

National University of Computer and Emerging Sciences



<Project Name>

**FYP Team**

Saad Nauman ………… 19L-0996

Muhammad Abdullah Umer ………… 19L-0997

Laiba Afzal ………… 19L-2351

**Supervised by**

**Mr. Waqas Manzoor**

**FAST School of Computing**

**National University of Computer and Emerging Sciences**

**[Lahore], Pakistan**

**October 2022**

Anti-Plagiarism Declaration

This is to declare that the above publication produced under the:

**Title: \_\_\_\_Your FYP Report Title Here\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

is the sole contribution of the author(s) and no part hereof has been reproduced on **as it is** basis (cut and paste) which can be considered as **Plagiarism**. All referenced parts have been used to argue the idea and have been cited properly. I/We will be responsible and liable for any consequence if violation of this declaration is determined.

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student 1

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Student 2

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Student 3

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Student 4

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Authors’ Declaration***

This states Authors’ declaration that the work presented in the report is their own, and has not been submitted/presented previously to any other institution or organization.

Abstract

An Abstract is a short summary of the work being reported. Three to five sentences describing the essence of the work. An abstract is a short, 50-125 words summary of a work. An Abstract should state: the purpose, findings, and conclusion of your work without commenting on or evaluating the work itself. **It should be only one paragraph.** Put the abstract on a separate page that follows the title page.

**Executive Summary**

The executive summary should be one to two pages’ overview of the information contained in the FYP report.  It should give the reader an easy reference, in a very brief form, to the important information contained in the report and explained in more detail in the body of the report. People reading the report will use this section as a reference during presentations.

Table of Contents

[Table of Contents i](#_Toc113958742)

[List of Tables iii](#_Toc113958743)

[List of Figures iv](#_Toc113958744)

[Chapter 1: Introduction 1](#_Toc113958745)

[1.1 Purpose of this Document 1](#_Toc113958746)

[1.2 Intended Audience 1](#_Toc113958747)

[1.3 Definitions, Acronyms, and Abbreviations 1](#_Toc113958748)

[Chapter 2: Project Vision 2](#_Toc113958749)

[2.1 Problem Domain Overview 2](#_Toc113958750)

[2.2 Problem Statement 2](#_Toc113958751)

[2.3 Problem Elaboration 2](#_Toc113958752)

[2.4 Goals and Objectives 2](#_Toc113958753)

[2.5 Project Scope 2](#_Toc113958754)

[2.6 Sustainable Development Goal (SDG) 2](#_Toc113958755)

[2.7 Constraints 3](#_Toc113958756)

[2.8 Business Opportunity 3](#_Toc113958757)

[2.9 Stakeholders Description/ User Characteristics 3](#_Toc113958758)

[2.9.1 Stakeholders Summary 3](#_Toc113958759)

[2.9.2 Key High-Level Goals and Problems of Stakeholders 3](#_Toc113958760)

[Chapter 3: Literature Review / Related Work 4](#_Toc113958761)

[3.1 Definitions, Acronyms, and Abbreviations 4](#_Toc113958762)

[3.2 Detailed Literature Review 4](#_Toc113958763)

[3.2.1 Related Research Work 1 4](#_Toc113958764)

[3.3 Literature Review Summary Table 4](#_Toc113958765)

[3.4 Conclusion 4](#_Toc113958766)

[Chapter 4: Software Requirement Specifications 5](#_Toc113958767)

[4.1 List of Features 5](#_Toc113958768)

[4.2 Functional Requirements 5](#_Toc113958769)

[4.3 Quality Attributes 5](#_Toc113958770)

[4.4 Non-Functional Requirements 5](#_Toc113958771)

[4.5 Assumptions 5](#_Toc113958772)

[4.6 Hardware and Software Requirements 5](#_Toc113958773)

[4.6.1 Hardware Requirements 5](#_Toc113958774)

[4.6.2 Software Requirements 5](#_Toc113958775)

[4.7 Use Cases 5](#_Toc113958776)

[4.7.1 Sample Use Case Name Here 5](#_Toc113958777)

[4.8 Graphical User Interface 7](#_Toc113958778)

[4.9 Database Design (*if required*) 7](#_Toc113958779)

[4.9.1 ER Diagram 7](#_Toc113958780)

[4.9.2 Data Dictionary 7](#_Toc113958781)

[4.10 Risk Analysis 7](#_Toc113958782)

[Chapter 5: Proposed Approach and Methodology 8](#_Toc113958783)

[Chapter 6: High-Level and Low-Level Design 9](#_Toc113958784)

