**Security Management System**

**Koushik Majumder**

Index

Contents

[Introduction: 5](#_Toc351927541)

[Objective 7](#_Toc351927542)

[System Analysis 8](#_Toc351927543)

[Identification of Need: 8](#_Toc351927544)

[Security Management System GUI: 9](#_Toc351927545)

[Visitor Register: 9](#_Toc351927546)

[Member Registration: 10](#_Toc351927547)

[Member Database Browser: 10](#_Toc351927548)

[Employee Management: 10](#_Toc351927549)

[Security Training: 10](#_Toc351927550)

[Video Surveillance Window: 11](#_Toc351927551)

[Alarm System: 11](#_Toc351927552)

[Emergency Handler: 11](#_Toc351927553)

[Security Management System Engine: 11](#_Toc351927554)

[Engine Controller: 12](#_Toc351927555)

[GUI Interactor: 12](#_Toc351927556)

[Database Controller: 12](#_Toc351927557)

[Photo & Video Surveillance Controller : 12](#_Toc351927558)

[Visitor Controller: 12](#_Toc351927559)

[Member Controller : 12](#_Toc351927560)

[Employee Controller : 12](#_Toc351927561)

[Alarm Controller : 13](#_Toc351927562)

[Data Entry Controller : 13](#_Toc351927563)

[Training Controller : 13](#_Toc351927564)

[Search Engine : 14](#_Toc351927565)

[Emergency Controller : 14](#_Toc351927566)

[Security Management System Database: 14](#_Toc351927567)

[Preliminary Investigation: 14](#_Toc351927568)

[Feasibility Study: 14](#_Toc351927569)

[Project Planning & Scheduling: 15](#_Toc351927570)

[Gantt chart 15](#_Toc351927571)

[Tracking Gantt 16](#_Toc351927572)

[Pert Chart 17](#_Toc351927573)

[Software requirement specifications (SRS): 18](#_Toc351927574)

[Hardware Requirement 19](#_Toc351927575)

[Software Requirement 19](#_Toc351927576)

[Hardware Used 20](#_Toc351927577)

[Software Used 20](#_Toc351927578)

[Functional Requirement 20](#_Toc351927579)

[Add Visitor 21](#_Toc351927580)

[Add Member 21](#_Toc351927581)

[Search member 21](#_Toc351927582)

[Make an appointment with a member 22](#_Toc351927583)

[Create an Preference 22](#_Toc351927584)

[Distress calling 22](#_Toc351927585)

[Emergency action 23](#_Toc351927586)

[Non-functional Requirements 23](#_Toc351927587)

[Technical Specification 24](#_Toc351927588)

[Software Engineering Paradigm applied 24](#_Toc351927589)

[Data models 24](#_Toc351927590)

[Context Diagram 24](#_Toc351927591)

[0-Level DFD 25](#_Toc351927592)

[1-Level DFD 26](#_Toc351927593)

[2-Level DFD 29](#_Toc351927594)

[Sequence diagrams 29](#_Toc351927595)

[Entity Relationship Model 29](#_Toc351927596)

[E-R Diagram 31](#_Toc351927597)

[Class Diagrams 32](#_Toc351927598)

[Activity Diagrams 33](#_Toc351927599)

[System Design 33](#_Toc351927600)

[Modularisation details 33](#_Toc351927601)

[DESIGN DOCUMENT: 33](#_Toc351927602)

[Manu Bar 34](#_Toc351927603)

[Tool Bar 34](#_Toc351927604)

[Expander Control 34](#_Toc351927605)

[Tree View 34](#_Toc351927606)

[Web Browser 35](#_Toc351927607)

[Text Block 35](#_Toc351927608)

[Media Element 35](#_Toc351927609)

[Image Control 36](#_Toc351927610)

[Ink Canvas Control 36](#_Toc351927611)

[Status Bar Class 36](#_Toc351927612)

[Database & Table Details 36](#_Toc351927613)

[Complete Structure 37](#_Toc351927614)

[Module Description 37](#_Toc351927615)

[User Interface Design 37](#_Toc351927616)

[Test Cases (Unit Test Cases and System Test Cases) 37](#_Toc351927617)

[Unit Test Cases: 37](#_Toc351927618)

[System Test cases: 38](#_Toc351927619)

[Coding 38](#_Toc351927620)

[Complete Project Coding 38](#_Toc351927621)

[Comments and Description of Coding segments 42](#_Toc351927622)

[Standardization of the coding 43](#_Toc351927623)

[Code Efficiency 43](#_Toc351927624)

[Error handling 43](#_Toc351927625)

[Parameters calling/passing 43](#_Toc351927626)

[Validation checks 43](#_Toc351927627)

[Testing 44](#_Toc351927628)

[Testing techniques and Testing strategies used 44](#_Toc351927629)

[Database & Data Integrity Testing 44](#_Toc351927630)

[Functional Testing: 44](#_Toc351927631)

[Regression Testing: 45](#_Toc351927632)

[User Interface Testing: 46](#_Toc351927633)

[Performance Profiling: 46](#_Toc351927634)

[Load Testing: 47](#_Toc351927635)

[Stress Testing: 48](#_Toc351927636)

[Volume Testing: 48](#_Toc351927637)

[Security & Access Control Testing: 48](#_Toc351927638)

[Failover & Recovery Testing: 49](#_Toc351927639)

[Configuration Testing: 49](#_Toc351927640)

[Installation/Deploy & Back out Testing: 49](#_Toc351927641)

[Post Production Testing: 49](#_Toc351927642)

[Unit Testing: 50](#_Toc351927643)

[Smoke Testing: 50](#_Toc351927644)

[Data Migration Testing: 50](#_Toc351927645)

[Testing Plan used 51](#_Toc351927646)

[Test reports for Unit Test Cases and System Test Cases 51](#_Toc351927647)

[Debugging and Code improvement: 51](#_Toc351927648)

[System Security measures: 52](#_Toc351927649)

[Database/data security: 52](#_Toc351927650)

[Creation of User profiles and access rights 52](#_Toc351927651)

[Estimation 53](#_Toc351927652)

[Cost Estimation of the Project along with Cost Estimation Model 54](#_Toc351927653)

[Estimation of development effort 54](#_Toc351927654)

[Estimation of development time 54](#_Toc351927655)

[Data Structure 55](#_Toc351927656)

[Reports (sample layouts should be placed) 58](#_Toc351927657)

[Future scope and further enhancement of the Project 58](#_Toc351927658)

[Bibliography 58](#_Toc351927659)

[Appendices (if any) 59](#_Toc351927660)

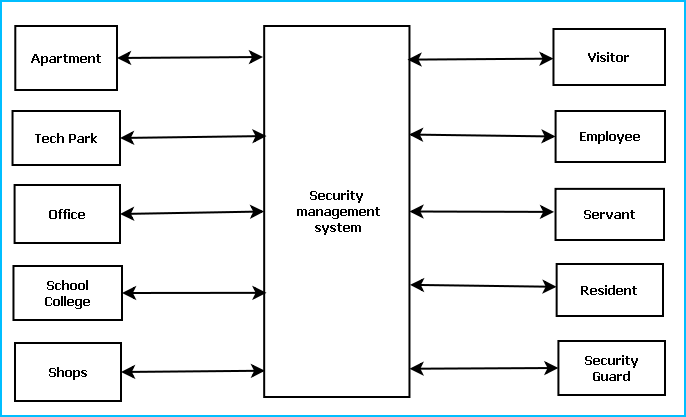
[Glossary. 59](#_Toc351927661)

# Introduction:

Security management system will provide a software solution for enhancing security measures in housing complexes, Apartments, companies and other organizations. Security is a foremost concern in every walk of our life. Every day lots of new and unknown people visits the organizations. We want to restrict some specific people from entering the premises and track the people entering and leaving the premises.

Now in most of the places paper book registers are used for tracking visitors. But it is not an efficient system. There are no proper ways of validating the data provided by visitor and searching a visitor’s entry history is a cumbersome job. Security management system will provide better ways validating visitor data and searching visitor’s previous entry history.

Security management system will track the security guards and other servants who enter premises for various reasons. Following diagram displays the users and coverage of the security management system.



To ensure better security, Security Management System will provide the following features:

* Residents or employee information database
* Video Surveillance
* Photo identity card generation
* GPS Tracker
* In time and out time registry, Visitor Traffic and History
* Visitor tracking and contact person confirmation
* Distress calling system
* Emergency action System
* Do Not Disturb criteria
* Visitor time out alert
* Security Guard Management
* Security training & Tutorial

## Objective

The objectives of this project are given below:

* Security management system will provide a software solution for enhancing security measures in various organizations.
* System will facilitate to restrict some specific people from entering the premises and track the people entering and leaving the premises.
* It will enable inhabitants to implement a automatic security system by which a centralized system can keep track with trace passers.
* Security management system will track the security guards and other servants who enter premises for various reasons.
* Security management system will provide better ways for validating visitor data and searching visitor’s previous entry history which will replace cumbersome paper work with efficient computerized datasheets.
* By accomplishing this project I could learn new technologies like .NET, C#, DirectX, XML and I am able to be involved with the complete software development lifecycle.

# System Analysis

## Identification of Need:

Need of security is the basic necessity of any individual. The feeling that you are safe and everything around you is all right is imperative for a peaceful living. But in this unsafe world, when crime, terror and threats are on their peak, how can one attain that sense of security? Here, security management system provides us with a solution and for this reason more and more people are installing them in order to stay safe and secure. Security Management System will enhance the security of apartments, offices, schools and business centres by providing an efficient system for tracking incoming & outgoing traffic. Now in most of the places paper book registers are used for tracking visitors. But it is not an efficient system. There are no proper ways of validating the data provided by visitor and searching a visitor’s entry history is a cumbersome job. Security management system will provide better ways validating visitor data and searching visitor’s previous entry history.Security management system will track the security guards and other servants who enter premises for various reasons.

## Preliminary Investigation:

Security is a foremost concern in every walk of our life. Every day lots of new and unknown people visits the organizations. We want to restrict some specific people from entering the premises and track the people entering and leaving the premises.

Now in most of the places paper book registers are used for tracking visitors. But it is not an efficient system. There are no proper ways of validating the data provided by visitor and searching a visitor’s entry history is a cumbersome job. Security management system will provide better ways validating visitor data and searching visitor’s previous entry history.

Security management system will track the security guards and other servants who enter premises for various reasons.

## Feasibility Study:

We have analysed all Technical, Economic, Legal, Operational, and Scheduling feasibility of Security management system.

### Technology and system feasibility

Our team has the technical expertise to handle completion of the project. We are using recommended human and economic factor, technologies to implement the solution properly.It has been already mentioned that ‘Security Management System’ is a desktop cum mobile based project. A lots of such types of projects have already been made or been running to make every day. So it is not even technically impossible rather difficult to build such a software. The technical software knowledge that is required is C#, Mysql,Nokia SDK,Nokia web tool and XML anyone who has a basic knowledge of computer software and has a good sense of design can almost be an expert in handling these software. As the project will precede one can easily make himself informative about ASP to do the project. So not very much technical capability is required by the project. We see that the project is technically feasible and worth to do because of the reasons specified above.

### Legal feasibility

The proposed system does not conflict with legal requirements, e.g. a data processing system complies with the local Data Protection Acts.

### Operational feasibility

We have proposed well defined system logic to solve the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. To know whether the proposed system is acceptable to the end user, they were subjected to a mini prototype. The users were asked to respond as to how they feel about the system. To a large extent the users were satisfied with the prototype, which I have shown to them, minor modifications were also done to closely match with the users’ requirement.To implement the user requirements in a software system is the true goal of a systems analyst. I rigorously followed the requirements of the user, what they want from the system and how it should help them. Prototyping results show that the proposed system is acceptable to the user.

### Schedule feasibility

Security management system will be completed as per the project planning. We have generated project planning and scheduling in terms of Gantt and pert chart.

