# Annex-C

**Project Report**

**Project Title**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Submitted by**

**Student Name** (Program-Roll#-Section)

**Student Name** (Program-Roll#-Section)

Session 2019-2023

**Supervised by**

**Teacher Name**



**Department of Computer Science & Information Technology**

**Lahore Garrison University**

**Lahore**

**Project Title**

A project submitted to the

Department of Computer Science

In

Partial Fulfillment of the Requirements for the

Bachelor’s / Master Degree in Computer Science

By

**Student Names**

**Internal Supervisor**

**Teacher Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Designation   
Department of Computer Science

**External Examiner** **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Chairperson**

**Dr. Muhammad Arfan Ali Nagra \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Assistant Professor   
Department of Computer Science

**COPYRIGHTS**

This is to certify that the project titled “**Project Title”** is the genuine work carried out by **Student Names,** student of BSCS of Computer Science Department, Lahore Garrison University, Lahore during the academic year 2015-19, in partial fulfilment of the requirements for the award of the degree of Bachelor of Computer Science and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

Student Name \_\_\_\_\_\_\_\_\_\_\_\_

Student Name \_\_\_\_\_\_\_\_\_\_\_

**DECLARATION**

This is to declare that the project entitled “**Project Title**” is an original work done by undersigned, in partial fulfilment of the requirements for the degree “Bachelor of Science in Computer Science” at Computer Science Department, Lahore Garrison University, Lahore.

All the analysis, design and system development have been accomplished by the undersigned. Moreover, this project has not been submitted to any other college or university.

**Group Members**

Student Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Supervisor**

Supervisor Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

**DEDICATION**

**ACKNOWLEDGEMENTS**

**Table of Contents**

To be added automatically through heading styles

**List of Tables**

To be added automatically through table captions styles

**List of Figures**

To be added automatically through figure captions

**List of Abbreviation**

This page will include all abbreviations used in the documentation and their full forms. The details are to be added by the student. (example given)

IP: Internet Protocol

ISO: International Systems Organization

**ABSTRACT**

The unique investment opportunity offers by Prize Bonds and also security provided by the government. Prize Bonds offer his holders with the excitement of lottery-style prize draws. Even their popularity, application offer limited scope and functionality for managing Prize Bonds. These applications mainly offer features like Prize Bond Number, Search and Prize Bonds Lists but does not focus on investors difficulty to find a Broker in this way investor unable find a broker who is near him. This creates a problem for investor to invest easily.

Challenges identified include absence of Broker information, limited market. The users are rely on limited market and unable to explore and find a Broker who is able to provide him maximum with their experience. Broker on the other hand know that if someone want to buy Prize Bonds from it then they have to go his office and the office is known by specific peoples

Prize Bond application proposed a comprehensive solution to provide its user to provide broker info with Location and benefits that a specific broker may able to provide its user. User also have the feature to save its bonds Number so that when the Bond Number appear in the List able to alarm his respected user. The scope of this application to provide a platform in which user and broker both take benefits. In this way the Broker have a Profile that show to more that specific peoples.

The approaches use in order to achieve the features take the Information from user and set as portfolio for the broker including the information that feel its client that this broker provide benefits to me. We show broker profile to users in this way the non-physical interaction of client with Broker may possible.

Existing application comparison with this application may force to believe on that this application find a problem and provide a solution to its users. Current applications not focus on the Broker and its client interaction difficulty. The proposed application with the previous features facilitate its user.

Evaluation of the application may be taken by various testing and also its users feedback and this help to ensure that the application its objective or not? These evaluation help to find its bugs and errors. In this way the application may able to achieve its objective.

In the conclusion part, the Prize Bond application focuses to solve the identified problem and help the user. By check and balance through its user feedback and testing the application achieve its objective .

# **Chapter 1**

## INTRODUCTION

Introduction is mostly written for non-specialists so that they can get an overview of the project without technical details. It should provide a brief overview of the project aims and structure of the solution. It should also specify what unmet need or problem the FYP caters for and who needs it. At the end of chapter, provide a summary of the report organization, chapter outlining what has been covered in this chapter and explain what comes in the following chapters.

# **Chapter 2**

## LITERATURE REVIEW

Introduction is mostly written for non-specialists so that they can get an overview of the project without technical details. It should provide a brief overview of the project aims and structure of the solution. It should also specify what unmet need or problem the FYP caters for and who needs it. At the end of chapter, provide a summary of the report organization, chapter outlining what has been covered in this chapter and explain what comes in the following chapters.

# **Chapter 3**

## PROBLEM DEFINITION

Describe the problem definition according to the project.

# **Chapter 4**

## SOFTWARE REQUIREMENT SPECIFICATION

Describe the SRS of project.

# **Chapter 5**

## METHODOLOGY

The methods, approaches, tools, techniques, algorithms, or other aspects of the solution are sufficiently discussed with sufficient details and supporting figures.

# **Chapter 6**

## DETAILED DESIGN AND ARCHITECTURE

### **6.1 System Architecture**

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. Don't 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 to achieve the required results. Don't forget to provide some sort of rationale for choosing this 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 subsystem should be included within the subsection that describes that component or subsystem.

#### **6.1.1 Architecture Design Approach**

Describe the architectural design approach.

#### **6.1.2 Architecture Design**

Provide and describe a figure that depicts the overall system architecture. Develop a modular program structure and explain the relationships between the modules to achieve the complete functionality of the system. This is a high-level overview of how responsibilities of the system were partitioned and then assigned to subsystems. Identify each high-level subsystem and the roles or responsibilities assigned to it. Describe how these subsystems collaborate with each other in order to achieve the desired functionality. Don’t go into too much detail about the individual subsystems. The main purpose is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together. Provide a diagram showing the major subsystems and data repositories and their interconnections. Describe the diagram if required.

