#### Project Title

#### 

#### Name 1 (Reg1)

#### Name 2 (Reg2)

#### Name 3 (Reg3)

***Supervised By***

**Supervisor Name**

***Fall 2022***

**BS Software Engineering**

**Department of Software Engineering**

**Capital University of Science & Technology, Islamabad**

|  |  |  |
| --- | --- | --- |
|  | Submission Form for Final-Year  PROJECT REPORT |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Version** | | V X.0 | |  | **NUMBER OF MEMBERS** |  |
|  | | | | | | |
| **TITLE** |  | | | | | |
|  | | | | | | |
| **SUPERVISOR NAME** | | |  | | | |

|  |  |  |
| --- | --- | --- |
| **MEMBER NAME** | **REG. NO.** | **EMAIL ADDRESS** |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **MEMBERS’ SIGNATURES** |  | |
|  |  | **Supervisor’s Signature** |
|  |  |
|  |  |

*Note 1: This paper must be signed by your supervisor*

*Note 2: The soft-copies of your project report, source codes, schematics, and executable should be delivered in a CD*

**APPROVAL CERTIFICATE**

This project, entitled as “Menu Drive (Insert Your Project Title Here) ” has been approved for the award of

**Bachelors of Science in Software Engineering**

**Committee Signatures:**

Supervisor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Mr.Qamar Uz Zaman)

Project Coordinator: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Mr. Ibrar Arshad)

Head of Department: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Dr. Nadeem Anjum)

**DECLARATION**

*I/We, hereby, declare that “No portion of the work referred to, in this project has been submitted in support of an application for another degree or qualification of this or any other university/institute or other institution of learning”. It is further declared thatthis undergraduate project, neither as a whole nor as a part thereof has been copied out from any sources, wherever references have been provided.*

|  |
| --- |
| **MEMBERS’ SIGNATURES** |
|  |
|  |
|  |

**ACKNOWLEDGEMENTS**

*It is usual to thank those individuals who have provided particularly useful assistance, technical or otherwise, during your project. Your supervisor will obviously be pleased to be acknowledged as he or she will have invested quite a lot of time overseeing your progress.*

**DEDICATION**

**This is an optional section**

*In this section you dedicate your project to anybody that you feel motivates you for hard work and putting effort for successful life.*

**Executive Summary**

This should be not more than one page in length (200 words approx.). The summary should allow the reader who is unfamiliar with the work to gain a swift and accurate impression of what the project is about, how it arose and what has been achieved.

*It is recommended, you write this section when the report is finished.*

**Table of Contents**

*This should give a complete list of what the report contains starting with the abstract (the title page is not included in the contents list).*

[Chapter 1 13](#_Toc111051919)

[Introduction 13](#_Toc111051920)

[1.1. Project Introduction 13](#_Toc111051921)

[1.2. Existing Examples / Solutions 13](#_Toc111051922)

[1.3. Business Scope 14](#_Toc111051923)

[1.4. Useful Tools and Technologies 14](#_Toc111051924)

[1.5. Project Work Break Down 14](#_Toc111051925)

[1.6. Project Time Line 15](#_Toc111051926)

[Chapter 2 16](#_Toc111051927)

[Requirement Specification and Analysis 16](#_Toc111051928)

[Requirement Specification 16](#_Toc111051929)

[2.1. Functional Requirements 17](#_Toc111051930)

[2.2. Non-Functional Requirements 17](#_Toc111051931)

[2.3. Selected Functional Requirements 18](#_Toc111051932)

[2.4. System Use Case Modeling 18](#_Toc111051933)

[2.5. System Sequence diagrams 21](#_Toc111051934)

[2.6. Domain Model 22](#_Toc111051935)

[Chapter 3 23](#_Toc111051936)

[System Design 23](#_Toc111051937)

[3.1. Software Architecture 24](#_Toc111051938)

[3.2. Class Diagram 25](#_Toc111051939)

[3.3. Sequence Diagram 26](#_Toc111051940)

[3.4. Entity Relationship Diagram 27](#_Toc111051941)

[3.5. Database Schema 28](#_Toc111051942)

[3.6. User Interface Design 29](#_Toc111051943)

[3.7. Software COTS 31](#_Toc111051944)

[Chapter 4 32](#_Toc111051945)

[Software Development 32](#_Toc111051946)

[4.1. Coding Standards 32](#_Toc111051947)

[4.2. Development Environment 33](#_Toc111051948)

[4.3. Software Description 33](#_Toc111051949)

[Chapter 5 35](#_Toc111051950)

[Software Testing 35](#_Toc111051951)

[5.1. Testing Methodology 35](#_Toc111051952)

[5.2. Testing Environment 35](#_Toc111051953)

