

Project Name



By

Student Name 1
xx-ARID-xxxx

Student Name 2
xx-ARID-xxxx

Student Name 3
xx-ARID-xxxx

Supervisor
Dr./Mr./Ms. Supervisor Name

**University Institute of Information Technology
PMAS-Arid Agriculture University Rawalpindi
Pakistan-20XX**

DECLARATION

The candidate confirms that the work submitted is their own and appropriate credit has been given where reference has been made to the work of others.(Just for Information and remove this paragraph)

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software documentation and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other. We will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

Student Name 1

Student Name 2

Student Name 3

CERTIFICATE OF APPROVAL

It is to certify that the final year project of BS (CS/IT/SE) “**Project title**” was developed by “**Student Name, Registration #**”, “**Student Name, Registration #**” and “**Student Name, Registration #**” under the supervision of “**Supervisor Name**” and that in their opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Science/**Information Technology/Software Engineering**.

Supervisor

(Examiner Name)
Examiner-I

(Examiner Name)
Examiner-II

Administrator UIIT

Executive Summary

In public places, there is often a need for monitoring people and different activities going on, which can be referred later for many reasons including security. Appointing humans for this task involves many problems such as increased employee hiring, accuracy problem, trust, no proof for later use, and also the fact that a human can remember things till a certain time limit. Talking about the current security system, they use dumb still cameras with a continuous recording facility irrespective of the fact that any event may happen or not. Moreover they are usually pointing at a specific user defined locations so more than one cameras are required to cover the entire region.

To prevent all these problems from prevailing, the CSCS is developed. It is a surveillance system, which provides solution to many of these problems. It is a stand-alone application which doesn't require any computer to operate. It monitors different situations using a camera which is able to rotate intelligently based on sensor messages and captures the scene in the form of video or photos later reference as well.

Customizable Surveillance Control System (CSCS) is a surveillance system that can be assigned a sensor type as in our case a heat sensor is used, it works accordingly, rotates the camera upon event detection and perform user defined actions like capturing video and stores them, for the future use.

It is an embedded system consisting of Linux fox kit with embedded a running server application also a camera, USB storage device and a sensor node base station is attached with fox kit. LAN communication is used by user to download the videos and to operate the system manually.

Acknowledgement

All praise is to Almighty Allah who bestowed upon us a minute portion of His boundless knowledge by virtue of which we were able to accomplish this challenging task. We are greatly indebted to our project supervisor “Dr. Kashif Sattar” **and our Co-Supervisor “Dr. Tariq Ali”** for personal supervision, advice, valuable guidance and completion of this project. We are deeply indebted to him/her/them for encouragement and continual help during this work. We are also thankful to our parents and family who have been a constant source of encouragement for us and brought us the values of honesty & hard work.

Student Name 1

Student Name 2

Student Name 3

Abbreviations

SRS	Software Requirement Specification
PC	Personal Computer

Table of Contents

Introduction.....	1
1.1 Brief	2
1.2 Relevance to Course Modules	2
1.3 Project Background.....	3
1.4 Literature Review.....	3
1.5 Methodology and Software Life Cycle.....	5
Problem Definition.....	6
2.1 Purpose.....	7
2.2 Product Functions	7
2.3 Proposed Architecture.....	7
2.4 Project Deliverables	8
2.5 Operating Environment.....	8
2.6 Assumptions and Dependencies	8
Requirement Analysis.....	9
3.1 Functional Requirements	10
3.2 Non – Functional Requirements	11
3.2.1 Usability	11
3.2.2 Reliability	11
3.2.3 Performance	11
3.2.4 Supportability.....	11
3.2.5 Design Constraints	11
3.2.6 Licensing Requirements.....	11
3.3 Use case Model	12
3.3.1 Use Case Diagarm.....	12
3.3.2 Actors Discription	16
3.3.3 Use Case Discription.....	17
The Design	21
4.1 UML Structural Diagrams	22
4.1.1 Component Diagram.....	22
4.1.2 System Component Diagram	24

4.1.3	Package Diagram	25
4.1.4	Deployment Diagram.....	26
4.2	UML Behavioral Diagrams.....	27
4.2.1	Activity Diagrams.....	27
4.2.2	State Machine Diagrams.....	29
4.3	UML Interaction Diagrams.....	30
4.3.1	Sequence Diagrams.....	30
4.4	Node Structure.....	31
4.5	Communication Design Protocol	32
Implementation	33
5.1	Communication Protocol Implementation	34
5.2	PC Application Implementation	36
5.3	Embedded Application Implementation	38
5.4	Wireless Sensor Application Implementation	51
Testing and Evaluation	52
6.1	Verification	53
6.1.1	Functional Testing	53
6.1.2	Static Testing	57
6.2	Validation.....	57
6.3	Usability Testing.....	57
6.4	Unit Testing	57
6.5	Integration Testing	57
6.6	System Testing	57
GUI	58
Future Work	70
References	72

List of Figures

Fig 1.1 Block Diagram.....	8
Fig 2.1 Use Case Diagram	9

List of Tables

Table 1.1 Data Table.....	8
Table 2.1 Results.....	9

Chapter 1: Introduction

This chapter provides the overview of the project. The first paragraph of every chapter should provide the chapter summary.

