Style Guidelines for Final Year Project ReportsTitle of the Project

**Senior Project**



Primary Advisor: **<Advisor Name>**

Secondary Advisor: **<Advisor Name>**

Presented by:

Student Reg# Student Name

Department of Computer Science

**Forman Christian College (A Chartered University)**

Topic of Project

By

NAME(S) OF PARTICIPTANT(s)

Project submitted to

Department of Computer Science,

Forman Christian College (A Chartered University),

Lahore, Pakistan.

in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE

IN

COMPUTER SCIENCE (Honors)

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Primary Project Advisor |  | Secondary Project Advisor |
|  |
| Senior Project Management Committee Representative |

# Abstract

<Despite the fact that an abstract is quite brief, it must do almost as much work as the multi-page paper that follows it. This means that it should in most cases include the following sections. Each section is typically a single sentence, although there is room for creativity. In particular, the parts may be merged or spread among a set of sentences. Use the following as a checklist for your abstract but do not write any headings in this abstract:

**Motivation:** *Why do we care* about the problem and the results? If the problem isn't obviously "interesting" it might be better to put motivation first; but if your work is incremental progress on a problem that is widely recognized as important, then it is probably better to put the problem statement first to indicate which piece of the larger problem you are breaking off to work on. This section should include the importance of your work, the difficulty of the area, and the impact it might have if successful. **Problem statement:** What *problem* are you trying to solve? What is the *scope* of your work (a generalized approach, or for a specific situation)? Be careful not to use too much jargon. In some cases it is appropriate to put the problem statement before the motivation, but usually this only works if most readers already understand why the problem is important. **Approach:** *How did you go about solving* or making progress on the problem? Did you use simulation, analytic models, prototype construction, or analysis of field data for an actual product? What was the *extent* of your work (did you look at one application program or a hundred programs in twenty different programming languages?) What important *variables* did you control, ignore, or measure? **Results:** *What's the answer?* Specifically, most good computer papers conclude that something is so many percent faster, cheaper, smaller, or otherwise better than something else. Put the result there, in numbers. Avoid vague, hand-waving results such as "very", "small", or "significant." If you must be vague, you are only given license to do so when you can talk about orders-of-magnitude improvement. There is a tension here in that you should not provide numbers that can be easily misinterpreted, but on the other hand you don't have room for all the caveats. **Conclusions:** *What are the implications* of your answer? Is it going to change the world (unlikely), be a significant "win", be a nice hack, or simply serve as a road sign indicating that this path is a waste of time (all of the previous results are useful). Are your results *general*, potentially generalizable or specific to a particular case? >

# Acknowledgement

<The acknowledgement comes here. Make sure it is not more than on page long.>

# List of Figures

<Provide a list of all figures in following format.

Figure 1 Figure 1 Title page number

Note: In the document, Titles of the figures should be written under the figures, along with the figure number (Figure 1: Xyz)>

# List of Tables

Provide a list of all tables in following format.

Table 1 Table 1 title page number

Note: Every table must bear a title and table number which should be written on the top of the table (Table 1: Abc)

Table of contents

[Abstract i](#_Toc61870443)

[Acknowledgement ii](#_Toc61870444)

[List of Figures iii](#_Toc61870445)

[List of Tables iv](#_Toc61870446)

[Chapter 1. Introduction 1](#_Toc61870447)

[1.1 Introduction 1](#_Toc61870448)

[1.2 Objectives 1](#_Toc61870449)

[1.3 Problem Statement 1](#_Toc61870450)

[1.4 Scope 1](#_Toc61870451)

[Chapter 2. Requirements Analysis 2](#_Toc61870452)

[2.1 Literature Review 2](#_Toc61870453)

[2.2 User Classes and Characteristics 2](#_Toc61870454)

[2.3 Design and Implementation Constraints 2](#_Toc61870455)

[2.4 Assumptions and Dependencies 2](#_Toc61870456)

[2.5 Functional Requirements 2](#_Toc61870457)

[2.6 Use Case Diagram 3](#_Toc61870458)

[2.7 Nonfunctional Requirements 4](#_Toc61870459)

[2.8 Other Requirements 4](#_Toc61870460)

[Chapter 3. System Design 5](#_Toc61870461)

[3.1 Application and Data Architecture 5](#_Toc61870462)

[3.2 Component Interactions and Collaborations 5](#_Toc61870463)

[3.3 System Architecture 5](#_Toc61870464)

[3.4 Architecture Evaluation 5](#_Toc61870465)