[5.3. Test Cases 36](#_Toc111051954)

[Chapter 6 37](#_Toc111051955)

[Software Deployment 37](#_Toc111051956)

[6.1. Installation / Deployment Process Description 37](#_Toc111051957)

**List of Figures**

*If the report contains figures or tables a list of these should be provided. The list should give the table or figure number, the title of the table or figure and the page number. If only a few tables and figures are present, they may be treated on one page. Remember that all figures and tables used must be referred in the text. For example “The class diagram shown in Figure 2.1 ....”*

**Figure1. 1: Work breakdown Structure**

**Figure 1.2: Sample Gant Chart**

**Figure 2.1: Sample Use case Diagram**

**Figure 2.2: System Sequence Diagram**

**Figure 2.3: Domain Model**

**Figure 3.1: Software Architecture Diagram**

**Figure 3.2: UML Class Diagram**

**Figure 3.3: Sequence Diagram**

**Figure 3.4: Entity Relationship Diagram**

**Figure 3.5: Database Schema**

**Figure 3.6: Common GUI elements**

**Figure 3.7: Example Login Page UI Design with description in text**

**List of Tables**

**Table 2.1: Functional Requirements**

**Table 2.2: Functional and Non-Functional Requirement**

**Table 2.1: Use Case 1**

**Table 2.2: Use Case 1**

**Table 3.1: Data Dictionary**

This page is kept blank

# 

# Chapter 1

# Introduction

*A good introduction should tell the reader what the project is about without assuming special knowledge and without introducing any specific material that might obscure the overview. Introduction is mostly written for readers’ who are not experts of the field.*

## Project Introduction

* *Brief description of the project: Briefly describe that what the project is. It explains the immediate context of the problem you want to solve. To start writing your introduction, first come up with a simple one-sentence summary of the goal of your project. Then elaborate the statement briefly.*
* *Introduction of the beneficiary of the project: In this section you will describe the organization for which you intend to build this project. The origination could be a company or a single user who would benefit from your application in a particular way.*

## Existing Examples / Solutions

*In this section you are supposed to provide the survey of the existing systems or products that belong to the same domain that you have indentified in the previous section. This survey should include history and working details of prominent systems/products that solve the similar problem that you intend to solve. You should select a relevant case study carefully and also identify any short comings or all those features that are missing in the existing solutions and you intend to provide in your project.*

## Business Scope

This section will describe the project from the business point of view. It will identify its business potential with the target market.

*You can identify the additional potential customers and emphasize on the possibility of deploying this project as a commercial product. You can identify the effort and cost of all the additional resources you need in your project to achieve this aim.*

## Useful Tools and Technologies

This section should mention possible technologies that could be used during the designing, development and testing of your project.

*You should mention the technologies that you intend to use with a brief but technical justification for your decision. Your discussion should include*

1. *What programming language is you using and why?*
2. *Which development environment do you intend to use and why?*
3. *What database (if any) is you using and why?*
4. *Which operating systems will support your software?*
5. *What network protocol (if any) is implemented and why?*

## Project Work Break Down

All projects require planning, including an outline of **who** on the team is doing **what** and **when**; thus, you will need to include a **Work Breakdown** chart. *You must identify all the components of the project and also specify how much time you will spend on each component. The justification should include your strengths and weakness from the project point of view and it should indicate that you have allocated appropriate time period for those modules that you find yourself as your weak points. A typical software project is divided into several parts as shown in Figure 1*

Software Project

Requirements Gathering Student A

Requirement Analysis Student A, B

Design Student A, B

ImplementationStudent A, B

Testing Student A, B

Initial requirements

Detailed Requirements

Identification of Functional-Non Functional Requirements

Use Cases Student A, B

Sequence Diagram

Architecture Student A

Class Diagram Student B

ER Diagram Student B

Data Flow Diagram

User Interface Design

Module 1 Student B

Module 2 Student A

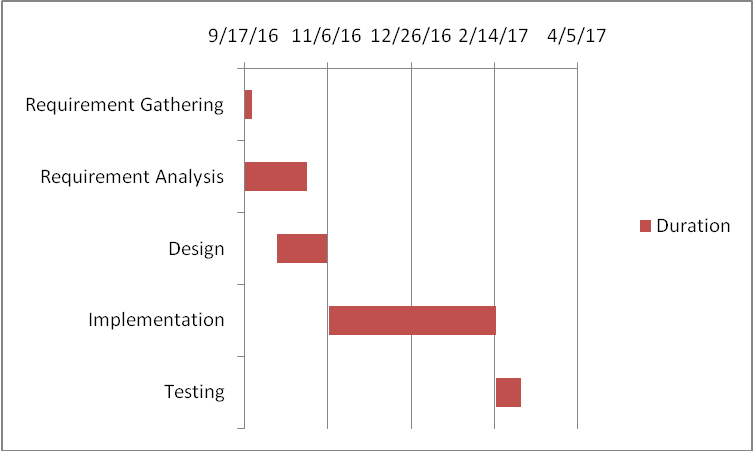
Sub Module Module 1 1 Day Student B

**Figure : Work breakdown Structure**

You are supposed to provide work breakdown structure of your project along with the assignment details that which group member will be performing which tasks.