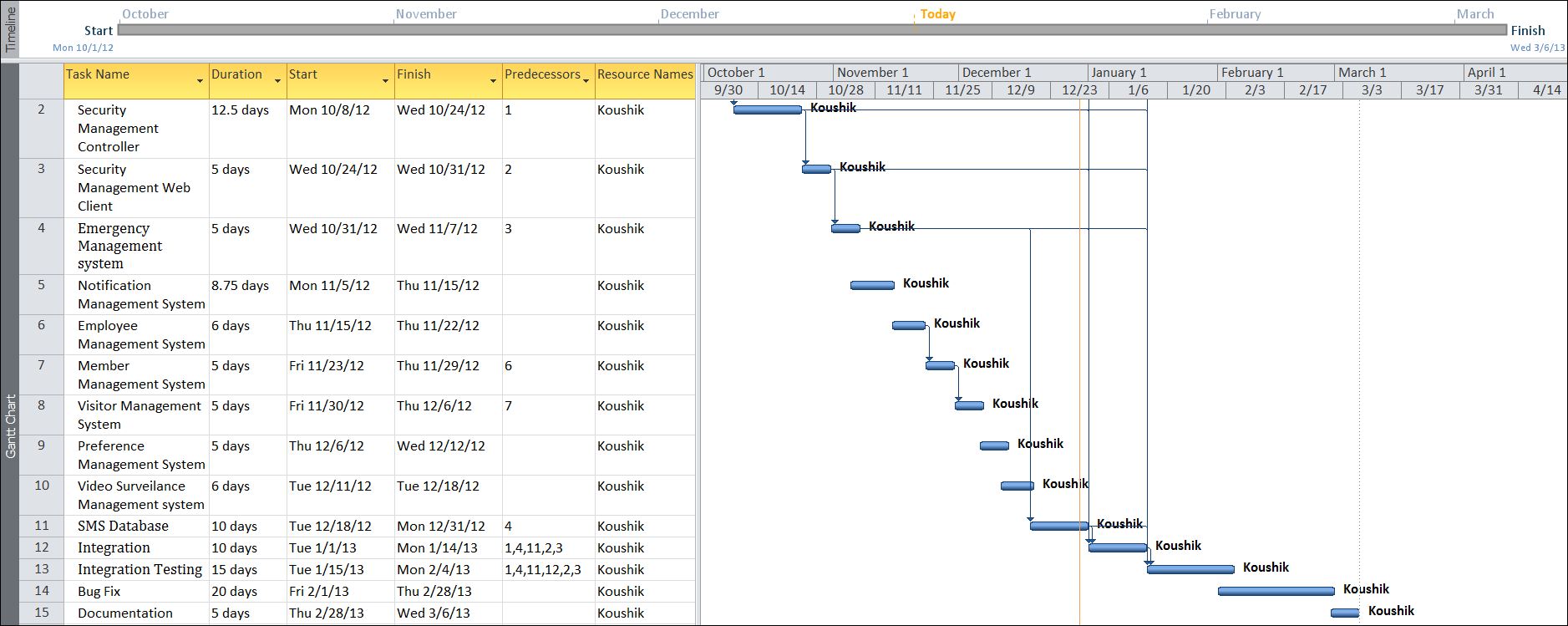
## Project Planning & Scheduling:

Planning is an on-going process that must consider what the whole organization is going to do and the role of computer-based information systems in achieving the organization’s goals. A plan determines the most important tasks to be done and set priorities to them. While carrying out my project I had to plan very minutely so that every step from analysis to implementation receives its due importance within time limits. I have to use some planning charts like GANTT CHART (Bar chart) and PERT CHART.

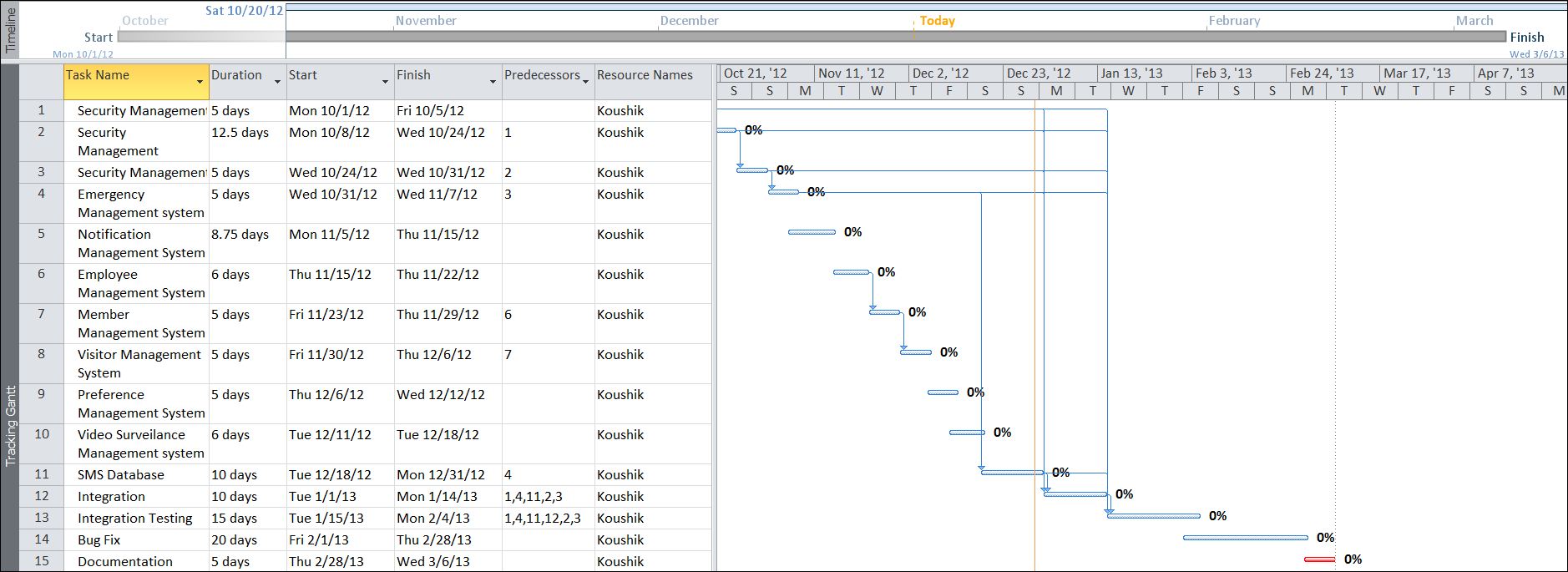
### Gantt chart

A GANTT chart is a Bar chart, which is perhaps the simplest form of formal project management. The bar chart is used almost exclusively for scheduling purposes and therefore controls only the time dimension of projects.

GANTT Charts (developed by Henry L. Gantt) are a project control technique that can be used for several purposes, including scheduling, budgeting and resource planning. A GANTT chart is bar chart, with each bar representing an activity. The bars are drawn against a time line. The length of each bar is proportional to the length of time planned for the activity.

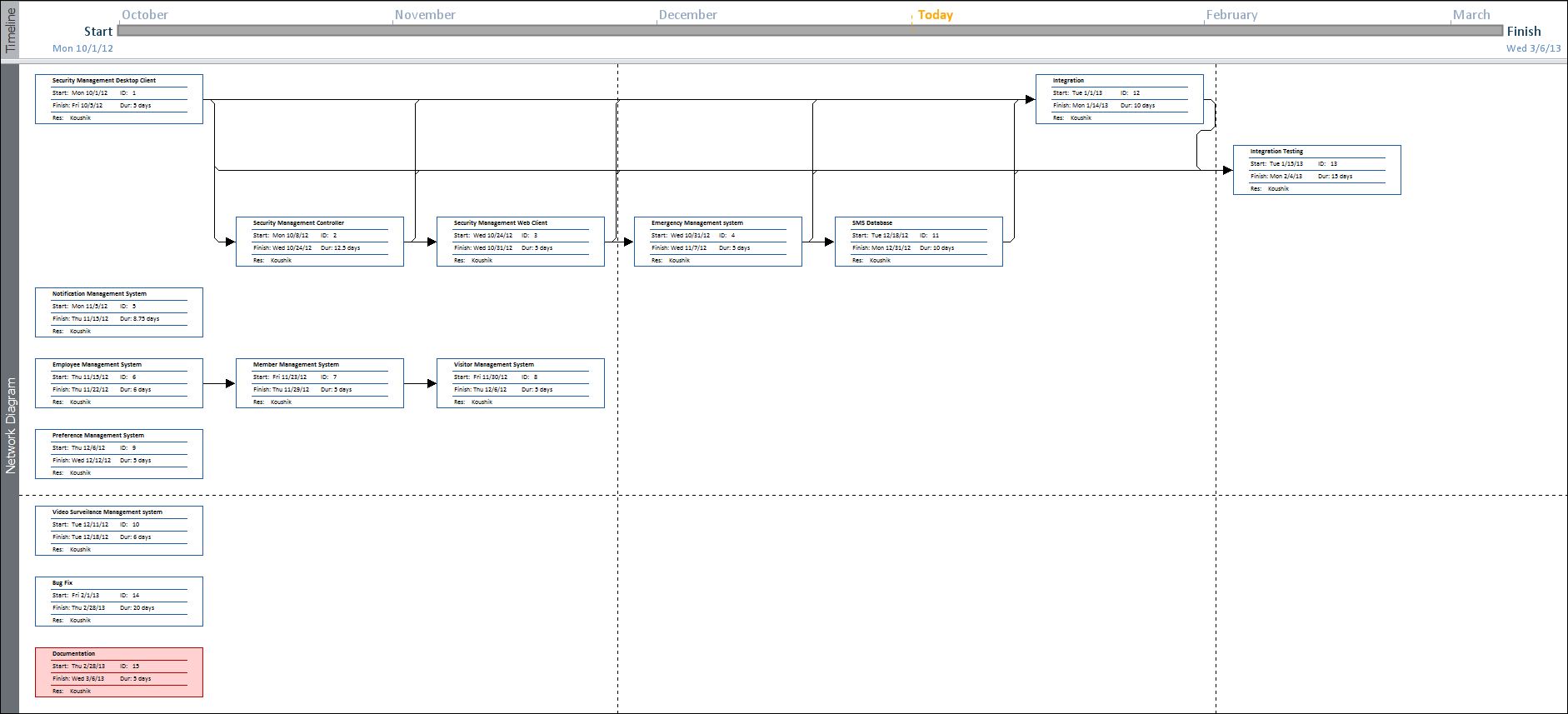
******

### Tracking Gantt

******

### Pert Chart

PERT chart (program evaluation review technique) can be both a Cost and a Time Management system; PERT is organised by events and activities or tasks. PERT has several advantages over bar charts and is likely to be used with projects that are more complex. One advantage of PERT is that it is a scheduling device that also shows graphically which tasks must be completed before others are begun. In addition, by displaying the various task paths, PERT enables the calculation of a critical path. Each path consists of combinations of tasks, which must be completed. The time and cost associated with each task along a path are calculated and the path that requires the greatest amount of elapsed time is the critical path. Calculation of critical path enables project managers to monitor this series of tasks more closely that others and to shift resources to it if it begins to fall behind schedule. PERT controls time and cost during the project and facilitate finding the right balance between completing it within the budget. PERT recognise that projects are complex, that some tasks must be completed before others can be started, and that the appropriate way to manage a project is to define and control each task. Because projects often fall behind schedule, PERT is designed to facilitate getting a project back on schedule. PERT is based in part on the premise that subjective estimates of the total completion time for a project are usually inferior to the sum of subjective estimates for each task. As with GANTT charts, to build a PERT chart for a project, one must first list all the activities required for completion of the project and estimate how long each will take. Then one must determine the dependence of the activities on each other.



## Software requirement specifications (SRS):

### Functional Requirement

#### Add Visitor

Introduction:

Register a new visitor to a particular member.

Input:

Relevant visitor data like name, address, contact number, blood group, date of birth.

Processing:

Admin will enter the data in the SMS and create a new visitor entry.

Output:

The SMS will allow a visitor I for visit the member and generate a temporary identity card.

#### Add Member

Introduction:

Anew resident can register as a member of the security system.

Input:

Member will enter data like name, address, job profile, emergency contact details.

Processing:

The SMS will check for availability and create the registration confirmation.

Output:

The **SMS** will generate a permanent ID for future reference.

#### Search member

Introduction:

Anyone can search for member.

Input:

He will enter data like name, job profile.

Processing:

The SMS will search for the requirement.

Output:

The SMS will display the search result.

#### Make an appointment with a member

Introduction:

A visitor can ask for an appointment with an existing member and member will confirm the appointment with an admin approval.

Input:

Admin will enter the name, time and duration of appointment, address and details of visitor etc. data in the SMS.

Processing:

The SMS will create a new appointment with a member approval and entry details to the scheduler.

Output:

The SMS will generate the appointment approval to the visitor and appointment details.

#### Create an Preference

Introduction:

A preference is meant for alerting security guards on specific emergency situations or to notify members about some information. It also restrict availability of a member.

Input:

Admin will enter the Event name, venue, requirements, availability of a member and date etc. data in the SMS.

Processing:

The SMS will create a new event entry.

Output:

The SMS will generate preference ID and details.

#### Distress calling

Introduction:

SMS will facilitate distress calling functionality to the members when they are in critical situations.

Input:

Member will initiate a request for help.

Processing:

SMS will receive and accept the request and check the privilege of the member. It will search for the available resources for taking action

Output:

SMS will send the securities to help and resolve the issue.

