Project Title

Project Team

Student 1	19I-1234
Student 2	19I-1234
Student 3	19I-1234

Session 2018-2022

Supervised by

Mr./ Ms./ Dr. Supervisor Name

Co-Supervised by

Mr./ Ms./ Dr. Supervisor Name

Department of Computer Science

National University of Computer and Emerging Sciences
Islamabad, Pakistan

June, 2022

Contents

List of Figures

List of Tables

Chapter 1

Introduction [AS PER SCOPE DOCUMENT]

The "Introduction" section sets the foundation for the entire report and should be carefully crafted to give the reader a clear understanding of the project's scope, objectives, and context. Here's what you can include:

Background:

Begin by providing background information on the broader topic or field your development project is related to. Highlight any existing problems, gaps, or opportunities that led to the motivation for your project. Include relevant references to previous research, development projects, or technologies. For example, you might want to cite studies or prior systems that failed to address the issues you're focusing on. [1].

1.1 Existing Solutions

For the Existing Solutions section, students should research and describe various solutions, methods, or approaches that have been previously developed to solve the problem they are addressing. This section should include an overview of the different techniques, technologies, or models that are currently in use or have been applied in the past. Students should critically assess the strengths and weaknesses of each solution, highlighting any gaps or limitations that their project aims to address. Additionally, this section should show an understanding of the state-of-the-art in the field and provide context for why the proposed solution is necessary or an improvement over existing ones.

Table 1.1: Comparison of Existing Solutions

System Name	System Overview	System Limitations
System 1	Brief description of what System 1 does and its primary features.	Highlight the shortcomings or issues with System 1.
System 2	Brief description of what System 2 does and its primary features.	Highlight the shortcomings or issues with System 2.
System 3	Brief description of what System 3 does and its primary features.	Highlight the shortcomings or issues with System 3.

1.2 Problem Statement

Clearly define the specific problem your project aims to solve. Explain why this problem is important and how solving it will contribute to the field. This section is critical as it sets the context for your work. This can include technical challenges, system inefficiencies, or user experience problems. Why you are developing this system? What problem does your software solve? (Usually in 10-16 sentences)

Example:

In recent years, the surge in online shopping has resulted in a significant increase in the demand for e-commerce platforms. However, most small to medium-sized businesses struggle to adopt sophisticated e-commerce systems due to high costs, complexity, and lack of customization. Current available solutions either require substantial financial investment or offer limited functionality, leading to inefficiencies in managing product inventories, processing orders, and handling customer interactions.

This project aims to address this issue by developing a cost-effective, customizable, and user-friendly e-commerce platform tailored specifically for small to medium-sized enterprises (SMEs). The platform will integrate key functionalities such as inventory management, order tracking, and customer relationship management into a single, unified system that can be easily deployed with minimal technical expertise.

1.3 Scope

The scope of this project defines what will be included and excluded, helping to set clear expectations for the reader. This section outlines the boundaries and limitations of the development project.

This can include:

- What the project will cover: You define the main features and functionality that the project will focus on, including the specific tools, systems, and technologies to be developed.
- What the project will not cover: This outlines the limitations, clarifying any areas that the project will not address (e.g., advanced features or post-deployment support).

(Usually in 14-18 sentences)

1.4 Modules

This section describes the key functional units or modules of the project, each designed to perform a specific task. The breakdown of the project into modules ensures that each part of the system is manageable and addresses specific objectives. Each module should be described in terms of its functionality and how it interacts with other modules in the system.

The modules should be logically organized to reflect the overall system workflow, ensuring that each module description is comprehensive but concise.

1.4.1 Module 1

[2-4 lines description of the module]

1. Feature 1
2. Feature 2

1.4.2 Module 2

[2-4 lines description of the module]

1. Feature 1
2. Feature 2

1.4.3 Example:

1.4.3.1 Inventory Management Module

This module focuses on inventory management, allowing businesses to efficiently track and manage their product stock levels. It integrates features for adding new products, updating stock information, and generating low-stock alerts.

1. Ability to add, update, and remove products from inventory.
2. Automated alerts for low stock levels to aid in restocking decisions.

1.5 Work Division

Chapter 2

Project Requirements [AS PER FYP1 MID REPORT]

This section outlines the necessary requirements for the successful completion of the project. Project requirements can be divided into two main categories: functional requirements, which describe the system's core operations, and non-functional requirements, which specify performance, security, and usability standards.

2.1 Use-case/Event Response Table/Storyboarding

This section describes how users will interact with the system through various scenarios, often referred to as use cases or event responses.
A use case represents a specific interaction between the user and the system, detailing the steps involved and the expected system responses. Each use case typically includes key components such as a unique identifier, a description of the interaction, the user or system actor involved, preconditions, the main flow of events, and postconditions.
Additionally, storyboarding can be used as a visual aid to depict the sequence of actions, illustrating how the user progresses through the system step by step. Storyboards are helpful for capturing the user experience and ensuring that the design aligns with the intended functionality.

Depending on the project, use cases, storyboarding or Event-response table can be chosen as appropriate methodologies to illustrate user interactions. Examples of all the approaches are provided in Appendix A.

2.2 Functional Requirements

This section describes the functional requirements of the system expressed in the natural language style. This section is typically organized by feature as a system feature name and specific functional requirements associated with this feature.

2.2.1 Module 1

Following are the requirements for module 1:

1. FR1
2. FR2

2.2.2 Module 2

Following are the requirements for module 2:

1. FR1
2. FR2

2.3 Non-Functional Requirements

This section specifies nonfunctional requirements. These quality requirements should be specific, quantitative, and verifiable. The following are some examples of documenting guidelines.

2.3.1 Usability

Usability requirements deal with ease of learning, ease of use, error avoidance and recovery, the efficiency of interactions, and accessibility. The usability requirements specified here will help the user interface designer create the optimum user experience. Example:

USE-1: The COS shall allow a user to retrieve the previous meal ordered with a single interaction.

2.3.2 Performance

State specific performance requirements for various system operations. If different functional requirements or features have different performance requirements, it's appropriate to specify those performance goals right with the corresponding functional requirements, rather than collecting them in this section. Example:

PER-1: 95% of webpages generated by the COS shall download completely within 4 seconds from the time the user requests the page over a 20 Mbps or faster Internet connection.

Chapter 3

System Overview [AS PER FYP1 MID REPORT]

Give a general description of the functionality, context, and design of your project. Provide any background information if necessary.

3.1 Architectural Design

Develop a modular program structure and explain the relationships between the modules to achieve the complete functionality of the system. This is a high-level overview of how the system's modules collaborate with each other to achieve the desired functionality.

Don't go into too much detail about the individual subsystems. The main purpose is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together.

Provide a diagram showing the major subsystems and their connections.

- In the initial design stage, create a Box and Line Diagram for a simpler representation of the systems.

- After finalizing the architecture style/pattern diagram (MVC, Client-Server, Layered, Multi-tiered), create a detailed mapping of modules/components to each part of the architecture.

To view examples of box and line diagrams and architecture styles, see Appendix C.

3.2 Data Design

Explain how the information domain of your system is transformed into data structures. Describe how the major data or system entities are stored, processed, and organized.