[6.1 System Overview 9](#_Toc113958785)

[6.2 Design Considerations 9](#_Toc113958786)

[6.2.1 Assumptions and Dependencies 9](#_Toc113958787)

[6.2.2 General Constraints 9](#_Toc113958788)

[6.2.3 Goals and Guidelines 10](#_Toc113958789)

[6.2.4 Development Methods 10](#_Toc113958790)

[6.3 System Architecture 10](#_Toc113958791)

[6.3.1 Subsystem Architecture 11](#_Toc113958792)

[6.4 Architectural Strategies 11](#_Toc113958793)

[6.4.1 Architectural Strategies Strategy-1 name or description 12](#_Toc113958794)

[6.4.2 Architectural Strategies Strategy-2 name or description 12](#_Toc113958795)

[6.5 Domain Model/Class Diagram 12](#_Toc113958796)

[6.6 Sequence Diagrams 12](#_Toc113958797)

[6.7 Policies and Tactics 12](#_Toc113958798)

[6.7.1 Policy/tactic-1 name or description 13](#_Toc113958799)

[6.7.2 Policy/tactic-2 name or description 13](#_Toc113958800)

[Chapter 7: Implementation and Test Cases 14](#_Toc113958801)

[7.1 Implementation 14](#_Toc113958802)

[7.1.1 Implementation of First Component/Algorithm 14](#_Toc113958803)

[7.2 Test case Design and description 14](#_Toc113958804)

[7.2.1 Sample Test case No.1 14](#_Toc113958805)

[7.3 Test Metrics 15](#_Toc113958806)

[7.3.1 Sample Test case Matric.No.1 15](#_Toc113958807)

[7.3.2 Sample Test case Metric.No.2 15](#_Toc113958808)

[Chapter 8: User Manual 16](#_Toc113958809)

[Chapter 9: Experimental Results and Discussion 17](#_Toc113958810)

[Chapter 10: Conclusion and Future Work 18](#_Toc113958811)

[References 19](#_Toc113958812)

[Appendix 20](#_Toc113958813)

[Appendix A: Guidelines 20](#_Toc113958814)

[Appendix B: Heading of Sample Appendix B 20](#_Toc113958815)

[Formatting Guidelines 21](#_Toc113958816)

[Chapter 1: Heading 1 21](#_Toc113958817)

[1.1 Heading 2 21](#_Toc113958818)

[1.1.1 Heading 3 21](#_Toc113958819)

[Tables and Figures 22](#_Toc113958820)

[Equations 23](#_Toc113958821)

[Header/Footer 23](#_Toc113958822)

[Other Formatting Guidelines 23](#_Toc113958823)

[References 23](#_Toc113958824)

List of Tables

[Table 1: History of Computing Devices 4](#_Toc113452247)

[Table 2: This is Sample table caption 22](#_Toc113452248)

[Table 3: This is Sample table caption 22](#_Toc113452249)

List of Figures

Figure 1: Sustainable Development Goals 2

Figure 2: List of Styles 22

Figure 3: IEEE Reference style 23

# Introduction

**This chapter is mandatory.** The introduction should contain a brief overview of the problem being addressed and the background information needed for the reader to understand the work being done and the reasoning behind it.

**The last paragraph of introduction chapter should contain an outline of the entire report. Summarize each chapter in one line to make the last paragraph.**

## Purpose of this Document

Specify the purpose of this Project.

## Intended Audience

## Definitions, Acronyms, and Abbreviations

List all important definitions, the acronyms and abbreviations used in this document.

**SDG:** Sustainable Development Goal

# Project Vision

**For each related work provide a paragraph of introduction and in the end a paragraph of conclusions.** Give a page break after the chapter ends. **This chapter is mandatory.**

Clearly specify the **goals and objectives** of the project along with the scope of the project. (You can make sub-heading of goals and objectives and scope of project).

## Problem Domain Overview

Describe in full detail what your system does. This section can be further divided into sub-sections, but basically a paragraph description of the system is required.

The system is based on the concept of carpooling i.e., a person is willing to share a ride with strangers. To further elaborate the idea, a person with a vehicle (driver) will be able to turn on his/her application that would use live GPS services of google maps and indicate the availability of the driver to other fellow people (riders). In addition, the driver will also announce the route he will be taking prior to the ride. If the route is feasible to the riders, the riders will be able to place a bid for the ride, and upon successful negotiation with the driver, the driver will pick up the rider/riders and drop them accordingly. Furthermore, the system will provide a platform to different local restaurants/brands to offer the passengers on route with custom deals/offers. If the passengers find the deals feasible, they might stop by to shop. This feature will also provide a platform to the restaurants/brands to advertise their products.