#### Emergency action

Introduction:

SMS will facilitate emergency action functionality to the members when they are in problem.

Input:

Member will initiate a request for help.

Processing:

SMS will receive and accept the request and check the privilege of the member. It will search for the available resources for taking action

Output:

SMS will send the securities to help and resolve the issue.

### Non-functional Requirements

* The application will be **self-dependent** and no dependency on other parties required.
* There will be a digital **backup** and restore system.
* There will be more **opportunity** to extend the application in various type of device in future.
* The response time will be low and the system will **response** fast.
* It will be very **user friendly** and **usable** by any person with minimal computer knowledge.
* In terms of **security** unauthorized access will be denied and register user will be able to change as necessary.
* It will be **efficient** as it reduces manual labor and searching.
* **SMS** will have user manual and help **documents**.
* It is designed such a way that it can be **maintained** with minimal effort.

## Software Engineering Paradigm applied

We have followed agile version of Model Driven Development (MDD). As the name implies, AMDD is the agile version of Model Driven Development (MDD). MDD is an approach to software development where extensive models are created before source code is written. A primary example of MDD is the Object Management Group (OMG)’s Model Driven Architecture (MDA) standard. With MDD a serial approach to development is often taken, MDD is quite popular with traditionalists, although as the RUP/EUP shows it is possible to take an iterative approach with MDD. The difference with AMDD is that instead of creating extensive models before writing source code you instead create agile models which are just barely good enough that drive your overall development efforts. AMDD is a critical strategy for scaling agile software development beyond the small, co-located team approach that we saw during the first stage of agile adoption.



Figure 1The AMDD lifecycle: Modeling activities throughout the lifecycle of a project

Above Figure depicts a high-level lifecycle for AMDD for the release of a system. First, let’s start with how to read the diagram. Each box represents a development activity. The envisioning includes two main sub-activities, initial requirements envisioning and initial architecture envisioning. These are done during iteration 0, iteration being another term for cycle or sprint. “Iteration 0” is a common term for the first iteration before you start into development iterations, which are iterations one and beyond (for that release). The other activities – iteration modeling, model storming, reviews, and implementation – potentially occur during any iteration, including iteration 0. The time indicated in each box represents the length of an average session: perhaps you’ll model for a few minutes then code for several hours. I’ll discuss timing issues in more detail below..



Figure 2AMDD Through the Agile Development Lifecycle.

Above Figure depicts how the AMDD activities fit into the various iterations of the agile software development lifecycle.  It's simply another way to show that an agile project begins with some initial modelling and that modelling still occurs in each construction’s iteration.

## Data models

### Context Diagram

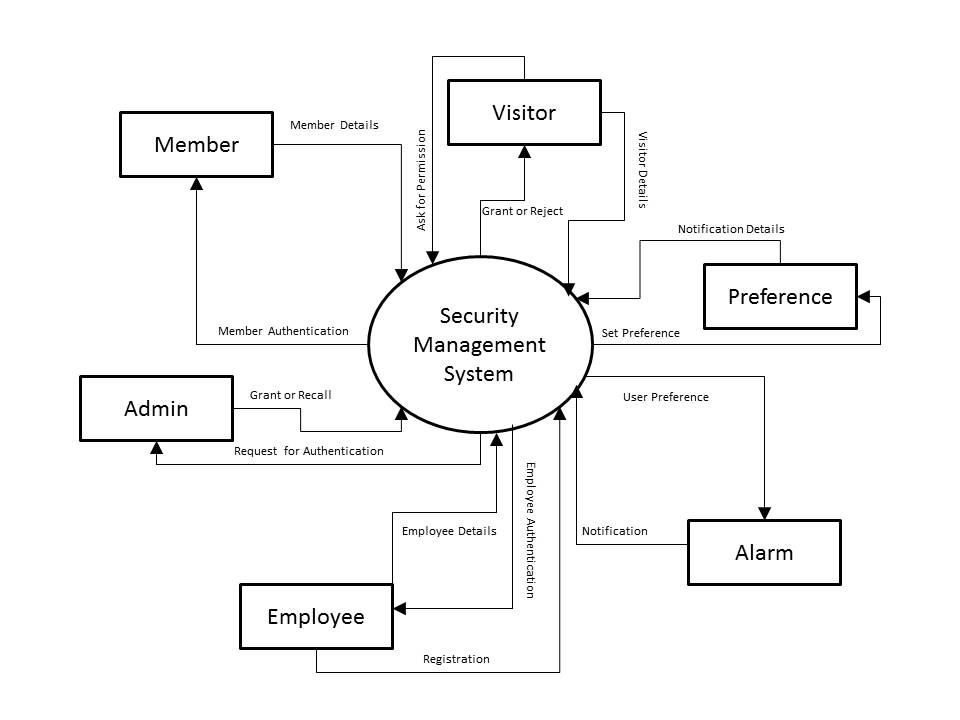


Figure 3: Context level diagram of SMS

### 0-Level DFD

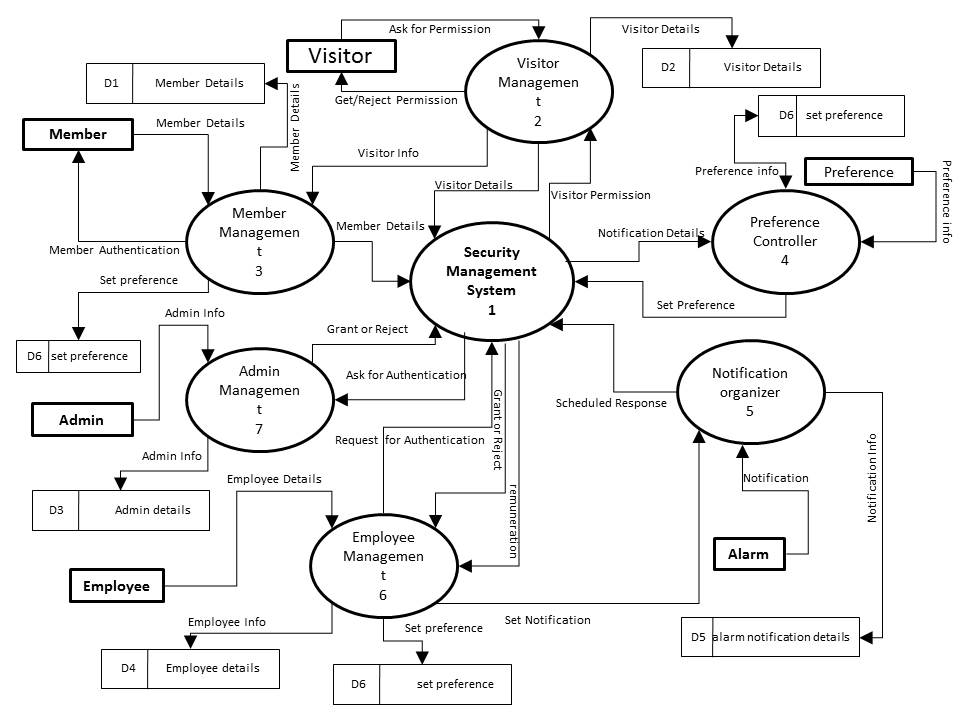


Figure 4: DFD level 0 of SMS

### 1-Level DFD

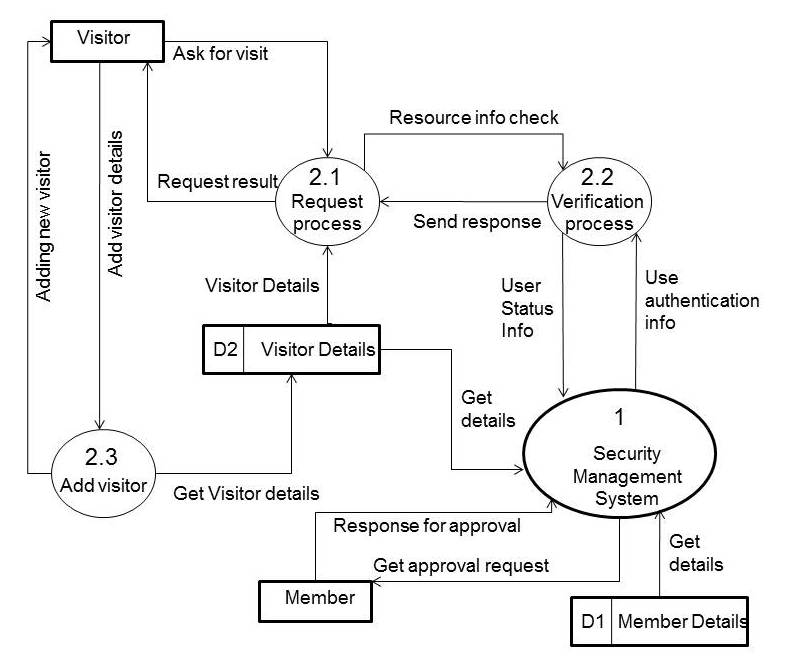
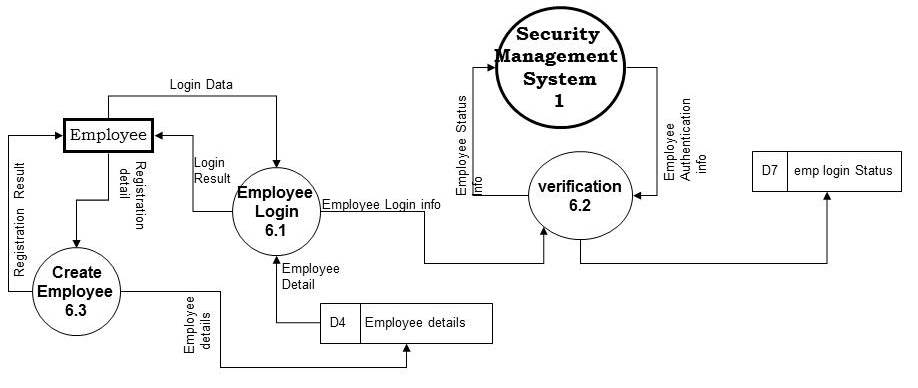