List any databases or data storage items.

3.3 Domain Model

The domain model is a conceptual representation of the various entities, their attributes, and the relationships between them within the system. It serves as a bridge between the requirements and the design of the software, providing a clear understanding of the business domain and how it interacts with the system being developed.

A domain model typically includes key components such as:

- **Entities**: These are the primary objects within the domain that hold data and represent real-world concepts. Each entity is defined by its attributes and behaviors.
- **Attributes**: Characteristics or properties of the entities that define their state. For example, a "Customer" entity might have attributes such as `customerID`, `name`, `email`, and `address`.
- **Relationships**: These illustrate how different entities interact with each other. Relationships can be one-to-one, one-to-many, or many-to-many, depending on how entities are associated.
- **Associations**: Connections between entities that help clarify their interdependencies. For instance, a "Product" may be associated with a "Category," indicating the type of product it belongs to.

By constructing a domain model, developers can better understand the system's requirements, facilitate communication among stakeholders, and guide the subsequent design and implementation phases. It also aids in identifying potential issues early in the development process, ensuring a more robust architecture.

An example of a domain model for an online shopping system is provided in Appendix B.

3.4 Design Models [UPTO THE CURRENT ITERATION ONLY]

Create design models as are applicable to your system. Provide detailed descriptions with each of the models that you add. Also ensure visibility of all diagrams.

Design Models for Object Oriented Development Approach

The applicable models for the project using object oriented development approach may include:

- Activity Diagram
- Class Diagram
- Class-level Sequence Diagram
- [OPTIONAL] State Transition Diagram (for the projects which include event handling and backend processes)

Design Models for Procedural Approach

The applicable models for the project using procedural approach may include:

- Activity Diagram
- Data Flow Diagram (data flow diagram should be extended to 2-3 levels. It should clearly list all processes, their sources/sinks and data stores.)
- System-level Sequence Diagram
- [OPTIONAL] State Transition Diagram (for the projects which include event handling and backend processes)

Examples of above diagrams are given in Appendix D.

Chapter 4

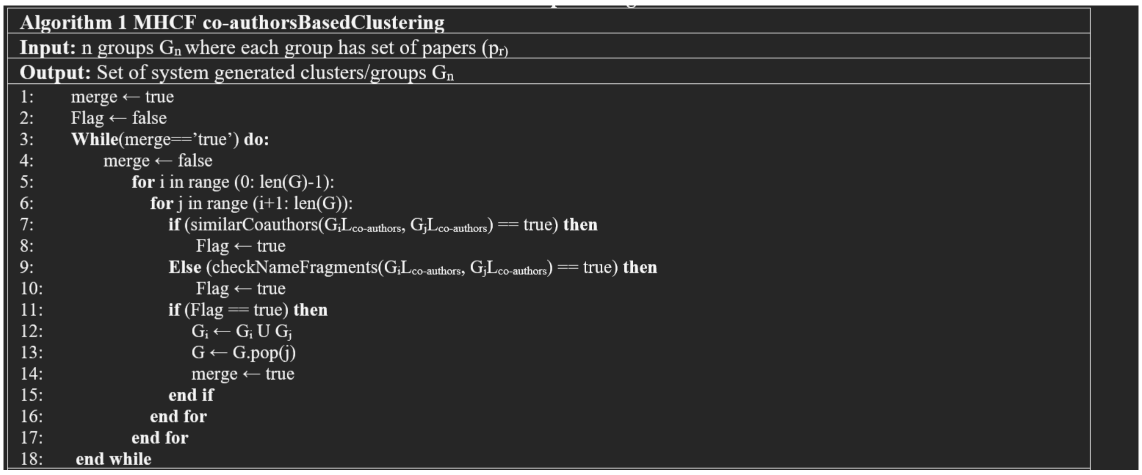
Implementation and Testing [UPTO THE CURRENT ITERATION ONLY]

Give a general description of the functionality, context, and design of your project. Provide any background information if necessary.

4.1 Algorithm Design

Mention the algorithm(s) used in your project to get the work done with regards to major modules. Provide a pseudocode explanation regarding the functioning of the core features. Following are few examples of algorithms/pseudocode.

Example:



Algorithm 1 MHCF co-authorsBasedClustering

Input: n groups G_n where each group has set of papers (p_r)

Output: Set of system generated clusters/groups G_n

```
merge ← true
Flag ← false
While(merge==’true’) do:
    merge ← false
        for i in range (0: len(G)-1):
            for j in range (i+1: len(G)):
                if (similarCoauthors($G_iL_{co-authors}$, $G_jL_{co-authors}$) == true) then
                    Flag ← true
                Else (checkNameFragments($G_iL_{co-authors}$, $G_jL_{co-authors}$) == true) then
                    Flag ← true
                if (Flag == true) then
                    $G_i$ ← $G_i$ U $G_j$
                    G ← G.pop(j)
                    merge ← true
                end if
            end for
        end for
end while
```

Figure 4.1: Example Of Algorithm Design

4.2 External APIs/SDKs

Describe the third-party APIs/SDKs used in the project implementation in the following table. Few examples of APIs are provided in the table.

API and version	Description	Purpose of usage	API endpoint/function/class used
Stripe (version 2020-08-27)	Credit Card payment integration	Sandbox used orders payment	stripe.paymentMethods.create
Cloudinary	Image and Video management	Uploading Product Images	https://api.cloudinary.com/v1

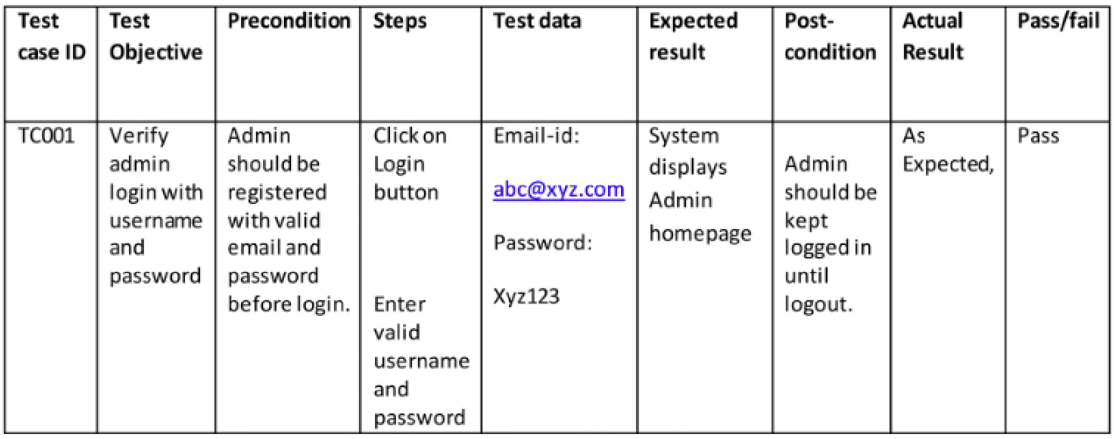
4.3 Testing Details

Once the system has been successfully developed, testing has to be performed to ensure that the system working as intended.