## Problem Statement

## Problem Elaboration

Describe the problem in detail and the various sub-problems you would work on.

## Goals and Objectives

Write goals and objectives here.

## Project Scope

Specify scope of your project here

## Sustainable Development Goal (SDG)

Identify at least one SDG area that you are targeting in this project. This has to be documented as a paragraph. The complete SDG list is shown in the Figure below.



Figure 1: Sustainable Development Goals

This figure represents all the SDG’s that can be target of a FYP

## Constraints

## Business Opportunity

## Stakeholders Description/ User Characteristics

Identify and briefly describe who the users of this system will be, and their roles.

### Stakeholders Summary

### Key High-Level Goals and Problems of Stakeholders

Notice how a section’s heading is changed to “heading 1”. It will show up in table of contents as a section under a chapter. Notice its numbering also. Similarly, subsections can have heading 2 and heading 3 styles.

NOTE: Subsections from **2.7 till 2.9** are optional for **Research** projects but compulsory for **Development** projects.

# Literature Review / Related Work

**For each related work provide a paragraph of introduction and in the end a paragraph of conclusions.** Give a page break after the chapter ends. **This chapter is mandatory.**

NOTE: For development projects describe related or similar work done by other teams and details of their methods/algorithms. For a research project a detailed literature survey is expected.

## Definitions, Acronyms, and Abbreviations

## Detailed Literature Review

This section will include detailed literature review of your problem area. Make different categories for different types of work done in the past. In addition to textual descriptions, make a summary table that describes each paper that you have read, along with references.

### Related Research Work 1

#### Summary of the research item (1 or 2 paragraphs)

#### Critical analysis of the research item (Strengths and Weaknesses)

#### Relationship to the proposed research work

## Literature Review Summary Table

The columns in the table depend upon your problem and should be specific to your project.

Table 1: History of Computing Devices

The summary of various computing devices invented in the past from 1833-1901 is presented here.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Name, reference | Inventor | Year | Input | Output | Description |
| 1. | Analytical Engine, [1] | Charles Babbage | 1833 | Punch cards | Printer, curve plotter, bell | First general purpose computer that had an arithmetical logic unit and could compute using conditional branching and loops. Also incorporated integrated memory. |

## Conclusion

Write the conclusion of the literature review.

# Software Requirement Specifications

Describe all modules of requirements and design in clear English text along with the necessary diagram and figures. Anyone reading your report should be able to reproduce your system/results after reading it. It describes functional requirements, design constraints, and other factors necessary to provide a complete and comprehensive description of the requirements for the software.

**For each chapter provide a paragraph of introduction and in the end a paragraph of conclusions.** Make sure no heading/sub heading is blank. Write text to introduce each section as well.

NOTE: This Chapter is optional for **Research** projects but compulsory for **Development** projects.

## List of Features

List all important features of your system here.

## Functional Requirements

The functional requirements fully describe the external behavior of the system. Identify and list each functionality and give a brief description, along with the user of each functionality.

## Quality Attributes

## Non-Functional Requirements

This section should describe all the non-functional requirements including: reusability, performance (how many maximum users can access it at a time), extensibility etc.

## Assumptions

List down all the assumptions, made for the specification.

## Hardware and Software Requirements

List the hardware and software requirements that will be required to develop and deploy the project.

### Hardware Requirements

### Software Requirements

## Use Cases

This section lists use cases or scenarios from the use-case model if they represent some significant, central functionality of the final system, or if they have a large architectural coverage—they exercise many architectural elements or if they stress or illustrate a specific, delicate point of the architecture.

### Sample Use Case Name Here

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | | Sample Use Case Name Here | | |
| Actors | | Admin, Business Owner, Store Manager | | |
| Summary | | The user shall provide their email and password on the login form and after successful verification, redirect the user to the home page. | | |
| Pre-Conditions | | The user must be in the database records either added by any of the authorized users or added manually by a developer.  The user must not already be logged in. | | |
| Post-Conditions | | The user’s session is successfully established and shall be redirected to the home page. | | |
| Special Requirements | | None | | |
| Basic Flow | | | | |
| Actor Action | | | **System Response** | |
| 1 | The user opens the login page. | | 2 | The login page is displayed asking for email and password. |
| 3 | The user enters valid email and password. | | 4 | The system verifies the email and password, establishes a session for the user and redirects the user to the home page. |
| **Alternative Flow** | | | | |
| 3 | The user enters invalid email or password. | | 4-A | The system responds with an error message: *Incorrect email or password entered.* |

**Note: A sample use case is given in the above table. You have to follow this tabular format for use cases. Following is the description of content required in this section:**

This section lists use cases or scenarios from the use-case model if they represent some significant, central functionality of the final system, or if they have a large architectural coverage—they exercise many architectural elements or if they stress or illustrate a specific, delicate point of the architecture.