Figure 5: DFD level 1 of SMS



**Figure 6: DFD level 1 of SMS**

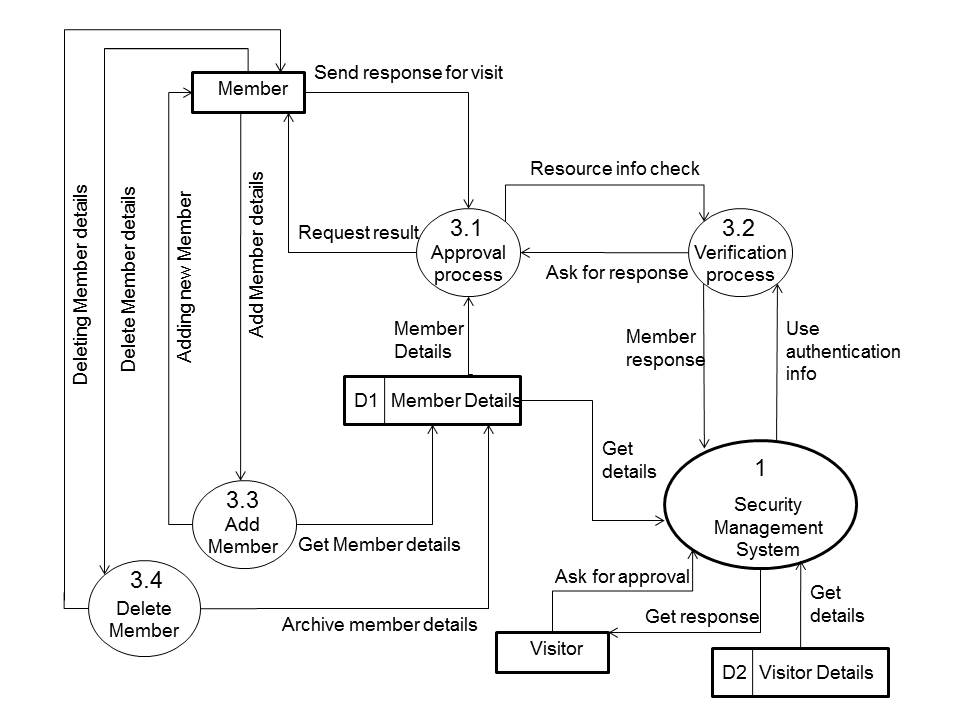


Figure 7: DFD level 1 of SMS

### 2-Level DFD

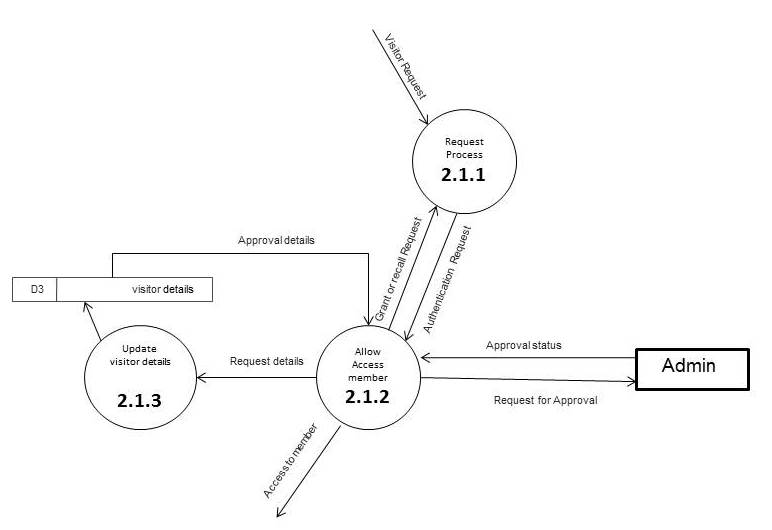
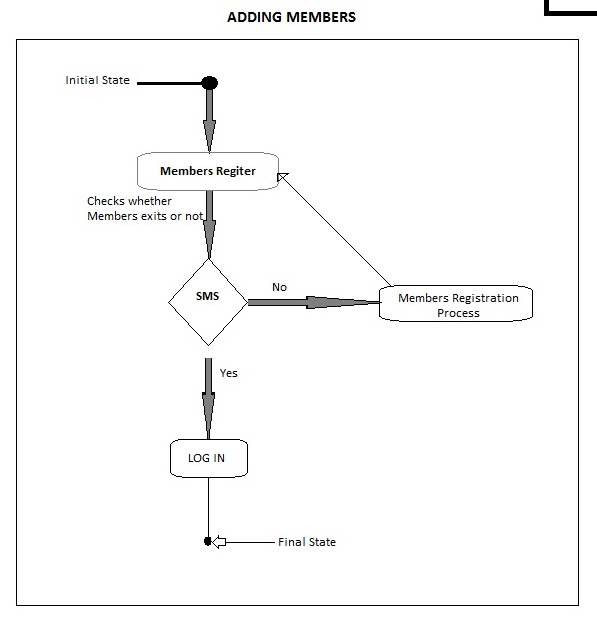
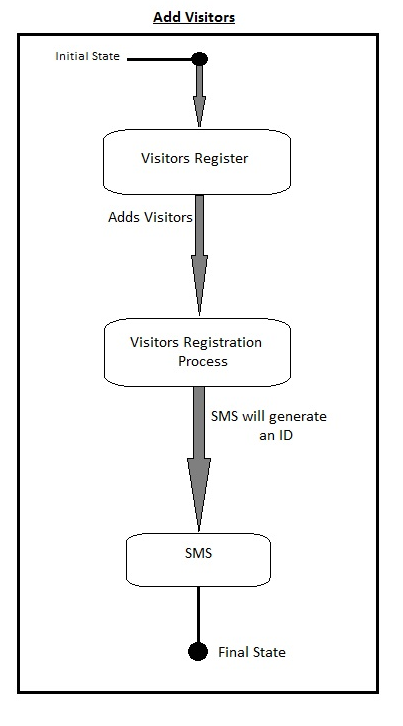
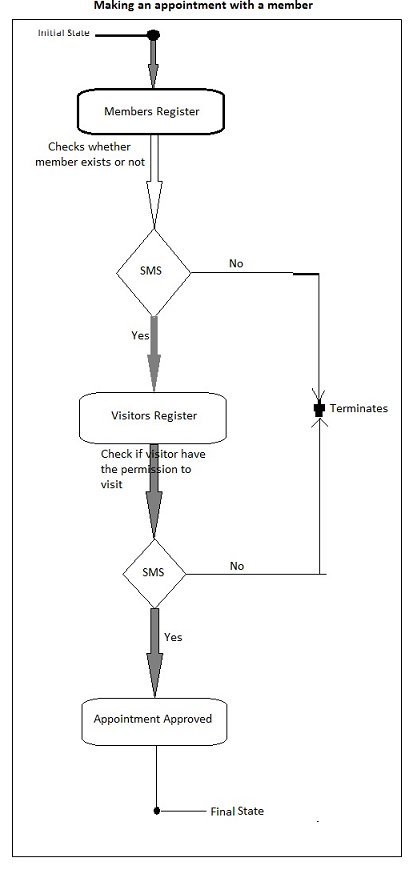


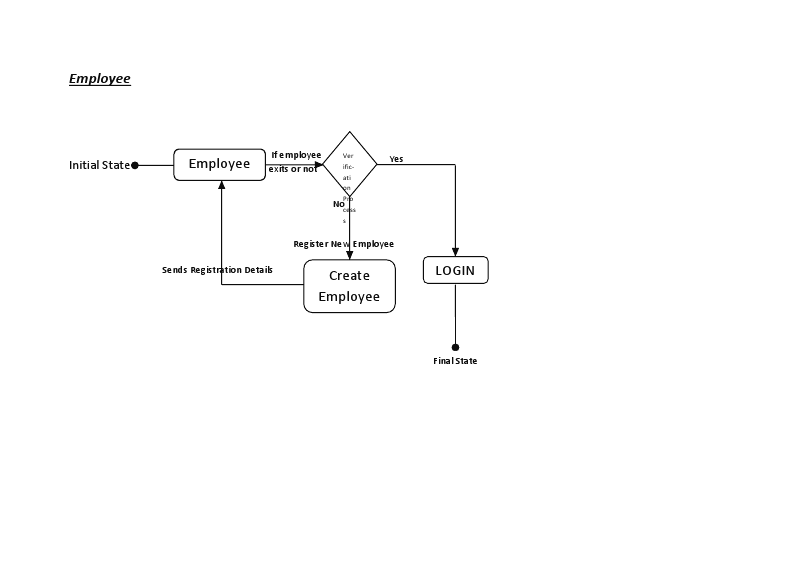
Figure 8: DFD level 2 of SMS

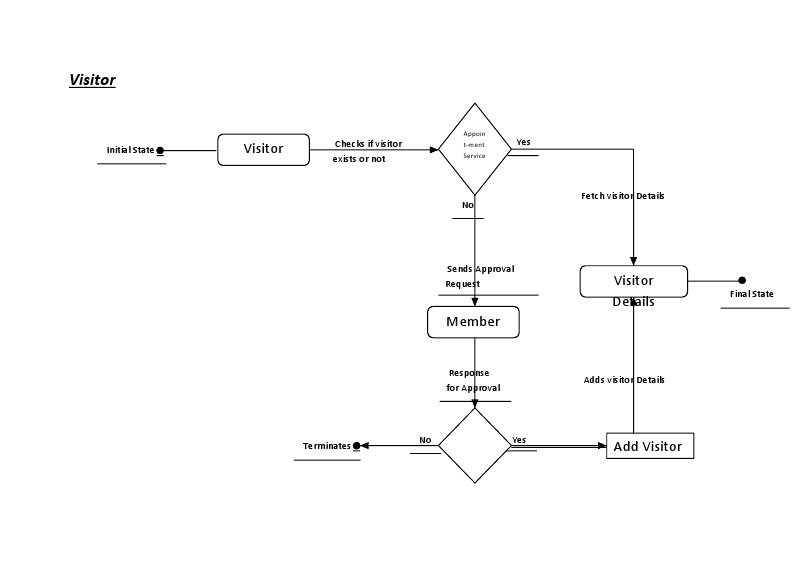
### Control Flow Diagram



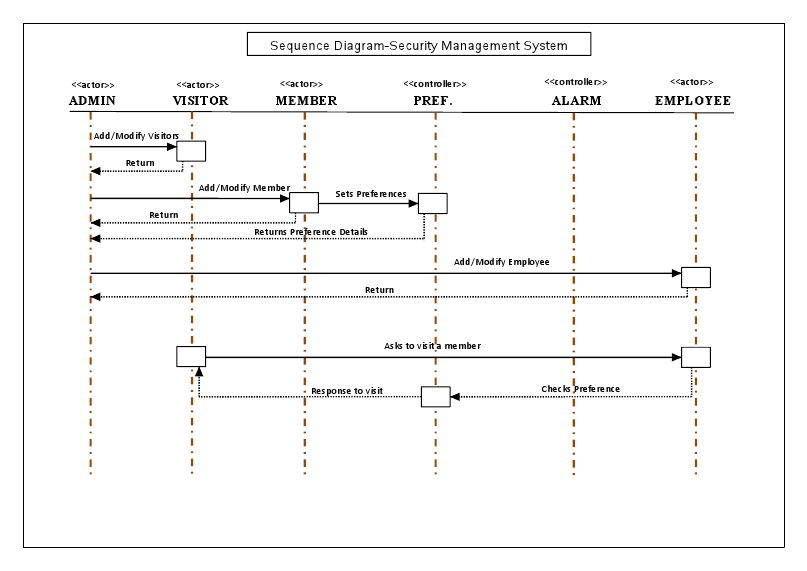








### Sequence diagrams



### Entity Relationship Model

We will design a RDBMS for File Management System. The entities and their attributes are listed below. Attributes in Bold letter is the unique key.

|  |  |
| --- | --- |
| **Entities** | **Attributes** |
| Visitor | **Visitor Id**, Name, Address , Contact Number |
| Visit Request | **Request Id,** type, purpose, time, duration, member id |
| Security Management System | **Organization Id** , Name, Address, Total member |
| Employee | **Employee Id**, Name, Admin Right |
| Member | **Member Id**, Name, address, contact number, emergency contact number |
| Member Preference | **Preference Id,** Type, Description |

**Relationship between Entities:**

Security Management System has Members🡪 1 : N

Security Management System has Visitors🡪 1 : N

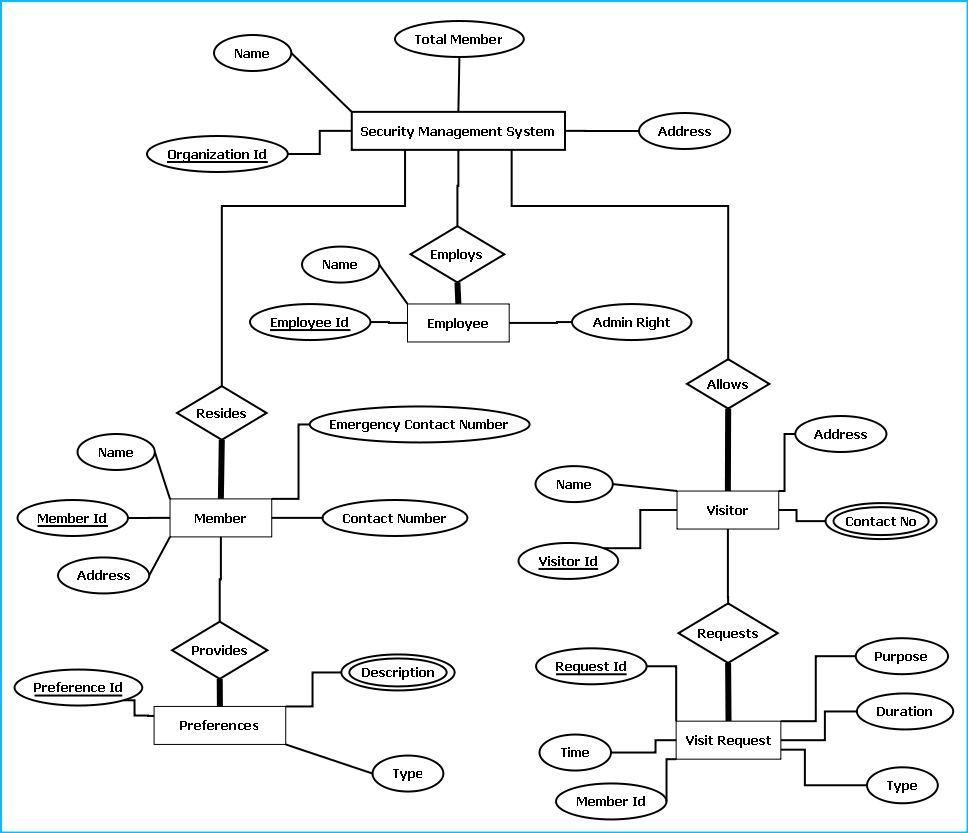
Security Management System has Employees🡪 1 : N