4.3.1 Unit Testing

Each unit test is designed to test a specific function or method independently from other components, helping to identify issues directly related to the functionality being tested.

Following is the example of Unit testing:



Test case ID	Test Objective	Precondition	Steps	Test data	Expected result	Post-condition	Actual Result	Pass/fail
TC001	Verify admin login with username and password	Admin should be registered with valid email and password before login.	Click on Login button Enter valid username and password	Email-id: abc@xyz.com Password: Xyz123	System displays Admin homepage	Admin should be kept logged in until logout.	As Expected,	Pass

Figure 4.2: Example for Unit Testing

Bibliography

[1] A Kolyshkin and S Nazarovs. Stability of slowly diverging flows in shallow water. *Mathematical Modeling and Analysis*, 2007.

Appendix A

Appendices

A.1 Appendix A

A.1.1 Use Case Diagram example (Online Shopping System)

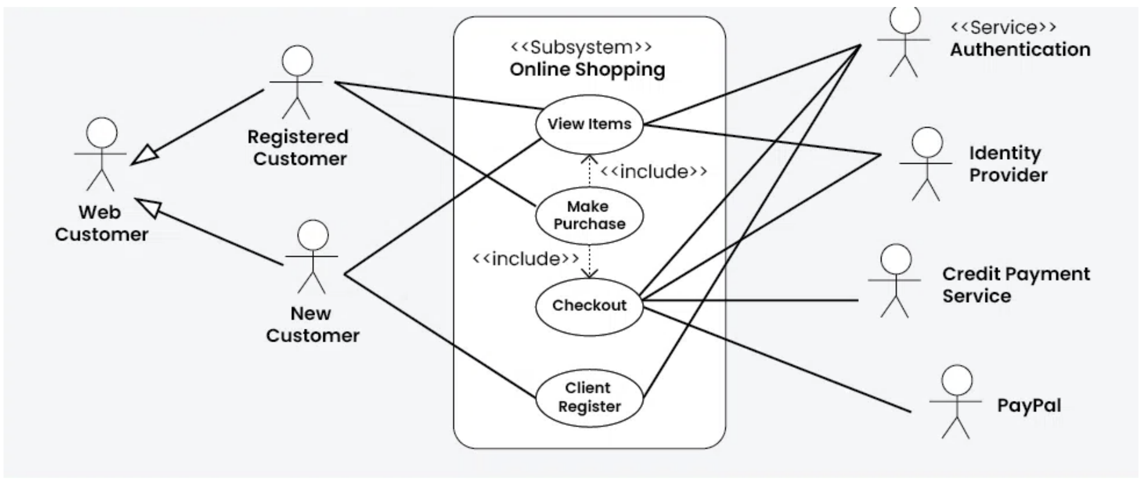
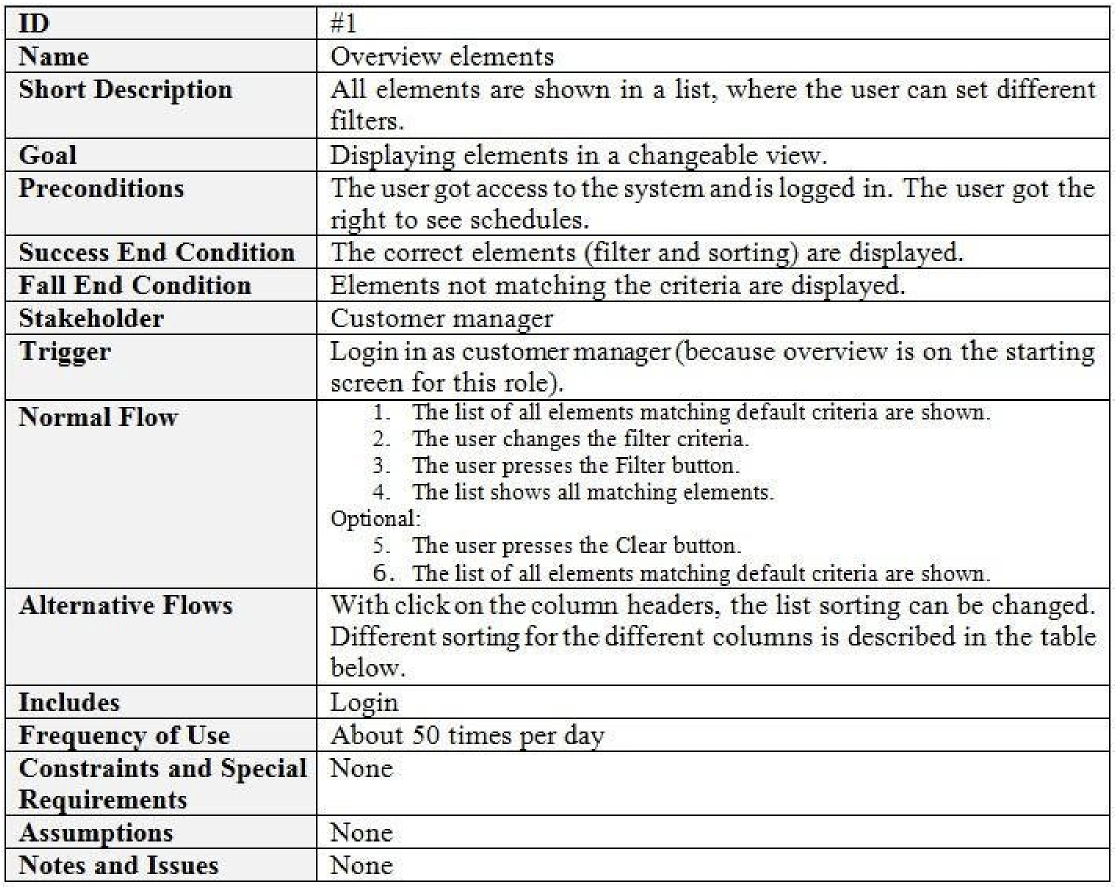


