SECJ2203: Software Engineering

# System Documentation (SD)

Project Title

Version X

Date

Faculty of Computing

Prepared by: <Team name>

# Revision Page

a. **Overview**

b. **Target Audience**

c. **Project Team Members**

| Member Name | Role | Task | Status |
|---|---|---|---|
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

d. **Version Control History**

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| 1.0 | Team leader 1 (Full Name) | Completed Chapter X, Section… | dd/mm/yyyy |
| … |  |  |  |
|  |  |  |  |

**Note:**

# Table of Contents

*Update this table of content accordingly*

# 1.

# Introduction

## 1.1 Purpose

## 1.2 Scope

## 1.3 Definitions, Acronyms and Abbreviation

| Term | Definition |
|---|---|
| Emotion Chatbot Analyzer | The software product being developed to analyze the user input in a chatbot conversation and determine the user's emotion based on the language and sentiment used in the conversation. |
| SRS | System Requirements Specification - a document that outlines the requirements and objectives of the software product being developed. |
| SDD | System Design Document - a document that describes the system architecture, components, and interfaces in detail. |
| STD | System Testing Document - a document that outlines the testing process and procedures for the software product being developed. |
| API | Application Programming Interface - a set of protocols and tools for building software applications. |
| UI | User Interface - the graphical layout of an application or website, including buttons, menus, and other visual elements. |
| UX | User Experience - the overall experience a user has while interacting with a product, including ease of use, efficiency, and satisfaction. |
| NLP | Natural Language Processing - a branch of artificial intelligence that deals with the interaction between computers and human language. |
| ML | Machine Learning - a type of artificial intelligence that allows software applications to learn and improve from experience without being explicitly programmed. |

## 1.4 References

*This subsection should:*
a) *Provide a complete list of all documents referenced elsewhere in the SD;*
b) *Identify each document by title, report number (if applicable), date, and publishing organization;*
c) *Specify the sources from which the references can be obtained.*

[Specify a complete list of references using a standardized reference format]

## 1.5 Overview

*This subsection should:*
a) *Describe what the rest of the SD contains;*
b) *Explain how the SD is organized.*

[Give an overview of the content of this SD document.]

## 2.

# Specific Requirements

## 2.1 **User characteristics**

*The "Emotion Chatbot Analyzer" software will be used by two main types of users: UTM students and UTM counselors.*

*2.1.1 UTM Students*

- *The UTM students who will use the software are expected to have basic computer skills, including familiarity with web-based applications.*

- *They may have varying levels of emotional intelligence and may experience a wide range of emotional states, from mild stress to severe anxiety or depression.*

- *Some students may be reluctant to seek counseling or may not have access to professional ccounseling services, making the chatbot a valuable resource for emotional support and guidance.*

*2.1.1  UTM Counsellors*

- *The UTM counsellors who will use the software are expected to have professional training and experience in counselling and psychology.*

- *They may have varying levels of technical expertise, but will receive training on how to use the software.*

- *They will use the software to access student emotion records and provide counselling services to students as needed.*

- *Both types of users will be required to provide personal information such as name and email address in order to register for the software. The software will be designed with privacy and security measures in place to protect user information.*

## 2.2 **System Features**

***Figure X.X: Use Case Diagram for <Name of the System>***

| Module | Function | Description |
|---|---|---|
| User Module | UC01 – Register | This use case allows students to sign up as a user for the mobile application. |
| | | |
| | | |