**Summary:** The description briefly conveys the role and purpose of the use case. A single paragraph will suffice for this description.

**Basic Flow:** This use case starts when the actor does something. An actor always initiates use cases. The use case describes what the actor does and what the system does in response. It is phrased in the form of a dialog between the actor and the system.

The use case describes what happens inside the system, but not how or why. If information is exchanged, be specific about what is passed back and forth. For example, it is not very illuminating to say that the actor enters customer information. It is better to say the actor enters the customer’s name and address. A Glossary of Terms is often useful to keep the complexity of the use case manageable⎯you may want to define things like customer information there to keep the use case from drowning in details.

Simple alternatives may be presented within the text of the use case. If it only takes a few sentences to describe what happens when there is an alternative, do it directly within the **Flow of Events** section. If the alternative flow is more complex, use a separate section to describe it. For example, an **Alternative Flow** subsection explains how to describe more complex alternatives.

A picture is sometimes worth a thousand words, though there is no substitute for clean, clear prose. If it improves clarity, feel free to paste graphical depictions of user interfaces, process flows or other figures into the use case. If a flow chart is useful to present a complex decision process, by all means use it! Similarly for state-dependent behavior, a state-transition diagram often clarifies the behavior of a system better than pages upon pages of text. Use the right presentation medium for your problem, but be wary of using terminology, notations or figures that your audience may not understand. Remember that your purpose is to clarify, not obscure.

**First Alternative Flow:** More complex alternatives are described in a separate section, referred to in the **Basic Flow** subsection of **Flow of Events** section. Think of the **Alternative Flow** subsections like alternative behavior⎯ each alternative flow represents alternative behavior usually due to exceptions that occur in the main flow. They may be as long as necessary to describe the events associated with the alternative behavior. When an alternative flow ends, the events of the main flow of events are resumed unless otherwise stated.

**Second Alternative Flow:** There may be, and most likely will be, a number of alternative flows in a use case. Keep each alternative flow separate to improve clarity. Using alternative flows improves the readability of the use case, as well as preventing use cases from being decomposed into hierarchies of use cases. Keep in mind that use cases are just textual descriptions, and their main purpose is to document the behavior of a system in a clear, concise, and understandable way.

**Special Requirements:** A special requirement is typically a nonfunctional requirement that is specific to a use case, but is not easily or naturally specified in the text of the use case’s event flow. Examples of special requirements include legal and regulatory requirements, application standards, and quality attributes of the system to be built including usability, reliability, performance or supportability requirements. Additionally, other requirements⎯such as operating systems and environments, compatibility requirements, and design constraints⎯should be captured in this section.

**Pre-Conditions:** A pre-condition of a use case is the state of the system that must be present prior to a use case being performed.

**Post-Conditions:** A post-condition of a use case is a list of possible states the system can be in immediately after a use case has finished.

## Graphical User Interface

This section should give the GUI dumps of each screen, with reference to the users. The navigation flow of each user is also required, and each GUI should mark the functionality/use case that it covers.

## Database Design (*if required*)

### ER Diagram

### Data Dictionary

## Risk Analysis

List and explain the risks that maybe encountered during the project. For e.g.: technical risks, business risks etc.

# Proposed Approach and Methodology

The proposed approach could be a Framework, Heuristic, Algorithm, Protocol, or a Mathematical-model.

**For each chapter provide a paragraph of introduction and in the end a paragraph of conclusions.** Not the programming code but the algorithmic and procedural details especially related to the hidden/ backend algorithms that are not covered in the design.

NOTE: This chapter is optional for **Development** projects but compulsory for **Research** projects.

# High-Level and Low-Level Design

Provide the high- and low-level design of your system in this chapter. Select the design that is appropriate for your project.

NOTE: This Chapter is optional for **Research** projects but compulsory for **Development** projects.

## System Overview

Provide a general description of the software system including its functionality and matters related to the overall system and its design (perhaps including a discussion of the basic design approach or organization). Feel free to split this discussion up into subsections (and sub subsections, etc. ...).

## Design Considerations

This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution.

