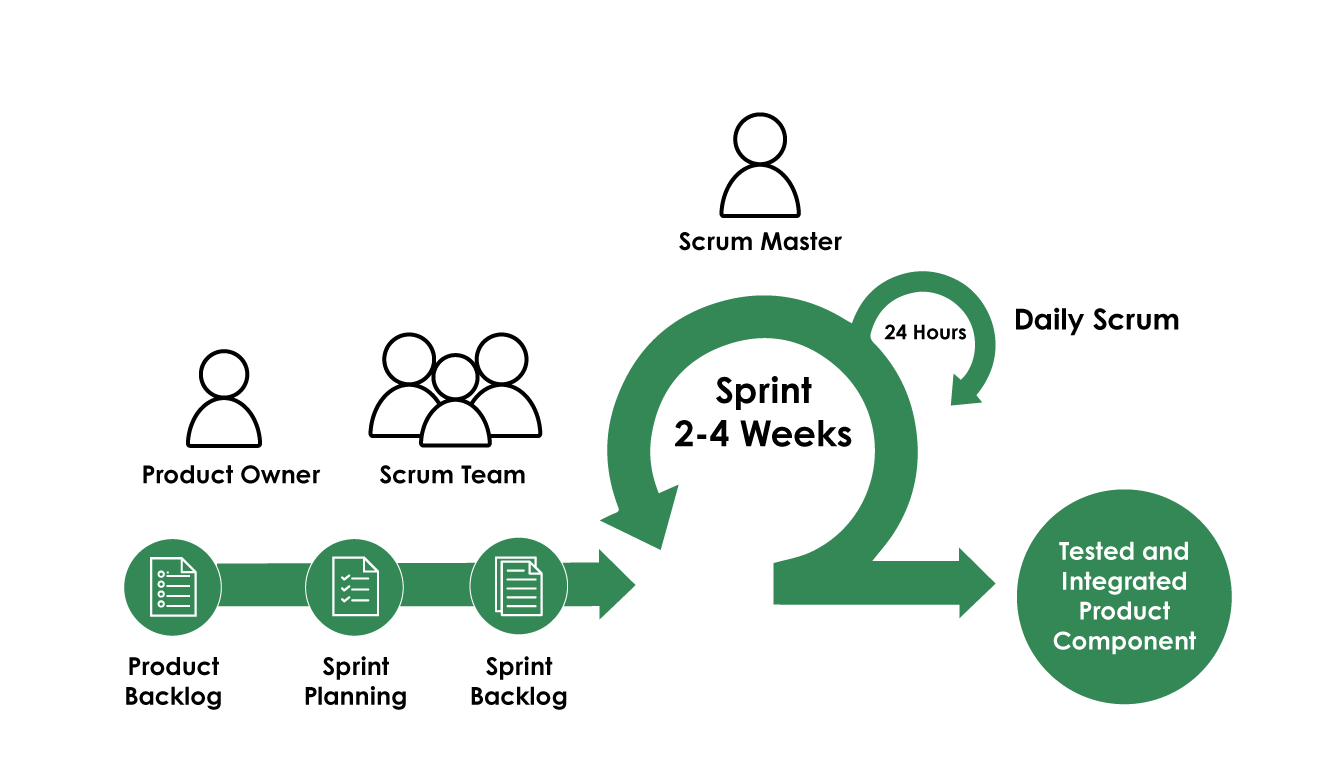
**Sales Report**



**Profitability Analysis**



### Software development life cycle model (justification on why this particular model is considered)



**Reasons:**

**a) Iterative Development:** Allows for incremental development of features, which is crucial for a complex system with multiple functionalities.

**b) Flexibility:** Can adapt to changing requirements or market needs, which is important for an e-commerce platform.

**c) Continuous Feedback:** Regular sprints and reviews allow for continuous improvement and alignment with user needs.

**d) Faster Time-to-Market:** Core features can be developed and released quickly, with additional features added in subsequent sprints.