*[Include domain model i.e. **class diagram** without the operation part – only attributes without details on visibility and type, explain each class and its attributes including the relationships among the classes, see the example.]*
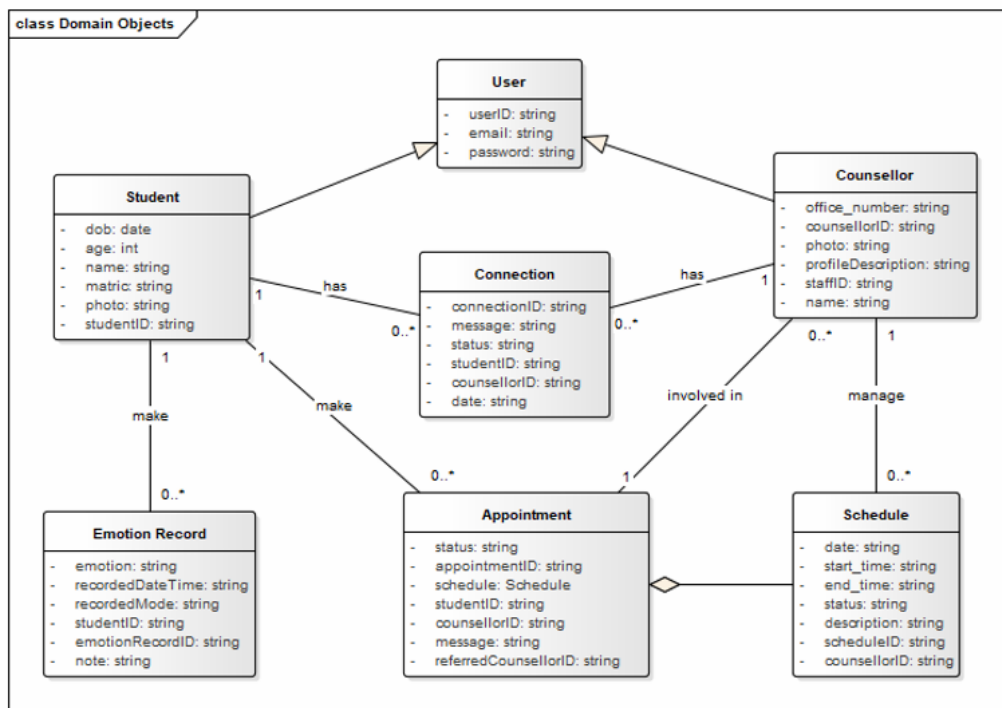


Figure X.X: Domain Model for <Name of the System>



uml-diagrams.org

8
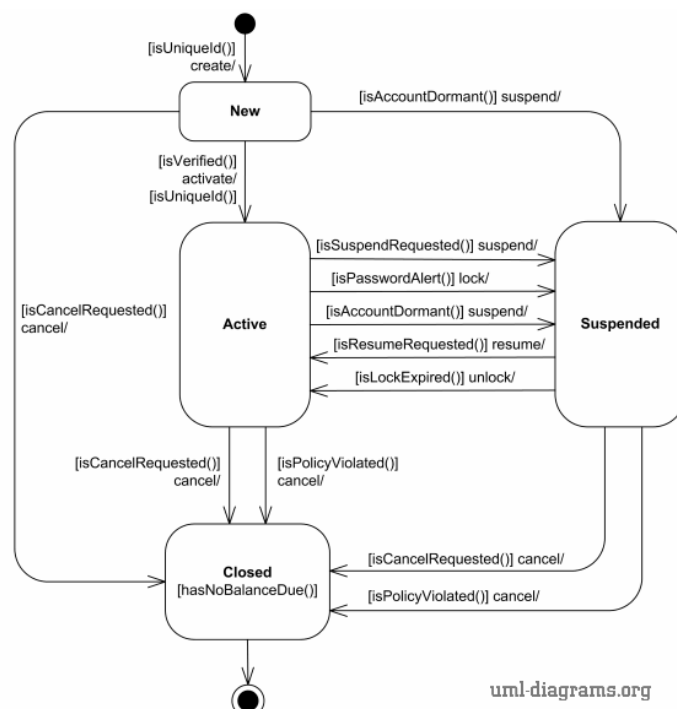
## 2.3 Use Case Details

### 2.3.1  UC001: Use Case <Name of Use Case 1>

*[Provide code for each use case such as UC001 and so on… Include Use Case Description for each use case as in the example below. If there are any alternative flows, add the rows accordingly. Otherwise, just remove the rows. Ensure each use case has its own unique ID and the name of the heading above corresponds to the name of use case in the use case diagram (refer to Figure X.X). Different scenarios of a use case require separate use case descriptions.]*

**Table X.X: Use Case Description for <Name of Use Case>**

| Use case: <Name of Use Case> |
| --- |
| **ID**: UCxxx |
| **Actors**: |
| **Preconditions**: |
| **Flow of events:**<br>1.<br>2.<br>… |
| **Postconditions:** |
| **Alternative flow *n*:** |
| **Postconditions:** |
| **Exception flow (if any):** |