#### **6.1.3 Subsystem Architecture**

Provide a decomposition of the subsystems in the architectural design. Supplement with text as needed. You may choose to give a functional description or an object-oriented description. For a functional description, put top level data flow diagram (DFD) and structural decomposition diagrams. For an OO description, put subsystem model, object diagrams, generalization hierarchy diagram(s) (if any), aggregation hierarchy diagram(s) (if any), interface specifications, and sequence diagrams here.

### **6.2 DETAILED SYSTEM DESING**

Most components described in the System Architecture section will require a more detailed discussion. Other lower-level components and subcomponents may need to be described as well. Each subsection of this section will refer to or contain a detailed description of a system software component. The discussion provided should cover the following software component attributes:

#### **6.2.1 Classification**

The kind of component, such as a subsystem, module, class, package, function, file, etc.

#### **6.2.2 Definition**

The specific purpose and semantic meaning of the component. This may need to refer back to the requirements specification.

#### **6.2.3 Responsibilities**

The primary responsibilities and/or behavior of this component. What does this component accomplish? What roles does it play? What kinds of services does it provide to its clients? For some components, this may need to refer back to the requirements specification.

#### **6.2.4 Constraints**

Any relevant assumptions, limitations, or constraints for this component. This should include constraints on timing, storage, or component state, and might include rules for interacting with this component (encompassing preconditions, post conditions, invariants, other constraints on input or output values and local or global values, data formats and data access, synchronization, exceptions, etc.)

#### **6.2.5 Composition**

A description of the use and meaning of the subcomponents which are a part of this component.

#### **6.2.6 Uses/Interactions**

Description of component collaboration with other components. What other components is this entity used by? What other components does this entity use (this would include any side-effects this entity might have on other parts of the system)? This concerns the method of interaction as well as the interaction itself. Object-oriented designs should include a description of any known or anticipated subclasses, super classes, and meta classes.

#### **6.2.7 Resources**

A description of all resources that are managed, affected, or needed by this entity. Resources are entities external to the design such as memory, processors, printers, databases, or a software library. This should include a discussion of any possible race conditions and/or deadlock situations, and how they might be resolved.

#### **6.2.8 Processing**

A description of precisely how this component goes about performing the duties necessary to fulfill its responsibilities. This should encompass a description of any algorithms used; changes of state; relevant time or space complexity; concurrency; methods of creation, initialization, and cleanup; and handling of exceptional conditions.

#### **6.2.9 Interface/Exports**

The set of services (resources, data, types, constants, subroutines, and exceptions) that are provided by this component. The precise definition or declaration of each such element should be present, along with comments or annotations describing the meanings of values, parameters, etc. For each service element described, include (or provide a reference) in its discussion a description of its important software component attributes (Classification, Definition, Responsibilities, Constraints, Composition, Uses, Resources, Processing, and Interface).

Much of the information that appears in this section is not necessarily expected to be kept separate from the source code. In fact, much of the information can be gleaned from the source itself (especially if it is adequately commented). This section should not copy or reproduce information that can be easily obtained from reading the source code (this would be an unwanted and unnecessary duplication of effort and would be very difficult to keep up to date). It is recommended that most of this information be contained in the source (with appropriate comments for each component, subsystem, module, and subroutine). Hence, it is expected that this section will largely consist of references to or excerpts of annotated diagrams and source code. Any referenced diagrams or source code excerpts should be provided at any design reviews.

#### **6.2.10 Detailed Subsystem Design**

Provide a detailed description of this software component (or a reference to such a description). Complex diagrams showing the details of component structure, behavior, or information/control flow may be included in the subsection devoted to that component (although, unless they are very large or complex, some of these diagrams might be more appropriately included in the System Architecture section. The description should cover any applicable software component attributes (some of which may be adequately described solely by a source code declaration or excerpt).

**Draw following diagrams according to your project**

1. Use case Diagram
2. ER Diagram
3. Architectural Diagram
4. Activity Diagram
5. Sequence Diagram
6. Component Diagram
7. State Machine Diagram
8. Class Diagram
9. Data Flow Diagram
10. Database Diagram

# **Chapter 7**

## IMPLEMENTATION AND TESTING

Explain the methods, tools and techniques used to develop the software. What kind of software and testing methodologies implemented? Explain core functionalities in narrative format. Controlled libraries, templates, code walkthroughs, explain how the proposed software has been evaluated and compared at runtime with the original specifications. The Accuracy, Performance and Scalability of the proposed software must be critically analyzed and should solve identified problem statement.

# **Chapter 8**

## RESULTS AND DISCUSSION

A comprehensive evaluation of the solution is presented with supporting figures and graphics. System testing is performed through a strong testing strategy and the test cases cover all the use cases.

# **Chapter 9**

## CONCLUSION AND FUTURE WORK

Include a 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 to prove that the solution really solves the problem with evidence on results findings. Provide an overview of the recommendations and include a future direction which is required as part of the future work.

## REFERENCES

A comprehensive list of references is cited using a standard format.

[1] I. Thompson, “Women and feminism in technical communication,” Journal of Business and Technical Communication, vol. 13, no. 2, pp.154–178, 1999. (**Journal Article**)

[2] M. S. MacNealy, Strategies for Empirical Research in Writing. Boston, MA: Allyn and Bacon, 1999. (**Book**)

[3] K. St.Amant, “Virtual office communication protocols: A system for managing international virtual teams,” in Proceedings of IEEE International Professional Communication Conference, 2005, pp. 703–717. (**Conference Paper**)