**e) Risk Mitigation:** Regular testing and reviews help identify and address issues early in the development process.

# Chapter 3: System Design

## Work breakdown structure (WBS)



## Activity diagram

**Sign Up**



**Login**



**Update Stocks**



**Stock Alert**



**Add Products**



**Supplier added a product**



**Export Order to CSV Excel**



**Generation of Invoice**



**Order Forwarding**



**Sales Details**



**Store Setup Domain Setup**



**Online Payment Store Owners**



**Online Payment for DroPilot Owners**



## Sequence diagram

**Sign Up**



**Login**



**Store Creation**



**Product Management**



Order Processing



Sales & Analytics View



## Software architecture

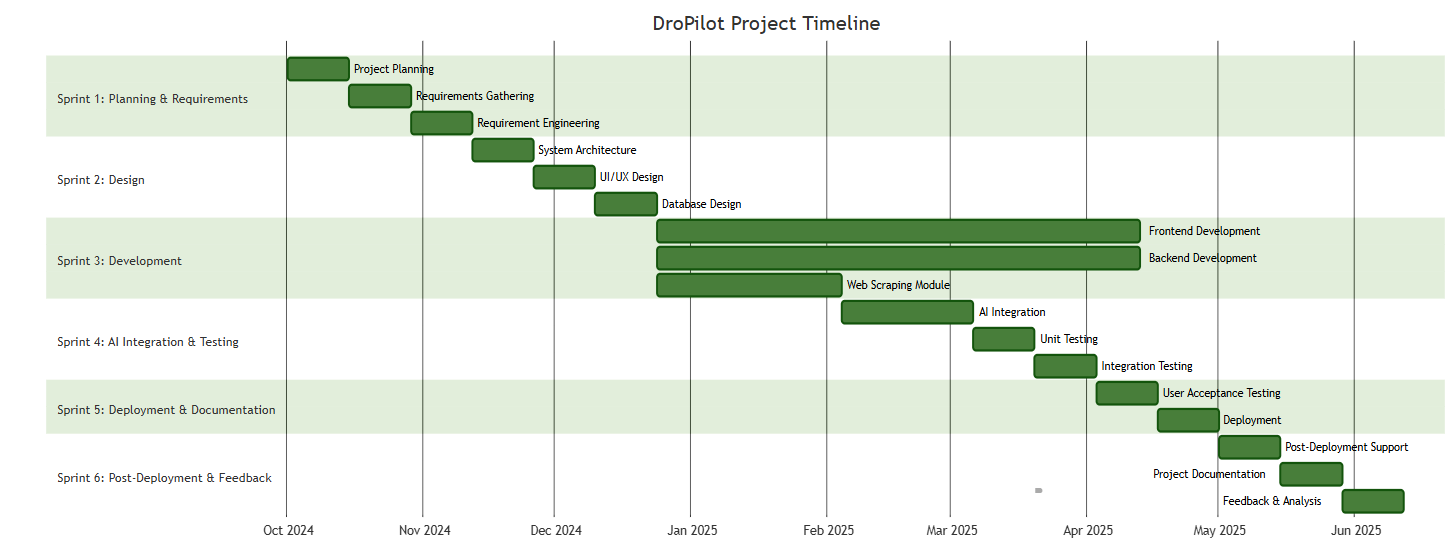


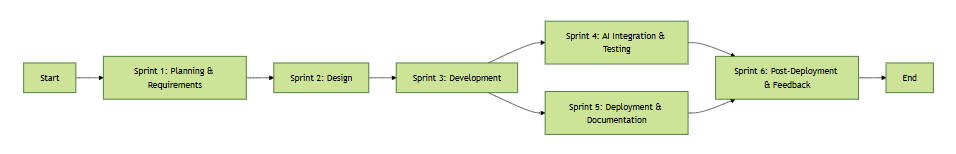
## Class diagram

## Database diagram



## Network diagram (Gantt chart)





## Collaboration diagram

**User Authentication**



**Store Creation**



**Order Processing**



**Product Management**



**Payment Processing**



**Analytics & Reporting**



# Chapter 4: System Testing

System testing is a type of software testing that focuses on testing a complete and integrated software product or system. It aims to validate that the system meets the specified requirements and is fit for purpose. System testing is usually done after the unit testing and integration testing phases, and it involves testing the system as a whole.