*[Include sequence diagram and activity diagram for each respective user case. See example below. May consider including different scenarios in different diagrams, if necessary, to avoid clutter.]*
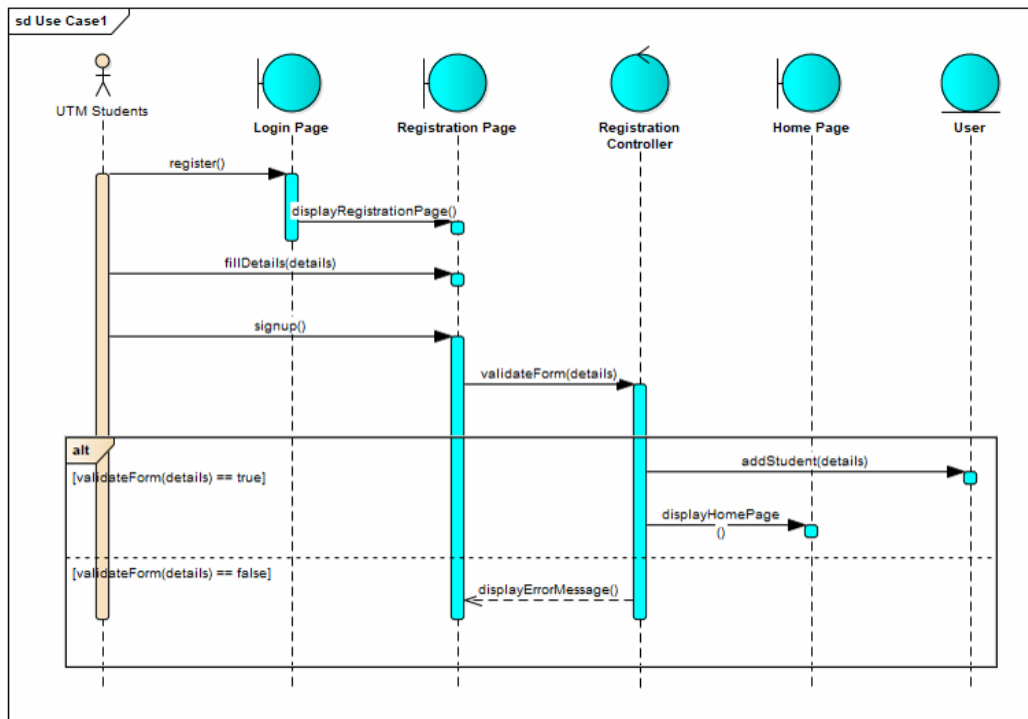
*Figure X.X: Sequence Diagram for <Name of Use Case/Scenario if more than one scenario>*
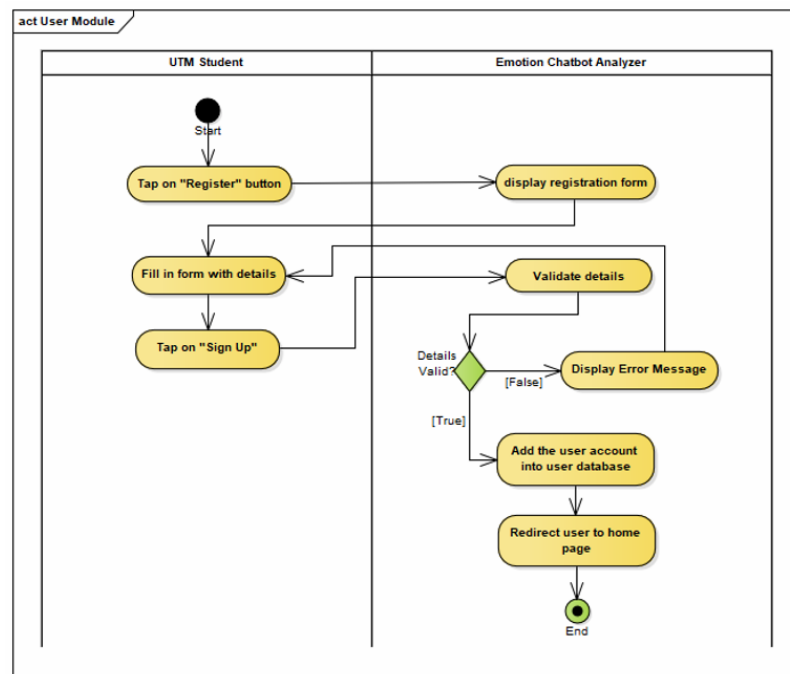