## Project Time Line

A Gantt chart outlines what aspects of the project will be completed and by when. It is an important component of good project management and something you will probably be asked to do as a part of your job. A sample Gantt chart is shown in Figure 2.



**Figure 1.2: Sample Gant Chart**

# Chapter 2

# Requirement Specification and Analysis

The emphasis of this chapter is on getting an idea of what the requirements are for the intended software. Students who are doing a research related project would provide literature survey for their problems. They are expected to understand the relevant papers and provide summary of the existing work presented in each research paper. Such students should consult their project supervisor for the detailed instructions related to this chapter.

*You should write SRS in precise, clear and plain language so that it can be reviewed by a business analyst or customer representative with minimal technical expertise. However it also contains analytical models (use case diagrams, entity relationship diagrams, data dictionary etc.), which can be used for the detailed design and the development of the software system*.

## Requirement Specification

Requirements specification involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users and documentation of all aspects of the project development process from start to finish. Requirements are a description of how a system should behave or a description of system properties or attributes. It can alternatively be a statement of 'what' an application is expected to do.

## 2.1. Functional Requirements

The Functional Requirements Specification documents the operations and activities that a system must be able to perform. The Functional Requirements Specification is described in such a way that anyone from non-technical audience can understand. Readers should understand the system, but no particular technical knowledge should be required to understand the document.

* Functional Requirements should include:
* Descriptions of data to be entered into the system
* Descriptions of operations performed by each screen
* Descriptions of work-flows performed by the system
* Descriptions of system reports or other outputs
* Who can enter the data into the system
* How the system meets applicable regulatory requirements

**Table 2.1: Functional Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| **S. No.** | **Functional Requirement** | **Type** | **Status** |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |

## 2.2. Non-Functional Requirements

Non-functional requirements cover all the remaining requirements, which are not covered by the functional requirements. They specify criteria that judge the operation of a system, rather than specific behaviors, for example: “Modified data in a database should be updated for all users accessing it within 2 seconds”. Some typical non-functional requirements include performance, scalability, availability, reliability, maintainability, usability and security.

*You are supposed to discuss both the functional and non-functional requirement for your project in the Part 1 mid term report. Functional requirements describe what the system should do while non-functional requirements describe how the system works. The Format for presenting these requirements is given in Table 2.*

**Table 2.2: Functional and Non-Functional Requirement**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Non Functional Requirements** | **Category** |
| 1 |  |  |
| 2 |  |  |
| 3… |  |  |

## Selected Functional Requirements

List of selected functional requirements for current iteration.