## Test cases

Test cases are sets of conditions or variables under which a tester will determine whether an application, system, or component is working correctly. Test cases are an important part of the software testing process as they help to ensure that an application, system, or component is functioning correctly and meeting the requirements of the stakeholders. They are typically created by the tester and reviewed by the development team to ensure that they are comprehensive and accurate.

**User Registration**

|  |  |
| --- | --- |
| Test Case ID | TC-01 |
| Application Name | DroPilot |
| Use Case(s) | User Registration |
| Created By |  |
| Test Case Description | Verify that a new user can register successfully using valid details. |
| Action/Test Steps | 1. Navigate to the Sign-Up page.  2. Enter valid details (Username, Email, Password).  3. Click on the "Sign Up" button. |
| Test Data | Username: test\_user  Email: test@example.com  Password: Password123 |
| Pre-Conditions | The user must not already have an account. |
| Post-Conditions | User should receive an email verification link. |
| Expected Result | Account is created, and email verification is sent. |
| Actual Result |  |
| Status |  |

**Login User**

|  |  |
| --- | --- |
| Test Case ID | TC-02 |
| Application Name | DroPilot |
| Use Case(s) | User Login |
| Created By |  |
| Test Case Description | Verify that a registered user can log in successfully. |
| Action/Test Steps | 1. Navigate to the Login page.  2. Enter valid credentials (Email, Password).  3. Click on the "Sign In" button. |
| Test Data | Email: test@example.com  Password: Password123 |
| Pre-Conditions | User must have an active account. |
| Post-Conditions | User is redirected to the dashboard. |
| Expected Result | User is logged in successfully. |
| Actual Result |  |
| Status |  |

**User Onboarding**

|  |  |
| --- | --- |
| Test Case ID | TC-03 |
| Application Name | DroPilot |
| Use Case(s) | User Onboarding |
| Created By |  |
| Test Case Description | Verify that the onboarding flow guides the user through the setup process. |
| Action/Test Steps | 1. Log in with valid credentials.  2. Follow the onboarding prompts.  3. Select a template and configure basic store settings. |
| Test Data | N/A |
| Pre-Conditions | User must log in for the first time. |
| Post-Conditions | User completes onboarding and is redirected to the dashboard. |
| Expected Result | Onboarding steps are successfully completed |
| Actual Result |  |
| Status |  |

**Payment Integration**

|  |  |
| --- | --- |
| Test Case ID | TC-04 |
| Application Name | DroPilot |
| Use Case(s) | Payment Integration |
| Created By |  |
| Test Case Description | Verify that the payment systems (e.g., PayPal, Stripe) are integrated successfully. |
| Action/Test Steps | 1. Navigate to the Payment Settings page.  2. Select a payment gateway (e.g., PayPal).  3. Enter valid API keys/credentials.  4. Save changes. |
| Test Data | PayPal API Key: abcd1234 |
| Pre-Conditions | The user must have an active account with the payment provider. |
| Post-Conditions | Payment gateway is successfully configured. |
| Expected Result | System accepts the API keys and confirms the integration. |
| Actual Result |  |
| Status |  |

