Final Year Project Manual

Table of Content

1. Final year project 4
2. Project Proposal 5
Project Title 5
Student Information 5
Problem Statement 5
Executive Summary 5
Introduction 5
Competitors/Competitive Analysis 5
Objectives 6
Motivation 6
Requirements 6
Features of Project 6
Architectural Design 6
Implementation Tools and Techniques 6
Project Plan 7
Version Control 7
References 7
3. Final year project Documentation 8
Chapter 1: Introduction to the Problem 8
Introduction 8
Purpose 8
Objective 8
Existing Solution 8
Proposed Solution 8
Chapter 2: Software Requirement Specification 8
1. Introduction 8
1.1. Purpose 8
1.2. Scope 8
1.3. Definitions, acronyms, and abbreviations 9
1.4. References 9

1.5. Overview 9
2. Overall description 9
1.2. Product perspective 9
1.3. Product functions 10
1.4. User characteristics 11
1.5. Constraints 11
1.6. Assumptions and dependencies 11
1.7. Apportioning of requirements 11
2. Specific requirements 12
2.1. Functional Requirement 12
2.2. Non-functional Requirements 12
3. Appendixes 12
4. index 12
Chapter 3: Use Case Analysis 13
Chapter 4: Design 15
4.1. Architecture Diagram 15
4.2. ERD with data dictionary 15
4.3. Data Flow diagram 16
• The level 0 16
• The level 1 17
4.4. Class Diagram 18
4.5. Activity Diagram 19
4.6. Sequence Diagram 20
4.7. Collaboration Diagram 21
4.8. State Transition Diagram 22
4.9. Component Diagram 23
4.10. Deployment Diagram 24
Chapter 5: Testing 25
5.1. Test Case Specifications 25
5.2. Black Box Test Cases 26
5.2.1. Equivalence Partitions (EP) 26
5.2.2. Boundary Value Analysis 26

5.2.3. Decision Table Testing 26
5.2.4. State transition Testing 27
Use Case Testing 27
5.3. White Box Test Cases 27
5.3.1. Cyclometric complexity 27
5.4. Performance testing 27
5.5. Stress Testing 27
5.6. System Testing 28
5.7. Regression Testing 28
5.7.1 Selecting Regression Tests 28
5.7.2. Regression Testing Steps 28
Chapter 6: Tools and Techniques 29
Chapter 7: Summary and Conclusion 29
Chapter 8: User Manual 29

1. Final year project

Final Year Projects (FYP) in Department of Computer Science, the university of Lahore are considered as group projects mandatory for every student in order to complete their Bachelor/Masters programs

Here are some important points to be considered

- All students (BSCS, BSSE, MCS/MIT) who enroll in final year project spring 2017 (Phase 1) are required to submit their proposal latest by February 15, 2017

- All students sire required to make group in their respective program only, BSCS students are not allowed to make group with BSSE students and vice versa

- Students are required to submit their transcript along with their project proposal to Mr. Soban Mamoon in admin office between 2pm-3pm till February 15, 2017

- No student having less than 97 credit hours is allowed to enroll final year project.

- No student with probation is allowed to enroll final year project

- Checklist for Proposal submission
    1) Project Proposal
    2) Allocation form
    3) Volunteer form
    4) Transcripts of each group member

Some of the pre-requisite which all students which are enrolling have studded in earlier semesters

- Introduction to Database
- Core-subjects of programming including Object Oriented Programming
- Introduction to Software Engineering
- Web Engineering
- Java Software Development Paradigm
- Mobile application development

2. Project Proposal

You can download your project proposal template from sites.google.com/site/uolcsfyp under the section of **forms**

The project proposal includes the following

Project Title

Project name must be self-defining what actually you project can do and should not be confusing.

Student Information

The detail of all group members in Block Letters Only including names, studentID, Email and the contact number

Problem Statement

The Problem Statement gives an explanation about the issue that is being addressed by the project. It also argues in favor of implementing the project in the proposed area in the existing conditions.

Executive Summary

The executive summary will be a brief introduction and justification of the proposal which needs to be shorter and more summarized when compared to an abstract. Most importantly, it should contain the problem which is the target of the project, and the solution to the problem in the proposed project. The attractive thinking will be maintained here to give reasons to the reader to finish reading of the whole proposal.

Introduction

- Relevance or importance of problem
- Background information to educate the reader
- Previous related work by others, literature review with credible sources

Competitors/Competitive Analysis

List down all the possible competitors of your product. That is, you need to list down all those products that are closely related to your product in terms of features, target audience, etc.

Objectives

Objectives are the final results to be achieved after the completion of your project. It includes all the modules which you want to add in this project

Motivation

Give a reason why is your problem interesting and important for the society

Requirements

This section includes all the software's you have to use in the implementing the project like

1. Android Studio for android development
2. NetBeans IDE or any other IDE for implementing the web development

Features of Project

Detailed features and the functionality of each of your module like

the registration module login, signup etc. are the functionalities

Architectural Design

Describe hardware, software, or network components as relevant and as understood at this time. Draw a high-level architecture diagram to illustrate the proposed system components and the relationships between them.

Implementation Tools and Techniques

This section includes the list of all the tools you have to use in the implementing the project like

1. Programming languages
   - Php for Web Application
   - Swift for IOS Application
   - Java for android for Android
2. Database programming using MySQL, Entity Framework and SQL Server 2014.