### Assumptions and Dependencies

Describe any assumptions or dependencies regarding the software and its use. These may concern such issues as:

* Related software or hardware
* Operating systems
* End-user characteristics
* Possible and/or probable changes in functionality

### General Constraints

Describe any global limitations or constraints that have a significant impact on the design of the system's software (and describe the associated impact). Such constraints may be imposed by any of the following (the list is not exhaustive):

* Hardware or software environment
* End-user environment
* Availability or volatility of resources
* Standards compliance
* Interoperability requirements
* Interface/protocol requirements
* Data repository and distribution requirements
* Security requirements (or other such regulations)
* Memory and other capacity limitations
* Performance requirements
* Network communications
* Verification and validation requirements (testing)
* Other means of addressing quality goals
* Other requirements described in the requirements specification

### Goals and Guidelines

Describe any goals, guidelines, principles, or priorities which dominate or embody the design of the system's software. Such goals might be:

* The KISS principle ("Keep it simple stupid!")
* Emphasis on speed versus memory use
* Working, looking, or "feeling" like an existing product

For each such goal or guideline, unless it is implicitly obvious, describe the reason for its desirability. Feel free to state and describe each goal in its own sub subsection if you wish.

### Development Methods

Briefly describe the method or approach used for this software design. If one or more formal/published methods were adopted or adapted, then include a reference to a more detailed description of these methods. If several methods were seriously considered, then each such method should be mentioned, along with a brief explanation of why all or part of it was used or not used.

## System Architecture

This section should describe both the internal architecture of the modules, as well as the external architecture of the system with other systems, if any. **Diagrammatic architecture is compulsory.**

This section should provide a high-level overview of how the functionality and responsibilities of the system were partitioned and then assigned to subsystems or components. Do not go into too much detail about the individual components themselves (there is a subsequent section for detailed component descriptions). The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.

At the top-most level, describe the major responsibilities that the software must undertake and the various roles that the system (or portions of the system) must play. Describe how the system was broken down into its components/subsystems (identifying each top-level component/subsystem and the roles/responsibilities assigned to it). Describe how the higher-level components collaborate with each other in order to achieve the required results. Don't forget to provide some sort of rationale for choosing this particular decomposition of the system (perhaps discussing other proposed decompositions and why they were rejected). Feel free to make use of design patterns, either in describing parts of the architecture (in pattern format), or for referring to elements of the architecture that employ them.

If there are any diagrams, models, flowcharts, documented scenarios or use-cases of the system behavior and/or structure, they may be included here (unless you feel they are complex enough to merit being placed in the Detailed System Design section). Diagrams that describe a particular component or sub-system should be included within the particular subsection that describes that component or subsystem.

**Note:** This section (and its subsections) really applies only to newly developed (or yet-to-be developed) portions of the system. If there are parts of the system that already existed before this development effort began, then you only need to describe the pre-existing parts that the new parts of the system depend upon, and only in enough detail sufficient to describe the relationships and interactions between the old parts and the new parts. Pre-existing parts that are modified or enhanced need to be described only to the extent that is necessary for the reader to gain a sufficient understanding of the nature of the changes that were made.

### Subsystem Architecture

If a particular component is one which merits a more detailed discussion than what was presented in the System Architecture section, provide that more detailed discussion in a subsection of the System Architecture section (or it may even be more appropriate to describe the component in its own design document). If necessary, describe how the component was further divided into subcomponents, and the relationships and interactions between the subcomponents (similar to what was done for top-level components in the System Architecture section).

If any subcomponents are also deemed to merit further discussion, then describe them in a separate subsection of this section (and in a similar fashion). Proceed to go into as many levels/subsections of discussion as needed in order for the reader to gain a high-level understanding of the entire system or subsystem (but remember to leave the gory details for the Detailed System Design section).

If this component is very large and/or complex, you may want to consider documenting its design in a separate document and simply including a reference to it in this section. If this is the option you choose, the design document for this component should have an organizational format that is very similar (if not identical to) this document.