Figure A.1: Use Case Diagram for the Online Shopping System

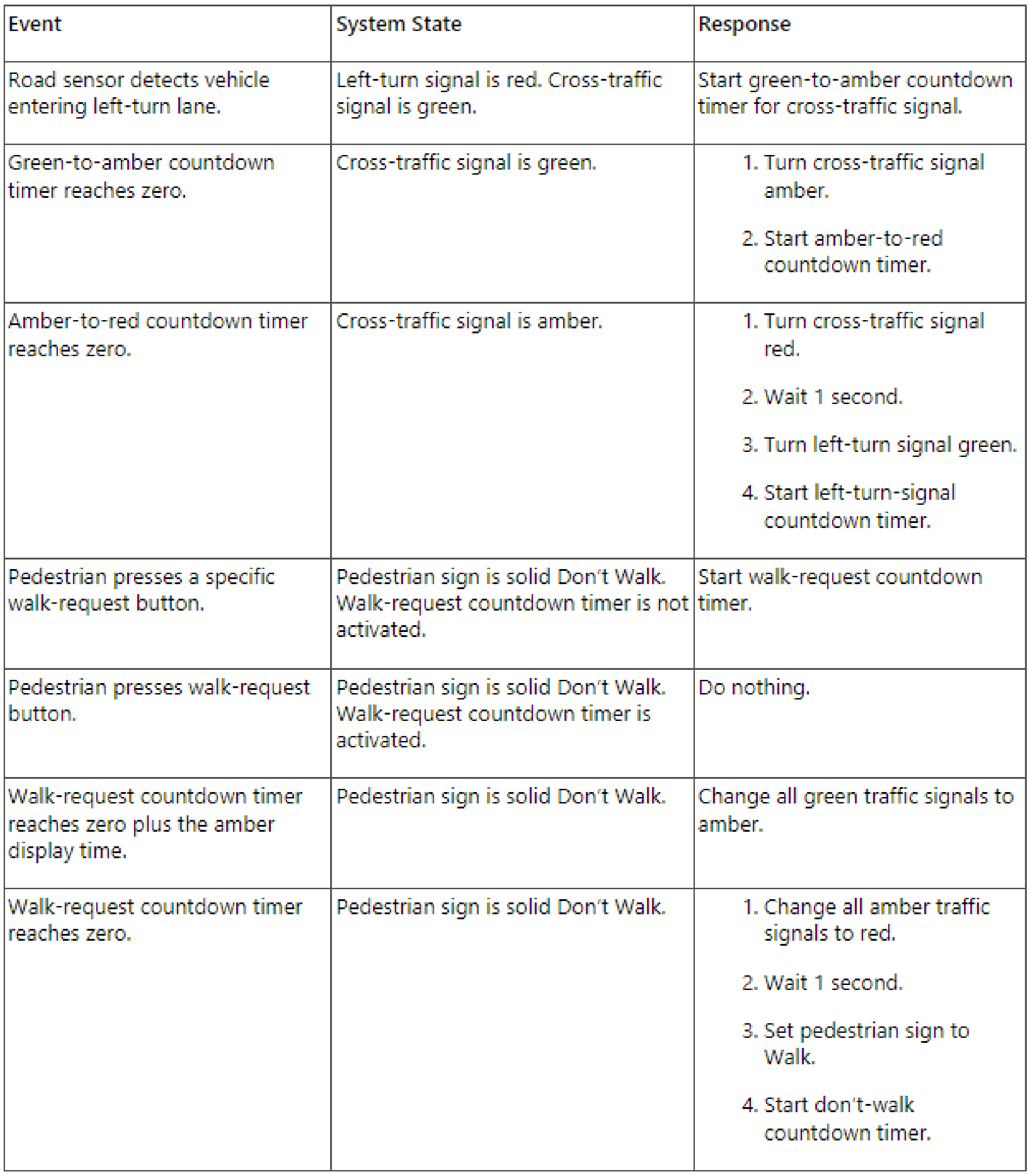
A.1.2 Detail Use Case Example



ID	#1
Name	Overview elements
Short Description	All elements are shown in a list, where the user can set different filters.
Goal	Displaying elements in a changeable view.
Preconditions	The user got access to the system and is logged in. The user got the right to see schedules.
Success End Condition	The correct elements (filter and sorting) are displayed.
Fall End Condition	Elements not matching the criteria are displayed.
Stakeholder	Customer manager
Trigger	Login in as customer manager (because overview is on the starting screen for this role).
Normal Flow	1. The list of all elements matching default criteria are shown. 2. The user changes the filter criteria. 3. The user presses the Filter button. 4. The list shows all matching elements. Optional: 5. The user presses the Clear button. 6. The list of all elements matching default criteria are shown.
Alternative Flows	With click on the column headers, the list sorting can be changed. Different sorting for the different columns is described in the table below.
Includes	Login
Frequency of Use	About 50 times per day
Constraints and Special Requirements	None
Assumptions	None
Notes and Issues	None

Figure A.2: Detail Use Case Example

A.1.3 Event-Response Table for a Highway Intersection



Event	System State	Response
Road sensor detects vehicle entering left-turn lane.	Left-turn signal is red. Cross-traffic signal is green.	Start green-to-amber countdown timer for cross-traffic signal.
Green-to-amber countdown timer reaches zero.	Cross-traffic signal is green.	1. Turn cross-traffic signal amber. 2. Start amber-to-red countdown timer.
Amber-to-red countdown timer reaches zero.	Cross-traffic signal is amber.	1. Turn cross-traffic signal red. 2. Wait 1 second. 3. Turn left-turn signal green. 4. Start left-turn-signal countdown timer.
Pedestrian presses a specific walk-request button.	Pedestrian sign is solid Don't Walk. Walk-request countdown timer is not activated.	Start walk-request countdown timer.
Pedestrian presses walk-request button.	Pedestrian sign is solid Don't Walk. Walk-request countdown timer is activated.	Do nothing.
Walk-request countdown timer reaches zero plus the amber display time.	Pedestrian sign is solid Don't Walk.	Change all green traffic signals to amber.
Walk-request countdown timer reaches zero.	Pedestrian sign is solid Don't Walk.	1. Change all amber traffic signals to red. 2. Wait 1 second. 3. Set pedestrian sign to Walk. 4. Start don't-walk countdown timer.