Project Plan

This section describes how the project will be managed, including a detailed plan with milestones. Specific items to include in this section are as follows:

1. Division of responsibilities and duties among team members.

2. Timeline with milestones with Gantt chart in which can be implemented in Microsoft project. For implementation, you can get the help from the MS Project manual which is uploaded in the **Documentation Template** section. The following are required elements of your Gantt chart

   - Project duration is from the date your project is enrolled to the completion date
   - Each milestone is to be labeled with a title.
   - Schedule all tasks not just **Design** or **Testing.** Break this schedule down to specific assignments.
   - Each task is to be labeled with a title and person or persons assigned to the task.
   - Subdivide larger items so that no task is longer than about one week
   - Link tasks which are dependent on the completion of a previous task.

Version Control

A table that will provide information of each time proposal was updated.

References

Give references to the resources you have consulted in finalizing your project topic.

3. Final year project Documentation

There are two phases for final year project documentation. The first phase of documentation includes four chapter which you have to submit. These chapter are detailed listed below

Chapter 1: Introduction to the Problem

Introduction

The Problem introduction gives an explanation about the issue that is being addressed by the project. It also argues in favor of implementing the project in the proposed area in the existing conditions

Purpose

Define the purpose why you are developing your project what is the need and how this system or project will help in the market or in life of the society

Objective

Define your objectives what you want to achieve after the completion of the project

Existing Solution

Explain the existing solution in the market what are the flows/problems with that existing solutions

Proposed Solution

Explain what you have proposed and how you tackle the problem in the existing products which are related to your project last but not the least why your product is better than the existing solution in the market

Chapter 2: Software Requirement Specification

1. Introduction

1.1. Purpose

This subsection should

1) Delineate the purpose of the SRS
2) Specify the intended audience for the SRS.

1.2. Scope

This subsection should

1) Identify the software product(s) to be produced by name
2) Explain what the software product(s) will, and, if necessary, will not do
3) Describe the application of the software being specified, including relevant benefits, objectives, and goals
4) Be consistent with similar statements in higher-level specifications the system requirements specifications, if they exist.

1.3.Definitions, acronyms, and abbreviations

This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.
So here what are definitions, acronyms, and abbreviations

Definitions: a statement of the exact meaning of some words which are used in the document.
Acronyms: name formed as an abbreviation from the initial components in a phrase or a word like NATO - The **N**orth **A**tlantic **T**reaty **O**rganization.
Abbreviations: a shortened form of a word or phrase like UOL is the abbreviation of The **U**niversity **o**f **L**ahore

1.4.References

This subsection should

1) Provide a complete list of all documents referenced elsewhere in the SRS
2) Identify each document by title, report number (if applicable), date, and publishing organization;
3) Specify the sources from which the references can be obtained.

1.5.Overview

Describe what the rest of the SRS contains

2. Overall description

1.2. Product perspective

This subsection of the SRS should put the product into perspective with other related products. If the product is independent and totally self-contained, it should be so stated here.
A block diagram showing the major components of the system, interconnections, and external inter-faces can be helpful.
This subsection is sub-divided into the following

- **System interfaces** (functionality of the software to the system requirement)
- **User interfaces** (define the layout of the user end including the screenshots)
- **Hardware interfaces** (define the software dependencies if any)
- **Software interfaces** (define the hardware dependencies if any)
- **Communications interfaces** (This should specify the various interfaces to communications such as local network protocols etc.)
- **Memory** (This should specify any applicable characteristics and limits on primary and secondary memory.)

- **Operations** (specify the normal and special operations like backup and recovery etc.)
- **Site adaptation requirements.** (Define the requirements for any data or initialization sequences that are specific to a given site, mission, or operational mode)

1.3. Product functions

Product functions include the functional requirement of your project

Now what is the functional requirements suppose you have a module of Account Registration in your project then the functional requirement for this module are Create Account, View Account, Delete Account, Update Account, Login Account, Logout Account etc. and many more according to the situation you are in.

The following things include in each of the functional requirement of your project

- ID
- Name
- Description
- Input
- Output
- Basic Work Flow
- Requirements (optional)

For example, yours create account functional requirement look like this and you have to follow this template for writing your functional requirements

ID:	FR_01			
Name:	Create Account			
Description	Input	Output	Requirements	Basic Work Flow
Enter details to create account	Name, Email, Password etc.	Account created	Internet Connectivity required	Enter correct information and click submit button System save the record in database

Table 1 Functional Requirement Create Account

1.4. User characteristics

This subsection of the SRS should describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise.

1.5. Constraints

This subsection of the SRS should provide a general description of any other items that will limit the developer's options. These include

- Regulatory policies
- Hardware limitations
- Interfaces to other applications
- Parallel operation
- Audit functions
- Control functions
- Higher-order language requirements;
- Signal handshake protocols
- Reliability requirements
- Criticality of the application
- Safety and security considerations

1.6. Assumptions and dependencies

This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS.

1.7. Apportioning of requirements

This subsection of the SRS should identify requirements that may be delayed until future versions of the system.

2. Specific requirements

This section will describe the functional and non-functional requirements of System at a sufficient level of detail for the designers to design a system satisfying the User requirements and testes to verify that the system satisfies the requirements.

2.1. Functional Requirement

In this section only describe the function/modules of your project

2.2. Non-functional Requirements

This sub-section includes the following

- Usability
- Reliability
- Performance
- Design Constraints
- Portability
- Maintainability
- License Agreement

3. Appendixes

4. index

Chapter 3: Use Case Analysis

This chapter includes all the use case diagrams of the functional requirements of your project along with the aggregated usecase diagram

The Usecase diagrams can be made by using Visual Paradigm

1. Select Diagram > New from the application toolbar.
2. In the New Diagram window, select Use Case Diagram.
3. Click Next.
4. Enter the diagram name and description.
5. Click OK.