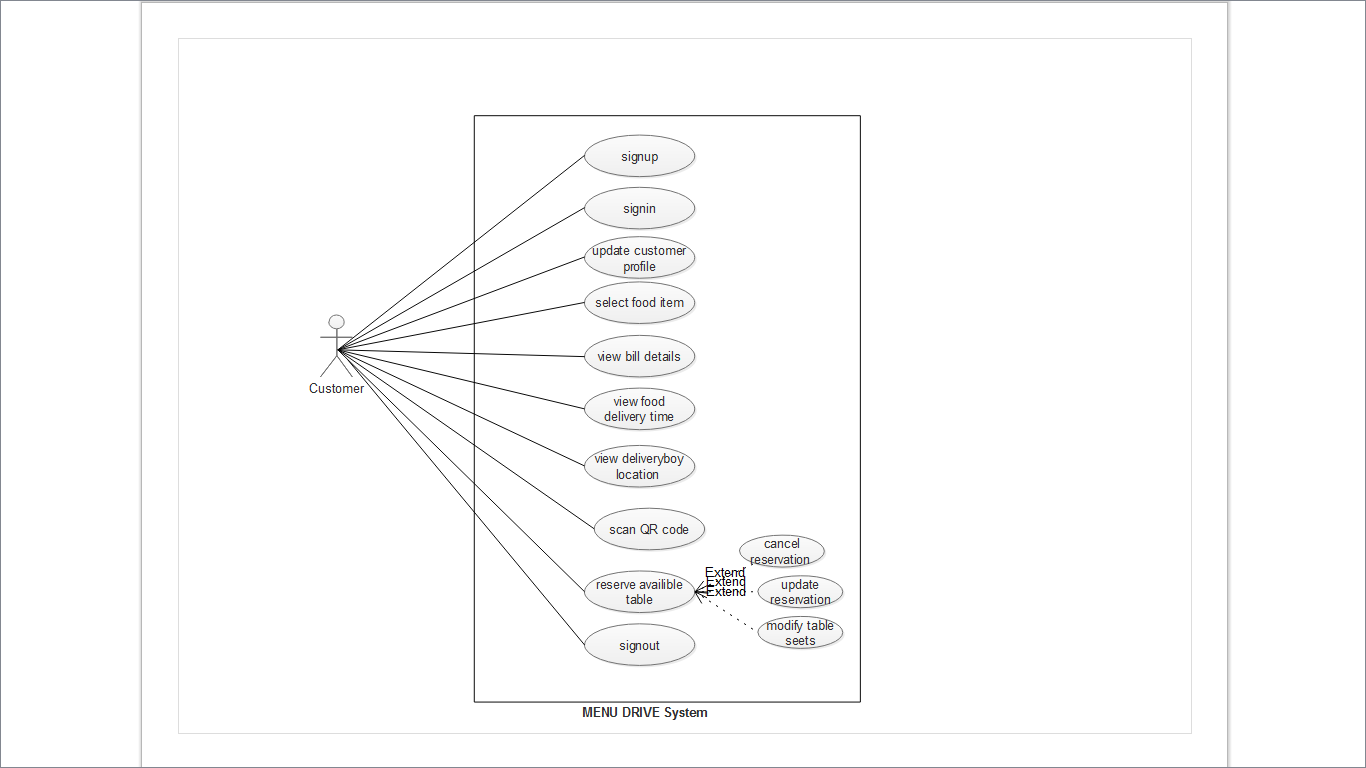
**Table 2.3: Selected Functional Requirement**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Functional Requirement** | **Type** |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |

## System Use Case Modeling

A use case defines a set of use-case instances, where each instance is a sequence of actions a system performs that yields an observable result of value to a particular actor. The functionality of a system is defined by different use cases, each of which represents a specific goal (to obtain the observable result of value) for a particular actor.

*You should develop fully dressed use cases. One way of conceptualize correct use case is by imaging the user interface of all the features of your project. This will help you to improve your project well in time.*

**

**Figure 2.1: Sample Use case Diagram**

**Use Case1 Title:**

*Describe the use case (expected behavior of the software) in the form of steps and sub steps in the format given below. You should also proved a brief description of user interface that will satisfy the requirement of each use case*

**Table 2.1: Use Case 1**

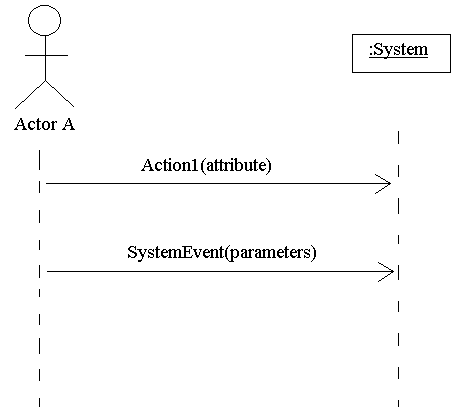
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case ID:** | [Repeat for multiple use cases] | | | | |
| **Use Case Name:** |  | | | | |
| **Created By:** |  | | **Last Updated By:** | |  |
| **Date Created:** |  | | **Last Revision Date:** | |  |
| **Actors:** | |  | | | |
| **Description:** | |  | | | |
| **Trigger:** | |  | | | |
| **Preconditions:** | |  | | | |
| **Post conditions:** | |  | | | |
| **Normal Flow:** | | Actor | | System | |
|  | |  | |
| **Alternative Flows:** | |  | | | |
| **Exceptions:** | |  | | | |

*Add further use cases in the given format*

## System Sequence diagrams

Sequence diagrams are created to show the sequence of events among user and the system to complete an action / use case. A sample is presented in Fig 2.2.

*You are required to provide SSD of all the uses cases that you have provided above.*

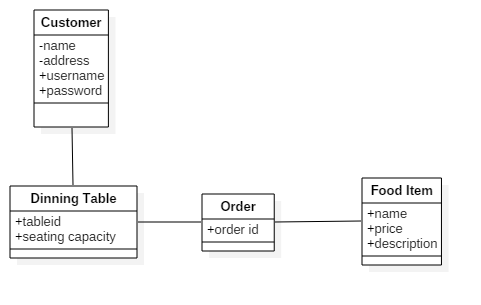


**Figure 2.2: System Sequence Diagram**

## Domain Model

Part of your initial architectural modeling efforts, particularly for a business application, will likely include the development of high-level domain model as you see in Fig. 2.3. This model should be very slim, capturing the main business entities and the relationships between them. Some people consider this type of model to be

an initial requirements model instead of an initial architecture model



**Figure 2.3: Domain Model**

# Chapter 3

# System Design

The purpose of this chapter is to provide information that is complementary to the code. Without an adequate design that delivers required function as well as quality attributes, the project will fail. But communicating architecture to its stakeholders is as important a job as creating it in the first place.