Figure A.3: Example of Event Response Table

A.1.4 Story Board Example For Android App

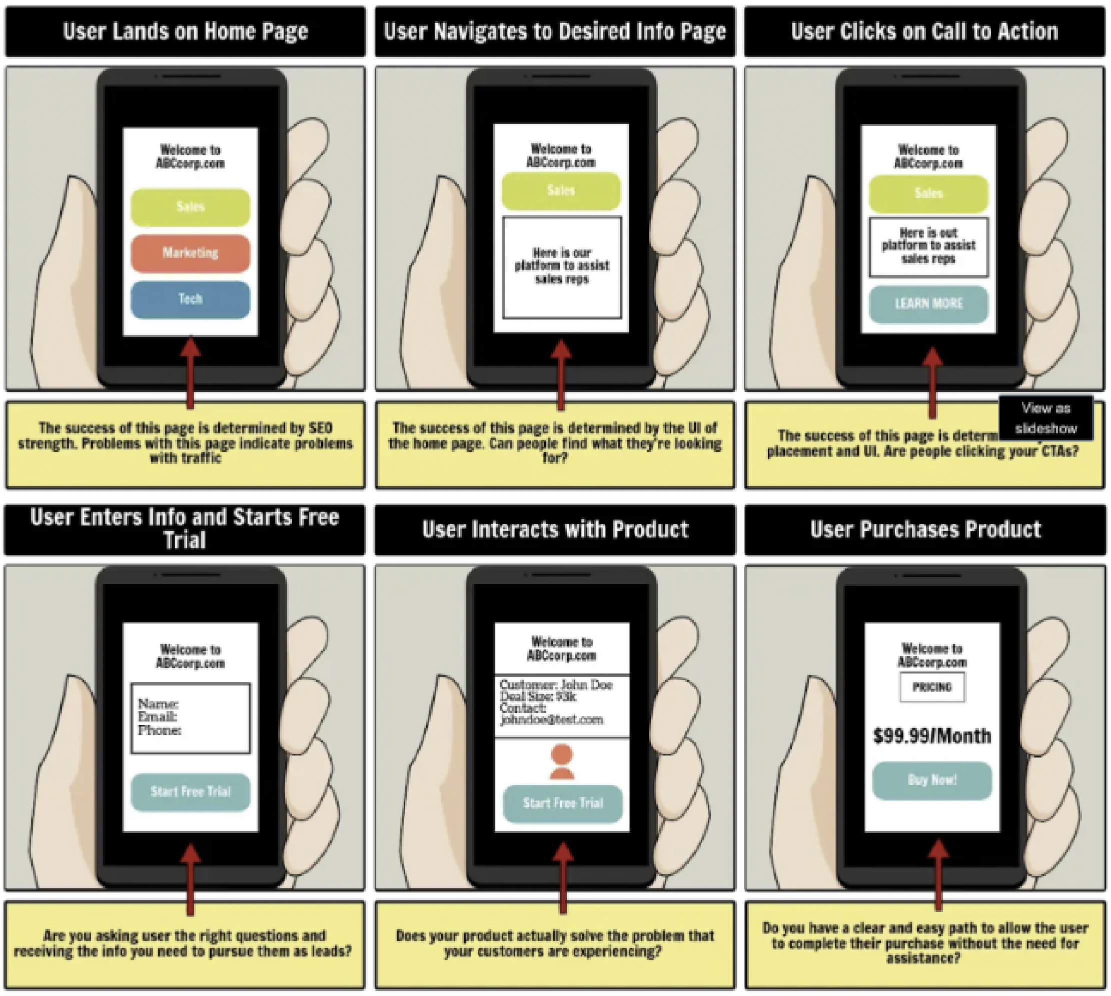

User Lands on Home Page
Welcome to ABCcorp.com
Sales
Marketing
Tech
The success of this page is determined by SEO strength. Problems with this page indicate problems with traffic
User Navigates to Desired Info Page
Welcome to ABCcorp.com
Sales
Here is our platform to assist sales reps
The success of this page is determined by the UI of the home page. Can people find what they're looking for?
User Clicks on Call to Action
Welcome to ABCcorp.com
Sales
Here is out platform to assist sales reps
LEARN MORE
The success of this page is determ
placement and UI. Are people clicking your CTAs?
View as slideshow
User Enters Info and Starts Free Trial
Welcome to ABCcorp.com
Name:
Email:
Phone:
Start Free Trial
Are you asking user the right questions and receiving the info you need to pursue them as leads?
User Interacts with Product
Welcome to ABCcorp.com
Customer: John Doe
Deal Size: 93k
Contact: johndoe@test.com
Start Free Trial
Does your product actually solve the problem that your customers are experiencing?
User Purchases Product
Welcome to ABCcorp.com
PRICING
$99.99/Month
Buy Now!
Do you have a clear and easy path to allow the user to complete their purchase without the need for assistance?

