**2013**

**SUSMITA PODDER**

**1/19/2013**



SCHOOL MANAGEMENT SYSTEM

Table of Contents

[INTRODUCTION AND OBJECTIVE 4](#_Toc346626317)

[INTRODUCTION 4](#_Toc346626318)

[OBJECTIVE 4](#_Toc346626319)

[SYSTEM ANALYSIS 4](#_Toc346626320)

[IDENTIFICATION OF NEED 5](#_Toc346626321)

[PRELIMINARY INVESTIGATION 5](#_Toc346626322)

[FEASIBILITY STUDY 5](#_Toc346626323)

[PROJECT PLANNING 5](#_Toc346626324)

[TRACKING GANTT 5](#_Toc346626325)

[PROJECT SCHEDULING 5](#_Toc346626326)

[PERT CHART (NETWORK DIAGRAM) 5](#_Toc346626327)

[GANTT CHART 6](#_Toc346626328)

[SOFTWARE REQUIREMENT AND SPECIFICATION 7](#_Toc346626329)

[FUNCTIONAL REQUIREMENTS 7](#_Toc346626330)

[TECHNICAL SPECIFICATION 10](#_Toc346626331)

[SOFTWARE ENGINEERING PARADIGM APPLIED 10](#_Toc346626332)

[DATA MODELS 10](#_Toc346626333)

[CONTEXT DIAGRAM 10](#_Toc346626334)

[DATA FLOW DIAGRAM (DFD) 11](#_Toc346626335)

[CONTROL FLOW DIAGRAM 15](#_Toc346626336)

[STATE DIAGRAM / SEQUENCE DIAGRAM 15](#_Toc346626337)

[ENTITY RELATIONSHIP MODEL 15](#_Toc346626338)

[CLASS DIAGRAM / CRC MODEL /COLLABORATION DIAGRAM / USE-CASE DIAGRAM / ACTIVITY DIAGRAM 17](#_Toc346626339)

[SYSTEM DESIGN 18](#_Toc346626340)

[MODULARISATION DETAILS 18](#_Toc346626341)

[SCHOOL MANAGEMENT SERVER 19](#_Toc346626342)

[SCHOOL MANAGEMENT CLIENT 19](#_Toc346626343)

[SCHOOL MANAGEMENT DATABASE 20](#_Toc346626344)

[DATA INTEGRITY AND CONSTRAINTS 20](#_Toc346626345)

[DATABASE AND TABLE DESIGN 20](#_Toc346626346)

[PROCEDURAL DESIGN / OBJECT ORIENTED DESIGN 21](#_Toc346626347)

[USER INTERFACE DESIGN 22](#_Toc346626348)

[TEST CASES 22](#_Toc346626349)

[UNIT TEST CASES 22](#_Toc346626350)

[SYSTEM TEST CASES 22](#_Toc346626351)

[CODING 23](#_Toc346626352)

[COMPLETE PROJECT CODING 23](#_Toc346626353)

[COMMENTS AND DESCRIPTION OF CODING SEGMENTS 23](#_Toc346626354)

[STANDARDIZATION OF THE CODING 23](#_Toc346626355)

[CODE EFFICIENCY 23](#_Toc346626356)

[ERROR HANDLING 23](#_Toc346626357)

[PARAMETERS CALLING / PASSING 23](#_Toc346626358)

[VALIDATION CHECKS 23](#_Toc346626359)

[TESTING 23](#_Toc346626360)

[TESTING TECHNIQUES AND TESTING STRATEGIES USED 23](#_Toc346626361)

[TESTING PLAN USED 23](#_Toc346626362)

[TESTING REPORTS 23](#_Toc346626363)

[UNIT TEST CASES 23](#_Toc346626364)

[SYSTEM TEST CASES 24](#_Toc346626365)

[DEBUGGING AND CODE IMPROVEMENT 24](#_Toc346626366)

[SYSTEM SECURITY MEASURES 24](#_Toc346626367)

[DATABASE / DATA SECURITY 24](#_Toc346626368)

[CREATION OF USER PROFILES AND ACCESS RIGHTS 24](#_Toc346626369)

[COST ESTIMATION 24](#_Toc346626370)

[COST ESTIMATION MODEL 24](#_Toc346626371)

[REPORTS 24](#_Toc346626372)

[FUTURE SCOPE AND FURTHER ENHANCEMENT 25](#_Toc346626373)

[BIBLIOGRAPHY 25](#_Toc346626374)

[APPENDICES 25](#_Toc346626375)

[GLOSSARY 25](#_Toc346626376)

# INTRODUCTION AND OBJECTIVE

## INTRODUCTION

The population of our country is increasing rapidly, but the resources for providing proper education to the children are limited. So we need to utilize our existing school management processes properly with the help of digital technologies.

Most of the schools are managed by a single authority like head master or a group of people like governing body. But in this fast paced world people tend to switch jobs. If the key people leave the school then the quality of education and school management deteriorates. There are so many dependencies on the school management personnel. We can eradicate this dependency by deploying a computerized solution for managing school.

Nowadays both of the parents are employed for most kids and they don’t have enough time to interact with teachers frequently. We need a computerized system to manage interaction between parents and teachers.

We will implement a school management system to address these issues and bring up an efficient system to manage activities of a school using single software.

## OBJECTIVE

School Management System is versatile and complete end-to-end school management software .School Management System is used to enhance the administrative efficiency of educational institutions. It is an interactive platform for all entities viz. Students, Teachers, Management, Parents. It is a simple yet powerful one point integrated platform that connects all the departments of an institution namely office, fee counter, library, hostel, stores, academics, activity center and so on.

# SYSTEM ANALYSIS

## IDENTIFICATION OF NEED

When I was a kid I saw my parents coming to the school for submitting the remuneration and other reasons. The entire data was written using pen and papers so undoubtedly it took lots of time completing even minor queries. Nowadays I see my relatives going to the school to admit their kids and they face huge problems as no proper procedure is followed even by larger and more reputed institutes. I have always felt that there is a need of a proper procedure which could not only save lots of time of both the school authority and the guardians, also easy to maintain the entire process. So I really feel that the modernization of technology could be used to minimize these problems. That is why a proper application or software is required to ease the task of the school administration and save the time of the parents as well.

## PRELIMINARY INVESTIGATION

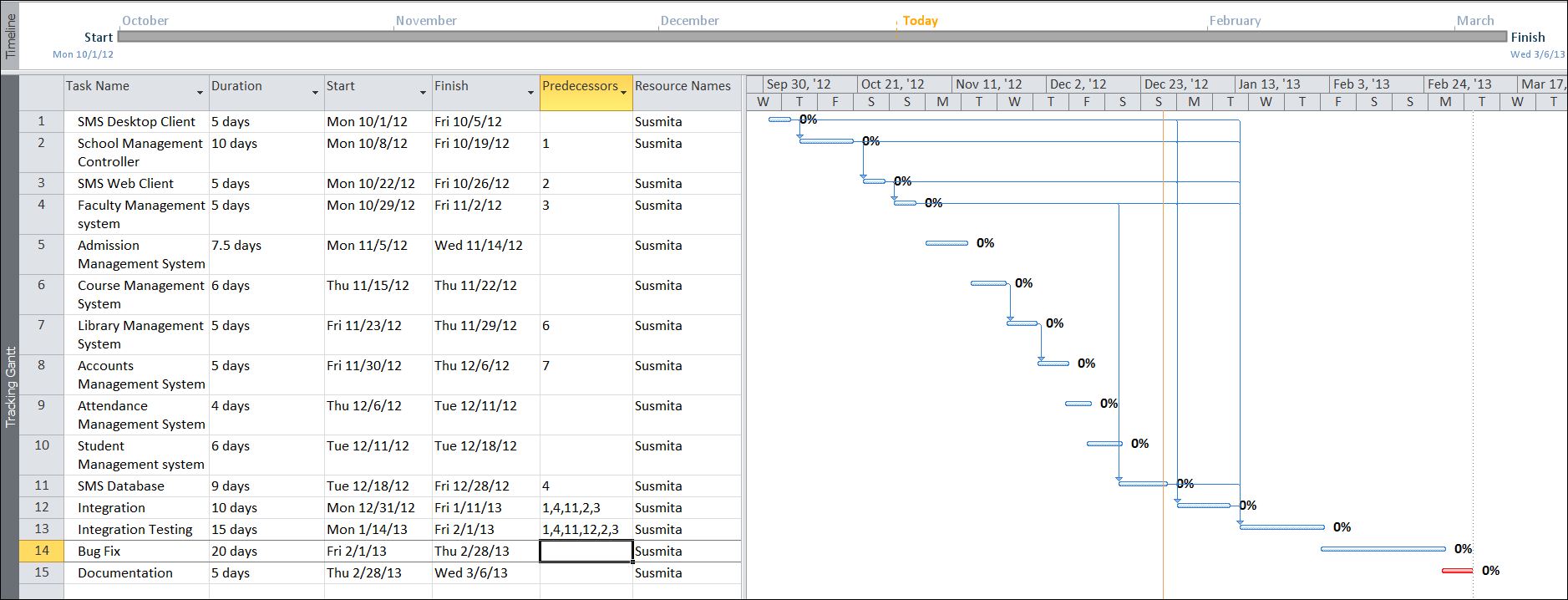
I started talking to my relatives and neighbors who have recently admitted their child to the schools or they are in the process of admitting them. Naturally, they are the ones who know the problems very well. I also spoke with some of school administrations in my locality about the problems they faced and what exactly would they want if they are given software like School Management System just to get an idea on what could the points be if we want to develop a new application to minimize school management related problems and difficulties. I obviously then consulted with some IT professionals and software engineers and expressed my view to them. I was really amazed to see their positive response on this matter and I got helps in many ways from them. After completing all this process I decided to develop this application for sure.

## FEASIBILITY STUDY

We all know that the numbers of school is growing as fast as Information Technology. Developing an application is very easy these days so people would love to use technology ease their tasks. There is need and there is solution so undoubtedly this software is going to be appreciated by the market.