Visitor does Requests 🡪 1 : N

Security Management System serves Requests 🡪 1 : N

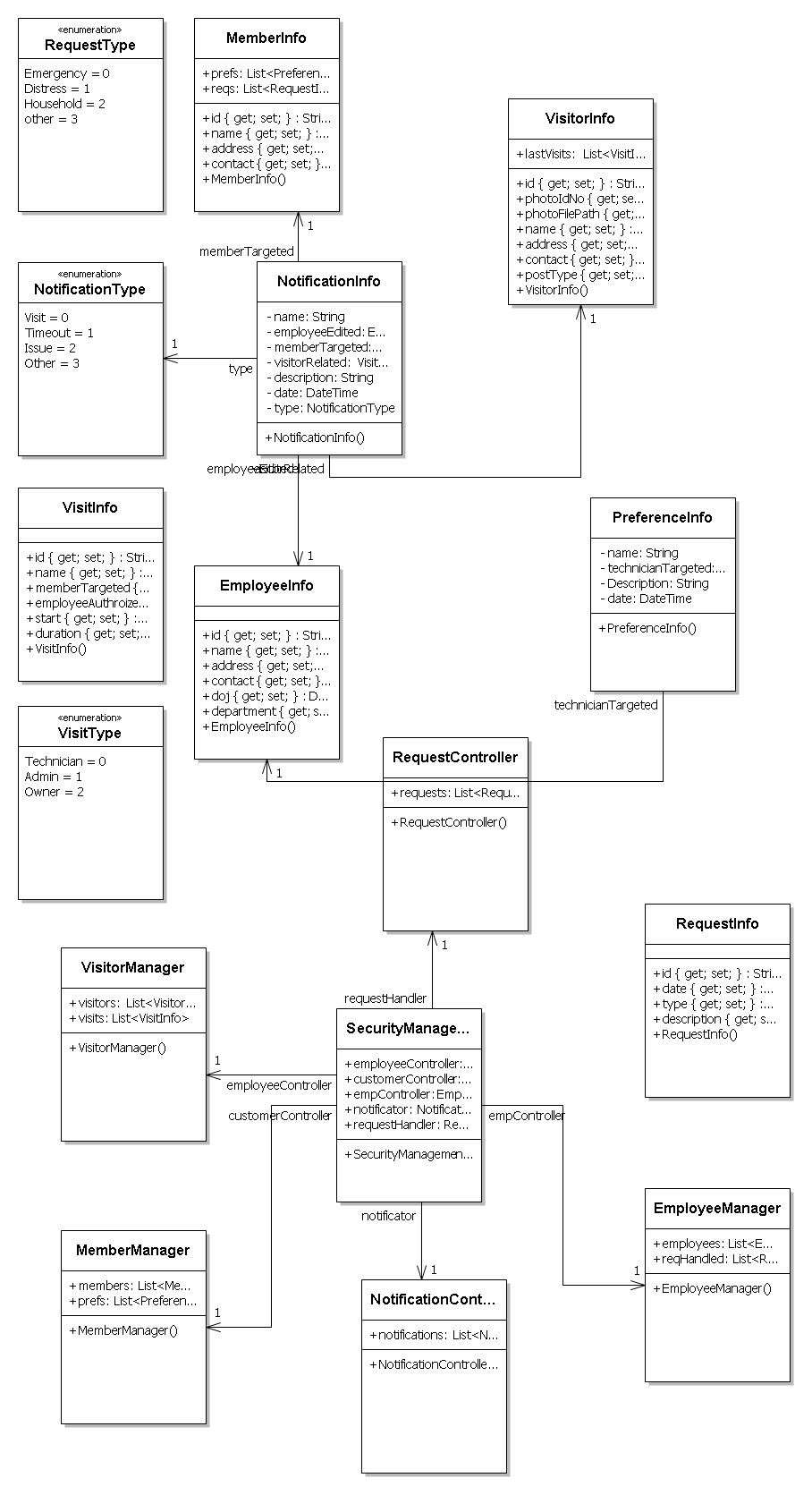
Members provides Preferences 🡪 M : N

### E-R Diagram



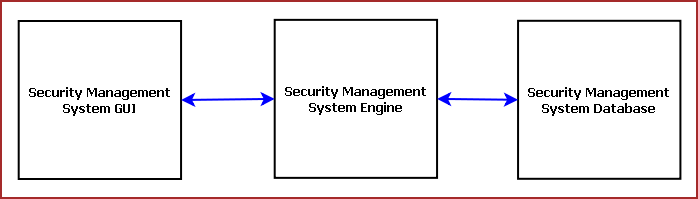
E-R Diagram of Security Management System

### Class Diagrams



# System Design

Security Management System will enhance the security of apartments, offices, schools and business centres by providing an efficient system for tracking incoming & outgoing traffic.



**Figure:** Components of Security Management System

Security Management System is divided into three main components. Such as:

* Security Management System GUI
* Security Management System Engine
* Security Management System Database

## Modularisation details

#### Security Management System GUI:

Security Management System GUI will display tutorial, navigator, online store interface. The main components of GUI are:

* Visitor Register
* Member Registration
* Member Database Browser
* Employee Management
* Security Training
* Video Surveillance Window
* Alarm System
* Emergency Handler

#### Visitor Register:

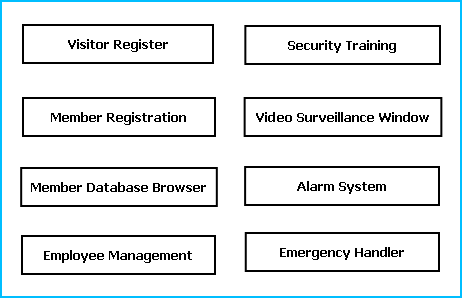
Visitor Register allows tracking visitors. It provides an interface to enter visitor details, purpose of visit, tentative time of stay, contact person. It will take the photograph of visitor and generate a photo ID card. The earlier visit histories can be browsed and it will suggest whether the visitor was blocked earlier in this organization.

#### Member Registration:

This module provides an interface for registering the members of the organization, residents of the apartment etc. This data helps validate the visitor provided inputs. Security management system will contact the member before allowing the access. If the visitor stays more than specified time limit then Security management system will enquire the reason

#### Member Database Browser:

Member Database Browser will provide information about the existing members of the organization and their contact information. It will help validating visitor data and handling emergency situations.



**Figure:** GUI Components

#### Employee Management:

This module will manage the employees responsible for security operations. It will keep track of employee attendance, duty hours, customer support quality and feedbacks.

#### Security Training:

This module is responsible for making the employees and members aware about security guidelines and actions to be taken on emergency situations. It will provide guideline booklet, presentation & video tutorials.

#### Video Surveillance Window:

Video Surveillance window displays the video of the area where the security cameras are installed. It can record the video for a specified time limit.

#### Alarm System:

Alarm system is meant for alerting security guards on specific emergency situations or to notify members about some information. All the major locations will have sound system installed and the roaming security guards also carry wireless devices.

#### Emergency Handler:

Emergency Handler takes appropriate actions on emergency situations. It will notify police, medical services or fire fighter depending on the emergency situations. It will can enable/ disable some services like lift service or power supply depending on the need.

#### Security Management System Engine:

Security Management Engine controls the overall system. It provides logical and tactical solutions for managing the whole system. The Security Management System is divided into 12 divided modular components. Such as:

* Engine Controller
* GUI Interactor
* Database Controller
* Photo & Video Surveillance Controller
* Visitor Controller
* Member Controller
* Employee Controller
* Alarm Controller
* Data Entry Controller
* Training Controller
* Search Engine
* Emergency Controller

#### Engine Controller:

This controls the overall interaction between all the backend modules. It schedules the priority of the actions in case of overlapping.

#### GUI Interactor:

It interacts with the GUI and polls GUI calls. It exposes APIs and events for GUI to use. GUI Interactor helps Engine to maintain wrapper around the Engine modules so that the GUI can be ported to any other framework without much changes in Engine code.

#### Database Controller:

It controls the database interactions. It forms query to fetch information from the database. It also sends data to be saved in database for future use.

#### Photo & Video Surveillance Controller :

This module processes the images and videos taken while surveillance and exposes APIs for viewing and Playing back. It also archives the video and images to save space.

#### Visitor Controller:

It keeps track of visitor information. It has an algorithm for maintaining Visitor history and rating the visitor’s trustworthiness. Visitor controller will trigger alarm if it senses any mismatch in visitor provided data.

#### Member Controller :

It saves information about the members of the organization. It validates the Visitor data and takes confirmation from the member to allow the visitor inside. Members can set DO NOT DISTURB mode, Preferred Visiting Time & Preferred visitors. It controls access to information of the members depending on the admin right of the user.

#### Employee Controller :

It controls the employee attendance, employee information and duty hours & duty locations. It can generate monthly reports on employee details to provide it to the security agencies for performance evaluation.

#### Alarm Controller :

Alarm controller manages the alert messages to be passed among Security Guards and Members. The situations when the alarm system will work are mentioned below:

If the visitor has exceeded the specific time limit the alarm will alert

If the ID card has GPS tracker and visitor has entered the restricted area it will alert the security guards in their wireless device.

If any member ask for emergency help. To alert the members about some declaration.

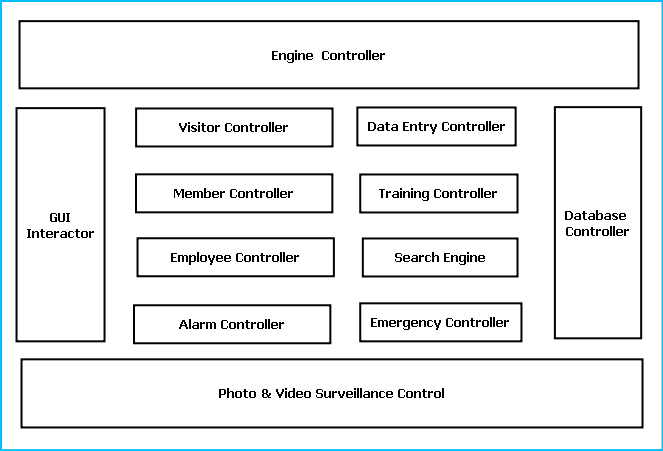


Figure: Components of Engine

#### Data Entry Controller :

It controls the data entered and passed to the engine. It validates the data depending on predefined inputs. If validation is failed it generates warning messages.

#### Training Controller :

This module provides the contents for Security Training and awareness campaigns. It provides audio visual output display to view them. It also allows printing security info flyers and booklets.

#### Search Engine :

Search engine enables rapid and efficient searching of data about visitors, employees and members. Search Engine prepares search indexes depending most accessed data. It improves the efficiency and performance of the software.

#### Emergency Controller :

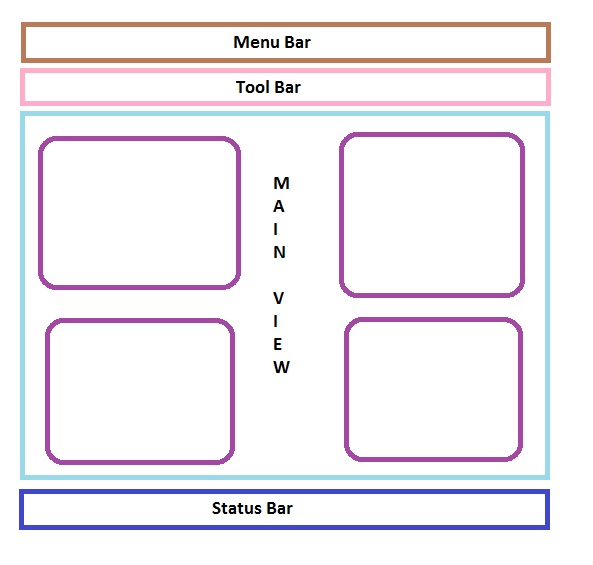
Emergency controller provides the algorithm for action to be taken on emergency situations. It will store emergency contacts, police station, hospital, ambulance & fire fighter contact number. And it will alert them via call, sms and email during emergency situations. It will also include intelligence to enable & disable critical services like opening exit doors, stopping lift services, shutting down power supply or starting emergency power backups.

#### Security Management System Database:

Security Management System will maintain a medium sized centralized database for storing information. We will design a RDBMS to manage the database and engine interaction. It will have optimized design and archive older data to save space and increase performance.

### DESIGN DOCUMENT:

The user interface is developed using WPF.