*Figure X.X: Activity Diagram for <Name of Use Case/Scenario if more than one scenario>*

### 2.3.2 UC002: Use Case <Name of Use Case 2>

### 2.3.3 UCn: Use Case <Name of Use Case n>

**…**

## 2.4 Performance and Other Requirements

*Non-functional requirements are the requirements that describe how the system should behave or operate, rather than what it should do. They address the characteristics or qualities of the system, such as its usability, reliability, performance, security, maintainability, and compatibility. Non-functional requirements are important to write in the SD because they help ensure that the system meets the stakeholders' needs and expectations.The ISO/IEC/IEEE 29148 standard suggests that non-functional requirements should be specified under three main categories: Software System Attributes, Performance and Other Requirements.*

*Example of Software System Attributes are as follows:*
*Software system attributes define the overall qualities or characteristics of the software. These attributes are the foundation on which the software is built, and they include the following:*
- *Usability: The ease with which users can interact with the system.*
- *Reliability: The ability of the system to perform its functions correctly and consistently.*
- *Maintainability: The ease with which the system can be modified, repaired, and enhanced.*
- *Portability: The ability of the system to run on different hardware and software platforms.*
- *Compatibility: The ability of the system to work with other systems and components.*

*Example of Performance is as follows:*
*Performance requirements define the system's capability to respond to user requests and handle data in a timely manner. These requirements include the following:*
- *Response Time: The time it takes for the system to respond to a user request.*
- *Throughput: The number of requests the system can handle in a given period of time.*
- *Capacity: The maximum number of users or amount of data that the system can handle.*
- *Availability: The percentage of time the system is operational and accessible to users.*

*Example of Other Requirements is as follows:*
*Other requirements refer to any non-functional requirements that do not fit under the categories of software system attributes or performance. These requirements include the following:*
- *Security: The protection of the system and its data from unauthorized access and malicious attacks.*
- *Safety: The ability of the system to operate safely without causing harm to users or the environment.*
- *Legal and Regulatory: Compliance with laws, regulations, and standards.*
- *Environmental: The impact of the system on the environment.*

## 2.5 Design Constraints

*Explain any constraints imposed by the organization where the software product will be used such as the system must adhere to certain organizational standard and other external non-functional requirements.*

*Example is as follows:*
*Environmental constraints: These constraints relate to the physical environment in which the system will operate, such as temperature, humidity, and lighting conditions. An example of an environmental constraint is: "The system must be designed to operate in temperatures ranging from 0°C to 40°C."*

*Hardware constraints: These constraints relate to the specific hardware components that will be used in the system, such as processors, memory, and storage. An example of a hardware constraint is: "The system must be designed to run on a minimum of 8GB of RAM."*

*Security constraints: These constraints relate to the security requirements for the system, such as data encryption, authentication, and access control. An example of a security constraint is: "The system must be designed to encrypt all sensitive user data using AES-256 encryption."*

*Compatibility constraints: These constraints relate to the compatibility requirements for the system, such as compatibility with specific software or hardware components. An example of a compatibility constraint is: "The system must be designed to be compatible with Windows 10 operating system."*