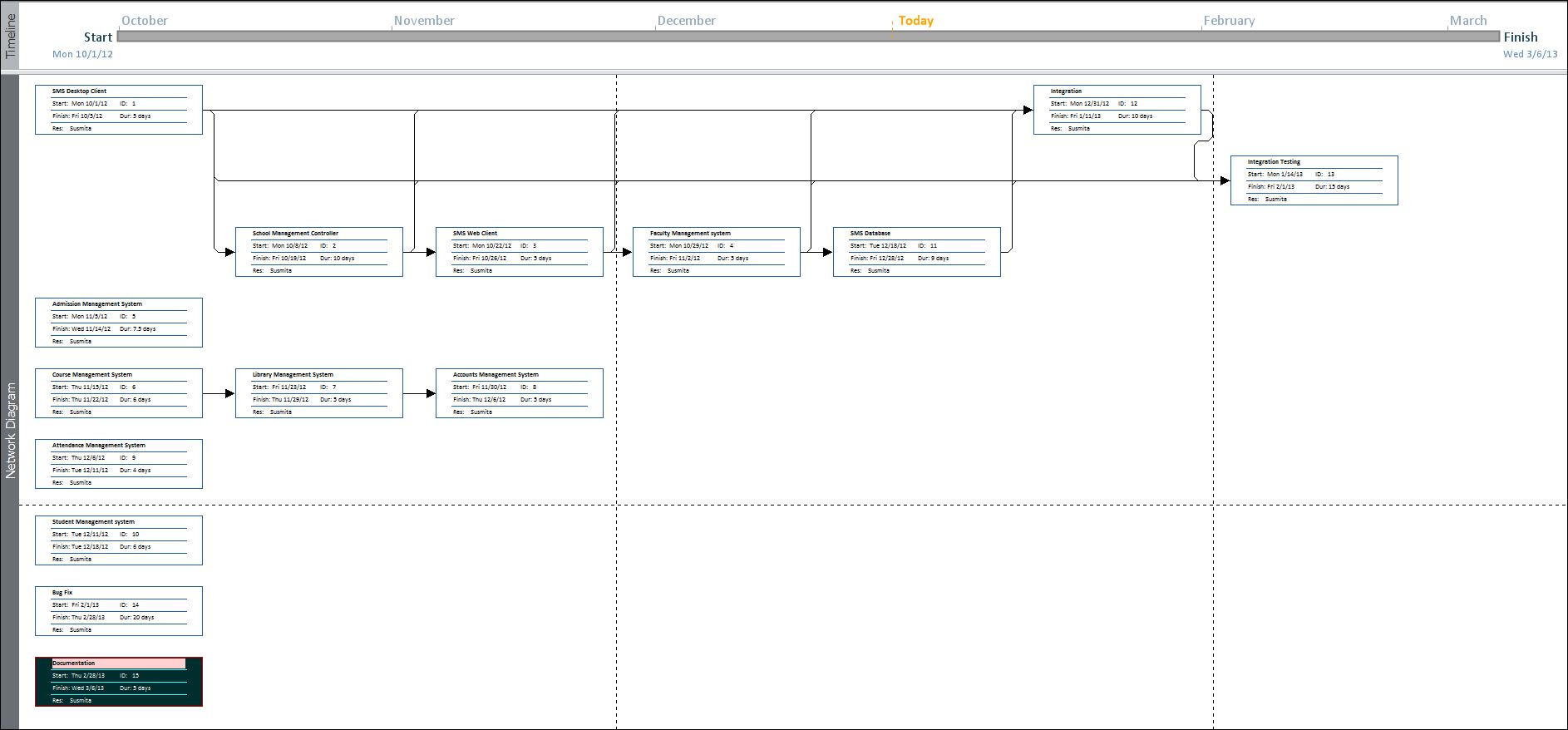
## PROJECT PLANNING

### TRACKING GANTT



## PROJECT SCHEDULING

### PERT CHART (NETWORK DIAGRAM)



### GANTT CHART



## SOFTWARE REQUIREMENT AND SPECIFICATION

### FUNCTIONAL REQUIREMENTS

#### View and Enter new Student and Employee information

**Introduction**

The details of a new student are stored into a student profile. Only School administration department can enter that details of new student but and teachers can only updated the existing student status. Details of employee are also kept into individual employee profile and it will also be updated by School Administration System.

**Input**

Relevant student and employee data like name, address, contact no., applying for which class.

**Processing**

Employee will enter data in SMS and create a new Student enrolment no, as well as a new code number for Employee.

**Output**

SMS will generate Enrolment no. for Student and Code no. for Employee .Details can be viewed later on whenever required.

#### View and Enter new timetable information

**Introduction**

Employee can view the time table and can also update the timetable information.

**Input**

Employee number, department number and Week range

**Processing**

Employee number and department number must be unique, and when entering timetable of any employee or teacher both values must be valid references. Week range must be between 1to 52.

**Output**

Teacher and Student both can see the time table.

#### Security

**Introduction**

Only the high level members of the School and Network manager will have access to the system for securing their important data from others.

**Input**

System username and password

**Processing**

The network operating system in the department will be used to enforce security. Another security level should also be incorporated to make the system more secure.

**Output**

All data are secured and that can be used in future.

#### Changing Password and Username

**Introduction**

Change existing username and password

**Input**

New username and password

**Processing**

Old username and password will be replaced by user provided new username and password after authenticating.

**Output**

Password and Username can be changed according to the Employee requirement whenever they want to change for better security of the System.

#### Mail Notification

**Introduction**

If holiday is declared suddenly, all students, teachers and employees are informed by sending them a mail.

**Input**

Student and Teacher’s name and email id.

**Processing**

Employee will enter the name, email id and reason of holidays in the SMS and it will generate a message.

**Output**

Employee and Students get a message from SMS.

#### Exam Grade Details

**Introduction**

Data sheets are prepared for individual class. And each datasheet is given via email to the student of the corresponding class.

**Input**

Student name, marks in individual subject, attendance, class performance**.**

**Processing**

Grade card can be generated for individual students. Exam administrators would need to be able to view, update, delete, print and add grade details.

**Output**

A printed Grade card can be given to the student as well as an email can be received by the student with Grade card.

#### Fees Details

**Introduction**

Fees details of all Students are kept and they are reminded after every 4 days after last date. Late Fee is also charged after last date.

**Input**

Student name, enrolment no, remaining fees, last date of deposit amount.

**Processing**

SMS automatically generates a message and send it to the student email id.

**Output**

Students are reminded after every 4 days after last date.

### non FUNCTIONAL REQUIREMENTS

* The software must have a easy to use graphic user interface as it is going to be used by all types of users, like professors and clerks.
* The software should be password protected to secure confidential data.
* The database must be encrypted so that no one could see the accounts information of the organization even if the database is somehow hacked.
* The application must be fast and flexible so that waiting time gets reduced.
* There must be a backup and restore feature so that the valuable data stays secure forever.

## 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.

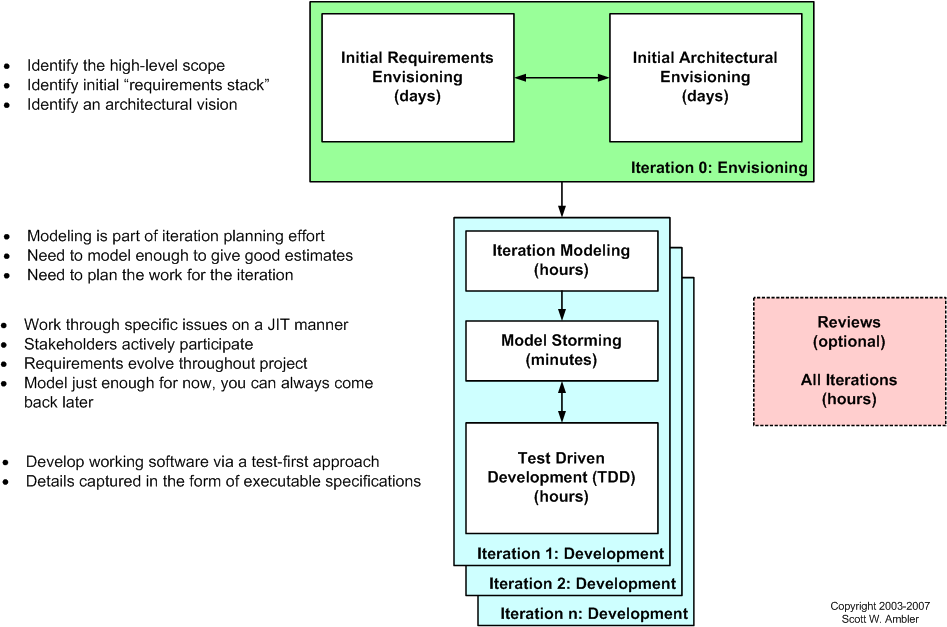


Figure 1The AMDD lifecycle: Modeling activities throughout the lifecycle of a project

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..

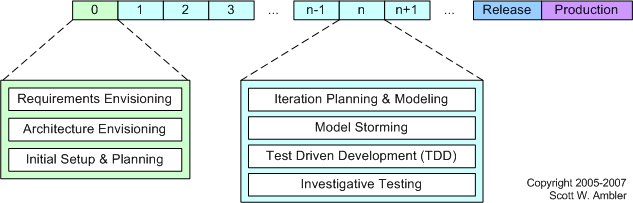
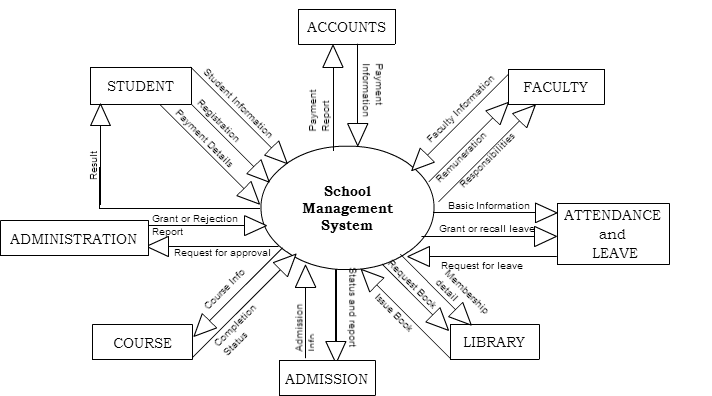


Figure 2AMDD Through the Agile Development Lifecycle.