#### Manu Bar

Menu bar contains the important features which can be accessed by shortcuts too. It is implemented using **Menu** control.

|  |
| --- |
| **System.Windows.Controls.Menu**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Represents a Windows menu control that enables you to hierarchically organize elements associated with commands and event handlers. |

#### Tool Bar

Toolbar lists down the mostly used features. It is implemented using **Tool Bar** control.

|  |
| --- |
| **System.Windows.Controls.ToolBar**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Provides a container for a group of commands or controls. |

#### Expander Control

Left navigation bar displays the tutorials available locally and online. This navigation bar is hide- able. It is implemented using the **Expander** control.

|  |
| --- |
| **System.Windows.Controls.Expander**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Represents the control that displays a header that has a collapsible window that displays content. |

#### Tree View

The local contents are displayed in a **Tree View** control. And every contents are displayed in a Tree View Item.

|  |
| --- |
| **System.Windows.Controls.TreeView**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Represents a control that displays hierarchical data in a tree structure that has items that can expand and collapse. |

#### Web Browser

Online contents are displayed in a **Web Browser** control. Users can select the content and purchase it from there.

|  |
| --- |
| **System.Windows.Controls.WebBrowser**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Hosts and navigates between HTML documents. Enables interoperability between WPF managed code and HTML script. |

#### Text Block

Textpane displays the Texts. This is implemented using a **Text Block** control.

|  |
| --- |
| **System.Windows.Controls.TextBlock**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Provides a lightweight control for displaying small amounts of flow content. |

#### Media Element

Video and audio playing is implemented using **Media Element** control.

|  |
| --- |
| **System.Windows.Controls.MediaElement**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Represents a control that contains audio and/or video. |

#### Image Control

Images are displayed using **Image** control.

|  |
| --- |
| **System.Windows.Controls.Image**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Represents a control that displays an image. |

#### Ink Canvas Control

Drawing pane is implemented using **Ink Canvas** control.

|  |
| --- |
| **System.Windows.Controls.InkCanvas**  **Namespace:**  [System.Windows.Controls](http://msdn.microsoft.com/en-us/library/system.windows.controls.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Defines an area that receives and displays ink strokes. |

#### Status Bar Class

Status bar displays the information about the application and the tutorial currently running. Its implemented using **Status Bar** Class.

|  |
| --- |
| **System.Windows.Controls.Primitives.StatusBar**  **Namespace:**  [System.Windows.Controls.Primitives](http://msdn.microsoft.com/en-us/library/system.windows.controls.primitives.aspx) **Assembly:**  Presentation Framework (in PresentationFramework.dll)  Represents a control that displays items and information in a horizontal bar in an application window. |

## Data integrity and constraints

We have used Integrity constraints in SMS to ensure accuracy and consistency of data in a relational database. Data integrity is handled in a relational database through the concept of referential integrity. There are many types of integrity constraints in **SMS** that play a role in referential integrity.

Codd initially defined two sets of constraints but, in his second version of the relational model, he came up with four integrity constraints:

### Entity integrity

In **SMS** we used various type of primary key and consciously we set the primary key property as not null. The entity integrity constraint states that no primary key value can be null. This is because the primary key value is used to identify individual tuples in a relation. Having null value for the primary key implies that we cannot identify some tuples.This also specifies that there may not be any duplicate entries in primary key column key row.

### Referential Integrity

The referential integrity constraint is specified between two relations and is used to maintain the consistency among tuples in the two relations. Informally, the referential integrity constraint states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation. It is a rule that maintains consistency among the rows of the two relations.

### Domain Integrity

**SMS** has various type of data field with set by default value of Null because if the value is not provided by the user, the vale will be set as null. The domain integrity states that every element from a relation should respect the type and restrictions of its corresponding attribute. A type can have a variable length which needs to be respected. Restrictions could be the range of values that the element can have, the default value if none is provided, and if the element can be NULL.

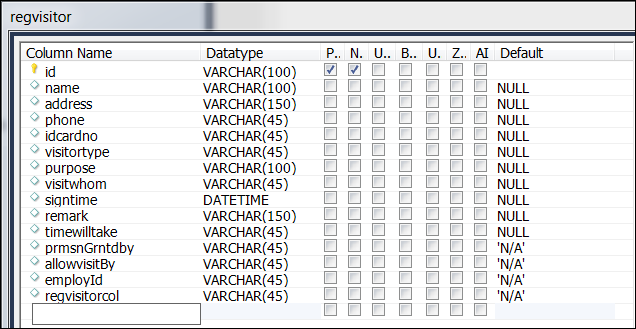
### User Defined Integrity

A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behaviour of the business.

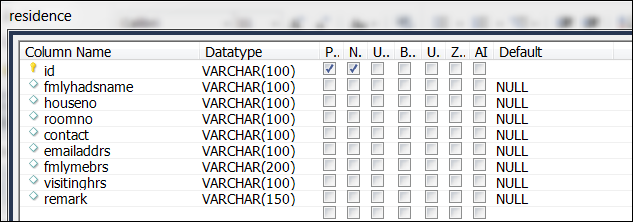
## Database & Table Details

We have used thye database named securityDB for storing data.Following are the tables are used in the databases for data manupulation:

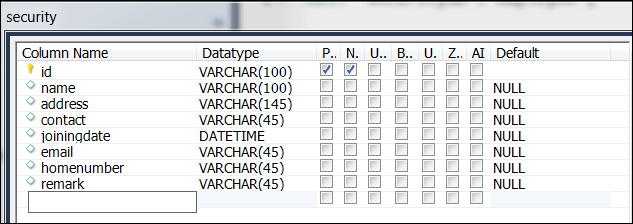
**Table::regvisitor**



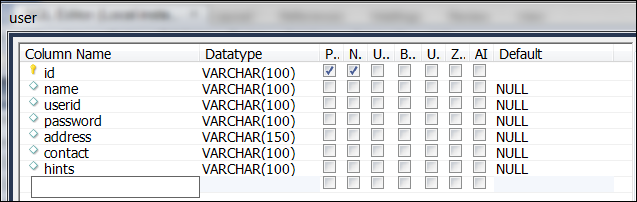
**Table::residence**



**Table::security**



**Table::user**



|  |  |
| --- | --- |
| **Tables** | **Attributes** |
| employee | `employee`.`id`, `employee`.`name`, `employee`.`contact`, `employee`.`email`, `employee`.`homenumber`, `employee`.`address`, `employee`.`joiningdate`, `employee`.`employeeType`, `employee`.`remark` |
| regvisitor | `regvisitor`.`id`, `regvisitor`.`name`, `regvisitor`.`address`, `regvisitor`.`phone`, `regvisitor`.`idcardno`, `regvisitor`.`visitortype`, `regvisitor`.`purpose`, `regvisitor`.`visitwhom`, `regvisitor`.`signtime`, `regvisitor`.`remark`, `regvisitor`.`timewilltake`, `regvisitor`.`prmsnGrntdby`, `regvisitor`.`allowvisitBy`, `regvisitor`.`employId`, `regvisitor`.`regvisitorcol` |
| residence | `residence`.`id`, `residence`.`fmlyhadsname`, `residence`.`houseno`, `residence`.`roomno`, `residence`.`contact`, `residence`.`emailaddrs`, `residence`.`fmlymebrs`, `residence`.`visitinghrs`, `residence`.`remark` |
| security | `security`.`id`, `security`.`name`, `security`.`address`, `security`.`contact`, `security`.`joiningdate`, `security`.`email`, `security`.`homenumber`, `security`.`remark` |
| User | `user`.`id`, `user`.`name`, `user`.`userid`, `user`.`password`, `user`.`address`, `user`.`contact`, `user`.`hints` |

## User Interface Design

### Main Window (Login Interface)

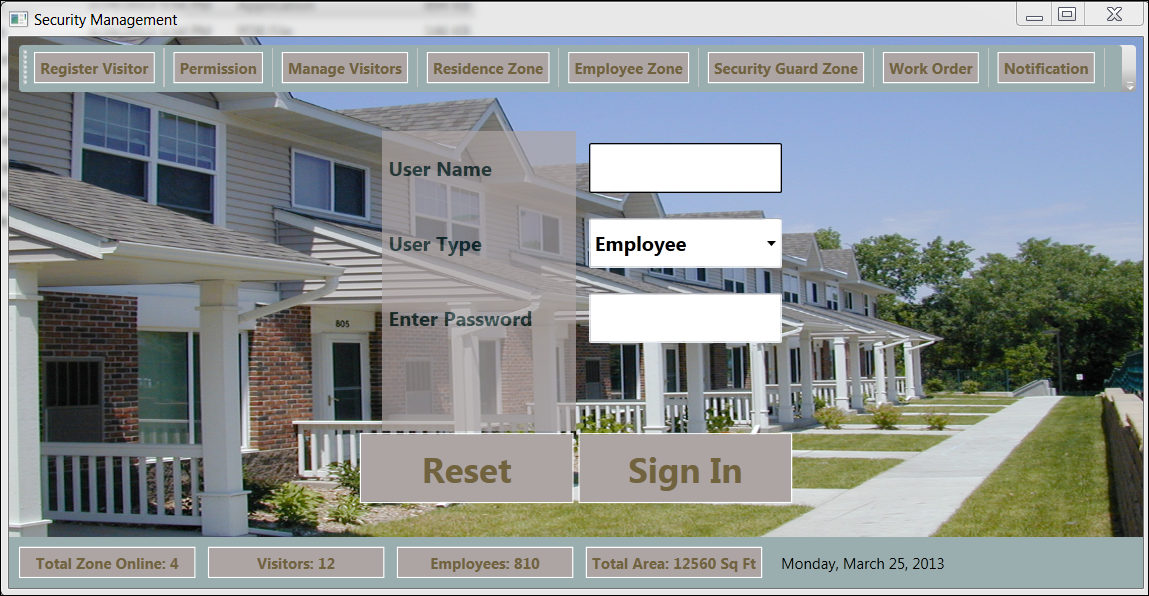


Figure 9 (Login Interface)

## Test Cases (Unit Test Cases and System Test Cases)

### Unit Test Cases:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test case Id | Description | Prerequisite | Test Data | Steps to produce | Result | Comments |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

### System Test cases:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test case Id | Description | Prerequisite | Test Data | Steps to produce | Result | Comments |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

# Coding

### Complete Project Coding

Mainwindow.xaml

|  |
| --- |
| <Window x:Class="SecurityManagementSystem.MainWindow"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  Title="Security Management" Height="848" Width="862" xmlns:my="clr-namespace:SecurityManagementSystem">  <DockPanel LastChildFill="True">  <Menu DockPanel.Dock="Top">  <MenuItem Header="File"></MenuItem>  <MenuItem Header="Settings"></MenuItem>  <MenuItem Header="Help"></MenuItem>  </Menu>  <ToolBar Margin="5" DockPanel.Dock="Top">  <Button Name="loginBtn" Click="loginBtn\_Click" Margin="5">Login</Button>  <Separator></Separator>  <Button Name="permBtn" Click="permBtn\_Click" Margin="5">Permission</Button>  <Separator></Separator>  <Button Name="reportBtn" Click="reportBtn\_Click" Margin="5">Reporting</Button>  <Separator></Separator>  <Button Name="regBtn" Click="regBtn\_Click" Margin="5">Register Visitor</Button>  <Separator></Separator>  <Button Name="visitorBtn" Click="visitorBtn\_Click" Margin="5">Manage Visitors</Button>  <Separator></Separator>  <Button Margin="5">Print</Button>  <Separator></Separator>  <Button Margin="5">Security</Button>  <Separator></Separator>  <Button Name="vidBtn" Click="vidBtn\_Click" Margin="5">Video Surveillance</Button>  <Separator></Separator>  <Button Margin="5">Internet</Button>  </ToolBar>  <StatusBar DockPanel.Dock="Bottom">  <UniformGrid Rows="1">  <Button Margin="5" Content="Total Zone Online: 4"></Button>  <Button Margin="5" Content="Visitors: 12"></Button>  <Button Margin="5" Content="Employees: 810"></Button>  <Button Margin="5" Content="Total Area: 12560 Sq Ft"></Button>  <Button Margin="5" Content="Time: 12:00"></Button>  </UniformGrid>  </StatusBar>  <UniformGrid>  <my:VideoSurveillanceControl x:Name="videoSurveillanceControl1" Margin="5"/><my:VideoSurveillanceControl />  <my:VideoSurveillanceControl x:Name="videoSurveillanceControl2" Margin="5"/>  <my:VideoSurveillanceControl x:Name="videoSurveillanceControl3" Margin="5"/>  </UniformGrid>  </DockPanel>  </Window> |

MainWindow.xaml.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  namespace SecurityManagementSystem  {  /// <summary>  /// Interaction logic for MainWindow.xaml  /// </summary>  public partial class MainWindow : Window  {  public MainWindow()  {  InitializeComponent();  }  private void visitorBtn\_Click(object sender, RoutedEventArgs e)  {  }  private void reportBtn\_Click(object sender, RoutedEventArgs e)  {  }  private void permBtn\_Click(object sender, RoutedEventArgs e)  {  }  private void regBtn\_Click(object sender, RoutedEventArgs e)  {  }  private void loginBtn\_Click(object sender, RoutedEventArgs e)  {  }  private void vidBtn\_Click(object sender, RoutedEventArgs e)  {  }  }  } |