## Architectural Strategies

Explain all strategies that you use for designing of the Architecture. Describe any design decisions and/or strategies that affect the overall organization of the system and its higher-level structures. These strategies should provide insight into the key abstractions and mechanisms used in the system architecture. Describe the reasoning employed for each decision and/or strategy (possibly referring to previously stated design goals and principles) and how any design goals or priorities were balanced or traded-off. Such decisions might concern (but are not limited to) things like the following:

* Use of a particular type of product (programming language, database, library, etc. ...)
* Reuse of existing software components to implement various parts/features of the system
* Future plans for extending or enhancing the software
* User interface paradigms (or system input and output models)
* Hardware and/or software interface paradigms
* Error detection and recovery
* Memory management policies
* External databases and/or data storage management and persistence
* Distributed data or control over a network
* Generalized approaches to control
* Concurrency and synchronization
* Communication mechanisms
* Management of other resources

### Architectural Strategies Strategy-1 name or description

### Architectural Strategies Strategy-2 name or description

Each significant strategy employed should probably be discussed in its own subsection, or (if it is complex enough) in a separate design document (with an appropriate reference here of course). Make sure that when describing a design decision that you also discuss any other significant alternatives that were considered, and your reasons for rejecting them (as well as your reasons for accepting the alternative you finally chose). Sometimes it will be most effective to employ the "pattern format" for describing a strategy.

## Domain Model/Class Diagram

In this subsection add the Class Diagram of your system. Class diagrams represents the structure design of your system.

## Sequence Diagrams

In this subsection add all sequence diagrams of your system components. Behavior Design of your system is defined through the diagrams such as sequence diagrams, interaction overview diagrams, flow diagram, data flow diagram, activity diagram, network automata/ graphs or stat machine, call graph.

## Policies and Tactics

Describe any design policies and/or tactics that do not have sweeping architectural implications (meaning they would not significantly affect the overall organization of the system and its high-level structures), but which nonetheless affect the details of the interface and/or implementation of various aspects of the system. Such decisions might concern (but are not limited to) things like the following:

* Choice of which specific product to use (compiler, interpreter, database, library, etc. ...)
* Engineering trade-offs
* Coding guidelines and conventions
* The protocol of one or more subsystems, modules, or subroutines
* The choice of a particular algorithm or programming idiom (design pattern) to implement portions of the system's functionality
* Plans for ensuring requirements traceability
* Plans for testing the software
* Plans for maintaining the software
* Interfaces for end-users, software, hardware, and communications
* Hierarchical organization of the source code into its physical components (files and directories).
* How to build and/or generate the system's deliverables (how to compile, link, load, etc. ...)

### Policy/tactic-1 name or description

### Policy/tactic-2 name or description

Each particular policy or set of tactics employed should probably be discussed in its own subsection, or (if it is large or complex enough) in a separate design document (with an appropriate reference here of course). Make sure that when describing a design decision that you also discuss any other significant alternatives that were considered, and your reasons for rejecting them (as well as your reasons for accepting the alternative you finally chose). For this reason, it may frequently be convenient to use one of the more popular "pattern formats" to describe a given tactic.

For this particular section, it may become difficult to decide whether a particular policy or set of tactics should be discussed in this section, or in the System Architecture section, or in the Detailed System Design section for the appropriate component. You will have to use your own "best" judgment to decide this. There will usually be some global policies and tactics that should be discussed here, but decisions about interfaces, algorithms, and/or data structures might be more appropriately discussed in the same (sub) section as its corresponding software component in one of these other sections.

# Implementation and Test Cases

**For each chapter provide a paragraph of introduction and in the end a paragraph of conclusions.** (Not the programming code but the algorithmic and procedural details especially related to the hidden/ backend algorithms that are not covered in the design)

## Implementation

Whatever implementation that you have done so far, please elaborate here.

Give clear details of the algorithms that were implemented along with the platform and the APIs which were used.

NOTE: For FYP-1, this chapter can be changed to description of prototype developed.

### Implementation of First Component/Algorithm

Write implementation of first component of your system here.

## **Test case Design and description**

**This section will be added in FYP-II.** Summarize the common attributes of test cases. This may include input constraints that must be true for every input in the set of associated test cases, any shared environmental needs, any shared special procedural requirements, and any shared case dependencies. The following scheme is recommended for describing test cases in detail.

### Sample Test case No.1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **<Software component Name>** | | | | | |
| **<Reference>** | | | | | |
| Test Case ID: | | *Reference Number* | QA Test Engineer: | | *Name of personnel* |
| Test case Version: | | *Version number* | Reviewed By: | | *Testing Team lead* |
| Test Date: | | *Date* | Use Case Reference(s): | | *Relation to use cases* |
| Revision History: | | *Refer to previous test case identity (if any)* | | | |
| Objective | | *Need and scope of the testing* | | | |
| Product/Ver/Module: | | *Refer to overall system being built and the place of this test case in it.* | | | |
| Environment: | | *Necessary and desired properties of the test environment. (hardware/software)* | | | |
| Assumptions: | | *Assumptions that might affect the testing process.* | | | |
| Pre-Requisite: | | *Necessary condition that needs to be fulfilled prior to the test case.* | | | |
| Step No. | Execution description | | | Procedure result | |
|  | *Events being tested.* | | | *Mention software response.* | |
| Comments: | | | | | |
| *Passed* *Failed* *Not Executed* | | | | | |

## Test Metrics

Summarize here the common ground of attributes of test case metrics.