[3.5 Component-External Entities Interface 5](#_Toc61870466)

[3.6 Screenshots/Prototype 5](#_Toc61870467)

[3.7 Other Design Details 6](#_Toc61870468)

[Chapter 4. Test Specification and Results 7](#_Toc61870469)

[4.1 Test Case Specification 7](#_Toc61870470)

[4.2 Summary of Test Results 7](#_Toc61870471)

[Chapter 5. Conclusion and Future Work 8](#_Toc61870472)

[5.1 Project summary 8](#_Toc61870473)

[5.2 Problems faced and lessons learned 8](#_Toc61870474)

[5.3 Future work 8](#_Toc61870475)

[References 9](#_Toc61870476)

[Appendix A Glossary 10](#_Toc61870477)

[Appendix B Deployment/Installation Guide 10](#_Toc61870478)

[Appendix C User Manual 10](#_Toc61870479)

[Appendix D Student Information Sheet 10](#_Toc61870480)

[Appendix E Plagiarism Free Certificate 11](#_Toc61870481)

[Appendix F Plagiarism Report 11](#_Toc61870482)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Introduction

<You are required to write a brief introduction of your project and provide background. In this section you should provide the context and initial knowledge of the domain. You should also highlight the significance of problem and provide motivation behind the work being done.

This complete section should be an improved version of the introduction provided in previous phases of the project. It is expected that there will be complete details available at this stage to provide in this section>

## Objectives

<Objectives are the final results that were to be achieved after the completion of your project. List all of them here.>

## Problem Statement

<Provide a short description of the software being specified. State the problem solved. Convert your language from future to past/present since the project is now complete. Also state whether your system is a program that applies certain algorithms to some application, a tool that is end product of a research, a dataset, a simulator developed as a result of research, an animated movie developed using graphics techniques, a system that achieves better performance than its competitors, or a software package that is useful in certain application etc.>

## Scope

< Describe the scope of the product that is covered.>

# 

# Requirements Analysis

## Literature Review

<Provide an overview to the projects background knowledge without too much in detail (stick to the scope of the project). The background can refer to previous work referenced from journals, articles, newspapers, or any academic literature providing evidence that the proposed problem is significant and real problem worth solving. If available, provide closely related work done within the project scope and the challenges or defects identified which can be considered as part of the new solution. Describe why you worked on this project in light of the literature review?>

## User Classes and Characteristics

<Describe the various user classes that you have identified. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the favored user classes from those who are less important to satisfy.>

## Design and Implementation Constraints

<Describe any items or issues that limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>

## Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that affect the requirements stated in the document. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

## Functional Requirements

<All functional requirements are expressed as use-cases. Fill out the following template for each use-case. Don’t really say “Use-Case 1.” State the use-case name in just a few words e.g. “Withdraw Cash from ATM”. A use-case may have multiple alternate courses of action.>

### Name of Use-Case 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-1 | |
| **Purpose** | | … | |
| **Priority** | | <Choose one from {High, Medium, Low}> | |
| **Pre-conditions** | | … | |
| **Post-conditions** | | … | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** |  | |  |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** |  | |  |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

Table 1: UC-1

### Name of Use-Case 2 (and so on)

## Use Case Diagram

<Provide the use case diagram>

## Nonfunctional Requirements

### Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

### Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

### Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

### Additional Software Quality Attributes

<Specify any additional quality characteristics for the product that are important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Other Requirements

<Define any other requirements not covered elsewhere in the document. These might include database requirements, external (hardware, software, or communication) interface requirements, internationalization requirements, legal requirements, and reuse objectives for the project.>

# System Design

## Application and Data Architecture

< Complete logical or physical model is expected. Diagrams/Tools that may be provided in this section include Component Diagram, ER Diagram, Class Diagram (with complete inheritance, composition, and association details), Activity Diagram, Decision Table etc. All these diagrams should have more details than the details provided in Phase 1 of the SDP. Research based projects may provide complete design of the proposed system. Describe each diagram briefly.>

## Component Interactions and Collaborations

<Provide interactions and collaborations between your system components/processing units. Diagrams/Tools that may be provided in this section include Design Level Sequence Diagram, Collaboration Diagram, Event Traces, Detailed DFD, Activity Diagram etc. All these diagrams should have more details than the details provided in Phase 1 of the SDP. Describe each diagram briefly.>

## System Architecture

<Provide the technical architecture of your system. Include a high-level architecture diagram that highlights major subsystems and components.>