There are two views that are considered while defining software architecture. There are specific design artifacts that belong to each view. Description of such artifacts is given below. *You may select the artifacts depending on the nature of your project.*

* Structural View
  + Architecture diagram
  + Module structure diagram
  + Component diagram
  + Class diagram
* Behavioral View
  + Sequence diagram
  + Activity diagram
  + State machine diagram

At a high level, a software architecture document includes:

1. An outline description of the software design, including major software components and their interactions.
2. A common understanding of requirements, constraints and principles that influence the architecture.
3. A description of the hardware and software platforms on which the system is built and deployed.
4. Explicit justification of how the architecture satisfies the above mentioned points.

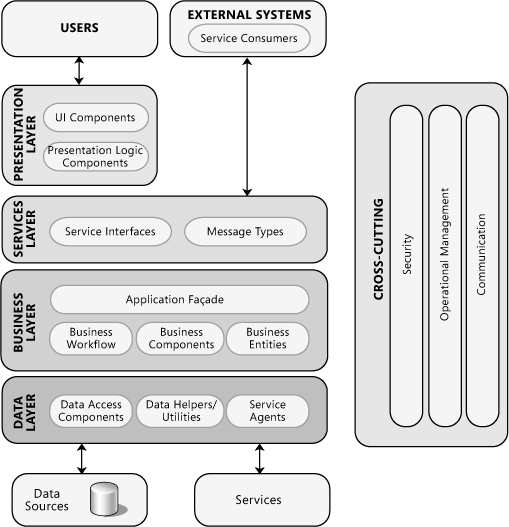
Design pattern is a description or template for how to solve a problem that can be used in many different situations. Object-oriented design patterns typically show relationships and interactions between classes or objects, without specifying the final application classes or objects that are involved.

*It is important that you justify its design, for example, by discussing the implications of constraints on your solution and different design choices, and then giving reasons for making the choices you did. At each stage of the design you should mention what kind of design patters have you followed while designing your system. You should identify which design pattern among the existing patterns are you following while designing your project.*

## Software Architecture

Software architecture is described as the organization or structure of a system, where the system represents a collection of components that accomplish a specific function or set of functions. When getting started with your design, keep in mind the key principles that will help you to create an architecture that adheres to proven principles, minimizes costs and maintenance requirements, and promotes usability and extendibility. The major decisions that you must make, and which help to ensure that you consider all of the important factors as you begin and then iteratively develop your architecture design are

* Determine the Application Type
* Determine the Deployment Strategy
* Determine the Appropriate Technologies
* Determine the Quality Attributes
* Determine the Crosscutting Concerns

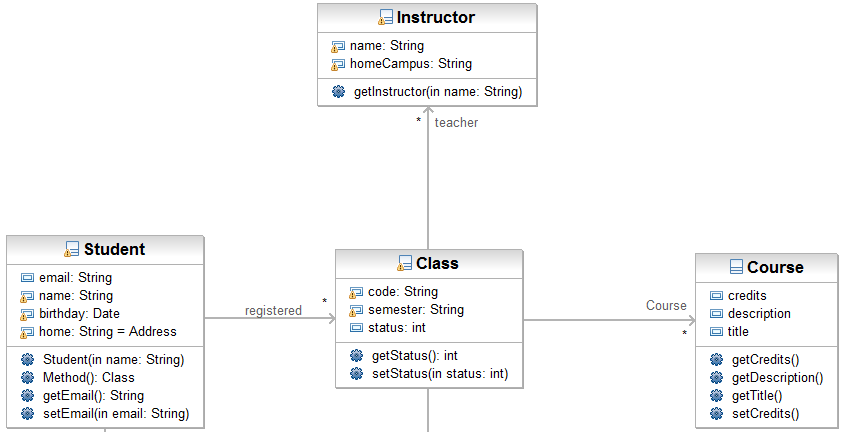


**Figure 3.1: Software Architecture Diagram**

*Fig. 3.1 illustrates common application architecture with components grouped by different areas of concern. You should think of architecture as the strategy for how you will build the system. An architectural “layer” is the top-level logical view, or an abstraction, of your design.*

## Class Diagram

Class Diagram as shown in Fig. 3.2 provides an overview of the target system by describing the objects and classes inside the system and the relationships between them. It provides a wide variety of usages; from modeling the domain-specific data structure to detailed design of the target system.