# 3. **System Architectural Design**

## 3.1 **Architecture Style and Rationale**

State here

## 3.2 **Component Model**
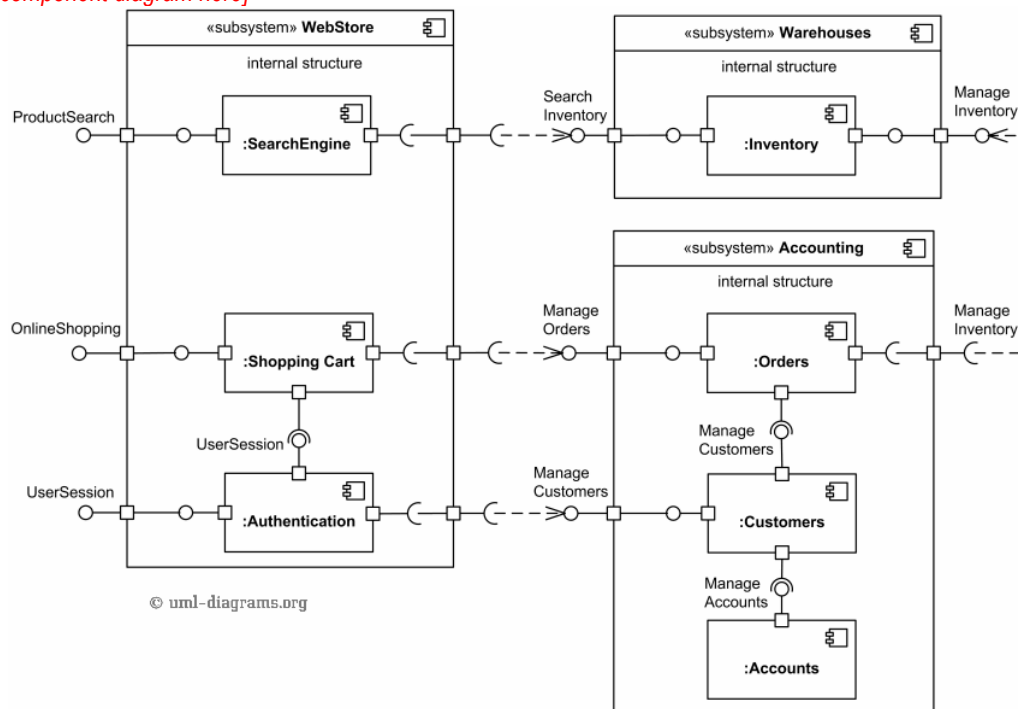
**Figure 3.1: Component Diagram of <Name of the System>**

4.

# Detailed Description of Components

*This section of SD serves as part of SDD that describes each module or subsystem in the project.*

## 4.1 Complete Package Diagram

*Include the overall package diagram of your system here. [Example package diagram for a Sale System adapted from Satzinger (2011)] Indicate the navigation visibility based on the dependency among classes in the design class diagram. If the diagram is too cluttered, simplify the classes by showing the class name only without showing the attributes and methods. The details can be shown in the following class diagram sub-section for respective subsystem/package.*
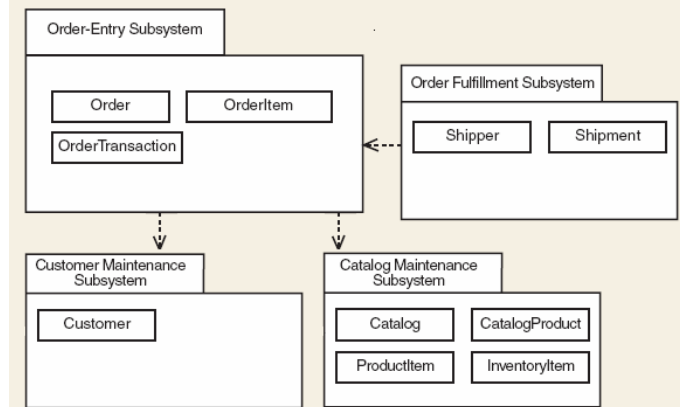


**Figure 4.1: Package Diagram for <Name of the System>**

## 4.2 Detailed Description

*For each subsystem/package there must be ONE class diagram and several sequence diagrams based on how many use cases you have in the subsystem/package. Use branching in sequence diagram to combine alternate flow in the same sequence diagram. If the diagram is cluttered, consider a new sequence diagram for respective scenario/alternate flow. The given example includes view, domain and data access layer in respective packages. Organise subsystem/package according to the chosen architectural style. Note that if you choose model-view-controller (MVC) or other architectural styles, then the packages should follow the selected styles. However, for the scope of this course, you may follow the example.*