The following things include in each of the functional requirement of your project

1. Usecase ID
2. Usecase Name
3. Description
4. Primary Actor
5. Secondary Actor
6. Pre-Condition
7. Post-Condition
8. Basic Flow
9. Alternate Flow

For example, yours create account use case look like this and you have to follow this template for writing your use cases

Usecase diagram for create account

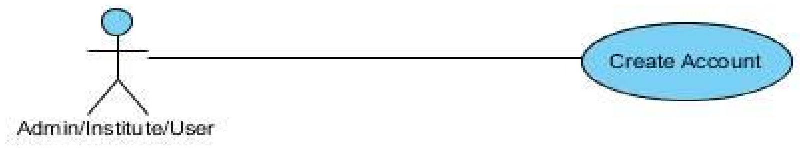


Figure 1 Usecase Diagram Create Account

Usecase diagram detail

Use Case ID	UC_01 (all ID should be in this sequence)	
Use Case Name	Create Account (Name of usecase here is create account)	
Description	Detail of this usecase	
Primary Actor	Actors associate with the usecase	
Secondary Actor		
Pre-Condition	What is required to do this function	
Post-Condition	What is the output of this function	
Basic Flow	Actor Action	System Action
	Flow of information	What would system do according to the information
Alternate Flow	Another way to work with this function	

Table 1 Usecase Create Account

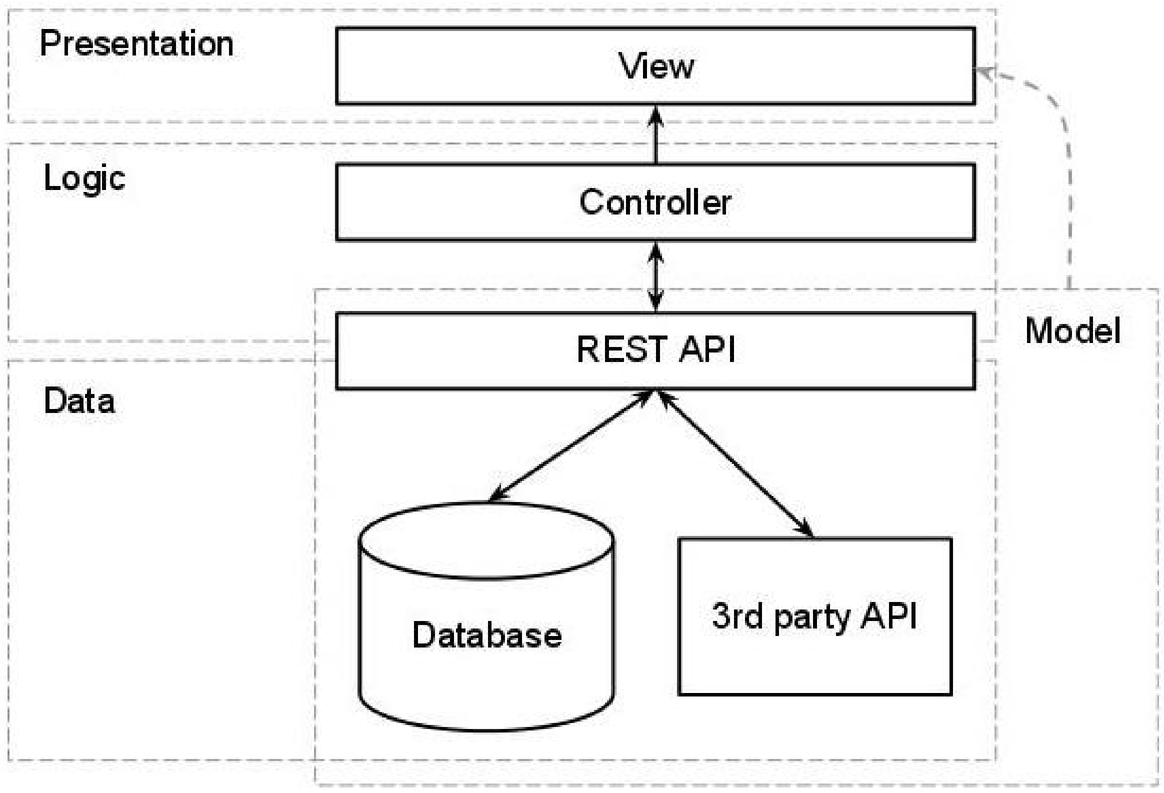
Chapter 4: Design

In this section, we provide the design analysis of our modules including the following designs

1. Architecture Diagram
2. ERD with data dictionary
3. Data Flow diagram
4. Class Diagram
5. Activity Diagram
6. Sequence Diagram
7. Collaboration Diagram
8. State Transition Diagram
9. Component Diagram
10. Deployment Diagram