**Payment Gateways Configuration**

|  |  |
| --- | --- |
| Test Case ID | TC-05 |
| Application Name | DroPilot |
| Use Case(s) | Configure Payment Gateway |
| Created By |  |
| Test Case Description | Verify that the user can configure a preferred payment gateway from the admin dashboard. |
| Action/Test Steps | 1. Access the admin dashboard.  2. Navigate to the Payment Configuration section.  3. Choose a preferred payment gateway and configure settings. |
| Test Data | Payment Gateway: PayPal |
| Pre-Conditions | User must have access to the admin dashboard. |
| Post-Conditions | Payment gateway settings are saved successfully. |
| Expected Result | |  | | --- | |  |  |  | | --- | | Payment gateway is configured as per user preferences. | |
| Actual Result |  |
| Status |  |

**Real-Time Stock Updates**

|  |  |
| --- | --- |
| Test Case ID | TC-06 |
| Application Name | DroPilot |
| Use Case(s) | Real-time Stock Updates |
| Created By |  |
| Test Case Description | Verify that stock levels are updated automatically through supplier APIs. |
| Action/Test Steps | 1. Configure supplier API integration.  2. Trigger stock synchronization.  3. Verify updated stock levels in the dashboard. |
| Test Data | Supplier API Key: xyz7890 |
| Pre-Conditions | User must have a supplier account and valid API credentials. |
| Post-Conditions | Stock levels are synchronized successfully. |
| Expected Result | Stock updates are reflected in real-time. |
| Actual Result |  |
| Status |  |

**Low Stock Alerts**

|  |  |
| --- | --- |
| Test Case ID | TC-07 |
| Application Name | DroPilot |
| Use Case(s) | Low Stock Alerts |
| Created By |  |
| Test Case Description | Verify that the system sends low stock alerts when thresholds are reached. |
| Action/Test Steps | 1. Set a low stock threshold for a product.  2. Reduce product stock below the threshold.  3. Check for alerts (e.g., email or notification). |
| Test Data | Product: T-Shirt  Threshold: 5 |
| Pre-Conditions | Stock levels must be above the threshold initially. |
| Post-Conditions | Low stock notifications are sent to the user. |
| Expected Result | System generates and sends low stock alerts. |
| Actual Result |  |
| Status |  |

**Order Processing**

|  |  |
| --- | --- |
| Test Case ID | TC-08 |
| Application Name | DroPilot |
| Use Case(s) | Order Processing |
| Created By |  |
| Test Case Description | Verify that the system successfully forwards order details to the supplier via API. |
| Action/Test Steps | 1. Place an order as a customer.  2. Confirm that the order is forwarded to the supplier.  3. Validate the transmitted order details. |
| Test Data | Order: Laptop  Customer: John Doe  Shipping: Standard |
| Pre-Conditions | Supplier's API must be active and reachable. |
| Post-Conditions | Order details are securely forwarded to the supplier, and confirmation is logged |
| Expected Result | Order is forwarded successfully, and the system confirms transmission |
| Actual Result |  |
| Status |  |

**Order Status Tracking**

|  |  |
| --- | --- |
| Test Case ID | TC-09 |
| Application Name | DroPilot |
| Use Case(s) | Order Status Tracking |
| Created By |  |
| Test Case Description | Verify that users can track real-time order statuses (e.g., Pending, Shipped, Delivered). |
| Action/Test Steps | 1. Place an order.  2. Check the order status in the admin panel.  3. Update the status (e.g., from Pending to Shipped) via the supplier's system. |
| Test Data | Order: Smartphone  Status Update: Shipped |
| Pre-Conditions | Order must already be placed in the system. |
| Post-Conditions | Status changes reflect correctly in the user’s dashboard. |
| Expected Result | Real-time updates are synced and displayed accurately in the system. |
| Actual Result |  |
| Status |  |