Figure A.4: Example of Story Board

A.2 Appendix B

A.2.1 Domain Model Example For Online Shopping

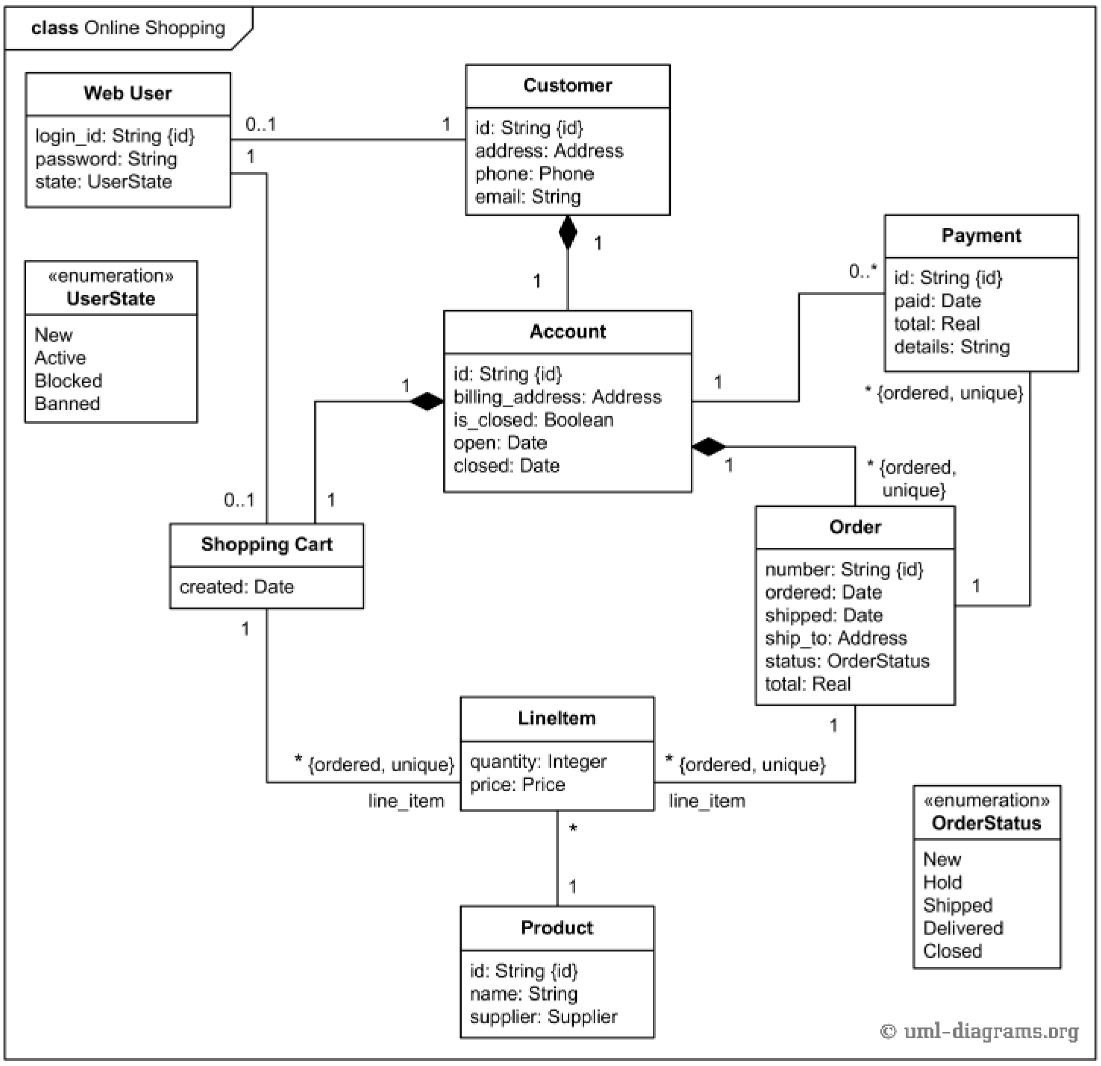


Figure A.5: Domain Model Example For Online Shopping Application

A.3 Appendix C

A.3.1 Box And Line Example For Online Shopping

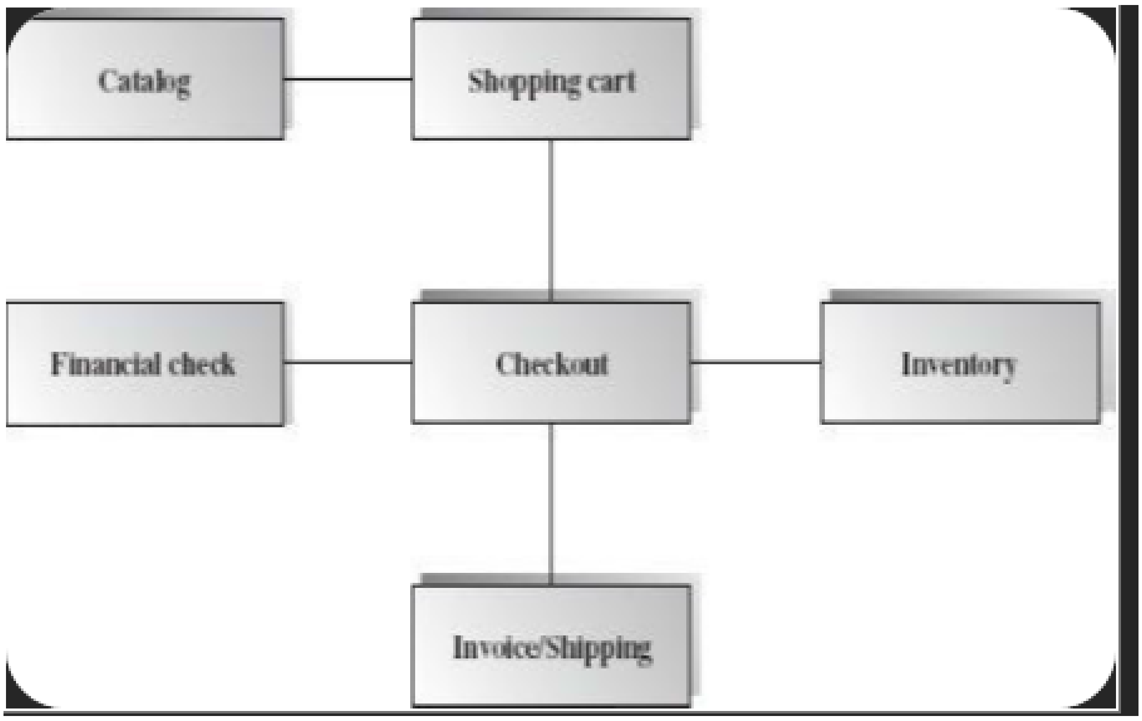


Figure A.6: Box and Line Diagram For Online Shopping Application

A.3.2 Architecture Pattern Example For Online Shopping

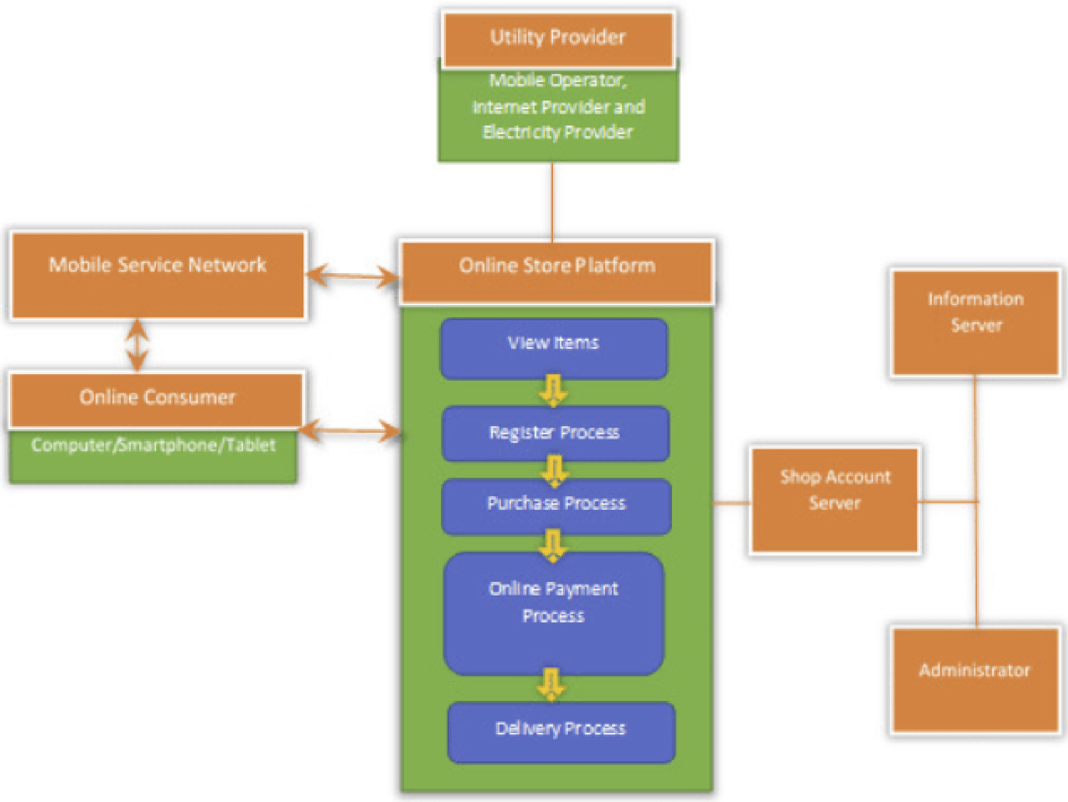


Figure A.7: Architecture Pattern For Online Shopping Application

A.4 Appendix D

A.4.1 Activity Diagram

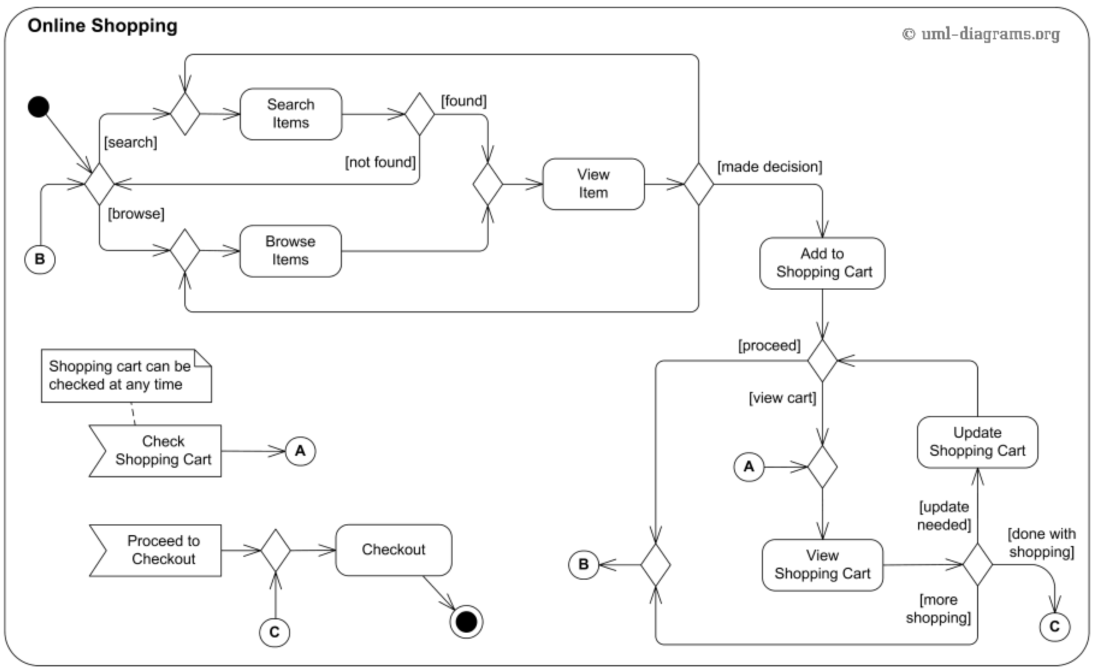


Figure A.8: Activity Diagram For Online Shopping Application

A.4.2 Class Diagram

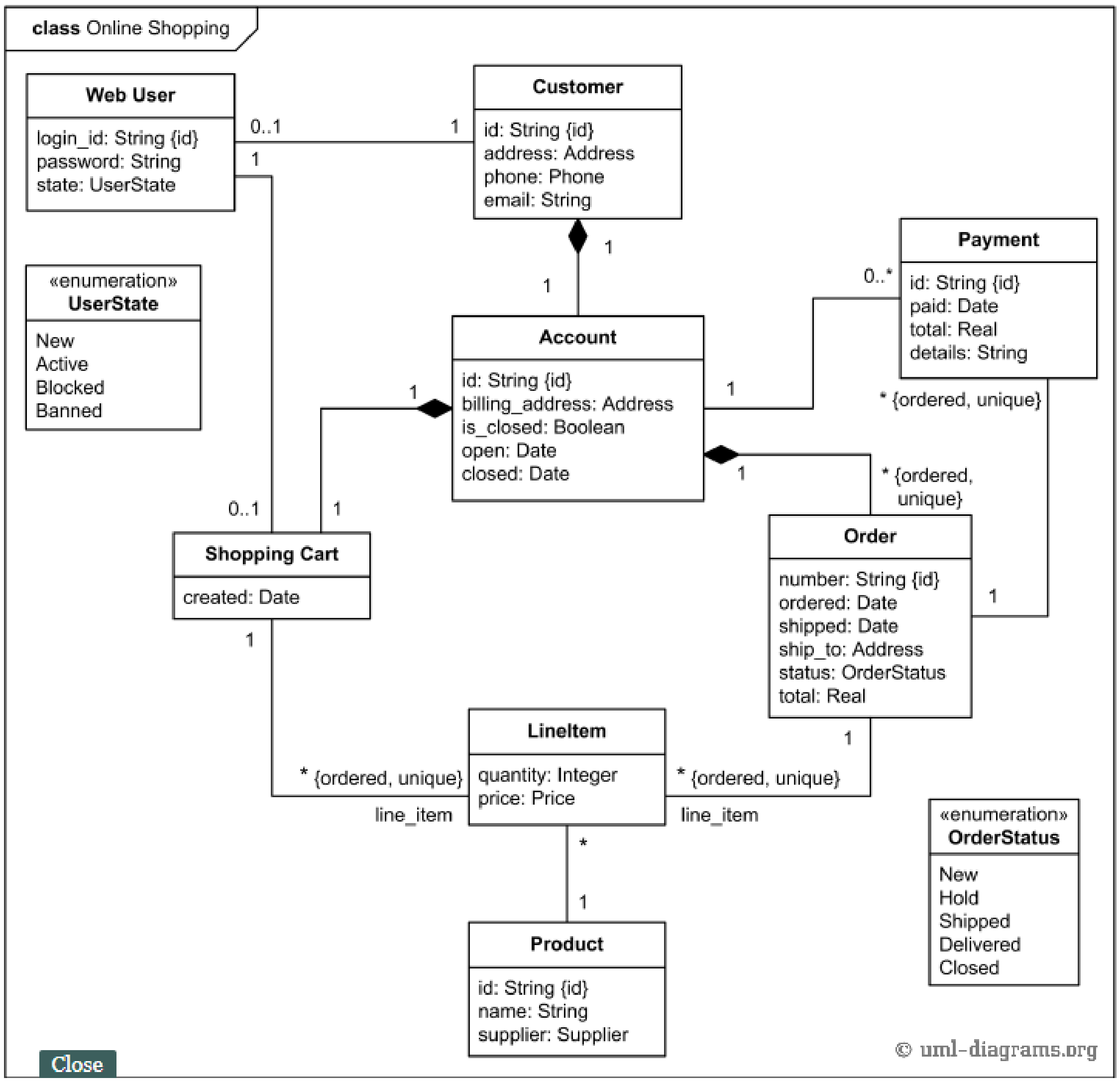


Figure A.9: Class Diagram For Online Shopping Application

A.4.3 Sequence Diagram

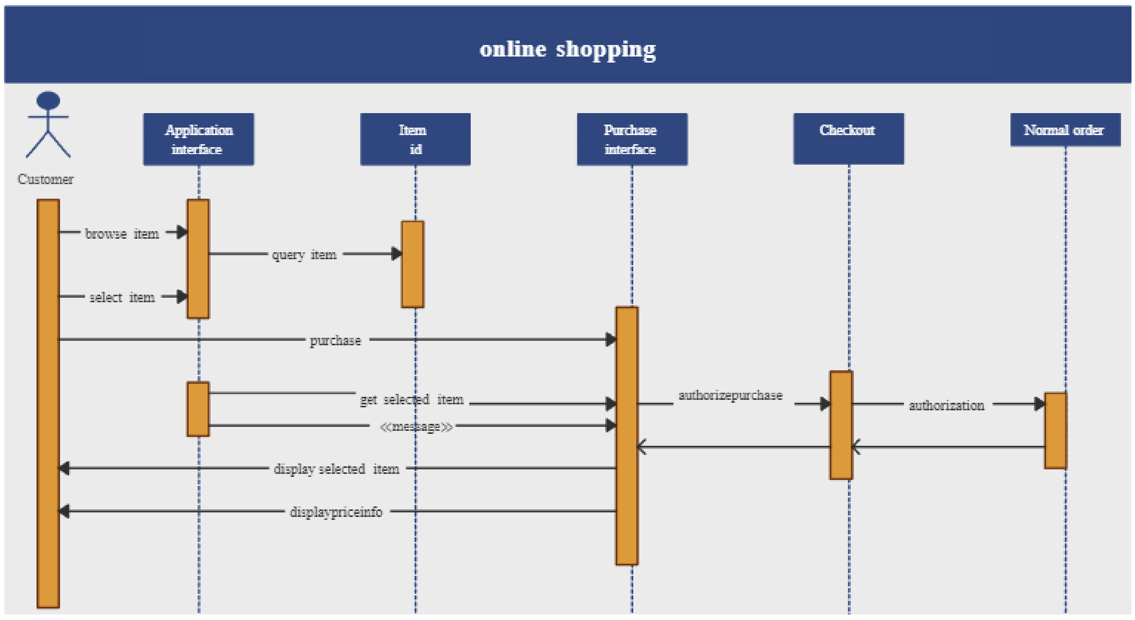


Figure A.10: Sequence Diagram For Online Shopping Application