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.

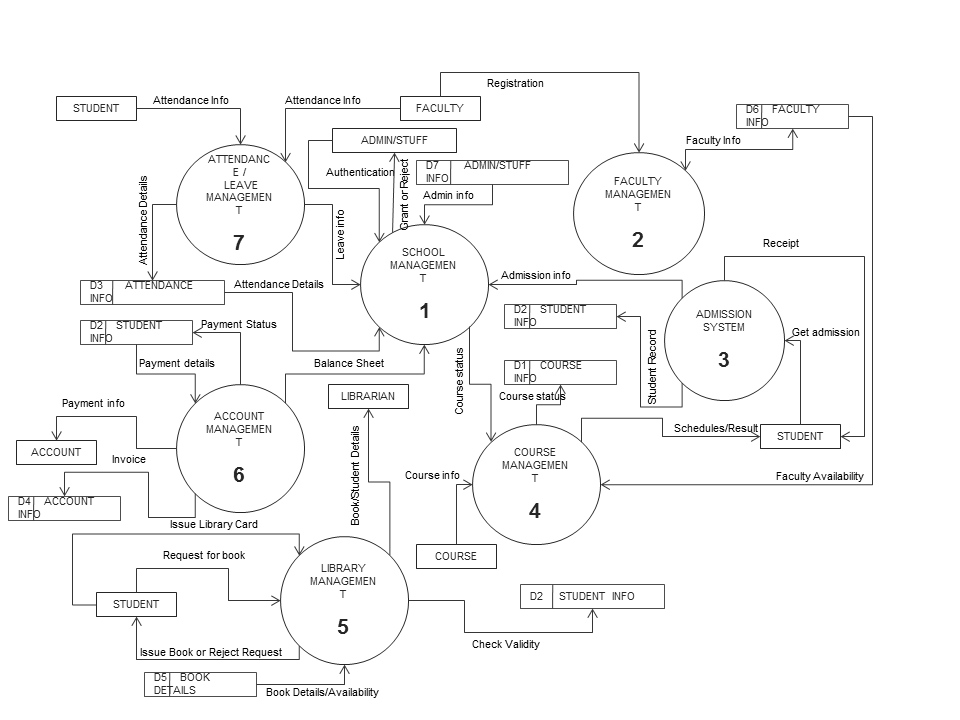
## DATA MODELS

### CONTEXT DIAGRAM

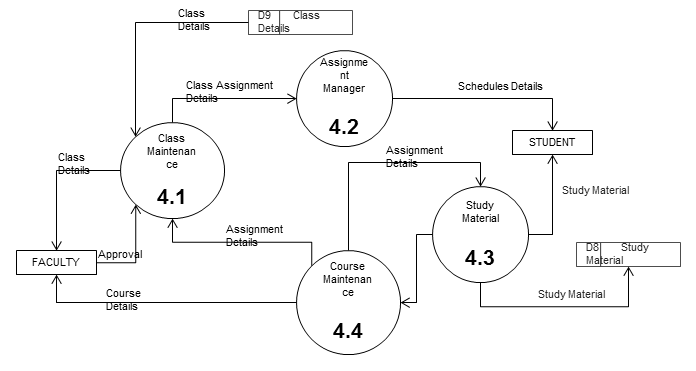


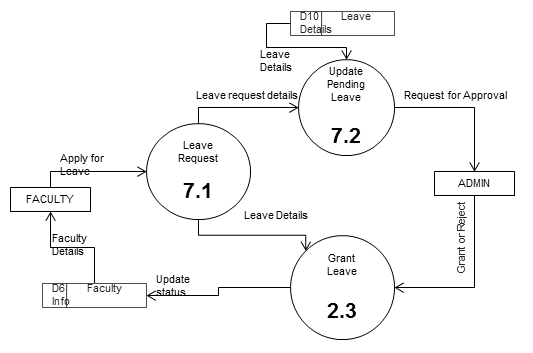
### DATA FLOW DIAGRAM (DFD)

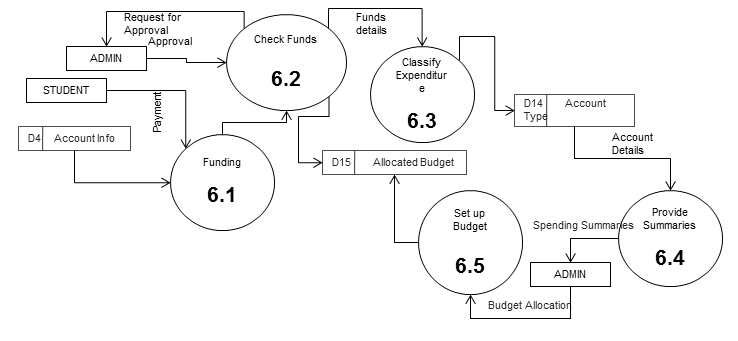
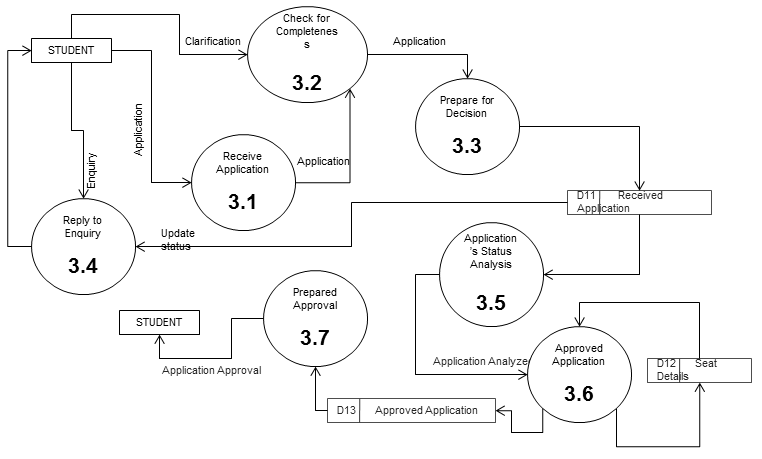
#### LEVEL 0 DFD