**Automatic Invoice Generation**

|  |  |
| --- | --- |
| Test Case ID | TC-10 |
| Application Name | DroPilot |
| Use Case(s) | Automatic invoice Generation |
| Created By |  |
| Test Case Description | Verify that the system generates and stores an invoice after an order is confirmed |
| Action/Test Steps | 1. Confirm an order.  2. Check the generated invoice in the user account. |
| Test Data | Order: Books  Price: $50 |
| Pre-Conditions | An order must be confirmed. |
| Post-Conditions | The invoice is generated and stored in the user’s account. |
| Expected Result | Invoice includes accurate details (product, price, customer, etc.) and follows proper formatting. |
| Actual Result |  |
| Status |  |

**Product Sourcing**

|  |  |
| --- | --- |
| Test Case ID | TC-11 |
| Application Name | DroPilot |
| Use Case(s) | Product Sourcing |
| Created By |  |
| Test Case Description | Verify that product data is fetched from supplier systems automatically. |
| Action/Test Steps | 1.Connect to a supplier’s API (e.g., AliExpress).  2.Fetch product data.  3.Validate the received product details. |
| Test Data | Supplier: AliExpress  Product: Shoes |
| Pre-Conditions | API integration with the supplier must be configured. |
| Post-Conditions | Product data is added to the store owner’s inventory. |
| Expected Result | Product data (name, description, pricing) is fetched and displayed accurately. |
| Actual Result |  |
| Status |  |

**AI-Enhanced Product Descriptions**

|  |  |
| --- | --- |
| Test Case ID | TC-12 |
| Application Name | DroPilot |
| Use Case(s) | AI-Enhanced Product Descriptions |
| Created By |  |
| Test Case Description | Verify that AI generates SEO-optimized product descriptions and allows user customization. |
| Action/Test Steps | 1. Select a product.  2. Generate an AI-enhanced product description.  3. Edit the AI-generated content and save it. |
| Test Data | Product: Backpack  Keywords: Durable, Waterproof |
| Pre-Conditions | The AI description generator must be functional. |
| Post-Conditions | Customizations are saved and displayed. |
| Expected Result | AI generates descriptions based on the given product and keywords. Users can edit and save their changes. |
| Actual Result |  |
| Status |  |

**Bulk Products Upload**

|  |  |
| --- | --- |
| Test Case ID | TC-13 |
| Application Name | DroPilot |
| Use Case(s) | Bulk Products Upload |
| Created By |  |
| Test Case Description | Verify that users can upload multiple products via a CSV/Excel file. |
| Action/Test Steps | 1. Upload a CSV file with product details.  2. Validate the uploaded data.  3. Check the system for errors and progress tracking. |
| Test Data | File: products.csv  Products: 100 |
| Pre-Conditions | The CSV file format must be correct. |
| Post-Conditions | Products are added to the system, and errors (if any) are displayed to the user |
| Expected Result | Products are uploaded successfully, and errors are handled appropriately. |
| Actual Result |  |
| Status |  |

**AI-Assisted Store Setup**

|  |  |
| --- | --- |
| Test Case ID | TC-14 |
| Application Name | DroPilot |
| Use Case(s) | AI-Assisted Store Setup |
| Created By |  |
| Test Case Description | Verify that the system uses AI to assist in creating an e-commerce store. |
| Action/Test Steps | 1. Access the AI-assisted store setup feature. 2. Input preferences for products, branding, and themes. 3. Check the generated product listings, descriptions, and images. 4. Verify the prompts provided for branding elements and recommendations for store layout and optimization. |
| Test Data | Preferences: - Industry: Fashion - Theme: Minimalist - Branding: Logo and pastel color palette |
| Pre-Conditions | User account must be active with the appropriate permissions. |
| Post-Conditions | The AI generates a functional draft store based on user inputs. |
| Expected Result | The system provides AI-generated product listings, branding suggestions, and layout recommendations based on user preferences. |
| Actual Result |  |
| Status |  |