A.4.4 State Transition Diagram

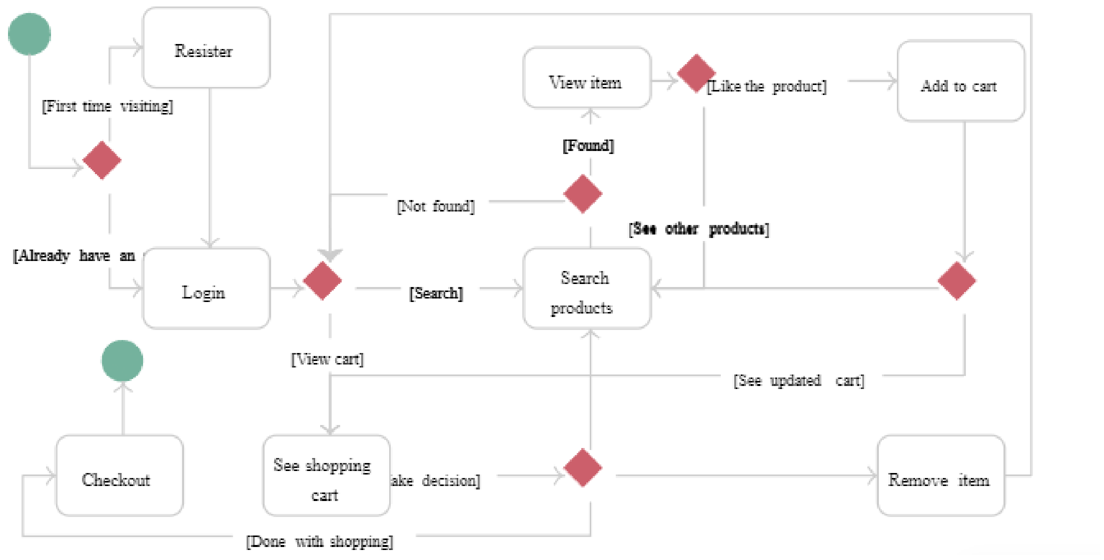


Figure A.11: State Transition Diagram For Online Shopping Application

A.4.5 Data Flow Diagram

Data Flow Diagram

Data flow diagram symbols, symbol names, and examples

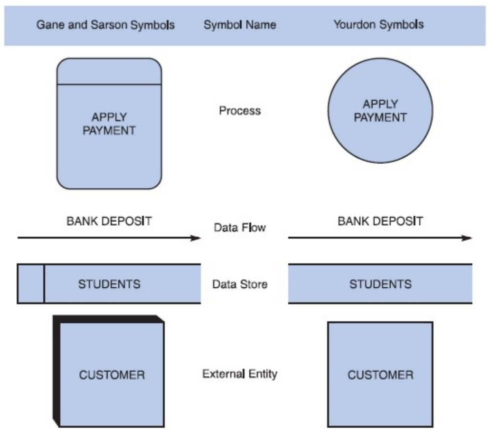


Figure B-5 Data flow diagram symbols, symbol names, and examples of the Gane and Sarson and Yourdon symbol sets.

Guidelines for Drawing DFDs

Step 1: Draw a Context Diagram: The first step in constructing a set of DFDs is to draw a context diagram. A **context diagram** is a top-level view of an information system that shows the system's boundaries and scope. Data stores are not shown in the context diagram because they are contained within the system and remain hidden until more detailed diagrams are created.

Example

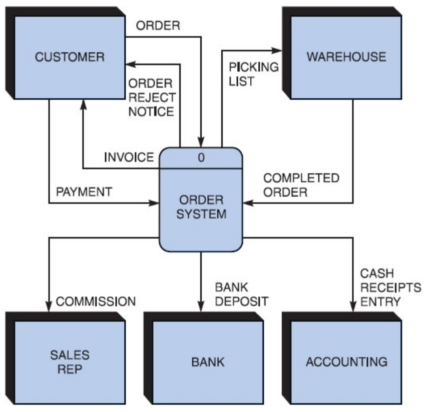


Figure B-6 Context diagram DFD for an order system.

