<Project Name>

Software Architecture Document

Version <1.0>

[The texts in blue are guidance for filling in the information for each section. Remove everything in blue when writing the document.

This template is a simplified version of the Software Architecture Document from the RUP model. ]

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

2. Architectural Goals and Constraints 4

3. Use-Case Model 4

4. Logical View 4

4.1 Component: abc 4

5. Deployment 4

6. Implementation View 4

Software Architecture Document

# Introduction

[The introduction of the **Software Architecture Document** provides an overview of the entire **Software Architecture Document**. It includes the purpose, scope, definitions, acronyms, abbreviations, references, and overview of the **Software Architecture Document**.]

# Architectural Goals and Constraints

## Security

Central security features are handled by the institute officials. They will be given full access features both in the application and database levels. Creating user accounts for staff and warehouse staff are done by the admin staff. After the staff or warehouse staff are added to the system, they will be provided with a default password, which can be changed by the user. All the passwords are encrypted in order to ensure higher security. Responses and commits made by the staff and warehouse staff can be seen only by the admin.

## Persistence

All the data will be saved in the central server. This is a rational database that implements the Third normal form (MySQL). In order to maintain ACID (Atomicity, Confidentially, Integrity, Durability) some measures have been taken such as encrypting passwords, using transactions for all database commits,…

## Reliability / Availability

The system will be subjected to several tesing operations (Unit testing, integration testing, system tesing) before being deployed in order to make sure that the system is reliable. The MySQL database server can respond to many number of clients at a given moment without losing consistency and data integrity.

## Portability and reuse

Even though Management sales SW is as a complete and standalone product, it is possible to extend the produce to integrate in a system. In order to maintain reusability, all the wfunctionalities are very well structured and layered. Best practices of rup are followed throughout the project and the project strictly adheres to OOP standards

## Development tools

The project incoporates many development tools.

Programming:

Database:

Diagram: Draw.IO, PowerDesigner,…

Database connection: MYSQL connector

# Use-Case Model

[This section includes the use case diagrams that are already modeled and presented in the use-case specification document.]

# Logical View

[This section describes the architecture with components and relationships among them. One or several diagrams showing the architecture are provided here. For each component, describe its responsibilities and/or services that are provided for other components. Each relationship should also indicate the means of communication, such as HTTP, HTTPS, Socket, LAN, Internet, etc.

The detail of each component is provided using the subsection below.]

## Component: abc

[This section provides details for the component named “abc”. You need to include class diagrams for this component and explain key classes.

For each component, create a section like this.]

# Deployment

[Leave this section blank for PA3.]

# Implementation View

[Leave this section blank for PA3.]