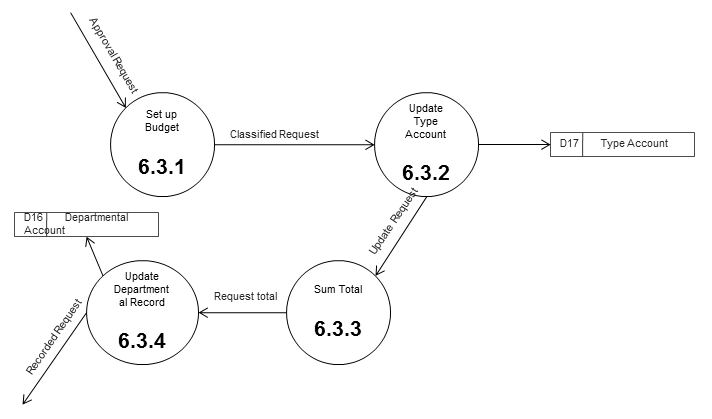
#### LEVEL 1 DFD



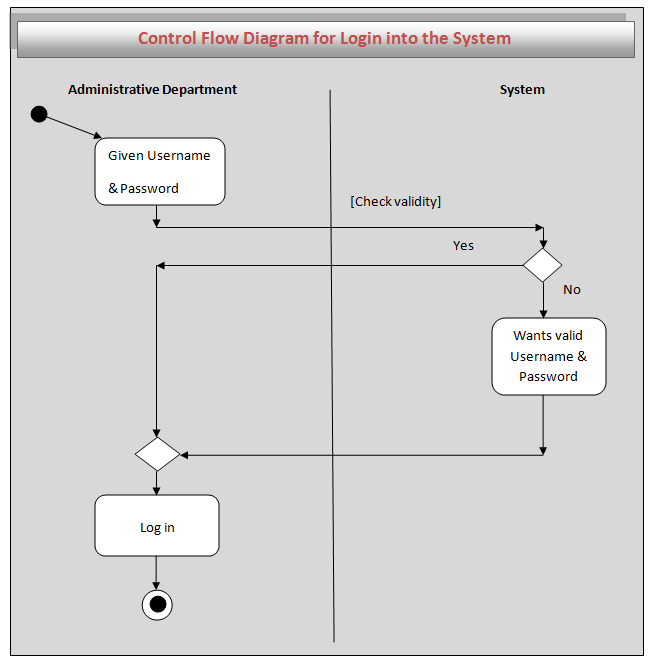


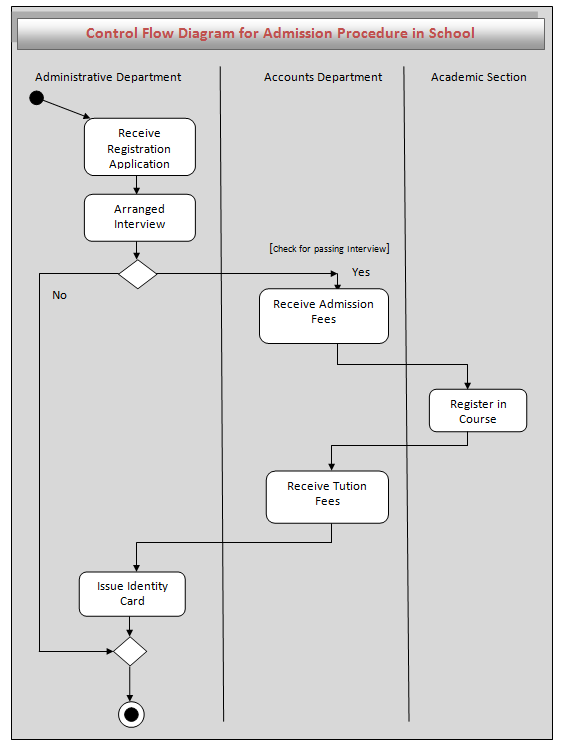


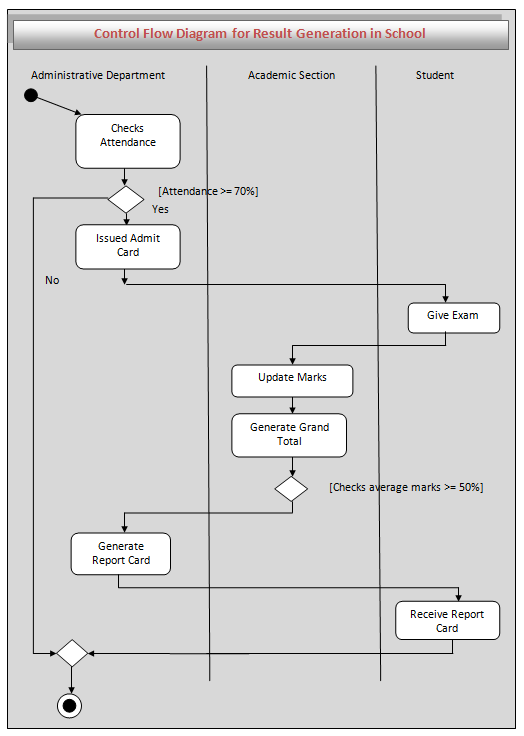
#### LEVEL 2 DFD



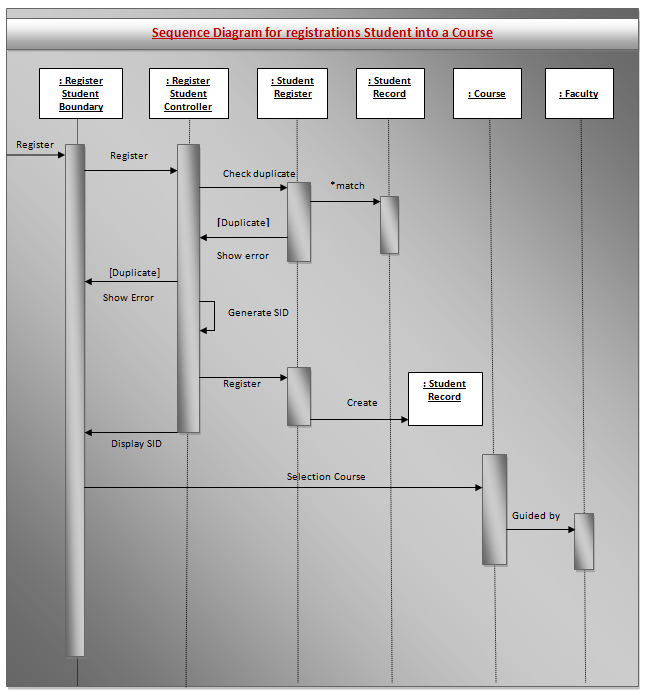
### CONTROL FLOW DIAGRAM







### SEQUENCE DIAGRAM



### ENTITY RELATIONSHIP MODEL

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

|  |  |
| --- | --- |
| **Entities** | **Attributes** |
| Student | **Student\_ID**, Student\_DOB, Student\_Name, Student\_Parent\_Name, Student\_Address, Student\_Admission\_Date, Student\_Course\_Name, Student\_Contact |
| Account | **Transaction\_ID**, Transaction\_Amount, Transaction\_Type, Transaction\_Reason,Account\_Balance |
| Admin | **Staff\_ID**, Staff\_Name, Staff\_Permission\_Level, Staff\_Address, Staff\_Admission\_Date, Staff\_Course\_Name, Staff\_Contract\_details, Staff\_Join\_Date, Staff\_Email, Staff\_Role |
| Books | **Book\_ID**, Book\_Name, Book\_Author, Purchase\_Date, Book\_Status, Book\_Description,Purchase\_Amount |
| Faculty | **Faculty\_ID**, Faculty\_Name, Faculty\_Address, Faculty\_Join\_Date, Faculty\_Course\_Under, Faculty\_Contact\_Details, Faculty\_Salary\_Details |
| Course | **Course\_ID**, Course\_Faculty, Course\_Name, Required\_Qualification, Course\_Fees, Course\_Admission\_Date, Students\_Under, Course\_Description |

**Relationship between Entities:**

School Management System has Courses 1 : N

School Management SystemhasStudents 1 : N

School Management System has Faculties1 : N

School Management System has Admin 1 : 1

Studenthas Attendance1 :1

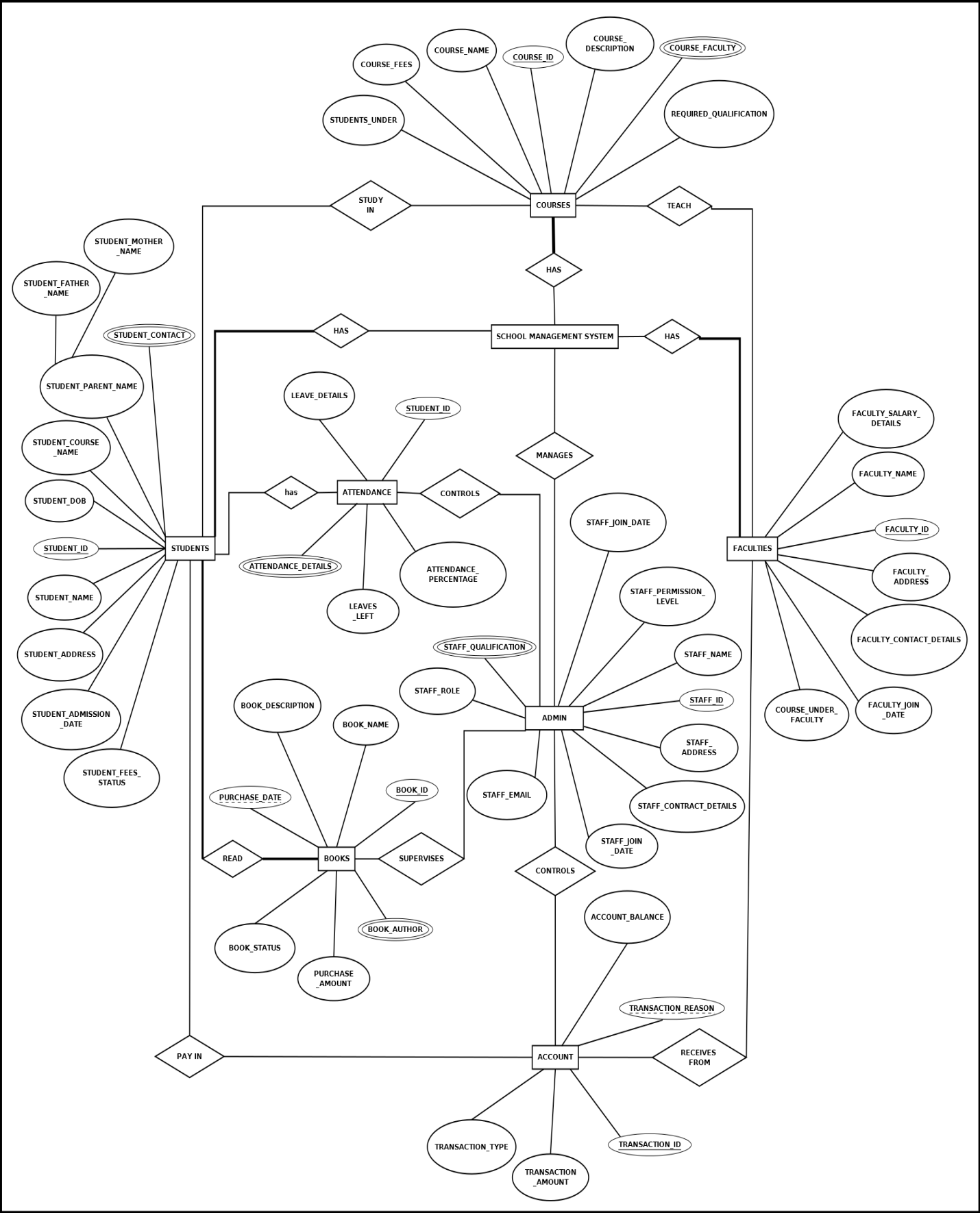
AdminChecksAttendance1 : 1

StudentsreadsBooksM : N

StudentspaysAccount1 : 1

AdminControlsAccount1 : 1

Students Studies in Course N:1

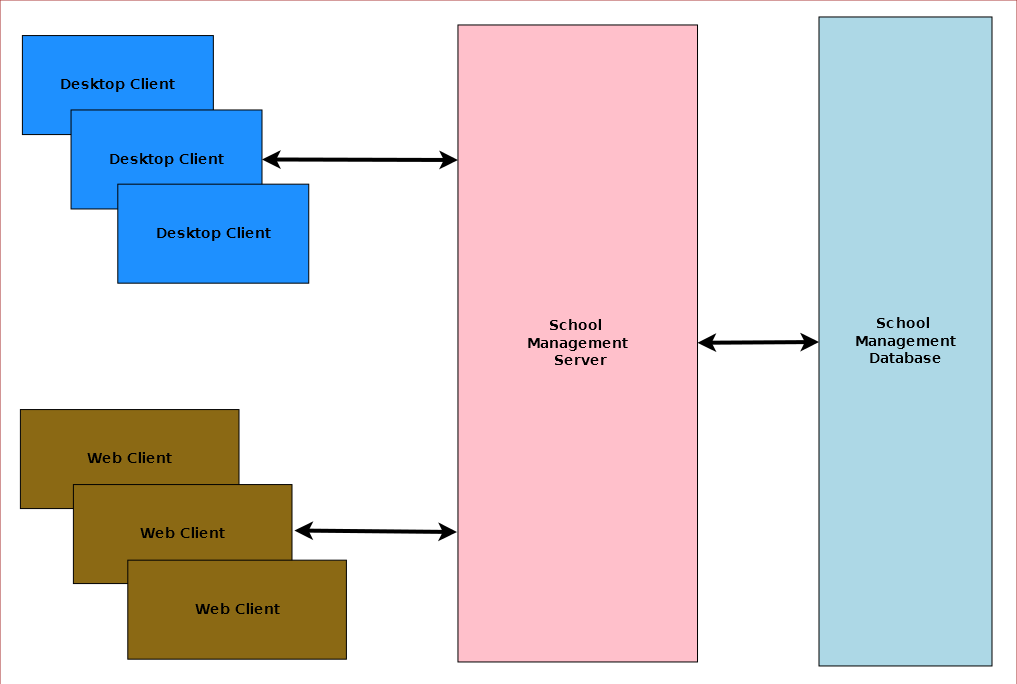


### CLASS DIAGRAM



# SYSTEM DESIGN

## MODULARISATION DETAILS



School Management System is divided three main modules such as:

1. School Management Server
2. School Management Client
3. School Management Database

### SCHOOL MANAGEMENT SERVER

School Management server is a singleton server designed provide services for school management system. It controls various activities required for the school management system. To manage these activities it has several sub modules such as:

1. Admission Management
2. Student Management
3. Faculty Management
4. Course Management
5. Attendance & Leave Management
6. Library Management
7. Accounts Management
8. Administration Management

### SCHOOL MANAGEMENT CLIENT

School Management System will provide two different clients for the convenience of the user. Desktop client is for doing bulk activities and faster tasks. Web client will allow instant access from anywhere and anytime.