1.1. Brief (sub Heading size 12 e.g. 1.1.1)

A very brief introduction of project work, outcome of your work, tools, methodology used & highlights of discussions in various chapters of report.

1.2. Relevance to Course Modules

A brief explanation of how your project is related to various courses studied during degree.

1.3. Project Background

It includes explanation of the idea behind the project. For example if the project is related to VoIP then this section describes that what is voice over IP & how it works.

1.4. Literature Review

This section describes current trends/ research/ products etc. related to your project.

1.5. Analysis from Literature Review (in the context of your project)

This section provides an analytical discussion of your work in comparison with discussion in literature review.

1.6. Methodology and Software Lifecycle for This Project

A brief discussion of methodology and SDLC model selected for this project.

1.6.1. Rationale behind Selected Methodology

Why you selected above methodology (such as structural and Object Oriented) and software life cycle for this project?

1.6.1.1.Rationale behind Selected Methodology

It is the example of third and last level heading. Please do not insert further levels in numbers. Use different format style e.g. italic to highlight the important text.

Chapter 2: Problem Definition

This chapter discusses the precise problem to be solved. It should extend to include the outcome.

2.1. Problem Statement

Problem statement goes here.

2.2. Deliverables and Development Requirements

Deliverables and development requirements.

2.3. Current System (if applicable to your project)

A brief description of an existing system. Figure 2.1 is the sample figure, please follow the same style of numbering and caption for the whole report.



Figure 2.1: Sample picture

The following table (Table 2.1) is sample table; please follow the same style of numbering and caption for the whole report.

Table 2.1: Sample Table

Header 1	Header 2	Header 3
Text	Text	Text

The following list style is the sample to consistently follow in the whole report.

- List items 1
- List items 2

Chapter 3: Requirement Analysis

Software Requirements Specification (SRS) report should be included in this chapter.

1.1.1. Use Cases

Use cases are a widely used and highly regarded format for capturing requirements. Before writing functional requirement use cases can help you to understand the requirements in the way user expect. Following table presents you not only the template to write use case(s) as well as guides you to write each section with example.

Use Case ID:	Enter a unique numeric identifier for the Use Case. e.g. UC-1.2.1
Use Case Name:	Enter a short name for the Use Case using an active verb phrase. e.g. Withdraw Cash
Actors:	[An actor is a person or other entity external to the software system being specified who interacts with the system and performs use cases to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product. Name the actor that will be initiating this use case (primary) and any other actors who will participate in completing the use case (secondary).]
Description:	[Provide a brief description of the reason for and outcome of this use case.]
Trigger:	[Identify the event that initiates the use case. This could be an external business event or system event that causes the use case to begin, or it could be the first step in the normal flow.]
Preconditions:	[List any activities that must take place, or any conditions that must be true, before the use case can be started. Number each pre-condition. e.g. 1. Customer has active deposit account with ATM privileges 2. Customer has an activated ATM card.]
Postconditions:	[Describe the state of the system at the conclusion of the use case execution. Should include both <i>minimal guarantees</i> (what must happen even if the actor's goal is not achieved) and the <i>success guarantees</i> (what happens when the actor's goal is achieved. Number each post-condition. e.g. 1. Customer receives cash 2. Customer account balance is reduced by the amount of the withdrawal and transaction fees]
Normal Flow:	[Provide a detailed description of the user actions and system responses that will take place during execution of the use case under normal, expected conditions. This dialog sequence will ultimately lead to accomplishing the goal stated in the use case name and]