VideoSurveillance.xaml

|  |
| --- |
| <UserControl x:Class="SecurityManagementSystem.VideoSurveillanceControl"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  mc:Ignorable="d"  d:DesignHeight="338" d:DesignWidth="402">  <Border CornerRadius="25" Background="Bisque" BorderBrush="Chocolate" BorderThickness="4">  <Grid Height="270" >  <Grid.RowDefinitions>  <RowDefinition Height="36\*" />  <RowDefinition Height="226\*" />  <RowDefinition Height="36\*" />  </Grid.RowDefinitions>  <StackPanel Orientation="Horizontal">  <Label Margin="20 0">Zone name</Label>  <ComboBox SelectedIndex="1" Width="150">  <ComboBoxItem> Zone One</ComboBoxItem>  <ComboBoxItem>Zone Two</ComboBoxItem>  <ComboBoxItem>Zone Three</ComboBoxItem>  <ComboBoxItem>Zone Four</ComboBoxItem>  </ComboBox>  </StackPanel>  <MediaElement x:Name="MediaEL" Grid.RowSpan="1" Grid.Row="1" LoadedBehavior="Manual" />  <StackPanel Orientation="Horizontal" Grid.Row="2" HorizontalAlignment="Center" Margin="8,0,8,5" Width="301">  <Button x:Name="btnRec" Content="Record" Click="btnRec\_Click"  Width="50" Height="25"/>  <Button x:Name="btnPlay" Content="Play" Click="btnPlay\_Click"  Width="50" Height="25"/>  <Button x:Name="btnStop" Content="Stop" Click="btnStop\_Click"  Width="50" Height="25"/>  <Button x:Name="btnMoveBackward" Content="Back" Click="btnMoveBackward\_Click"  Width="50" Height="25"/>  <Button x:Name="btnMoveForward" Content="Forward" Click="btnMoveForward\_Click"  Width="50" Height="25"/>  <Button x:Name="btnOpen" Content="Open" Click="btnOpen\_Click"  Width="50" Height="25"/>  </StackPanel>  </Grid>  </Border>  </UserControl> |

videoSurveilance.xaml.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  namespace SecurityManagementSystem  {  /// <summary>  /// Interaction logic for VideoSurveillanceControl.xaml  /// </summary>  public partial class VideoSurveillanceControl : UserControl  {  public VideoSurveillanceControl()  {  InitializeComponent();  IsPlaying(false);  }  private void btnRec\_Click(object sender, RoutedEventArgs e)  {  }  #region IsPlaying(bool)  private void IsPlaying(bool bValue)  {  btnStop.IsEnabled = bValue;  btnMoveBackward.IsEnabled = bValue;  btnMoveForward.IsEnabled = bValue;  btnPlay.IsEnabled = bValue;  }  #endregion  #region Play and Pause  private void btnPlay\_Click(object sender, RoutedEventArgs e)  {  IsPlaying(true);  if (btnPlay.Content.ToString() == "Play")  {  MediaEL.Play();  btnPlay.Content = "Pause";  }  else  {  MediaEL.Pause();  btnPlay.Content = "Play";  }  }  #endregion  #region Stop  private void btnStop\_Click(object sender, RoutedEventArgs e)  {  MediaEL.Stop();  btnPlay.Content = "Play";  IsPlaying(false);  btnPlay.IsEnabled = true;  }  #endregion  #region Back and Forward  private void btnMoveForward\_Click(object sender, RoutedEventArgs e)  {  MediaEL.Position = MediaEL.Position + TimeSpan.FromSeconds(10);  }  private void btnMoveBackward\_Click(object sender, RoutedEventArgs e)  {  MediaEL.Position = MediaEL.Position - TimeSpan.FromSeconds(10);  }  #endregion  #region Open Media  private void btnOpen\_Click(object sender, RoutedEventArgs e)  {  Microsoft.Win32.OpenFileDialog ofd = new Microsoft.Win32.OpenFileDialog();  ofd.Filter = "Video Files (\*.wmv)|\*.wmv";  if (ofd.ShowDialog() == true)  {  MediaEL.Source = new Uri(ofd.FileName);  btnPlay.IsEnabled = true;  }  }  #endregion    }  } |

### Comments and Description of Coding segments

#### SecurityManagementSystemEngine

##### Classes

|  |  |  |
| --- | --- | --- |
|  | Class | Description |
| Public class | [EmployeeInformation](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1499411819.htm) | EmployeeInformation Class  Namespace: SecurityManagementSystemEngine  Assembly: SecurityManagementSystemEngine (in SecurityManagementSystemEngine.dll)  Syntax  public class EmployeeInformationPublic Class EmployeeInformation  public ref class EmployeeInformation  Inheritance Hierarchy  Object  SecurityManagementSystemEngine..::..EmployeeInformation |
| Public class | [ResidenceInformation](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1896257898.htm) | ResidenceInformation Class  Namespace: SecurityManagementSystemEngine  Assembly: SecurityManagementSystemEngine (in SecurityManagementSystemEngine.dll)  Syntax  public class ResidenceInformationPublic Class ResidenceInformation  public ref class ResidenceInformation  Inheritance Hierarchy  Object  SecurityManagementSystemEngine..::..ResidenceInformation |
| Public class | [SecurityInformation](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1430692446.htm) |  |
| Public class | [SecurityManagementController](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1249593192.htm) |  |
| Public class | [EmployeeManager](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1421830386.htm) |  |
| Public class | [MemberManager](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\809995051.htm) |  |
| Public class | [VisitorManager](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\243654530.htm) |  |
| Public class | [NotificationController](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\294197890.htm) |  |
| Public class | [RequestController](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\204423621.htm) |  |
| Public class | [MemberInfo](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\74971646.htm) |  |
| Public class | [EmployeeInfo](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\240363170.htm) |  |
| Public class | [VisitorInfo](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\2046260361.htm) |  |
| Public class | [VisitInfo](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\616313253.htm) |  |
| Public class | [PreferenceInfo](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1075729497.htm) |  |
| Public class | [NotificationInfo](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\356377396.htm) |  |
| Public class | [RequestInfo](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\661856587.htm) |  |
| Public class | [VisitorInformation](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\277981642.htm) |  |

##### Enumerations

|  |  |  |
| --- | --- | --- |
|  | Enumeration | Description |
| Public enumeration | [VisitType](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\377480514.htm) |  |
| Public enumeration | [NotificationType](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1892524425.htm) |  |
| Public enumeration | [RequestType](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\2124579274.htm) |  |

#### SecurityManagementSystem

##### Classes

|  |  |  |
| --- | --- | --- |
|  | Class | Description |
| Public class | [App](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\396371478.htm) | Interaction logic for App.xaml |
| Public class | [EmployeeZone](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\86473862.htm) | Interaction logic for EmployeeZone.xaml |
| Public class | [MainWindow](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1772753063.htm) | Interaction logic for MainWindow.xaml |
| Public class | [ManageVisitors](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1007970578.htm) | Interaction logic for ManageVisitors.xaml |
| Public class | [Notifications](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\350137988.htm) | Interaction logic for Notifications.xaml |
| Public class | [Permission](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\617715709.htm) | Interaction logic for Permission.xaml |
| Public class | [RegisterVisitor](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1060379016.htm) | Interaction logic for RegisterVisitor.xaml |
| Public class | [ResidenceZone](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\687487572.htm) | Interaction logic for ResidenceZone.xaml |
| Public class | [SecurityGuardZone](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\490729504.htm) | Interaction logic for SecurityZone.xaml |
| Public class | [VideoSurveillance](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1871061189.htm) | Interaction logic for VideoSurveillance.xaml |
| Public class | [VideoSurveillanceControl](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1810321452.htm) | Interaction logic for VideoSurveillanceControl.xaml |
| Public class | [VisitorsInside](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1620329095.htm) | Interaction logic for VisitorsInside.xaml |
| Public class | [Workorder](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\451183183.htm) | Interaction logic for Workorder.xaml |

#### SecurityManagementSystem.Properties Namespace

##### Classes

|  |  |  |
| --- | --- | --- |
|  | Class | Description |
| Protected class | [Settings](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\360785546.htm) |  |
| Protected class | [Resources](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\380891724.htm) | A strongly-typed resource class, for looking up localized strings, etc. |

#### SecurityManagementSystemStyle.Properties Namespace

##### Classes

|  |  |  |
| --- | --- | --- |
|  | Class | Description |
| Protected class | [Settings](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\205283726.htm) |  |
| Protected class | [Resources](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\1381608571.htm) | A strongly-typed resource class, for looking up localized strings, etc. |

#### SecurityManagementSystemStorage Namespace

##### Classes

|  |  |  |
| --- | --- | --- |
|  | Class | Description |
| Public class | [SecurityManagementSystemStorageInteraction](file:///C:\Users\chandra\Documents\GitHub\SecurityManagementSystem\Help\html\336999373.htm) |  |

### Standardization of the coding

### Code Efficiency

We started working on the project keeping in mind that we must develop it in a way that it not only provides a very easy to use GUI but also provide a fast and flexible service to the users. We know that a particular work can be done in more than one ways. We have tried all the options and then chose the one which provides the fastest and most secure performance. First of all, we have used the latest technologies of Microsoft like visual studio 2010 as IDE and WPF as GUI to keep our application’s performance few steps ahead. We have studies all the rules of software development life cycle and applied them to keep our application flexible. We have given special attention to the storage related codes. We have avoided all the unnecessary database codes and kept them as short as possible without harming our purpose so that insertion, updating, deletion and fetching of data take place flexibly. You can see the result as a user; our application does all the works very smoothly.

### Error handling

### Parameters calling/passing

### Validation checks

# Testing

## Testing techniques and Testing strategies used

SMS application will be tested using following strategies to ensure that the application succeeds to complete all the functional and non functional requirements.

### Database & Data Integrity Testing

The databases and the database processes should be tested as a subsystem within the SMS Application. These subsystems should be tested with the target-of-test’s User Interface as the interface to the database.

|  |  |
| --- | --- |
| Test Objective: | Ensure that data is stored correctly, audits can be performed, access is controlled |
| Technique: | SQL queries will be executed in the DB to verify the data content and correctness. |
| Completion Criteria: | All planned tests have been executed.  All defects that have been identified have been resolved  All resolutions have been implemented. |

### Functional Testing:

Function testing focuses on any requirements for test that can be traced directly to use cases or business functions and business rules. The goals of these tests are to verify proper data acceptance, processing, and retrieval, and the appropriate implementation of the business rules. This type of testing is based upon black box techniques; that are verifying the application and its internal processes by interacting with the application via the Graphical User Interface (GUI) and analyzing the output or results. Identified below is an outline of the function testing recommended for SMS:

|  |  |
| --- | --- |
| Test Objective: | Ensure proper target-of-test functionality, including business process validation. |
| Technique: | Execute each use case, use-case flow, or function, using valid and invalid data, to verify the following:  The expected results occur when valid data is used.  The appropriate error or warning messages are displayed when invalid data is used.  Business rules are properly applied.  Black Box end to end testing of configured processes. Manual validation of required and optional fields. |
| Completion Criteria: | All planned tests have been executed.  All defects that have been identified have been resolved  All resolutions have been implemented. |

### Regression Testing:

Regression testing focuses on software functionality that may have been previously working however through subsequent changes may have been inadvertently impacted. The goals of these tests are to verify that the broader impact of changes has been verified. Identified below is an outline of the regression testing recommended for each application(s)/module(s) of SMS.

|  |  |
| --- | --- |
| Test Objective: | Ensure that previously passed test cases continue to pass as the new system development is deployed and that surrounding systems that may be impacted by a change are still functioning as expected. |
| Technique: | Execute previous passed testing suites to ensure the following:  The expected results occur when valid data is used.  The appropriate error or warning messages are displayed when invalid data is used.  Each business rule is properly applied. |
| Completion Criteria: | • All planned regression tests have been executed.  • All identified defects have been resolved. |

### User Interface Testing:

User Interface (UI) testing verifies a user’s interaction with the software. The goal of UI testing is to ensure that the User Interface provides the user with the appropriate access and navigation through the functions of the target-of-test. In addition, UI testing ensures that the objects within the UI function as expected and conform to corporate or industry standards. Most of this testing will have been done during functional testing. The areas of focus will be on design, layout and navigation of the screens.

|  |  |
| --- | --- |
| Test Objective: | UI testing will verify the screens and the layouts and navigation |
| Technique: | Verify the design and layout of the screen.  Identify the integration links.  Test the functioning of the links – that the proper page is displayed and correct messages, pop-ups are shown when they need to be displayed etc  Validation of general navigation |
| Completion Criteria: | All navigation test cases have been executed.  All screens have been verified as per design and layouts  All defects that have been identified have been resolved. |

### Performance Profiling:

Performance profiling is a performance test in which response times, transaction rates, and other time-sensitive requirements are measured and evaluated. The goal of Performance Profiling is to verify performance requirements have been achieved. Performance profiling is implemented and executed to profile and tune performance behaviours as a function of conditions such as workload or hardware configurations

|  |  |
| --- | --- |
| Test Objective: | The purpose of performance profiling is to ensure the performance of the SMS application is up to the desired level. |
| Technique: | Use a subset of Test Procedures developed for Function and Business Cycle Testing.  Modify data files to increase the number of transactions or the scripts to increase the number of iterations each transaction occurs.  This will be done by using Load Runner or Quick Test Professional (QTP). |
| Completion Criteria: | Single Transaction or single user: Successful completion of the test scripts without any failures and within the expected or required time allocation per transaction.  Results are recorded and a performance baseline is created for the major logical functions within the scenarios listed above.  All performance defects are reviewed and triaged to an acceptable resolution. |

.