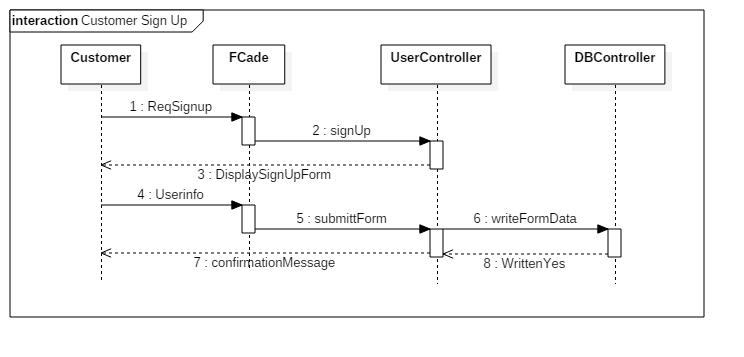


**Figure 3.2: Class Diagram**

## Sequence Diagram

Sequence diagrams, when used in conjunction with class diagrams; provide an extremely effective communication mechanism. UML sequence diagrams as shown in Fig. 3.3 are used to show how objects interact in a given situation.

*You can use a class diagram to illustrate the relationships between the classes, and the sequence diagram lets you show the messages sent among the instances of these classes and the order in which they are sent. When an object sends a message to another object, it implies that the two classes have a relationship that must be shown on a class diagram.*

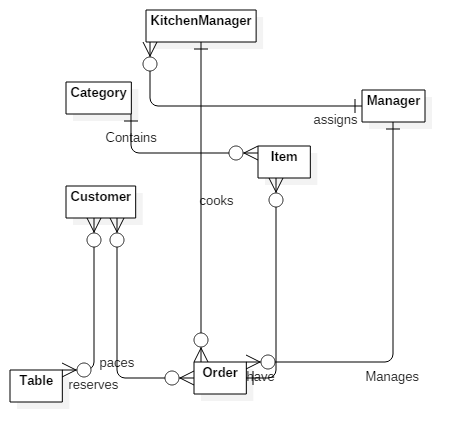


**Figure 3.3: Sequence Diagram**

## 3.4. Entity Relationship Diagram

Entity relationship model diagram (ERD) is a conceptual representation of the data in a software system. During detail design this model is mapped in to the physical database model. There are different diagramming conventions available for creating ER diagrams. A sample ERD is shown in Fig. 3.4.

**.***At this stage you may not be able to provide a comprehensive ERD therefore you may provide the refined ERD later in design chapter of the document where you will provide the fully attributed ERD such as primary keys, cardinality constraints etc.*



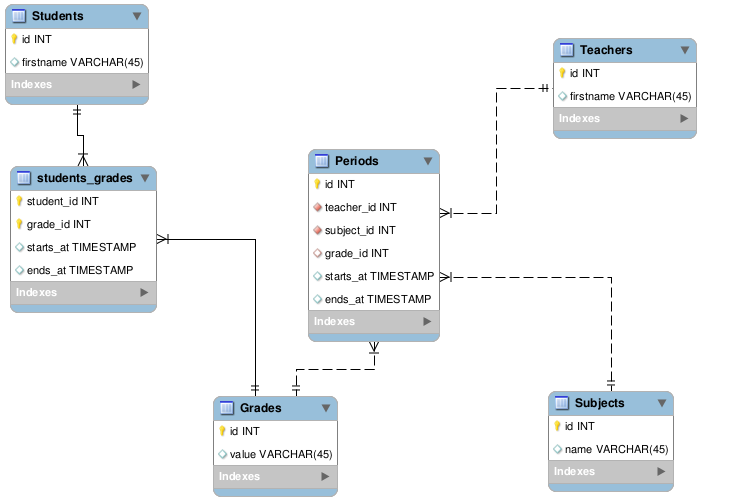
## 

**Figure 3.4: Entity Relationship Diagram**

## Database Schema

A database schema represents the logical configuration of all or part of a relational database. It can exist both as a visual representation and as a set of rules known as integrity constraints that govern a database. These rules are expressed in a data definition language, such as SQL. A database schema indicates how the entities that make up the database relate to one another, including tables, views, stored procedures, and more. A database scheme includes information related to primary and secondary keys, normalization and indexing.

You may present database scheme using front end tool of any DBMS or any other design tools such as Visio or Enterprise Architecture. A sample database scheme is shown in Fig. 3.5.