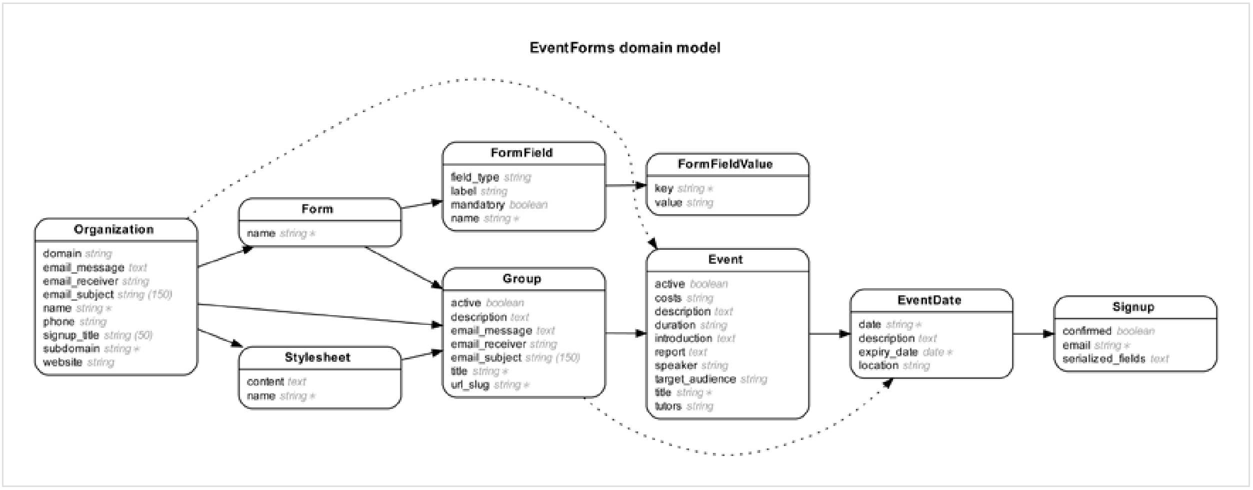
4.1. Architecture Diagram

Define the graphical representation of the concepts, their principles, elements and components that are part of your project.



4.2. ERD with data dictionary

Entity Relationship Diagram with complete relations with dependencies of your project