### 4.2.1 P001: <Name of Package 1> Subsystem



**Figure 4.2: Package Diagram for <Name of Package 1> Subsystem**

#### 4.2.1.1 Class Diagram

*Include class diagram to represent all classes in the respective subsystem/package. Include the controller classes.*
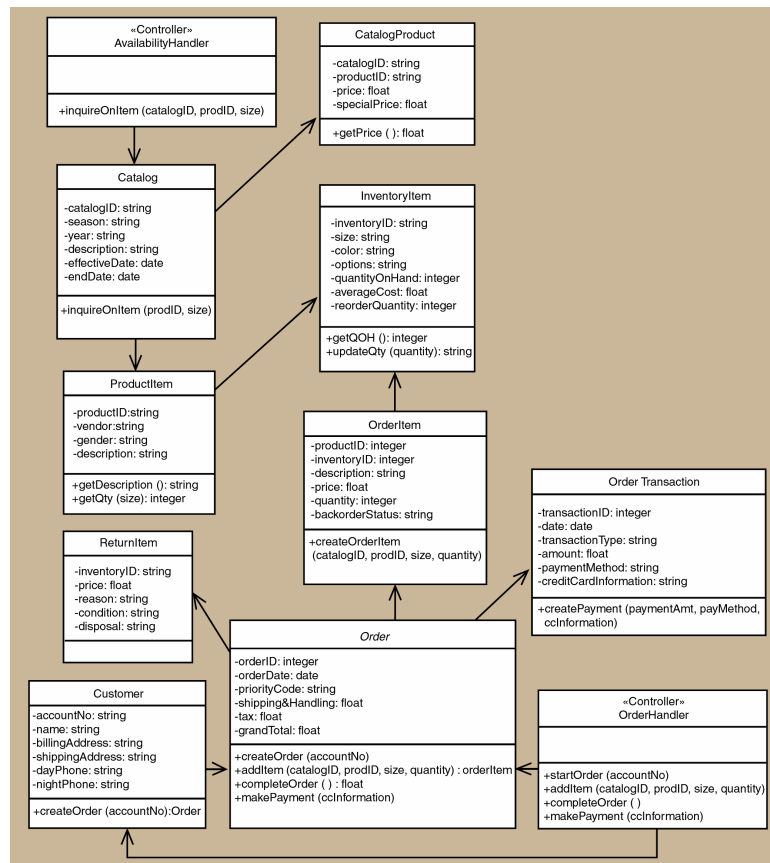
**Figure 4.3: Class Diagram for <Name of Package 1> Subsystem**

*List all methods in a table for respective entity (add the table accordingly), then write its algorithm. Example of algorithm as shown below.*
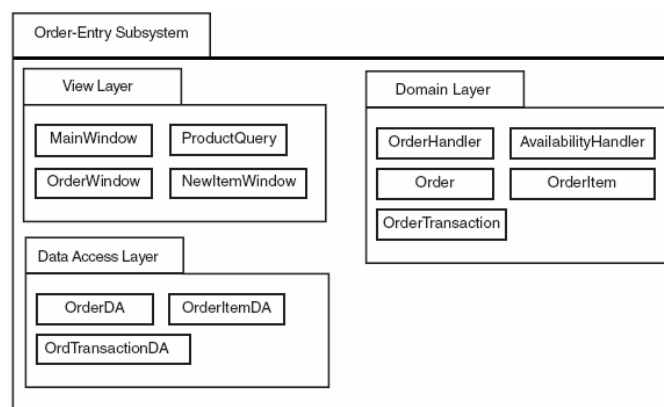
*Step1: Start*

*Step2: Read/input A and B*

*Step3: If A greater than B then C=A*

*Step4: If B greater than A then C=B*

*Step5: Print C*

*Step6: End*

| Entity Name | e.g. Order |
|---|---|
| Method Name | e.g. createOrder |
| Input | |
| Output | |
| Algorithm | 1. Start<br>2. …<br>3. End |

## 4.2.1.2     Sequence Diagram

*Include sequence diagram for each <u>respective use case in your package</u>. In these examples only sequence diagram Create New Phone Order Scenario and Cancel an Order Scenario are shown. Include the final sequence diagram that comprises view layer, controller, and its problem domain (entity) and data access layer. Provide a unique code for each scenario of*