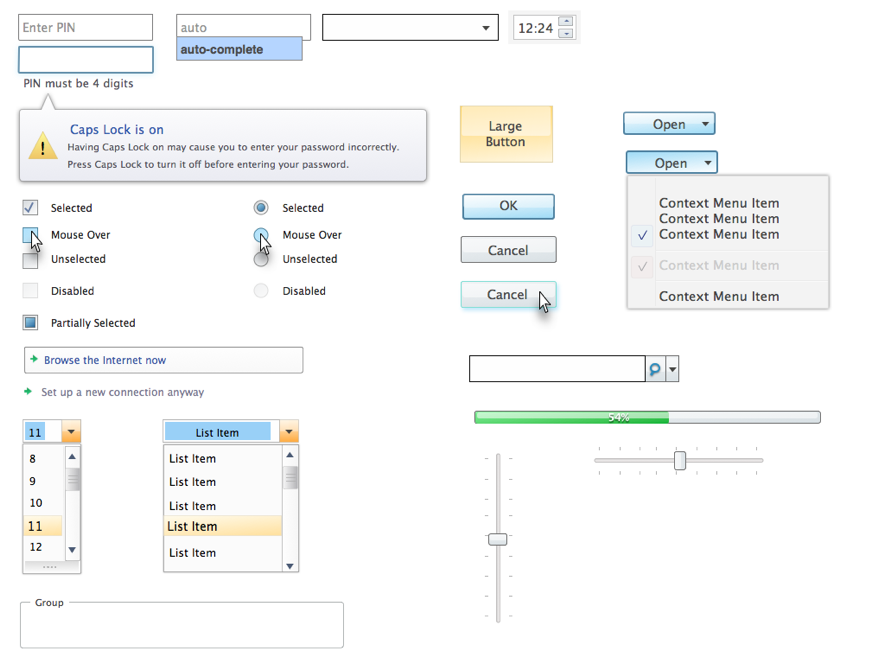
.

**Figure 3.5: Database Schema**

## User Interface Design

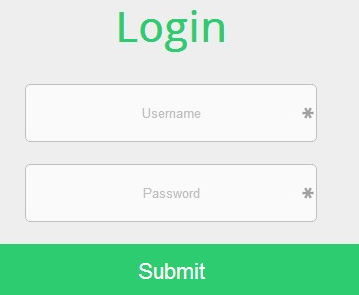
User Interface (UI) Design focuses on anticipating what users might need to do and ensuring that the interface has elements that are easy to access, understand, and use to facilitate those actions. UI brings together concepts from interaction design, visual design, and information architecture.

*You should describe the UI design in such a way that it remains simple and consistent along different views. Common GUI elements are shown in the Fig. 3.6. You should describe the UI design of each page.*



**Figure 3.6: Common GUI elements**

*Example Login Page as shown in Fig. 3.7 will contain one text field and one password field. Max length of text field is 8 and min is 4 whereas maximum length off password is 6 and minimum is 3.*



**Figure 3.7: Example Login Page UI Design with description in text**

## Software COTS

If you use COTS (Components Off the Shelf, also named SOUP, Software Of Unknown Provenance), list them here.

For each COTS, describe:

* Its identification and version
* Its purpose
* Where it comes from: manufacturer, vendor, university …
* Whether it is maintained by a third party or not
* If this is an executable,
  + What are the hardware / software resources it uses
  + Whether it is insulated in the architecture and why

# Chapter 4

# Software Development

The Implementation section is similar to the Specification and Design section in that it describes the system, but it does so at a finer level of detail, down to the code level. This section is about the realization of the concepts and ideas developed earlier. It can also describe any problems that may have arisen during implementation and how you dealt with them.

**Make sure that the system design corresponds to the implementation of the project. If there is no relationship between design and implementation, it may downgrade your score in FYP.**

You should also mention any unforeseen problems you encountered when implementing the system and how and to what extent you overcame them. Common problems are:

* Difficulties involving existing software, because of e.g.,
  + Its complexity,
  + Lack of documentation;
* Lack of suitable supporting software
* Overambitious project aims.

A seemingly disproportionate amount of project time can be taken up in dealing with such problems. The Implementation section gives you the opportunity to show where that most of the effort has been spent.

## Coding Standards

*Describe the indention, declaration,naming convention and statement standard used while coding the project.*

## Development Environment

In this section you will provide the reason behind using all the existing tools and technologies that you may have used during the development of your project. This includes development environment that you have used. How have you deployed the development environment? What different kind of packages you have used? Are there any third party libraries involved etc?

## Software Description

In this section you will identify major modules of the software that you have produced. You will show the class diagram of these major modules for this section. Typical subheadings of this section can be

**Login process**

**Loading Data**

**Data processing**

**Report generation**

**…..**

Moreover you will also discuss the logic that you have implemented in the code of those modules with the help of code snippets as shown below in the examples. Do not attempt to describe all the code in the system, and do not include large pieces of code in this section.