	<p>description.</p> <ol style="list-style-type: none"> 1. Customer inserts ATM card 2. Customer enters PIN 3. System prompts customer to enter language performance English or Spanish 4. System validates if customer is in the bank network 5. System prompts user to select transaction type 6. Customer selects Withdrawal From Checking 7. System prompts user to enter withdrawal amount 8. ... 9. System ejects ATM card]
Alternative Flows: [Alternative Flow 1 – Not in Network]	<p>[Document legitimate branches from the main flow to handle special conditions (also known as extensions). For each alternative flow reference the branching step number of the normal flow and the condition which must be true in order for this extension to be executed. e.g. Alternative flows in the <i>Withdraw Cash</i> transaction:</p> <p>4a. In step 4 of the normal flow, if the customer is not in the bank network</p> <ol style="list-style-type: none"> 1. System will prompt customer to accept network fee 2. Customer accepts 3. Use Case resumes on step 5 <p>4b. In step 4 of the normal flow, if the customer is not in the bank network</p> <ol style="list-style-type: none"> 1. System will prompt customer to accept network fee 2. Customer declines 3. Transaction is terminated 4. Use Case resumes on step 9 of normal flow <p>Note: Insert a new row for each distinctive alternative flow.]</p>
Exceptions:	<p>[Describe any anticipated error conditions that could occur during execution of the use case, and define how the system is to respond to those conditions.</p> <p>e.g. Exceptions to the Withdraw Case transaction</p> <p>2a. In step 2 of the normal flow, if the customer enters an invalid PIN</p> <ol style="list-style-type: none"> 1. Transaction is disapproved 2. Message to customer to re-enter PIN 3. Customer enters correct PIN 4. Use Case resumes on step 3 of normal flow]
Includes:	<p>[List any other use cases that are included (“called”) by this use case. Common functionality that appears in multiple use cases can be split out into a separate use case that is included by the ones that need that</p>

	common functionality. e.g. steps 1-4 in the normal flow would be required for all types of ATM transactions- a Use Case could be written for these steps and “included” in all ATM Use Cases.]
Special Requirements:	[Identify any additional requirements, such as nonfunctional requirements, for the use case that may need to be addressed during design or implementation. These may include performance requirements or other quality attributes.]
Assumptions:	[List any assumptions that were made in the analysis that led to accepting this use case into the product description and writing the use case description. e.g. For the <i>Withdraw Cash</i> Use Case, an assumption could be: The Bank Customer understands either English or Spanish language.]
Notes and Issues:	[List any additional comments about this use case or any remaining open issues or TBDs (To Be Determined) that must be resolved. e.g. 1. What is the maximum size of the PIN that a user can have?]

3.2. Functional Requirements

Many people have a hard time finding their way in a new building. We considered this situation a useful test case for a location-based service provided through our mobile augmented reality system. The system should guide a user on the way through the building by showing him directions.

3.3. Non-Functional Requirements

Many people have a hard time finding their way in a new building. We considered this situation a useful test case for

Chapter 4: Design and Architecture

This chapter will discuss the design and architecture of your system.

4.1. System Architecture

Explain and justify the choice of system architecture for your project.

4.2. System Design

As system design varies from system to system, therefore you are required to explore which design pattern is suitable for your system. For guidelines an IEEE Recommended Practice for Software Design Descriptions (section 5 and 6) is provided with this template.

Chapter 5: Implementation

This chapter will discuss implementation details supported by UML diagrams (if applicable). You will not put your source code here. Any of the following sections may be included based on your project.

5.1. Component Diagram

Present and explain component diagrams of your project.

5.2. Network and Protocol Choice

It goes here.

5.3. Choice of Object Middleware

RMI vs. CORBA vs. DCOM etc.

5.4. User Interface

Details about user interface. (**must enter user interfaces here**)

Chapter 6: Testing and Evaluation

This chapter may include the following sections.

6.1. Verification

Verification section.

6.2. Validation

Validation section.

6.3. Usability Testing

Usability testing section.

6.4. Module / Unit Testing

Unit testing.

6.5. Integration Testing

Integration testing.

6.6. System Testing

System testing.

6.7. Acceptance Testing

Acceptance testing.

6.8. Stress Testing

Stress testing.

6.9. Hardware Configuration for Testing

Hardware configuration.

6.10. Evaluation

Evaluation section.

6.11. Deployment

Evaluation section.

6.12. Maintenance

Evaluation section.

Chapter 7: Conclusion and Future Work

This chapter concludes the project and highlights future work.

7.1. Conclusion

Conclusion section.

7.2. Future Work

Future work section.

References

References to any book, journal paper or website should properly be acknowledged. Please consistently follow the style. The following are few examples of different resources i.e. journal article, book, and website.

- 1 Lyda M.S. Lau, Jayne Curson, Richard Drew, Peter Dew and Christine Leigh, (1999), Use Of VSP Resource Rooms to Support Group Work in a Learning Environment, ACM 99, pp-2. (Journal paper example)
- 2 Hideyuki Nakanishi, Chikara Yoshida, Toshikazu Nishimura and Tsuru Ishada, (1996), FreeWalk: Supporting Casual Meetings in a Network, pp 308-314 (paper on web)
<http://www.acm.org/pubs/articles/proceedings/cscw/240080/p308-nakanishi.pdf>
- 3 Ali Behforooz & Frederick J. Hudson, (1996), Software Engineering Fundamentals, Oxford University Press. Chapter 8, pp255-235. (book reference example)
- 4 Page Author, Page Title, <http://www.bt.com/bttj/archive.htm>, Last date accessed. (web site)