### SCHOOL MANAGEMENT DATABASE

School Management System will have a unified database for storing all the information. It can be a networked database or a database situated in the server machine.

## 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

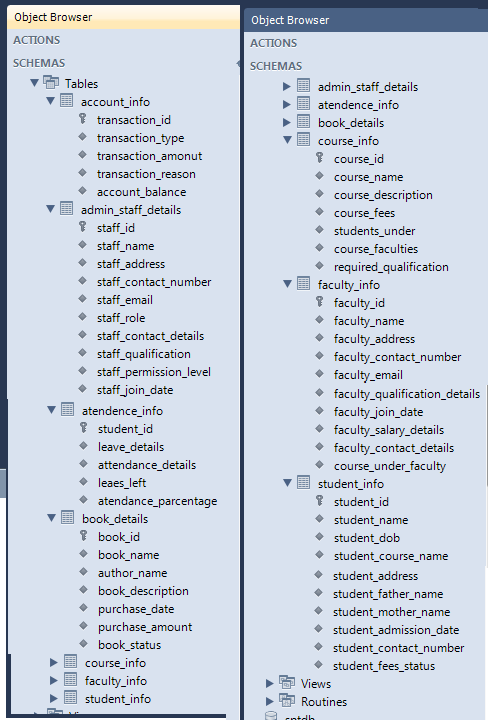
**SMS** has various type of data field with set by default value of Null because if the value is not provided by the user, the vale will be set as null. The domain integrity states that every element from a relation should respect the type and restrictions of its corresponding attribute. A type can have 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 AND TABLE DESIGN

The database used for this software is called **smsdb**. A screenshot from the MySQl workbench is given below. It shows the tables and its columns. The first row is the primary key.



## PROCEDURAL DESIGN / OBJECT ORIENTED DESIGN

## USER INTERFACE DESIGN

### Desktop Application User Interface

#### For Windows(C#.net)

##### Main window



##### Login window



##### After Login



##### Add Teacher



##### View Teacher



##### Add STUDENT



##### View STUDENT



##### Submit Admission form

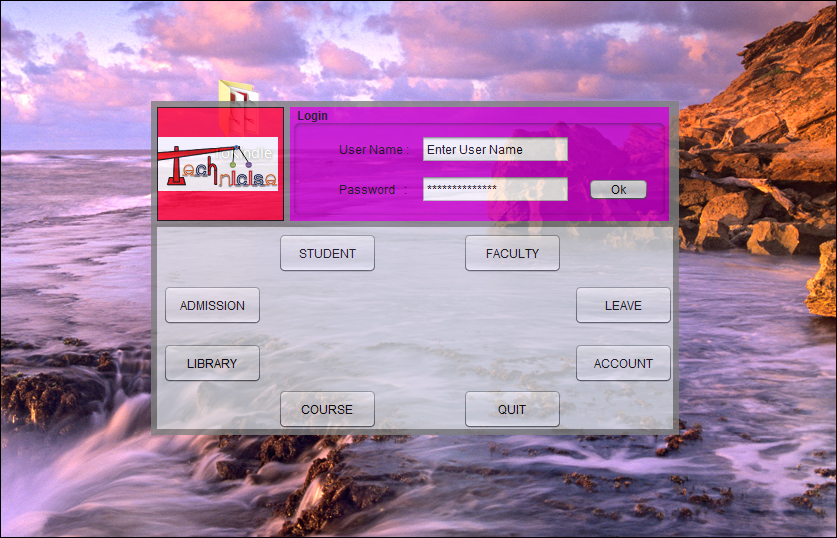


##### View Admission form

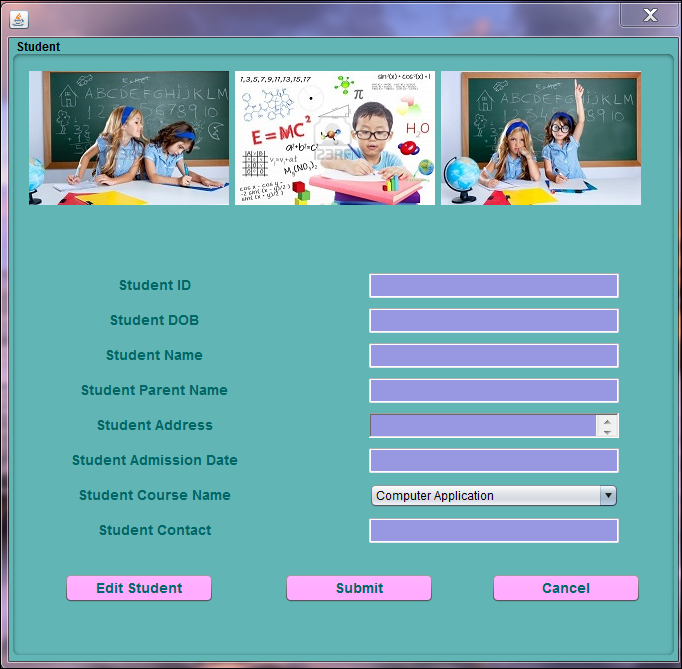


#### For LINUX (java)

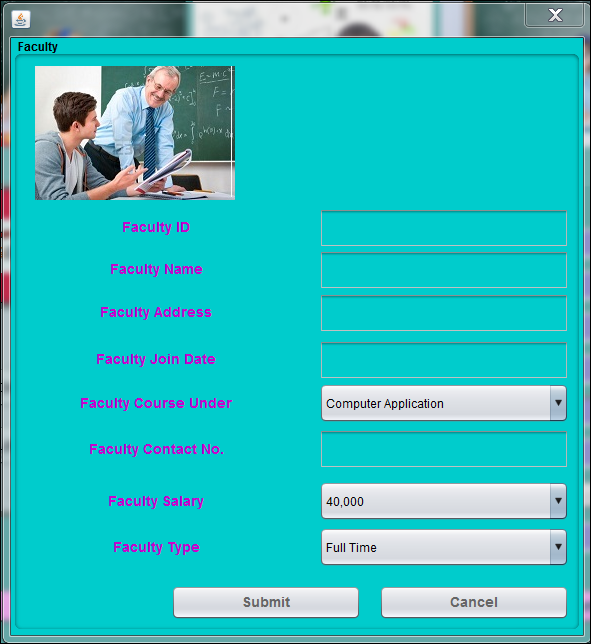
##### Main window



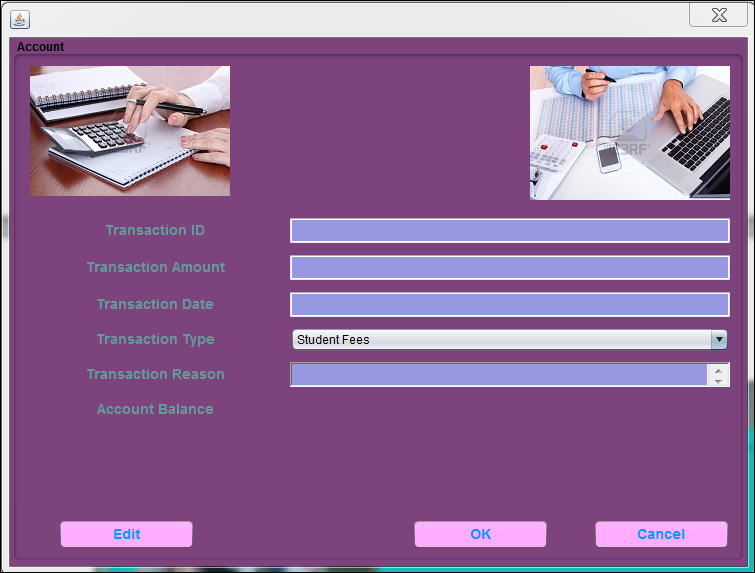
##### Student Window



##### Faculty Window

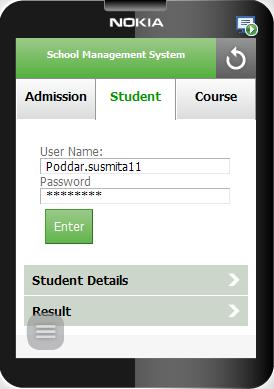


##### Account Window

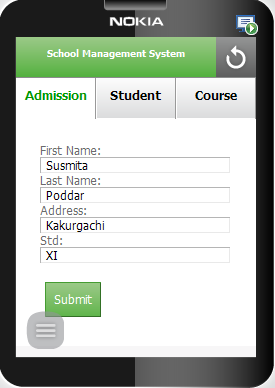


### Mobile App Interface

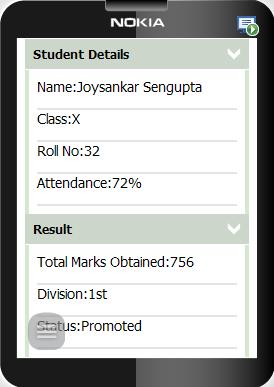
##### Mainwindow



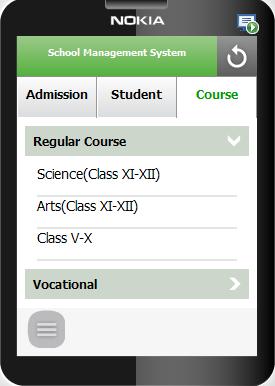
##### Admission Window:

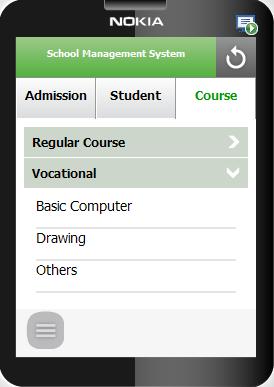


##### Student Window

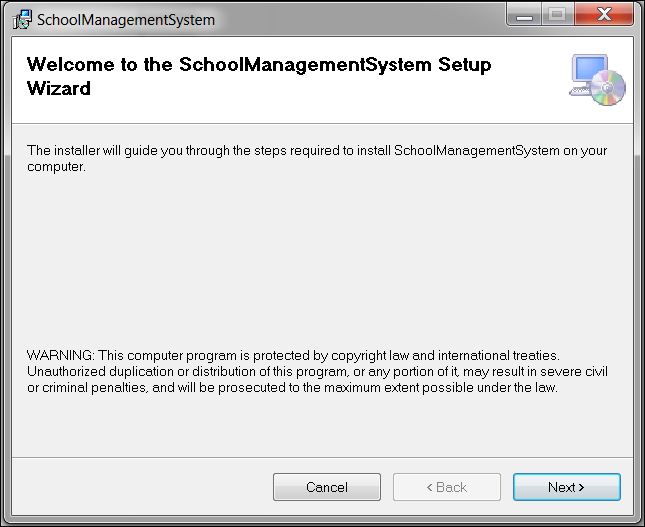


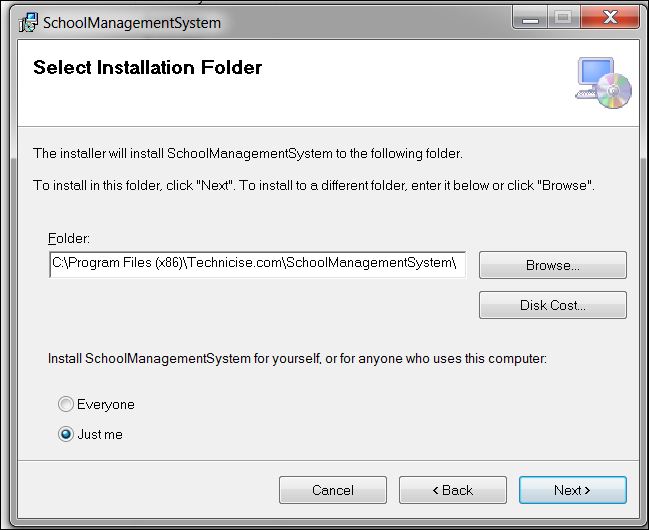
##### Course Details

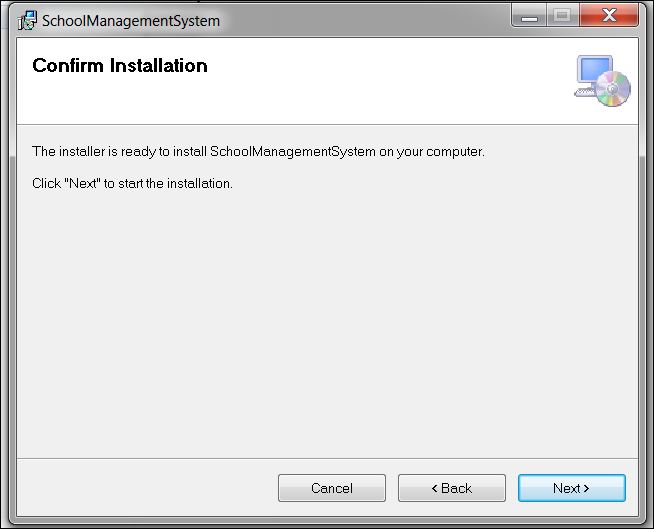


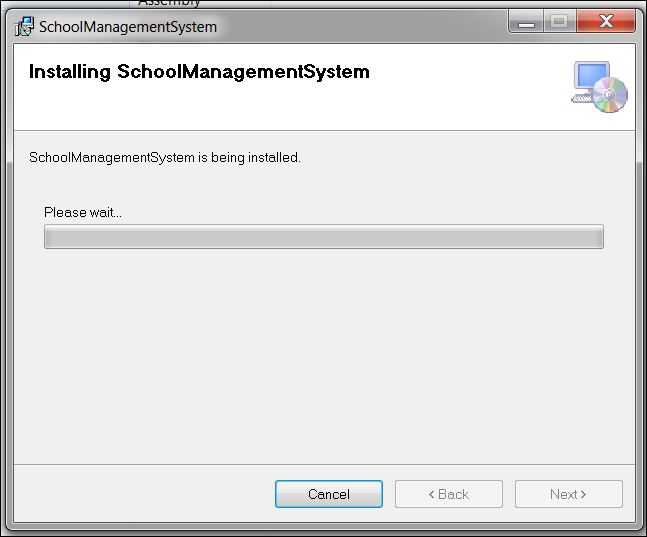


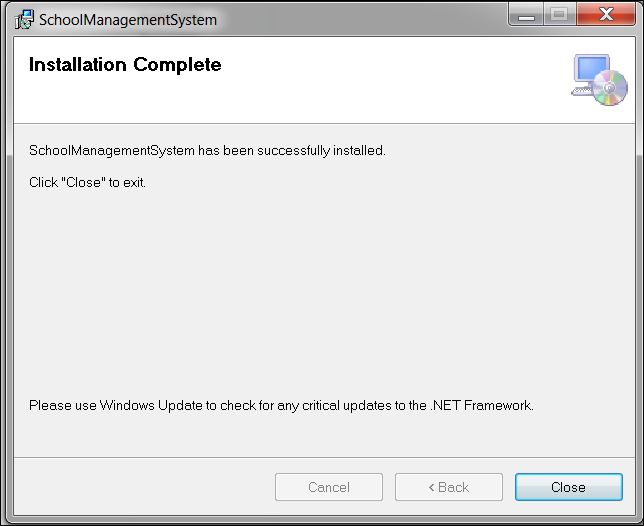
##### Installer ScreenShot:











## TEST CASES

### UNIT TEST CASES

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TEST CASE ID | TEST NAME | TEST ITEM | TEST DESCRIPTION | STEP NAME | DESCRIPTION | EXPECTED RESULT | TESTED BY |
| SMS – 001 | Check Successful Login for SMS. | Login | The  purpose  of this  test is to  verify that  the User  Id and  Password  of user is  Valid. | Step1 | Insert wrong User  Id and Password.  And Click on Login  Button. | SMS will  display error message. And  Failed to Login. | Susmita |
| SMS – 002 |  |  |  | Step 2 | Insert Wrong  User Id and valid  Password. And  Click on Login  Button. | SMS will display error message. And  Failed to Login. | Susmita |
| SMS – 003 |  |  |  | Step 3 | Insert Valid User  Id and Wrong  Password. And  Click on Login  Button. | SMS will  display error message. And  Failed to Login. | Susmita |
| SMS – 004 |  |  |  | Step 4 | Insert Nothing  in User Id and  Password fields.  And Click on Login  Button. | SMS will  display error  message. And  Failed to Login. | Susmita |
| SMS – 005 |  |  |  | Step 5 | Insert Nothing in  User Id and insert  Valid Password  fields. And Click  on Login Button. | SMS will  display error  message. And  Failed to Login. | Susmita |
| SMS – 006 |  |  |  | Step 6 | Insert Nothing  in Password and  insert Valid User  Id fields. And Click  on Login Button. | SMS will  display error  message. And  Failed to Login. | Susmita |
| SMS – 007 |  |  |  | Step 7 | Insert Nothing in  User Id and insert  invalid Password  fields. And Click  on Login Button. | SMS will  display error  message. And  Failed to Login. | Susmita |