* Are especially critical to the operation of the system.
* You feel might be of particular interest to the reader for some reason
* Illustrate a nonstandard or innovative way of implementing an algorithm, data structure, etc.

**Snippet 1**

#define SWAP(type, x, y) \

do { \

type temp;

temp = x;

x = y;

y = temp; \

} while(1)

**Description**: This function takes 2 arguments. Then we have an infinite loop that swaps the value of the two passed variables.

You are not allowed to include the complete source code of the software how ever you can include important functions of your major modules to discuss the logic of your code.

**Snippet 2**

#define pop(type, Top) \

type temp;

temp = Top.item;

Top = Top.next;

return temp;

**Description**: This function pops the top of the stack. It places the top pointer to the next item of the stack and return the popped item..

You are not allowed to include the complete source code of the software how ever you can include important functions of your major modules to discuss the logic of your code.

# Chapter 5

# Software Testing

Software Testing is the most crucial part of Software Development Process. It is the investigation or evaluation of a software component, improving them, and finding bugs and defects. Testing is usually done by executing a system in such a way that it identifies any gaps, errors, or missing requirements in contrary to the actual requirements.

## 5.1. Testing Methodology

It is essential to have a testing plan in place to ensure that the product delivered is robust and stable, and is delivered on a predictable timeline.

*In this section you will discuss the reason of various testing techniques that you have used to test the software you have created such as integration testing, component testing and system testing etc.*

## 5.2. Testing Environment

*Describe and discuss the reason to use the selected testing environment.*

## 5.3. Test Cases

*You should describe how you demonstrated that the system works as intended (or not, as the case may be). Include comprehensible summaries of the results of all critical tests that were carried out. You might not have had the time to carry out any full rigorous tests you may not even got as far as producing a testable system. However, you should try to indicate how confident you are about whatever you have produced, and also suggest what tests would be required to gain further confidence*

* **Test Case 1**
* Test case description
* How test case was generated
* Expected result of the test case
* Actual result of the test case

**Table 5.1: Test Case 1**

|  |  |
| --- | --- |
| Date: 06 June 2017 |  |
| *System:* Menu Drive |  |
| *Objective:* View location of delivery boy | *Test ID:*1 |
| *Version:*1 | *Test Type:* Unit testing |
| *Input:*  Longitude=33.7294  Latitude=73.0931 | |
| *Expected Result:*return Islamabad location. | |
| *Actual Result:* passed | |

### 

# Chapter 6

# Software Deployment

## 6.1. Installation / Deployment Process Description

*In this section you have to provide step-by-step guide for the installation of the software produced with the help of screen shots.*

**REPORT APPROVAL CERTIFICATE**

The report of the project, “Title” has been approved based on the following evaluation guideline.

Project Evaluation Guidelines

|  |  |
| --- | --- |
| **Artifacts Guidelines** | |
| Analysis and Design artifacts are syntactically correct (use-case model, SSDs, domain model, class diagram, SDs, ERDs, Flow charts, Activity Diagram, DFDs) |  |
| Consistency and traceability have been maintained among different artifacts |  |
| **General Guidelines** | |
| Formatting (font style, indentation) is according to the FYP template and consistent throughout the document |  |
| Captions are added to all the figures and tables. Figure captions must be placed below each figure, and table captions must be provided above the table |  |
| Each figure or table is followed by some text describing what it represents |  |

|  |  |  |
| --- | --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Name & Signature  (Examiner 1) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Name & Signature  (Examiner 2) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Name & Signature  (Examiner3) |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Name & Signature (Supervisor) |  |

**References**

All the documents, papers, articles and WebPages that you have taken help from must be cited in the references section

**Book**

Author(s), Book *Title*. Place of publication: Publisher, year, volume, page number(s).

Example: [1] W.K. Chen, *Linear Networks and Systems*. Belmont, CA: Wadsworth, 1993, pp. 123-35.

**Webpage**

Author(s) and/or organization, date of publish or date the page was last updated, title of web page document, website address that provides a direct link to the document, and the date you last accessed the document

Example: Winston, J 1999, *A look at referencing,* AAA Educational Services, accessed 20 October 2015, <http://www.aaa.edu.au/aaa.html>. United Nations Web Services 2006,

**Research Paper**

*Author(s), "Article title,* Journal Title*, vol., no., page number(s), Month year.*

Example: [2] G. Pevere, "Infrared Nation, *International Journal of Infrared Design*, vol. 33, pp. 56-99, Jan. 1979.

*If you need to reference any item that is not the the list, you should consult IEEE citation format available at the following link* [*http://library.queensu.ca/book/export/html/5846*](http://library.queensu.ca/book/export/html/5846)

**Appendix**

**User Manual of the software**