### Load Testing:

Load testing is a performance test which subjects the target-of-test to varying workloads to measure and evaluate the performance behaviours and ability of the target-of-test to continue to function properly under these different workloads. The goal of load testing is to determine and ensure that the system functions properly at the expected maximum workload. Additionally, load testing evaluates the performance characteristics, such as response times, transaction rates, and other time sensitive issues.

|  |  |
| --- | --- |
| Test Objective: | The purpose of load testing is to verify performance behaviour time for designated transactions or business cases under varying workload conditions. |
| Technique: | Use a subset of Test Procedures developed for Function and Business Cycle Testing.  Scripts will be executed to simulate the peak load for 1 hour and report will be generated and analysed.  This will be done using Load Runner. |
| Completion Criteria: | Multiple transactions or multiple users: Successful completion of the test scripts without any failures and within acceptable time allocation.  Results are recorded and a performance baseline is created for the major business functions within the scenarios listed above.  All load testing defects are reviewed and triaged to an acceptable resolution. |

### Stress Testing:

Stress testing is a type of performance test implemented and executed to find errors due to low resources or competition for resources. Low memory or disk space may reveal defects in the target-of-test that aren't apparent under normal conditions. Other defects might result from competition for shared resources like database locks or network bandwidth. Stress testing can also be used to identify the peak workload the target-of-test can handle, which is often beyond the production workload.

### Volume Testing:

Volume Testing subjects the target-of-test to large amounts of data to determine if limits are reached that cause the software to fail. Volume Testing also identifies the continuous maximum load or volume the target-of-test can handle for a given period. For example, if the target-of-test is processing a set of database records to generate a report, a Volume Test would use a large test database and check that the software behaved normally and produced the correct report.

### Security & Access Control Testing:

Security and Access Control Testing focus on following key areas of security:

Application-level security, including access to the Data or Business Functions

Application-level security ensures the authentication and authorization of a user. Authentication ensures that the user is a valid user of the system and authorization ensures that the user has the proper privileges to perform specific tasks on desired resources of the system. Testing will be conducted to validate the rules by taking into considerations the various roles applicable for the system.

### Failover & Recovery Testing:

Failover and Recovery Testing ensures that the target-of-test can successfully failover and recover from a variety of hardware, software or network malfunctions with undue loss of data or data integrity.

Failover testing ensures that, for those systems that must be kept running, when a failover condition occurs, the alternate or backup systems properly “take over” for the failed system without loss of data or transactions.

Recovery testing is an antagonistic test process in which the application or system is exposed to extreme conditions, or simulated conditions, to cause a failure, such as device Input/ Output (I/O) failures or invalid database pointers and keys. Recovery processes are invoked and the application or system is monitored and inspected to verify proper application, or system, and data recovery has been achieved.

### Configuration Testing:

Configuration testing verifies the operation of the target-of-test on different software and hardware configurations. In most production environments, the particular hardware specifications for the client workstations, network connections and database servers vary. Client workstations may have different software loaded⎯for example, applications, drivers, and so on⎯and at any one time, many different combinations may be active using different resources.

### Installation/Deploy & Back out Testing:

Installation testing has two purposes. The first is to ensure that the software can be installed under different conditions⎯such as a new installation, an upgrade and a complete or custom installation⎯under normal and abnormal conditions. Abnormal conditions include insufficient disk space, lack of privilege to create directories, and so on. The second purpose is to verify that, once installed; the software operates correctly and can be backed out successfully. This usually means running a number of the tests that were developed for Function testing before and after the back out.

### Post Production Testing:

The purpose of Post production testing is to verify that, once deployed, the software operates correctly. This usually means running a number of the tests that were developed for Function Testing ensuring that no data is changed/ modified in production. Typically, the business SME’s assist with Post production testing.

### Unit Testing:

Unit testing will take place within the construction phase of the project. After application module has been built to meet design specifications, each component (screen, view, interface, conversion program, etc.) will be tested individually to help confirm that it functions properly as an individual unit. Basic performance testing will also be completed during unit test to resolve obvious issues with performance prior to the System Testing.

The resource responsible for development will conduct testing of their module using the unit test conditions defined by the developer based on detailed design documents. The final step of unit test will be a review by the team lead to obtain their signoff on the component test checklist.

### Smoke Testing:

|  |  |
| --- | --- |
| Test Objective: | Verifies the major functionality at high level in order to determine if further testing is possible. |
| Technique: | After initial deployment to the test environment validate all critical components of the application prior to proceeding with testing. |
| Completion Criteria: | Navigation through the application at high level is possible, testing can continue. |

### Data Migration Testing:

This is the process of testing to verify whether or not the data migration (or conversion) has been successfully completed. The testing process will be carried out by running SQL scripts on both the source and destination databases.

The fields which are present in the new data Model in the Destination DB(s) will be migrated from the existing systems source DB(s) to Destination DB(s).

|  |  |
| --- | --- |
| Test Objective: | The objective of this test is to verify that data migration is successful from source DB(s) to destination DB(s). |
| Technique: | The Team is notified before the data migration.  Team runs queries on the source DB and fetches the data.  Data Migration is done.  Mapped data needs to be determined.  Team runs the queries on the Destination DB and fetches the data.  Cross verification of the data is done to see that data fetched from the old database is same as the data fetched from the new database.  Verification of the table structure.  Verification of record counts.  Verification of the data formatting. |
| Completion Criteria: | Data fetched from the Source DB(s) and Destination DB(s) matches.  The record count in the Source and the Destination databases should be equal.  No data are truncated.  Data formatting is proper (if required at any instance).  All defects that have been identified have been resolved. |

## Testing Plan used

## Test reports for Unit Test Cases and System Test Cases

## Debugging and Code improvement:

# System Security measures:

## Database/data security:

It encrypts the data stored in the database so that even if someone succeeds to hack the database still not much harm could be done.

The application will use Google open-id authentication for web interface.

## Creation of User profiles and access rights

The software requires a predefined username and password to login.

It allows a guest login as well which lets a guest user user this application with very limited access to the user data.

## Estimation



# Cost Estimation of the Project along with Cost Estimation Model

We used the basic COCOMO model, which gives an approximate estimate of our **SMS** project parameters. The basic COCOMO estimation model is given by the following expressions:

Effort = a1 \* (KLOC)a2 PM

Tdev = b1 \* (Effort)b2 months

Where

KLOC is the estimated size of the software product expressed in Kilo Lines of Code a1, a2, b1, b2 are constants for each category of software products.

Tdev is the estimated time to develop the software, expressed in months.

Effort is the total effort required to develop the software product, expressed in person-month (PM).

Our project is semidetached type, because the development team consists of a mixture of experienced and inexperienced staff like my guide and me. Team members may have limited experience on related system but may be unfamiliar with aspects of the system being developed.

## Estimation of development effort

For our Semi-detached class software product **SMS**, the formula for estimating the effort based on the code size is shown below:

Semi-detached **SMS**: Tdev = 3.0\*(KLOC)1.12 PM

## Estimation of development time

For our Semi-detached class software product **SMS**, the formula for estimating the development time based on the effort is given below:

Semi-detached **SMS**: Tdev = 2.5\*(Effort)0.35 months

Assume that the size of a Semi-detached CES product has been estimated to be 4,000 lines of source code. Assume that the average salary of software engineer(me) is Rs. 15,000 per month.

Assume that the size of our

The basic COCOMO estimation formula for CES semidetached software:

Our Effort = 3.0 \* (4)1.12 PM

= 14 PM

Normal Development time = 2.5 \* (14)0.35 months

= 6 months

Cost required to develop the product = Rs. 6 \* 15,000

= Rs. 90,000

# Data Structure

|  |
| --- |
| **MemberInfo** |
| public class MemberInfo  {  public string id { get; set; }  public string name { get; set; }  public string address { get; set; }  public string contact { get; set; }  public List<PreferenceInfo> prefs;  public List<RequestInfo> reqs;  } |

|  |
| --- |
| **EmployeeInfo** |
| public class EmployeeInfo  {  public string id { get; set; }  public string name { get; set; }  public string address { get; set; }  public string contact { get; set; }  public DateTime doj { get; set; }  public string department { get; set; }  } |

|  |
| --- |
| **VisitorInfo** |
| public class VisitorInfo  {  public string id { get; set; }  public string photoIdNo { get; set; }  public string photoFilePath { get; set; }  public string name { get; set; }  public string address { get; set; }  public string contact { get; set; }  public VisitType postType { get; set; }  public List<VisitInfo> lastVisits;  } |

|  |
| --- |
| **VisitInfo** |
| public class VisitInfo  {  public string id { get; set; }  public string name { get; set; }  public MemberInfo memberTargeted { get; set; }  public EmployeeInfo employeeAuthroized { get; set; }  public DateTime start { get; set; }  public DateTime duration { get; set; }  } |

|  |
| --- |
| **PreferenceInfo** |
| public class PreferenceInfo  {  string name;  EmployeeInfo technicianTargeted;  string Description;  DateTime date;  } |

|  |
| --- |
| **NotificationType** |
| public enum NotificationType  {  Visit,  Timeout,  Issue,  Other  } |

|  |
| --- |
| **NotificationInfo** |
| public class NotificationInfo  {  string name;  EmployeeInfo employeeEdited;  MemberInfo memberTargeted;  VisitorInfo visitorRelated;  string description;  DateTime date;  NotificationType type;  } |

|  |
| --- |
| **RequestType** |
| public enum RequestType  {  Emergency,  Distress,  Household,  other  } |

|  |
| --- |
| **RequestInfo** |
| public class RequestInfo  {  public string id { get; set; }  public DateTime date { get; set; }  public RequestType type { get; set; }  public string description { get; set; }  } |

|  |
| --- |
| **SecurityManagementController** |
| public class SecurityManagementController  {  public VisitorManager employeeController;  public MemberManager customerController;  public EmployeeManager empController;  public NotificationController notificator;  public RequestController requestHandler;    } |

|  |
| --- |
| **EmployeeManager** |
| public class EmployeeManager  {  public List<EmployeeInfo> employees;  public List<RequestInfo> reqHandled;  } |

|  |
| --- |
| **MemberManager** |
| public class MemberManager  {  public List<MemberInfo> members;  public List<PreferenceInfo> prefs;  } |

|  |
| --- |
| **VisitorManager** |
| public class VisitorManager  {  public List<VisitorInfo> visitors;  public List<VisitInfo> visits;  } |

|  |
| --- |
| **NotificationController** |
| public class NotificationController  {  public List<NotificationInfo> notifications;  } |

|  |
| --- |
| **RequestController** |
| public class RequestController  {  public List<RequestInfo> requests;  } |

|  |
| --- |
| **VisitType** |
| public enum VisitType  {  Technician,  Admin,  Owner  } |

# Reports (sample layouts should be placed)

# Future scope and further enhancement of the Project

* Now it will display the text based RSS feeds and link of the multimedia contents. We will display the Multimedia contents like Video, Audio & Image in future.
* To support UNIX / Linux Based Operating systems.
* To Support Mobile Operating systems for Symbian, Meego & Android.

# Bibliography

* <http://en.wikipedia.org>
* <http://msdn.microsoft.com/en-us/>
* <http://www.microsoft.com/en-us/default.aspx>
* <http://www.codeplex.com/>
* <http://stackoverflow.com/>
* <http://www.codeguru.com/>
* <http://www.w3schools.com>
* <http://blogs.technicise.com/>
* [www.mysql.org](http://www.mysql.org)

# Appendices (if any)

# Glossary.