### Sample Test case Matric.No.1

|  |  |
| --- | --- |
| Metric | Purpose |
| Number of Test Cases | Total number of test cases that you have developed for your system. |
| Number of Test Cases Passed | The number of test cases that successfully passed |
| Number of Test Cases Failed | The number of test cases that failed |
| Test Case Defect Density | (No of test cases failed \* 100)  No of test cases executed |
| Test Case Effectiveness | No of defects detected using test cases \*100  Total number of defects detected |
| Traceability Matrix | Traceability is the ability to determine that each feature has a source in requirements and each requirement has a corresponding implemented feature. |

### Sample Test case Metric.No.2

# User Manual

**This chapter will be added in FYP-II.** Design a complete user manual of your application and add that in this chapter.

NOTE: This Chapter is optional for **Research** projects but compulsory for **Development** projects.

# Experimental Results and Discussion

**This chapter will be added in FYP-II.** Give proper analysis and discussion of experimental results (in plain English text) along with tables of results. **For each chapter provide a paragraph of introduction and in the end a paragraph of conclusions.**

NOTE: This Chapter is optional for **Development** projects but compulsory for **Research** projects.

# Conclusion and Future Work

**This chapter is mandatory.** Give conclusions and summary of the work done. What were your findings and what were the results? Discuss in detail whether the scope of your project was entirely covered or not and whether the objectives of the project were met or not. What challenges did you face and what has been left out and why?

Sum up all the conclusions of all the chapters here to make a final conclusion chapter. Do not repeat any text, just summarize it in different words.

Give recommendations for future work also. How your project can be further enhanced or improved? Future recommendations if someone wants to work on it.

**For FYP-1 it is mandatory to list down a plan of the work to be done for FYP-2.**

# References

**This chapter is mandatory.** List all important sources of information which have been consulted for this project.

# Appendix

## Appendix A: Guidelines

This section should include all supporting information from the project that was not included in the body of the report.  You should include surveys, complex statistical calculations, certain detailed tables and other such information in an appendix.  The information presented in this section is important to support the work presented in the body of the report but would make it more difficult to read and understand if presented within the body of the report.

Cite the appendix items in the report narrative (write "see Appendix A") and organize appendices (e.g., Appendix A, Appendix B,

Any tables, figures, forms, or other materials that are not totally central to the analysis but that need to be included are placed in the Appendix.

## Appendix B: Heading of Sample Appendix B

Following is a sample code with “code” style format.

Void SampleFunction(){

Print “Hello World.”;

}

# Formatting Guidelines

This document also serves as style guide for final year project reports. In order to give a similar high-quality appearance to all final year software project reports this template uses a collection of predefined Microsoft Word formatting styles. **These styles should be used without modification or replacement.** Font in the document is ***“Time New Roman”.*** This template provides following styles:

* **Title** – the main title style
* **Title2** – the subtitle style
* **Body Text** – style for paragraphs
* **Caption** – the style for a figure or table caption
* **Table Description** – the style for description of table, it must be added after caption.
* **Figure Description** - the style for description of figure, it must be added after caption.
* **Code** – the style for program source code

**int x** = 10; // Writing important code

* **Table Header Row** – Style for the header row of table
* **Table Grid** – the style for the data rows in the tables
* **Reference** – The style for references
* **Bullets** – The style for the bullet lists
* **Numbered** **List**– Style for numbered lists

All Heading styles with different level numbers are listed below.

# Heading 1

## Heading 2

### Heading 3

#### Heading 4

##### Heading 5

###### Heading 6

Heading 7

Heading 8

Heading 9

## Tables and Figures

Tables and figures should be centered horizontally. The caption button should be used to insert caption for both the figures and tables. All figures and tables must be numbered properly. Always refer to tables and figures according to their numbers. A table or figure can be cited as follows: ‘see Table1’ or ‘as shown in Table1’. The caption of table should be centered above the table and figure caption should be centered below the figure. Place the tables/figures close to their reference. Use “Table Header Row” and ‘Table Grid’ style for table’s header and data rows respectively. It is compulsory to provide brief description of table/figure after its caption. Styles for table and figure descriptions are “Table Description” and “Figure Description” respectively.