## Architecture Evaluation

<Describe the reason(s) behind the decision related to selection of infrastructure/technology used. Discuss pros and cons of the selected technology/infrastructure and the alternative technology/infrastructure. For example why have your group selected a particular API from a list of APIs? Why have your group preferred a particular framework over an alternative framework.>

## Component-External Entities Interface

< Express the communication between components of your system and components of other systems such as NADRA system, bank, credit card verification system, third party multi game server etc. Use appropriate diagram to show the interaction in a better manner.>

## Screenshots/Prototype

### Workflow

<Describe complete workflow of your system. Swim-lane diagram may be used. This section should be an improved version of the section presented in previous phases>

### Screens

< Include all screenshots of your complete software application’s graphical user interface.>

## Other Design Details

<Describe all design details not covered in previous sections. Add subsections as required. There can be details regarding Research Oriented, Game Oriented, or Hardware based projects that have not been covered in this document before, those details can be provided in this section. For example research based projects may use this section to present their results and analysis; hardware based projects may use this section to describe interface dependencies and issues etc.>

# Test Specification and Results

## Test Case Specification

< Fill out the following template for each test case, also add any additional test cases that were not part of Phase 3or 4 document. Provide separate tables for input data with each test case if applicable. Research based projects may need to replace this test specification with their own test mechanism.>

|  |  |
| --- | --- |
| **Identifier** | TC-1 |
| **Related requirements(s)** | <Include use-case identifier(s) for functional requirement(s) and document section/sub-section number(s) for other requirement(s).> |
| **Short description** | … |
| **Pre-condition(s)** | … |
| **Input data** | … |
| **Detailed steps** | … |
| **Expected result(s)** | … |
| **Post-condition(s)** | … |
| **Actual result(s)** |  |
| **Test Case Result** |  |

Table 6.1: TC-1

## Summary of Test Results

<Provide in tabular form the defects found in each of your software modules. For example see Table 6.2 below.>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Module Name** | Test cases run | Number of defects found | Number of defects corrected so far | Number of defects still need to be corrected |
| **Module 1 (for example Bill Calculation Module, Speech Processing Unit** | TC1, TC2,… |  |  |  |
| **Module 2** | … |  |  |  |
|  | … |  |  |  |
| **Complete System** | <Sum all of the above> | <Sum all of the above> | <Sum all of the above> | <Sum all of the above> |

Table 6.2: Summary of All Test Results

# Conclusion and Future Work

## Project summary

<Include a brief summary of how the proposed solution is going to/has addressed the problem statement specified in the introduction section. Provide an overview of what kind of evaluations were undertaken in order to prove that the solution really solves the problem with evidence on results findings.>

## Problems faced and lessons learned

<Provide the details of problems faced during one year of tenure to complete the project. Problems can be technical, financial and motivational. List down all the lesson learned. >

## Future work

<Provide an overview of the recommendations and Include a future directions which is required as part of the future work.>

# References

<List all books, conference papers, journal articles, websites, etc. used in preparing the content of this document.All of the references should be alphabetically ordered.

Journals

Author Name/s (Surname, initial), year of publication in-parenthesis, title of the article, name of the journal, volume number, issue number (in parenthesis) followed by a colon and page numbers

Books

Author Name/s (Surname, initial), year of publication in-parenthesis, title of the book, publisher’s name, place of publication, page numbers

Reference from Internet

Name of the Author/s (If Known), Title of the topic followed by complete web address.>

# Glossary

<Define all the terms necessary to properly interpret the document, including acronyms and abbreviations.>

# Deployment/Installation Guide

<Provide a list of instructions such that users of your system can deploy and install your system on their own>

# User Manual

<Provide a manual such that users of your system can use your system after installation. In business software applications, where groups of users have access to only a sub-set of the application's full functionality, a user guide may be prepared for each group. There should be step by step instruction for each user class.>

# Student Information Sheet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Roll No | Name | Email Address (FC College) | Frequently Checked Email Address | Personal Cell Phone Number |
|  |  |  |  |  |
|  |  |  |  |  |

# Plagiarism Free Certificate

This is to certify that, I am \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ S/D/o \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, group leader of FYP under registration no \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ at Computer Science Department, Forman Christian College (A Chartered University), Lahore. I declare that my Final year project report is checked by my supervisor and the similarity index is \_\_\_\_\_\_\_\_% that is less than 20%, an acceptable limit by HEC. Report is attached herewith as Appendix F. To the best of my knowledge and belief, the report contains no material previously published or written by another person except where due reference is made in the report itself.

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

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

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

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

Senior Project Management Committee Representative: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

# Plagiarism Report