### SYSTEM TEST CASES

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TEST CASE ID | TEST NAME | TEST ITEM | TEST DESCRIPTION | STEP NAME | DESCRIPTION | EXPECTED RESULT | TESTED BY |
| SMS – 001 | Check Login | Login | It is to check that Login works properly. | Step1 | Click on Login  button after  inserting invalid  User id and  password from  DNBSN. | Login failed  to SMS.  And can’t able  to use the  feature. | Susmita |
| SMS – 002 |  |  |  | Step 2 | Click on Login  button after  inserting valid  User id and  password from  SMS. | Successfully  Login to  SMS. And  can able to use  the feature. | Susmita |
| SMS – 003 | Check  Successful  Registration  for New  SMS  Student. | Admission | The  purpose  of this test  is to verify  that the  all new  connection could be  creating  new  Account  By  Registration. | Step 1 | Click on  Admission button. | New Admission window opened. | Susmita |
| SMS – 004 |  | Submit |  | Step 2 | Click on  Submit  button after  inserting invalid  information from  SMS. | Registration  failed to  SMS. And  can’t able to use the feature. | Susmita |
| SMS – 005 |  |  |  | Step 3 | Click on  Submit  button after  inserting valid  information from  SMS. | Registration  Successfully  done to  SMS. | Susmita |
| SMS – 006 |  |  |  | Step 4 | Click on Login  button after  inserting newly  created valid User  id and password from SMS. | Successfully  Login to by  new User Id  And password  SMS. And can able to use the feature. | Susmita |
| SMS – 007 | Check  Successful  Registration  for New  SMS  Faculty. | Faculty | The  purpose  of this test  is to verify  that the  all new  connection could be  creating  new  Account  By  Registration. | Step 1 | Click on  Faculty button. | New Faculty window opened. | Susmita |
| SMS – 008 |  | Submit |  | Step 2 | Click on  Submit  button after  inserting invalid  information of Faculty from  SMS. | Registration  failed to  SMS. And  can’t able to use the feature. | Susmita |
| SMS – 009 |  |  |  | Step 3 | Click on  Submit  button after  inserting valid  information of Faculty from  SMS. | Registration  Successfully  done to  SMS. | Susmita |
| SMS – 010 |  |  |  | Step 4 | Click on Login  button after  inserting newly  created valid User  id and password from SMS. | Successfully  Login to by  new User Id  And password  SMS. And can able to use the feature. | Susmita |
| SMS – 011 | Check  for New  SMS  Course | Course | The  purpose  of this test  is to verify  that all the  registered student has to be under a Course  By  Registration. | Step 1 | Click on  Course button. | New Course window opened. | Susmita |
| SMS – 012 |  | Submit |  | Step 2 | Click on  Submit  button after  inserting invalid  information of Course from  SMS. | Registration  failed to  SMS. And  can’t able to use the feature. | Susmita |
| SMS – 013 |  | EditCourse |  | Step 3 | Click on EditCourse Button to edit the invalid information. And then press the Submit button. | Successfully edited the invalid Course information. Registration under Course can be done. | Susmita |
| SMS – 014 |  |  |  | Step 4 | Click on  Submit  button after  inserting valid  information of Course from  SMS. | Successfully added the details of the Course into SMS. | Susmita |
| SMS – 015 | Check for registered Student | Student | The  purpose  of this test  is to verify  that all the  registered student information successfully updated to SMS. | Step 1 | Click Student to open Student window. | Successfully opened Student window. | Susmita |
| SMS – 016 |  | Submit |  | Step 2 | Click on  Submit  button after  inserting invalid  information of Student from  SMS. | Can’t able to update the details of the Student for future use. | Susmita |
| SMS – 017 |  | EditStudent |  | Step 3 | Click on EditStudent Button to edit the invalid information of the Student. And then press the Submit button. | Successfully edited the invalid Student information. Can able to update the details of the Student for future use. | Susmita |
| SMS – 018 |  |  |  | Step 4 | Click on  Submit  button after  inserting valid  information of Student from  SMS. | Successfully added the details of the Student into SMS. | Susmita |
| SMS – 019 | Check for Books issuing for Student in Library | Library | The  purpose  of this test  is to verify  for the valid information of the Book registered into Library in SMS. | Step 1 | Click Library to open Library window. | Successfully opened Library window. | Susmita |
| SMS – 020 |  | Submit |  | Step 2 | Click on  Submit  button after  inserting invalid  information of Books from  SMS. | Can’t able to issue Books for the Student. | Susmita |
| SMS – 021 |  | EditBook |  | Step 3 | Click on EditBook Button to edit the invalid information of the Books. And then press the Submit button. | Successfully added the details of the Book into SMS. | Susmita |
| SMS – 022 |  |  |  | Step 4 | Click on  Submit  button after  inserting valid  information of Book from  SMS. | Successfully added the details of the Book into SMS. Can able to issue Books for the Student. | Susmita |
| SMS – 023 | Check for Transaction in various way in Accounts. | Account | The  purpose  of this test  is to verify  that every transaction is inserting in SMS. | Step 1 | Click Account to open Account window. | Successfully opened Account window. | Susmita |
| SMS – 024 |  | OK |  | Step 2 | Click on  OK  button after  inserting invalid  Transaction details from  SMS. | Can’t able to insert Transaction into SMS. | Susmita |
| SMS – 025 |  | Edit |  | Step 3 | Click on Edit Button to edit the invalid information of the Transaction. And then press the OK button. | Successfully added the details of the Transaction into SMS. | Susmita |
| SMS – 026 |  |  |  | Step 4 | Click on  OK  button after  inserting valid  information of Transaction from  SMS. | Successfully added the details of the Transaction into SMS. Can able to use Accounts in future. | Susmita |
| SMS – 027 | Check for Leaves made by Student in School. | Leave | The  purpose  of this test  is to check weather Student has a required Attendance in School or not. | Step 1 | Click Leave to open Leave window. | Successfully opened Leave window. | Susmita |
| SMS – 028 |  | Submit |  | Step 2 | Click on Submit  button after  inserting invalid  information of the Student from  SMS. | Can’t able to see Attendance information of the Student from SMS. | Susmita |
| SMS – 029 |  | Edit |  | Step 3 | Click on Edit Button to edit the invalid information of the Student. And then press the Submit button. | Successfully shows the attendance details of the Student from SMS. | Susmita |
| SMS – 030 |  |  |  | Step 4 | Click on  Submit  button after  inserting valid  information of Student from  SMS. | Successfully shows the attendance details of the Student from SMS. Can able to calculate the Percentage Attendance for allowing into Exam. | Susmita |
| SMS – 031 | Check Login Student from Mobile. | Student | The purpose of this test is that to Login Student in SMS any time from Mobile. | Step 1 | Enter user id and password of the student. Click on Enter  Button after inserting invalid user id and password of the Student for SMS in Mobile. | Can’t able to Login into SMS on Mobile. | Susmita |
| SMS – 032 |  |  |  | Step 2 | Enter user id and password of the student. Click on Enter  Button after inserting valid user id and password of the Student for SMS in Mobile. | Can able to Login into SMS on Mobile. | Susmita |
| SMS – 033 | Check Student Status from Mobile. | Status | The purpose of this test is that to show the status of the Student any time from Mobile. | Step 1 | Enter Student ID, Student Name, Student Course Name of the Student. Click on Enter  button after  inserting invalid  information of the Student from  SMS on Mobile. | Can’t able to show the status of the Student from Mobile. | Susmita |
| SMS – 034 |  |  |  | Step 2 | Enter Student ID, Student Name, Student Course Name of the Student. Click on Enter  button after  inserting valid  information of the Student from  SMS on Mobile. | Can able to show the status of the Student from Mobile | Susmita |
| SMS – 035 | Check Student Result from Mobile. | Result | The purpose of this test is that to show the Result of the Student any time from Mobile. | Step 1 | Click on Result  button after  inserting invalid  information of the Student from  SMS on Mobile. | Can’t able to show the Result of the Student from Mobile | Susmita |
| SMS – 036 |  |  |  | Step 2 | Click on Enter  button after  inserting valid  information of the Student from  SMS on Mobile. | Can able to show the Result of the Student from Mobile | Susmita |