Press **Ctrl + Shift + S** to see list of styles mentioned above. Figure 1 shows the Apply Style window displaying the list of styles. Select any text then press **Ctrl + Shift + S**, the Apply Style window will show you the current style applied on that text and if required, you can change the style by selecting any other style from the “Style Name” dropdown.

This is brief description of above figure.

Figure 2: List of Styles

Table 2: This is Sample table caption

This is brief description of following Table.

|  |  |  |  |
| --- | --- | --- | --- |
| Header row | Header row | Header row | Header row |
| Row1 col1 | Row1 col2 | Row1 col3 | Row1 col4 |
| Row2 col1 | Row2 col2 | Row2 col3 | Row2 col4 |

Table 3: This is Sample table caption

This is brief description of following Table.

|  |  |  |  |
| --- | --- | --- | --- |
| Header row | Header row | Header row | Header row |
| Row1 col1 | Row1 col2 | Row1 col3 | Row1 col4 |
| Row2 col1 | Row2 col2 | Row2 col3 | Row2 col4 |

## Equations

Use equation editor to write equations in this report. Use last button of the custom tool bar to invoke equation editor. Similar to tables and figures, equations should also be aligned centered horizontally. Number all equations and insert them in parenthesis. Below is a sample equation and its reference number. An equation can be referenced like this: ‘it is clear from (1)’.

 (1)

## Header/Footer

Notice the headers in this document, before Introduction (i.e., the main content of this document) page numbers are in roman numerals. The page numbers of the actual content start with Arabic numerals i.e. 1, 2, 3 and so on. All of the **odd numbered pages** contain title of your project while the **even numbered pages** contain the section heading (i.e., chapter’s name) in the headers.

## Other Formatting Guidelines

* Keep 2-4 GUIs in one page. Consume as much space as possible. Do not leave most of page blank unnecessarily.
* Do not break tables (or use cases) in multiple pages unless the table is too large to fit in one page.
* Re-arrange the content i.e., text, images and tables properly to meet above two guidelines.

## References

Always refer to the source of information by inserting the reference number in square brackets like this [5]. The reference numbers can either be added at the end of the sentence or within the sentence without changing the punctuation of sentence. A reference can also be cited as follows: ‘as Ruskey [2] mentioned’. List each source only once on your reference page.



Figure 3: IEEE Reference style

This figure represents the styling information for adding references in IEEE format

**Following is a list of sample reference for various type of sources in IEEE format.**

1. P.M. Morse and H. Feshback, *Methods* of *Theoretical Physics*. New York: McGraw Hill, 1953. **//Format for Book**
2. S.K. Kenue and J.F. Greenleaf, “Limited angle multifrequency deification tomography,” *IEEE Trans. Sonics Ultrason*., vol. SU-29, no. 6, pp. 213-2 17, July 1982. **//Format for Journal Article**
3. B. Tsikos, “Segmentation of 3-D scenes using multi-modal interaction between machine vision and programmable mechanical scene manipulation,” Ph.D. dissertation, Univ. of Pennsylvania, BCE Dept., Philadelphia, 1987. [Add if applicable: University Microfilms, Inc., University of Michigan, Ann Arbor, Michigan.] **//Format for Dissertation or thesis**
4. R. Finkel, R. Taylor, R. Bolles, R. Paul, and J. Feldman, “An overview of AL, programming system for automation,” in *Proc. Fourth Int. Joint Conf Artif. Intell*., pp. 758-765, Sept. 3-7, 1975. **//Format for Proceedings paper**
5. “Technology threatens to shatter the world of college textbooks, *The Wall Street Journal*, vol 91, pp. Al, A8, June 1, 1993. **//Format for Newspaper article**
6. R. Cox and J. S. Turner, “Project Zeus: design of a broadband network and its application on a university campus,” Washington Univ., Dept. of Comp. Sci., Technical Report WUCS-91-45, July 30, 1991. **//Format for Technical Report**
7. M. Janzen, *Instant Access Accounting*. Computer software. Nexus Software, Inc IBM-PC, 1993. **//Format for** **Software**
8. Fuminao Okumura and Hajime Takagi, “Maglev Guideway On the Yamanashi Test Line,” *http://www.rtri.or.jp/rd/maglev2/okumura.html*, October 24, 1998. **//Format for** **World Wide Web** (give author and title if named)
9. “AT&T Supplies First CDMA Cellular System in Indonesia,” http://www.att.com/press/1095/951011.nsa.html, Feb 5, 1996. **//Format for World Wide Web**