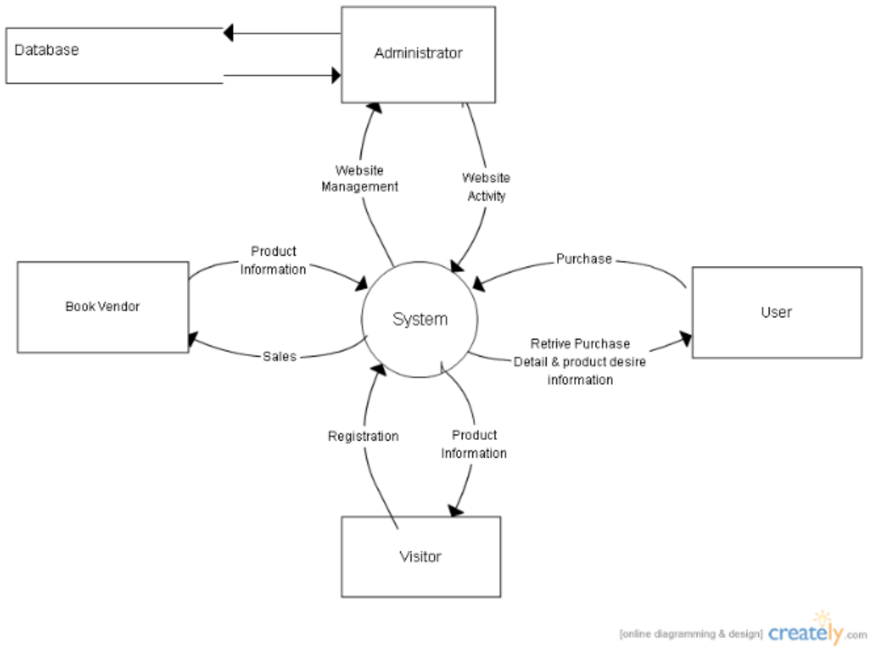


4.3. Data Flow diagram

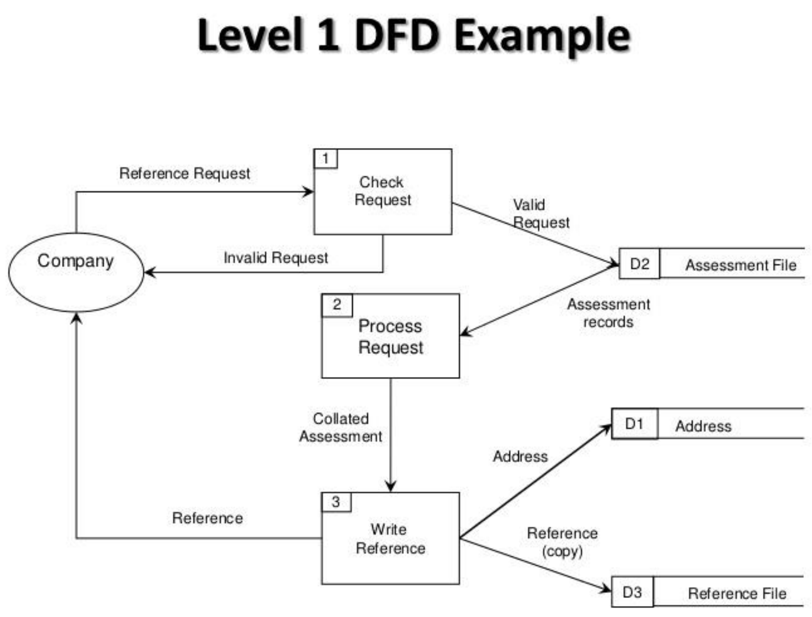
Data flow diagram includes two levels

- **The level 0**

  The flow of information inside the system is defined in this level

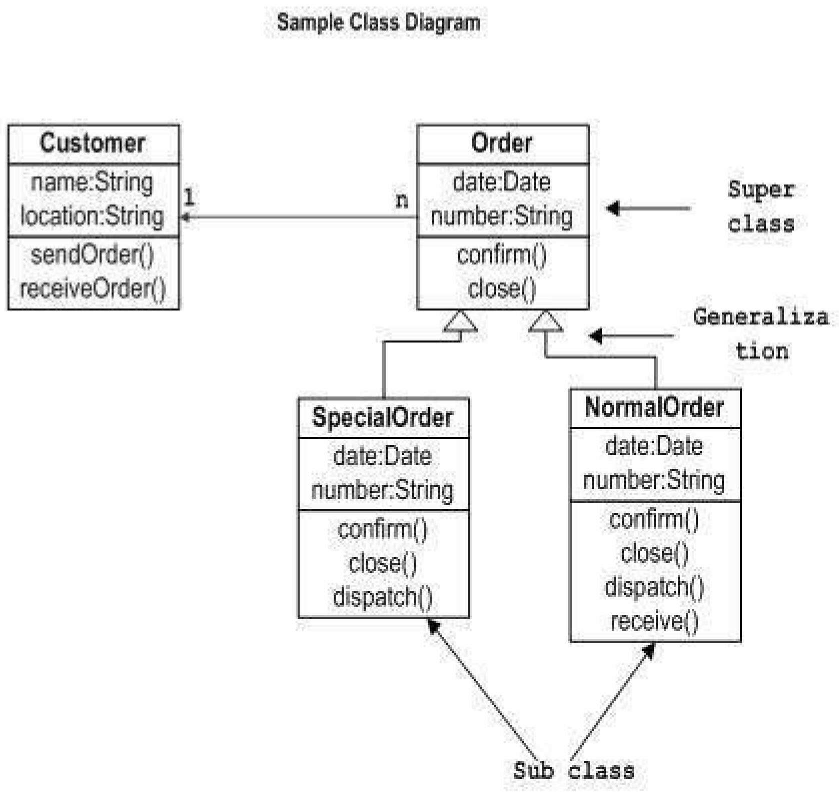


- **The level 1**
  The flow of information outside the system is defined in this level



4.4. Class Diagram

Describe the structure of a project by showing the systems classes, their attributes, operations (or methods), and the relationships among objects.



4.5. Activity Diagram

This diagram includes all the activity diagrams of the functional requirements of your project along with the aggregated activity diagram

The Activity diagrams can be made by using Visual Paradigm

1. Select **Diagram > New** from the application toolbar.
2. In the **New Diagram** window, select **Activity Diagram**.
3. Click **Next**.
4. Enter the diagram name and description.
5. Click **OK**.
6.

For example, yours create account activity look like this and you have to follow this template for writing your activity diagrams

Activity diagram for create account

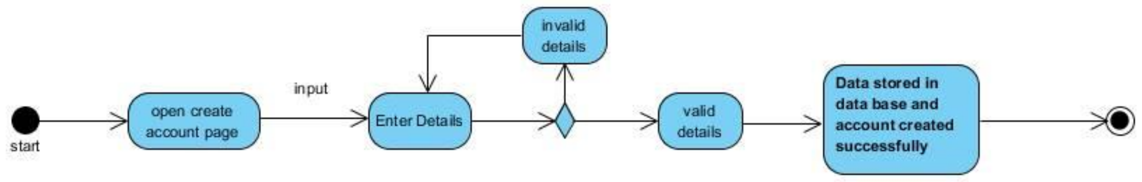


Figure 1 Activity Diagram Create Account

4.6. Sequence Diagram

This diagram includes all the Sequence diagrams of the functional requirements of your project along with the aggregated Sequence diagram

The Sequence diagrams can be made by using Visual Paradigm

1. Select **Diagram** > **New** from the application toolbar.
2. In the **New Diagram** window, select **Sequence Diagram**.
3. Click **Next**.
4. Enter the diagram name and description.
5. Click **OK**.
6.

For example, yours create account Sequence look like this and you have to follow this template for writing your Sequence diagrams