**Template Selection**

|  |  |
| --- | --- |
| Test Case ID | TC-15 |
| Application Name | DroPilot |
| Use Case(s) | Template Selection |
| Created By |  |
| Test Case Description | Verify that the user can select and preview templates based on industry and design style. |
| Action/Test Steps | 1. Navigate to the template selection page. 2. Browse through the template categories. 3. Preview selected templates. 4. Finalize and apply a template to the store. |
| Test Data | Template Type: Industry – Fashion Design Style: Minimalist |
| Pre-Conditions | Template options must be available in the system. |
| Post-Conditions | The selected template is applied to the store. |
| Expected Result | The system categorizes, previews, and applies the template successfully. |
| Actual Result |  |
| Status |  |

**Customization Options**

|  |  |
| --- | --- |
| Test Case ID | TC-16 |
| Application Name | DroPilot |
| Use Case(s) | Customization Options |
| Created By |  |
| Test Case Description | Verify that the system allows users to modify store layouts, colors, fonts, and design in real-time, with the ability to save and revert changes. |
| Action/Test Steps | 1. Navigate to the store customization section. 2. Modify layout, colors, and fonts using the provided interface. 3. Check for live preview of changes. 4. Save the changes and verify their persistence. 5. Revert changes to default settings. 6. Test the addition of custom CSS and HTML for advanced customization. |
| Test Data | Layout: Two-column design Colors: Blue and white theme Fonts: Roboto |
| Pre-Conditions | The store must have an active customization module. |
| Post-Conditions | Customizations are successfully applied and saved. |
| Expected Result | The system provides a user-friendly interface for modifications, saves changes, and supports reverting and advanced customization. |
| Actual Result |  |
| Status |  |

**Domain Integration**

|  |  |
| --- | --- |
| Test Case ID | TC-17 |
| Application Name | DroPilot |
| Use Case(s) | Domain Integration |
| Created By |  |
| Test Case Description | Verify that users can integrate custom domains or subdomains and connect external domains with SSL certificates. |
| Action/Test Steps | 1. Navigate to the domain management section. 2. Select an option to use a custom domain or subdomain. 3. Follow the step-by-step instructions to connect an external domain. 4. Verify SSL certificate activation for secure browsing. 5. Test domain management features such as updating or removing a domain. |
| Test Data | Custom Domain: www.mystore.com |
| Pre-Conditions | The store must have domain management capabilities, and the user must own the domain. |
| Post-Conditions | The custom domain is successfully integrated and secured with SSL. |
| Expected Result | The system integrates domains seamlessly, provides SSL certificates, and supports management tasks. |
| Actual Result |  |
| Status |  |

**Sales Report**

|  |  |
| --- | --- |
| Test Case ID | TC-18 |
| Application Name | DroPilot |
| Use Case(s) | Sales Report |
| Created By |  |
| Test Case Description | Verify that the system generates sales reports with detailed metrics and visualizations. |
| Action/Test Steps | 1. Navigate to the analytics and reporting section. 2. Select the sales report for a specific time frame (e.g., weekly). 3. Verify metrics like total revenue, best-selling products, and average order value. 4. Export the report in PDF format. 5. Check for visual representations such as graphs and charts. |
| Test Data | Time Frame: Weekly Report Format: PDF |
| Pre-Conditions | Sales data must be available for the selected time frame. |
| Post-Conditions | The user receives a comprehensive report with accurate data. |
| Expected Result | The system generates a report with correct metrics, visual representations, and export functionality. |
| Actual Result |  |
| Status |  |

**Product Profitability Analysis**

|  |  |
| --- | --- |
| Test Case ID | TC-19 |
| Application Name | DroPilot |
| Use Case(s) | Product Profitability Analysis |
| Created By |  |
| Test Case Description | Verify that the system provides detailed profitability insights for products. |
| Action/Test Steps | 1. Navigate to the analytics and reporting section. 2. Open the product profitability analysis feature. 3. Verify the gross and net profit metrics. 4. Check for detailed cost breakdowns including product price, shipping fees, and supplier fees. 5. Identify the most profitable products and suggested optimization areas. |
| Test Data | Product: Sneakers Time Frame: Monthly |
| Pre-Conditions | Accurate cost and sales data must be available. |
| Post-Conditions | Insights on profitability and cost breakdown are displayed. |
| Expected Result | The system provides clear and actionable insights on product profitability. |
| Actual Result |  |
| Status |  |