Step 2: Draw a Diagram 0 DFD: To show the detail inside the black box, you create DFD diagram 0. **Diagram 0** zooms in on the system and shows major internal processes, data flows, and data stores. Diagram 0 also repeats the entities and data flows that appear in the context diagram. When you expand the context diagram into DFD diagram 0, you must retain all the connections that flow into and out of process 0.

Example

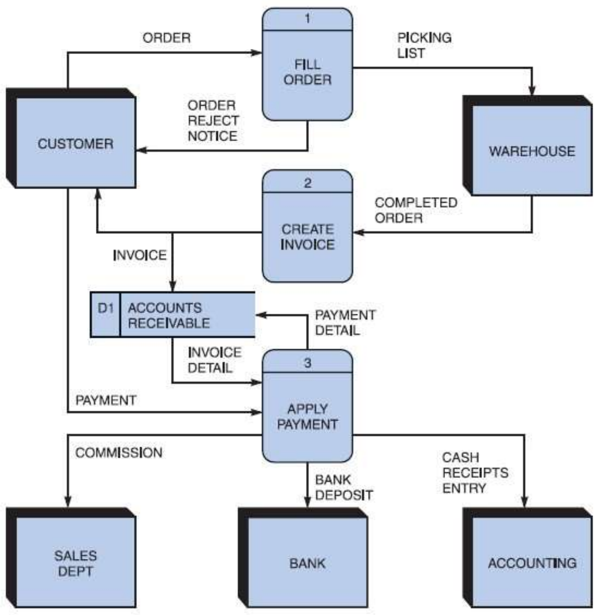


Figure B-7 Diagram 0 DFD for the order system.

Step 3: Draw the Lower-Level Diagrams:

To create lower-level diagrams, you must use leveling and balancing techniques. **Leveling** is the process of drawing a series of increasingly detailed diagrams, until all functional primitives are identified.

Leveling Example

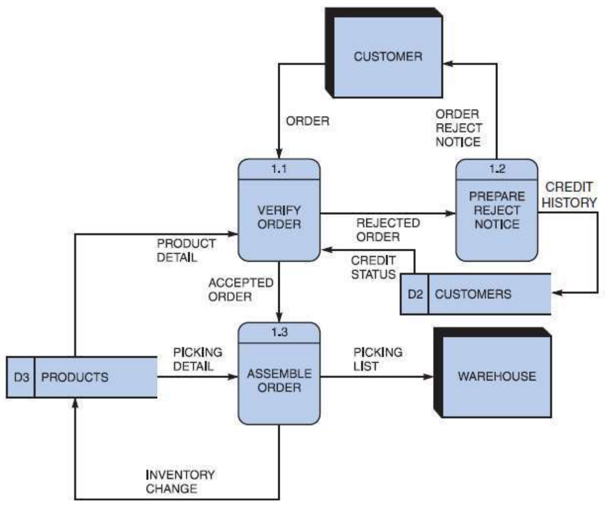


Figure B-8 Diagram 1 DFD shows details of the FILL ORDER process in the order system.