15

a) SD001: Sequence diagram for Create New Phone Order



**Figure 4.4: Sequence Diagram for <Create New Phone Order Scenario>**

b) SD002: Sequence diagram for Create Cancel an Order Scenario
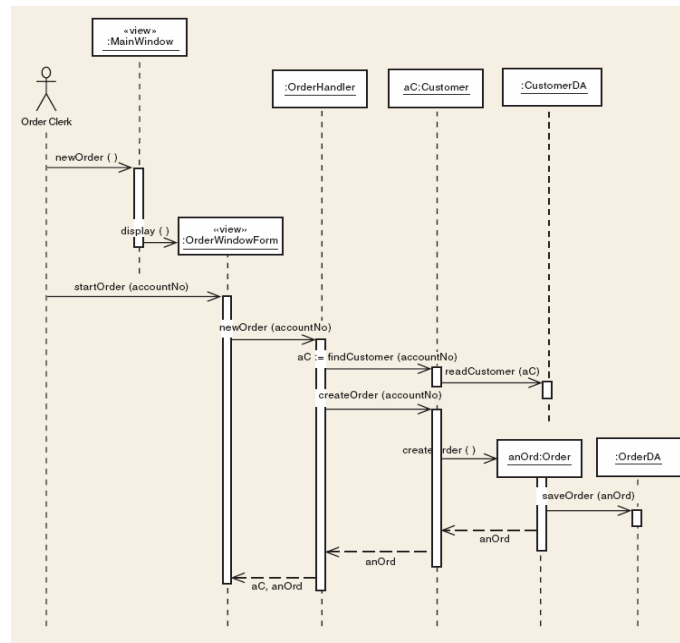


**Figure 4.5: Sequence Diagram for <Cancel an Order scenario>**

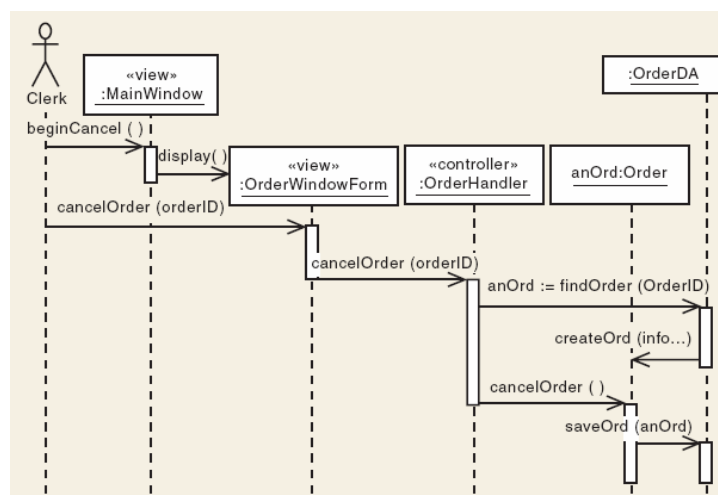4.2.2   **P002: <Name of Package 2> Subsystem**

4.2.2.1        **Class Diagram**

4.2.2.2        **Sequence Diagram**

### 4.2.3  P003: <Name of Package *n*> Subsystem

#### 4.2.3.1        Class Diagram

#### 4.2.3.2        Sequence Diagram

# 5.

# Data Design

## 5.1 Data Description

*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 the database(s) or data storage items. For a small system, there is normally only one database. It consists of all the tables in which for object-oriented they are classes/objects as also stated in the domain model. Use table for the listing.*

The major data or systems entities are stored into a relational database named as…, processed and organized into *n* entities as listed in Table 5.1.

**Table 5.1: Description of Entities in the Database**

| No. | Entity Name | Description |
|-----|-------------|-------------|
|     |             |             |
|     |             |             |

## 5.2 Data Dictionary

*Alphabetically list the system entities or major data along with their types and descriptions (class/object, attributes). Focus on classes in domain layer; omit the controller/handler class. Use tables for easy listing as shown below.*

### 5.2.1 Entity: <First Entity>

| Attribute Name | Type | Description |
|----------------|------|-------------|
|                |      |             |
|                |      |             |

### 5.2.2 Entity: <Second Entity>

| Attribute Name | Type | Description |
|----------------|------|-------------|
|                |      |             |
|                |      |             |

# 6.

# User Interface Design

## 6.1 Overview of User Interface

The interface includes…

## 6.2 Screen Images

Insert the screenshot here

**Figure**      **6.1:**

**Interface for <State the Feature/Function>**

# 7.

# Requirements Matrix

The sequence diagrams for each use case vs. corresponding classes (entities) are listed as in Table 7.1.