## Unit / integration / acceptance testing

# Chapter 5: Conclusion

## Problems faced and lessons learned

While developing DroPilot, our aim was to leverage the latest technologies to create a robust and efficient dropshipping automation system. However, we encountered several challenges:

* Limited knowledge and experience in using advanced technologies like AI and cloud-based systems.
* Understanding and implementing best practices in API integration and automation processes.
* Initial difficulty in organizing and managing a large-scale project with diverse features.

To address these issues, we enrolled in various online courses on platforms such as Coursera, Code with Mosh, and Udemy. These courses provided the foundational knowledge and hands-on experience required to begin our research and development effectively.

Throughout this project, we gained invaluable skills and knowledge, including:

* **Time Management:** Learning to prioritize tasks and adhere to deadlines efficiently.
* **Adaptability:** Gaining the ability to handle evolving requirements and incorporate changes into the project seamlessly.
* **Technological Proficiency:** Exploring and applying cutting-edge tools and technologies related to dropshipping, web scraping, and automation.
* **Domain Expertise:** Developing an in-depth understanding of dropshipping operations and the associated technical challenges.
* **Teamwork:** Enhancing team collaboration by actively listening, sharing ideas, and supporting each other to achieve common goals.

This experience has strengthened our ability to work cohesively as a team, ensuring effective communication and coordination.

## Project summary

DroPilot is designed for individuals and businesses seeking to streamline and automate their dropshipping operations. The platform provides:

* A comprehensive suite of tools for product research, supplier integration, and order management.
* AI-powered features for recommending trending products and analyzing performance metrics.
* A virtual assistant to facilitate customer support and manage operational tasks efficiently.

The project ensures users can save time, reduce manual effort, and enhance their business performance in a competitive market.

## Future work

In the future, we plan to enhance DroPilot by implementing the following features:

* **Real-Time Data Integration:** Linking the application to supplier and marketplace databases for real-time inventory and pricing updates.
* **Integrated Payment Gateway:** Introducing secure payment portals for order processing and premium subscriptions.
* **Membership Plans:** Offering tiered membership plans that unlock advanced analytics, exclusive supplier data, and premium automation features.
* **Expanded AI Capabilities:** Enhancing AI to provide more personalized recommendations and predictive analytics for market trends.
* **Mobile Application:** Developing a mobile-friendly version of DroPilot for on-the-go access and management.

# References

References to any book, journal paper or website should properly be acknowledged, For example.

1. Lyda M.S. Lau, Jayne Curson, Richard Drew, Peter Dew and Christine Leigh, (1999), Use Of VSP Resource Rooms to Support Group Work in a Learning Environment, ACM 99, pp-2. (Journal paper example)
2. Hideyuki Nakanishi, Chikara Yoshida, Toshikazu Nishmora and Turu Ishada, (1996), FreeWalk: Supporting Casual Meetings in a Network, pp 308-314 (paper on web)
   1. <http://www.acm.org/pubs/articles/proceedings/cscw/240080/p308-nakanishi.pdf>
3. Ali Behforooz & Frederick J.Hudson, (1996), Software Engineering Fundamentals, Oxford University Press. Chapter 8, pp255-235. (book reference example)
4. <http://www.bt.com/bttj/archive.htm> (web site)

APPENDICES

1. ANY OTHER SUPPORTING SCHEDULES or DOCUMENTS

Appendix A

*Include here the 1st page of Turnitin Report*

Every supervisor has his/her own Turnitin account. If not then the supervisors are requested to get the account from Library as soon as possible.