Sequence diagram for create account

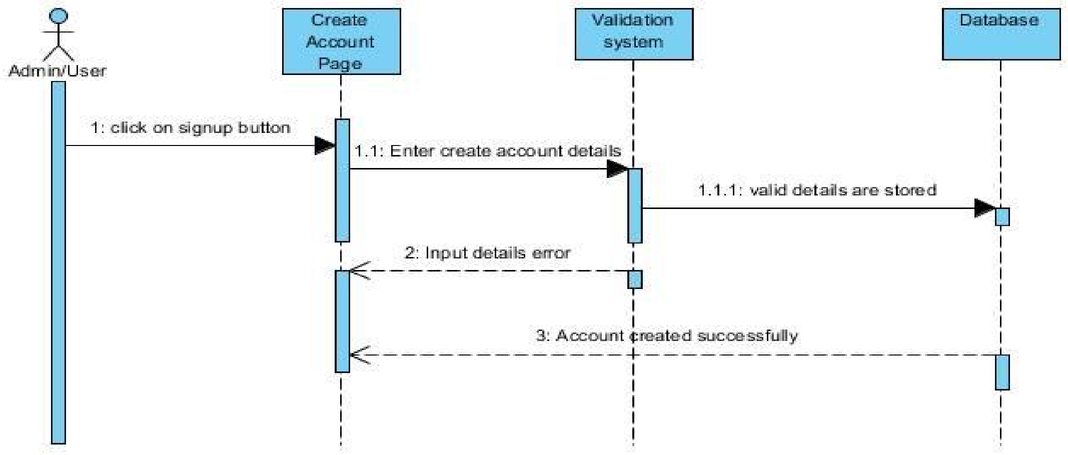
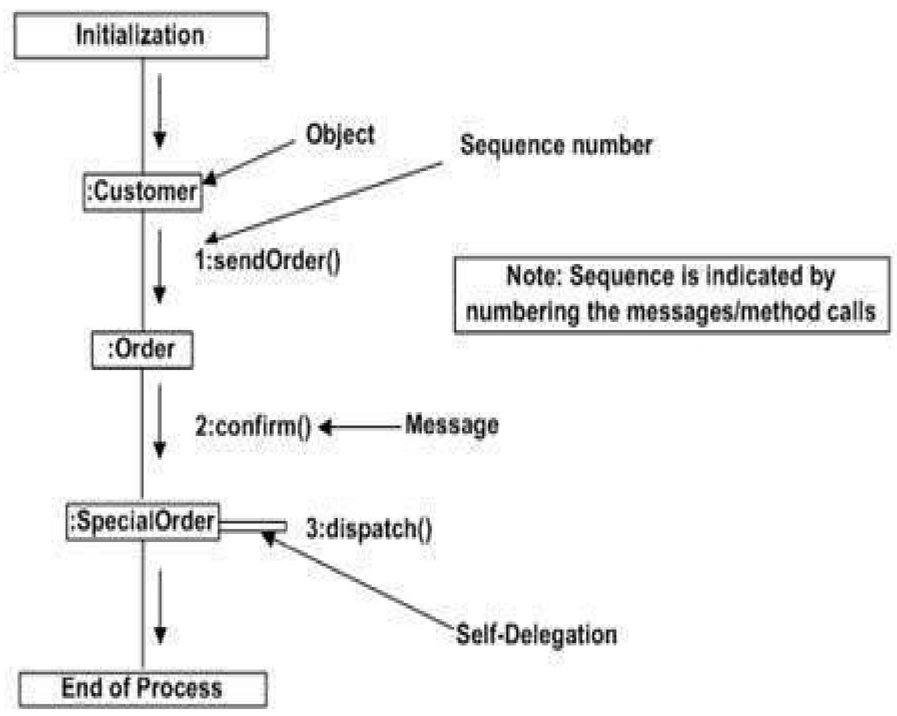

Admin/User
Create Account Page
Validation system
Database
1: click on signup button
1.1: Enter create account details
1.1.1: valid details are stored
2: Input details error
3: Account created successfully

Figure 1 Sequence Diagram Create Account

4.7. Collaboration Diagram

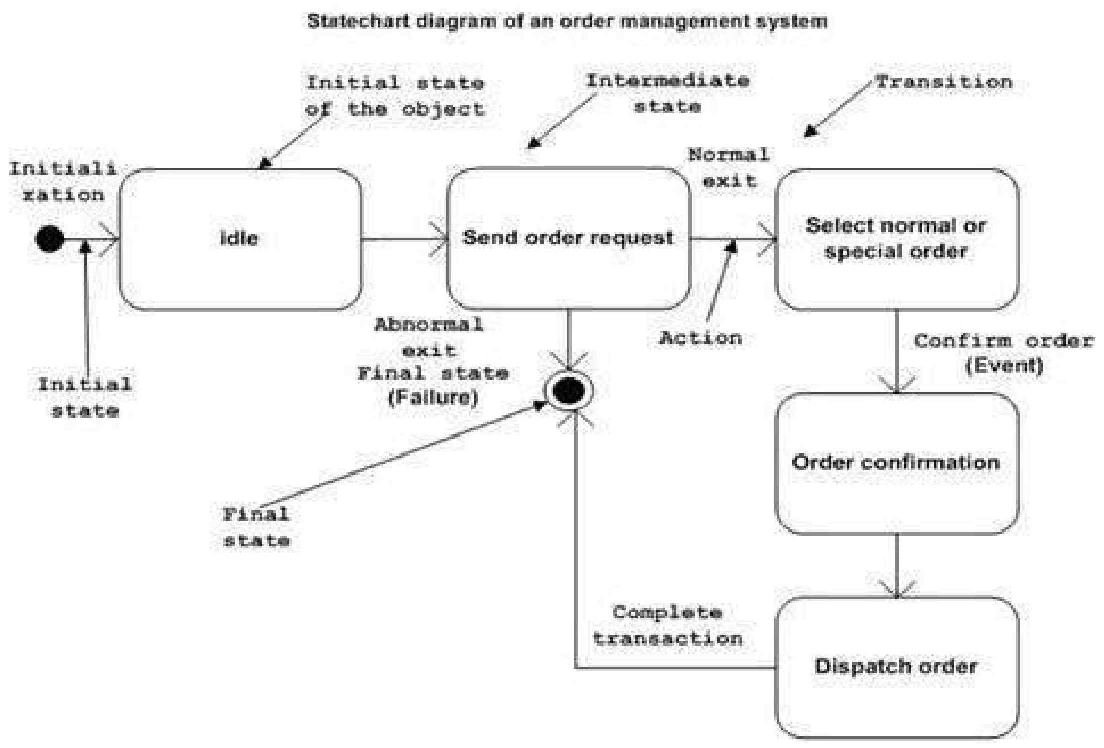
It shows the object organization as shown below. Here in collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram.