### Table 7.1: Description of Entities in the Database

|  | Customer | Order | Catalog | CatalogProduct | ProductItem | OrderItem | ReturnItem | InventoryItem | … |
|---|---|---|---|---|---|---|---|---|---|
| P001, UC001, SD001 | X | X |  |  |  |  |  |  |  |
| P001, UC001, SD002 |  | X |  |  |  |  |  |  |  |
| P002, UC002, SD003 |  |  | X | X |  |  |  |  |  |
| P003, UC003, SD004 |  |  |  |  | X | X | X | X |  |
| P004, UC004, SD005 |  | X |  |  |  |  |  | X |  |
| … |  |  |  |  |  |  |  |  |  |

8.

# Test Cases

## 8.1 TC001: Test <Name of Package 1> Subsystem: <Name of Use Case (UC001)>

This test contains the following test cases:
- (a) TC001_01: Test <Scenario of sequence diagram1 (SD001)>
- (b) TC001_02: Test <Scenario of sequence diagram2 (SD002)>
- (c) …

### 8.1.1 TC001_01: Test <state scenario of sequence diagram1 (SD001)>

This test contains the following alternate and exception scenarios (if any):
- (a) TC001_01_01: Test <alternate scenario1 of sequence diagram1 (SD001)>
- (b) TC001_01_02: Test <exception scenario1 of sequence diagram1 (SD001)>
- (c) …

| Test Case ID | BU_001 | Test Case Description | Test the Login Functionality in Banking | | |
|---|---|---|---|---|---|
| Created By | Mark | Reviewed By | Bill | Version | 2.1 |
| | | | | | |
| QA Tester's Log | | Review comments from Bill incorprate in version 2.1 | | | |
| | | | | | |
| Tester's Name | Mark | Date Tested | 1-Jan-2017 | Test Case (Pass/Fail/Not | Pass |

| S # | Prerequisites: | | S # | Test Data | |
|---|---|---|---|---|---|
| 1 | Access to Chrome Browser | | 1 | Userid = mg12345 | |
| 2 | | | 2 | Pass = df12@434c | |
| 3 | | | 3 | | |
| 4 | | | 4 | | |

| Test Scenario | Verify on entering valid userid and password, the customer can login |
|---|---|

| Step # | Step Details | Expected Results | Actual Results | Pass / Fail / Not executed / Suspended |
|---|---|---|---|---|
| 1 | Navigate to http://demo.guru99.com | Site should open | As Expected | Pass |
| 2 | Enter Userid & Password | Credential can be entered | As Expected | Pass |
| 3 | Click Submit | Cutomer is logged in | As Expected | Pass |

### 8.1.2 TC001_02: Test <Scenario of sequence diagram2 (SD002)>

### 8.1.3 TC001_n: Test <Scenario of sequence diagram n (…)>

## 8.2 TC002: Test <Name of Package 2> Subsystem: <Name of Use Case (UC002)>

*List all test cases before providing the details for the second use case in module1. Include the sub-sections accordingly.*

This test contains the following test cases:

    (a) TC002_01: Test <Scenario of sequence diagram4 (SD004)>
    (b) TC002_02: Test <Scenario of sequence diagram5 (SD005)>
    (c) …

## 8.3 TC003: Test <Name of Package 3> Subsystem: <Name of Use Case (UC003)>

*List all test cases before providing the details for the first use case in module2. Include the sub-sections accordingly.*

This test contains the following test cases:

    (a) TC003_01: Test <Scenario of sequence diagram6 (SD006)>
    (b) TC003_02: Test <Scenario of sequence diagram7 (SD007)>
    (c) …

## Appendix A: Traceability Matrix

| Test Case ID | Use Case ID/ Sequence Diagram ID | Package ID |
|---|---|---|
| TC001 for <Name of Package 1> Subsystem<br>• TC001_01<br>• TC001_02 | UC001<br>• SD001<br>• SD002 | P001 |
| TC002 for <Name of Package 2> Subsystem<br>• TC002_01<br>• TC002_02 | UC002<br>• SD004<br>• SD005 | P001 |
| TC003 for <Name of Package 3> Subsystem<br>• TC003_01<br>• TC003_02 | UC003<br>• SD006<br>• SD007 | P002 |
| … | | |