Collaboration diagram of an order management system



4.8. State Transition Diagram

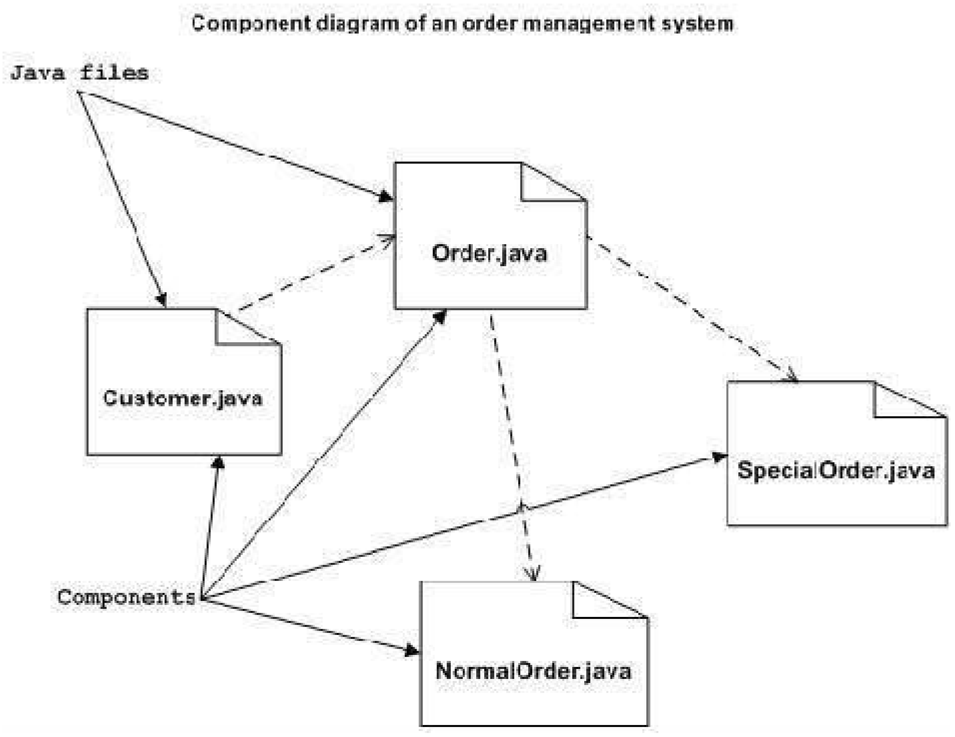
State Transition diagram is used to describe the states of different objects in its life cycle. So, the emphasis is given on the state changes upon some internal or external events. These states of objects are important to analyze and implement them accurately



4.9. Component Diagram

Component diagrams are used to describe the physical artifacts of a system. This artifact includes files, executables, libraries etc.

So, the purpose of this diagram is different, Component diagrams are used during the implementation phase of an application. But it is prepared well in advance to visualize the implementation details.

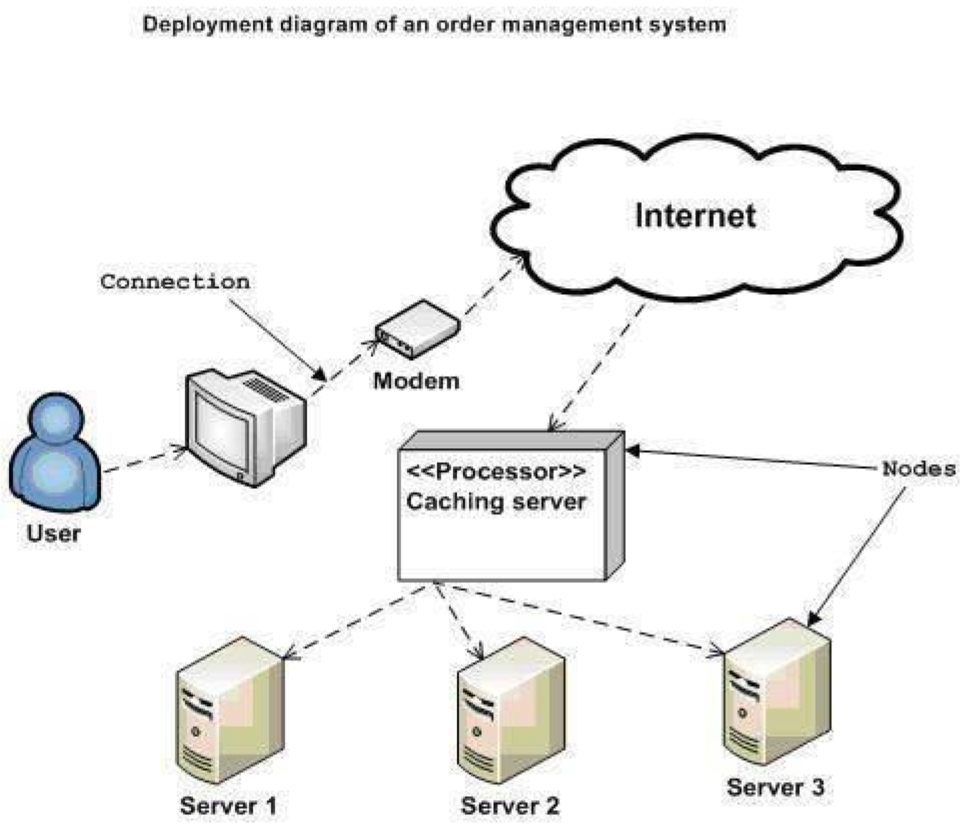


4.10. Deployment Diagram

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware’s used to deploy the application.

Deployment diagrams are useful for system engineers. An efficient deployment diagram is very important because it controls the following parameters

1. Performance
2. Scalability
3. Maintainability
4. Portability



Chapter 5: Testing

5.1. Test Case Specifications

This Testing phase includes all the Test Cases of the functional requirements of your project

For example, your login account test case look like this and you have to follow this template for writing your project test cases

Test Case for login account

Positive Test Case	
ID	TC_LOGIN_SUCCESS
Priority	High
Description	To verify user authentication to system.
Reference	Functional Requirement reference
Users	Administrator.
Pre-requisites	A System is online. B User must have active login credentials provided by system administrator. C User has internet access.
Steps	A Open the web link to system. B Enter login id C Enter Password. D Press Login.
Input	Login id and password
Expected result	Successfully enters the system and main home page opens.
Status	Tested, passed.

Negative Test Case	
ID	TC_LOGIN_FAILURE
Priority	High
Description	To verify user authentication to system.
Reference	Functional Requirement reference
Users	Administrator.
Pre-requisites	A System is online. B User must have active login credentials provided by system administrator. C User has internet access.
Steps	A Open the web link to system. B Enter login id. C Enter Password. D Press Login.
Input	Incorrect Login id or password or deactivated credentials.
Expected result	Does not allows access to system features and notifies the error.
Status	Tested, passed.

5.2. Black Box Test Cases

Black box testing also known as Behavioral Testing, is a software testing method in which the internal structure/ design/ implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Behavior or performance errors
- Initialization and termination errors

5.2.1. Equivalence Partitions (EP)

Equivalence class partitioning (EP) is a very widely used method to decrease the number of possible test cases that are required to test a system.

Let see the example of Equivalence Partitions for login Authentication

Variables	**Valid Classes**	**Invalid Classes**
Username	1. Only username "admin". 2. Case in-sensitive. 3. Compulsory field.	1. Alphabets, digits and symbols other than "admin". 2. Empty Field.
Password	1. Length should be greater than 5 characters. 2. May contain symbols, alphabets [a-z A-Z] and digits [0-9].	1. Length less than 5 characters. 2. Empty field.

5.2.2. Boundary Value Analysis

A boundary value is an input or output value on the border of an equivalence partition, includes minimum and maximum values at inside and outside boundaries. Normally Boundary value analysis is part of stress and negative testing.

5.2.3. Decision Table Testing

Decision Table is a testing method, which aims to ensure that each one of the possible branch from each decision point is executed at least once and thereby ensuring that all reachable code is executed.

5.2.4. State transition Testing

State Transition testing, a black box testing technique, in which outputs are triggered by changes to the input conditions or changes to 'state' of the system. In other words, tests are designed to execute valid and invalid state transitions.

Use Case Testing

Use Case Testing is a functional black box testing technique that helps testers to identify test scenarios that exercise the whole system on each transaction basis from start to finish.

5.3. White Box Test Cases

White box testing is a testing technique, that examines the program structure and derives test data from the program logic/code. The other names of glass box testing are clear box testing, open box testing, logic driven testing or path driven testing or structural testing.

5.3.1. Cyclometric complexity

Cyclometric complexity is a source code complexity measurement that is being correlated to a number of coding errors. It is calculated by developing a Control Flow Graph of the code that measures the number of linearly-independent paths through a program module.

Lower the Program's cyclometric complexity, lower the risk to modify and easier to understand.

5.4. Performance testing

Performance testing, a non-functional testing technique performed to determine the system parameters in terms of responsiveness and stability under various workload. Performance testing measures the quality attributes of the system, such as scalability, reliability and resource usage.

5.5. Stress Testing

Stress testing a Non-Functional testing technique that is performed as part of performance testing. During stress testing, the system is monitored after subjecting the system to overload to ensure that the system can sustain the stress.

The recovery of the system from such phase (after stress) is very critical as it is highly likely to happen in production environment.

5.6. System Testing

System Testing (ST) is a black box testing technique performed to evaluate the complete system the system's compliance against specified requirements. In System testing, the functionalities of the system are tested from an end-to-end perspective.

System Testing is usually carried out by a team that is independent of the development team in order to measure the quality of the system unbiased. It includes both functional and Non-Functional testing.

5.7. Regression Testing

Regression testing a black box testing technique that consists of re-executing those tests that are impacted by the code changes. These tests should be executed as often as possible throughout the software development life cycle.

Types of Regression Tests:

- **Final Regression Tests**
  A "final regression testing" is performed to validate the build that hasn't changed for a period of time. This build is deployed or shipped to customers.

- **Regression Tests**
  A normal regression testing is performed to verify if the build has NOT broken any other parts of the application by the recent code changes for defect fixing or for enhancement.

5.7.1 Selecting Regression Tests

- Requires knowledge about the system and how it affects by the existing functionalities.
- Tests are selected based on the area of frequent defects.
- Tests are selected to include the area, which has undergone code changes many a times.
- Tests are selected based on the criticality of the features.

5.7.2. Regression Testing Steps

Regression tests are the ideal cases of automation which results in better **R**eturn **on** **I**nvestment (ROI).

- Select the Tests for Regression.
- Choose the apt tool and automate the Regression Tests
- Verify applications with Checkpoints
- Manage Regression Tests/update when required
- Schedule the tests
- Integrate with the builds
- Analyze the results

Chapter 6: Tools and Techniques

This chapter includes the following

- Languages you are using in the development
- Applications and tools
- Libraries and Extensions if any

Chapter 7: Summary and Conclusion

This chapter include the summary and the conclusion

Chapter 8: User Manual

User manual is an important part of an application documentation. It is a guide for users on operations within the application. Follow section of this document will provide detailed guide on how to use features of application.

let take the login function as an example

This is the login authentication screen from where the user can log in to the system. User must have

1. Active username and password.
2. Web login address.

Steps:

1. Open login webpage
2. Enter login details.
3. Click Login.
4. Done.