# CODING

## COMPLETE PROJECT CODING

### Desktop application Coding:

#### C#.net

|  |
| --- |
| MainWindow.xaml |
| <Window x:Class="SMSUI.MainWindow"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  Title="School Management System" MinHeight="560" MinWidth="1000" WindowState="Maximized" Background="#FF6B736D" xmlns:my="clr-namespace:SMSUI">  <Window.Resources>  <ResourceDictionary>  <ResourceDictionary.MergedDictionaries>  <ResourceDictionary Source="/SmsStyle;component/ControlStyle.xaml" />  </ResourceDictionary.MergedDictionaries>  </ResourceDictionary>  </Window.Resources>  <DockPanel LastChildFill="True">    <Image DockPanel.Dock="Top" Name="companyNameImage" StretchDirection="Both" Stretch="Uniform" HorizontalAlignment="Stretch" VerticalAlignment="Top" Source="/MmmUI;component/Images/name.jpg" />  <UniformGrid Width="300" DockPanel.Dock="Left" Columns="1">  <Button Content="Sign In" Name="loginBtn" Style="{StaticResource MainBtnStyle}" ></Button>    <Button Content="Teachers" Name="TeacherBtn" Style="{StaticResource MainBtnStyle}" Click="TeacherBtn\_Click" IsEnabled="False"></Button>  <Button Content="Students" Name="StudentsBtn" Style="{StaticResource MainBtnStyle}" Click="StudentsBtn\_Click" IsEnabled="False"></Button>  <Button Content="Admission" Name="admissionBtn" Style="{StaticResource MainBtnStyle}" Click="admissionBtn\_Click" IsEnabled="False"></Button>  <Button Content="Attendance" Name="AttendanceBtn" Style="{StaticResource MainBtnStyle}" Click="Attendance\_Click" IsEnabled="False"></Button>  <Button Content="Settings" Name="settingsBtn" Style="{StaticResource MainBtnStyle}" Click="settingsBtn\_Click" IsEnabled="False"></Button>  <Button Content="Notice" Name="noticeBtn" Style="{StaticResource MainBtnStyle}" Click="noticeBtn\_Click" IsEnabled="False"></Button>  <Button Content="Help" Name="helpBtn" Style="{StaticResource MainBtnStyle}" Click="helpBtn\_Click" IsEnabled="False"></Button>  <Button Content="Sign Out" Name="logoutBtn" Style="{StaticResource MainBtnStyle}" Visibility="Collapsed" Click="logoutBtn\_Click"></Button>  </UniformGrid>  <UniformGrid DockPanel.Dock="Right" Name="mainUniGrid" Background="#FFC08E5D">  <DockPanel LastChildFill="True" Name="loginDP">  <Image DockPanel.Dock="Top" Name="schoolImage" StretchDirection="Both" Stretch="Uniform" HorizontalAlignment="Stretch" VerticalAlignment="Top" Source="/MmmUI;component/Images/banner1.jpg" Height="340" Width="1000"/>  <UniformGrid DockPanel.Dock="Bottom" Columns="4">  <Label></Label>  <Label Style="{StaticResource labelStyle}" >User Name</Label>  <TextBox Height="40" Name="userNameTxtbox" Style="{StaticResource textboxStyle}" ></TextBox>  <Label></Label>  <Label></Label>  <Label Style="{StaticResource labelStyle}" >User Type</Label>  <ComboBox Height="40" Name="userTypeCombobox" IsEditable="True" SelectedIndex="2" VerticalContentAlignment="Center" HorizontalContentAlignment="Left" >  <ComboBoxItem>Admin</ComboBoxItem>  <ComboBoxItem>Faculty</ComboBoxItem>  <ComboBoxItem>Stuff</ComboBoxItem>  </ComboBox>  <Label></Label>  <Label></Label>  <Label Style="{StaticResource labelStyle}">Enter Password</Label>  <PasswordBox Height="40" Name="passwordBox" Style="{StaticResource passwordboxStyle}"></PasswordBox>  <Label></Label>  <Label></Label>  <Button Name="resetBtn" Style="{StaticResource MainBtnStyle}" Margin="2">Reset</Button>  <Button Name="dologinBtn" Style="{StaticResource MainBtnStyle}" Margin="2" Click="dologinBtn\_Click">Sign In</Button>  </UniformGrid>  </DockPanel>  </UniformGrid>  <!--<TabControl Name="controlsTab" DockPanel.Dock="Bottom" TabStripPlacement="Bottom" Background="#37000000" >  <TabItem Name="homeTab" >  <Image ></Image>  </TabItem>  <TabItem Name="loginTab">  <UniformGrid Columns="2">  <Label >User Name</Label>  <TextBox Name="userNameTxtbox" ></TextBox>  <Label >User Type</Label>  <ComboBox Name="userTypeCombobox" >  <ComboBoxItem>User</ComboBoxItem>  <ComboBoxItem>Manager</ComboBoxItem>  <ComboBoxItem>Owner</ComboBoxItem>  </ComboBox>  <Label >Enter Password</Label>  <PasswordBox Name="passwordBox" Margin="10" Height="60" VerticalContentAlignment="Center"></PasswordBox>  <Button Name="dologoutBtn" IsEnabled="False" Margin="10" >Sign Out</Button>  <Button Name="dologinBtn" Margin="10" >Sign In</Button>  </UniformGrid>  </TabItem>  <TabItem Name="empTab">  <UniformGrid>  <Button Name="addEmployeeBtn" Content="Add Employee" ></Button>  <Button Name="viewEmployeeBtn" >View Employee</Button>  <Button Name="editEmployeeBtn" >Edit Employee</Button>  <Button Name="closeEmpBtn" >Close</Button>  </UniformGrid>  </TabItem>  <TabItem Name="StudentssTab">  <UniformGrid>  <Button Name="addStudentssBtn" >Add Studentss</Button>  <Button Name="viewStudentssBtn" >View Studentss</Button>  <Button Name="editStudentssBtn" >Edit Studentss</Button>  <Button Name="closeStudentssBtn" >Close</Button>  </UniformGrid>  </TabItem>  <TabItem Name="clientsTab">  <UniformGrid>  <Button Name="addClientsBtn" >Add Clients</Button>  <Button Name="viewClientsBtn" >View Clients</Button>  <Button Name="editClientsBtn" >Edit Clients</Button>  <Button Name="closeClientsBtn" >Close</Button>  </UniformGrid>  </TabItem>  </TabControl>-->  </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 SMSUI  {  /// <summary>  /// Interaction logic for MainWindow.xaml  /// </summary>  public partial class MainWindow : Window  {  public MainWindow()  {  InitializeComponent();  }      private void TeacherBtn\_Click(object sender, RoutedEventArgs e)  {  SMSUI.Teacher Teacherobj = new SMSUI.Teacher();  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(Teacherobj);    }  private void StudentsBtn\_Click(object sender, RoutedEventArgs e)  {  SMSUI.Student Studentobj = new SMSUI.Student();  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(Studentobj);  }  private void Attendance\_Click(object sender, RoutedEventArgs e)  {  SMSUI.Attendance Attendanceobj = new SMSUI.Attendance();  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(Attendanceobj);  }  private void admissionBtn\_Click(object sender, RoutedEventArgs e)  {  SMSUI.AdmissionForm AdmissionFormobj = new SMSUI.AdmissionForm();  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(AdmissionFormobj);    }  private void settingsBtn\_Click(object sender, RoutedEventArgs e)  {  SMSUI.Settings Settingsobj = new SMSUI.Settings();  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(Settingsobj);  }  private void helpBtn\_Click(object sender, RoutedEventArgs e)  {  SMSUI.Help Helpobj = new SMSUI.Help();  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(Helpobj);  }  private void closeBtn\_Click(object sender, RoutedEventArgs e)  {  this.Close();  }  private void noticeBtn\_Click(object sender, RoutedEventArgs e)  {  SMSUI.Notice Noticeobj = new SMSUI.Notice();  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(Noticeobj);  }  private void dologinBtn\_Click(object sender, RoutedEventArgs e)  {  if (userNameTxtbox.Text.Equals("1") && userNameTxtbox.Text.Equals("1"))  {  TeacherBtn.IsEnabled = true;  StudentsBtn.IsEnabled = true;  admissionBtn.IsEnabled = true;  AttendanceBtn.IsEnabled = true;  settingsBtn.IsEnabled = true;  noticeBtn.IsEnabled = true;  helpBtn.IsEnabled = true;  logoutBtn.Visibility = Visibility.Visible;  loginBtn.Visibility = Visibility.Collapsed;  SMSUI.Attendance Attendanceobj = new SMSUI.Attendance();  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(Attendanceobj);  userNameTxtbox.Text = "";  userNameTxtbox.Text ="";  }  else  MessageBox.Show("Enter correct Info");  }  private void logoutBtn\_Click(object sender, RoutedEventArgs e)  {  TeacherBtn.IsEnabled = false;  StudentsBtn.IsEnabled = false;  admissionBtn.IsEnabled = false;  AttendanceBtn.IsEnabled = false;  settingsBtn.IsEnabled = false;  noticeBtn.IsEnabled = false;  helpBtn.IsEnabled = false;  loginBtn.Visibility = Visibility.Visible;  logoutBtn.Visibility = Visibility.Collapsed;  mainUniGrid.Children.Clear();  mainUniGrid.Children.Add(loginDP);  }  }  } |

#### java

|  |
| --- |
| SmsGUI.java |
| /\*//GEN-FIRST:event\_admBtnActionPerformed  \* To change this template, choose Tools | Templates//GEN-LAST:event\_admBtnActionPerformed  \* and open the template in the editor.  \*/  package my.sms;  import java.awt.Color;  import java.awt.GraphicsDevice;  import static java.awt.GraphicsDevice.WindowTranslucency.TRANSLUCENT;  import java.awt.GraphicsEnvironment;  import java.awt.GridBagLayout;  import javax.swing.JDialog;  import javax.swing.JOptionPane;  /\*\*  \*  \* @author ibm  \*/  public class SmsGUI extends javax.swing.JFrame {  /\*\*  \* Creates new form SmsGUI  \*/  public SmsGUI() {  super("SmsGUI");  setLayout(new GridBagLayout());  setSize(300,200);  setLocationRelativeTo(null);  setUndecorated(true);  getContentPane().setBackground(Color.GRAY);  initComponents();  }  /\*\*  \* This method is called from within the constructor to initialize the form.  \* WARNING: Do NOT modify this code. The content of this method is always  \* regenerated by the Form Editor.  \*/  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">  private void initComponents() {    jLabel1 = new javax.swing.JLabel();  loginPanel = new javax.swing.JPanel();  jLabel3 = new javax.swing.JLabel();  jTextField1 = new javax.swing.JTextField();  jLabel4 = new javax.swing.JLabel();  jPasswordField1 = new javax.swing.JPasswordField();  jButton1 = new javax.swing.JButton();  logoPanel = new javax.swing.JPanel();  jLabel2 = new javax.swing.JLabel();  activityPanel = new javax.swing.JPanel();  stdtBtn = new javax.swing.JButton();  fclBtn = new javax.swing.JButton();  crsBtn = new javax.swing.JButton();  exitBtn = new javax.swing.JButton();  admBtn = new javax.swing.JButton();  libBtn = new javax.swing.JButton();  lveBtn = new javax.swing.JButton();  accBtn = new javax.swing.JButton();  java.util.ResourceBundle bundle = java.util.ResourceBundle.getBundle("my/sms/Bundle"); // NOI18N  jLabel1.setText(bundle.getString("SmsGUI.jLabel1.text")); // NOI18N  setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  loginPanel.setBackground(new java.awt.Color(204, 0, 204));  loginPanel.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.getString("SmsGUI.loginPanel.border.title"))); // NOI18N  jLabel3.setText(bundle.getString("SmsGUI.jLabel3.text")); // NOI18N  jTextField1.setText(bundle.getString("SmsGUI.jTextField1.text")); // NOI18N  jTextField1.addActionListener(new java.awt.event.ActionListener() {  @Override  public void actionPerformed(java.awt.event.ActionEvent evt) {  jTextField1ActionPerformed(evt);  }  });  jLabel4.setText(bundle.getString("SmsGUI.jLabel4.text")); // NOI18N  jPasswordField1.setText(bundle.getString("SmsGUI.jPasswordField1.text")); // NOI18N  jButton1.setText(bundle.getString("SmsGUI.jButton1.text")); // NOI18N  org.jdesktop.layout.GroupLayout loginPanelLayout = new org.jdesktop.layout.GroupLayout(loginPanel);  loginPanel.setLayout(loginPanelLayout);  loginPanelLayout.setHorizontalGroup(  loginPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(loginPanelLayout.createSequentialGroup()  .add(35, 35, 35)  .add(loginPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)  .add(jLabel3, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .add(jLabel4, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  .addPreferredGap(org.jdesktop.layout.LayoutStyle.UNRELATED)  .add(loginPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)  .add(jTextField1)  .add(jPasswordField1, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, 149, Short.MAX\_VALUE))  .add(18, 18, 18)  .add(jButton1, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, 61, Short.MAX\_VALUE)  .addContainerGap())  );  loginPanelLayout.setVerticalGroup(  loginPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(loginPanelLayout.createSequentialGroup()  .add(loginPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)  .add(jTextField1, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .add(jLabel3))  .addPreferredGap(org.jdesktop.layout.LayoutStyle.UNRELATED)  .add(loginPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)  .add(jLabel4)  .add(jPasswordField1, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .add(jButton1, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 23, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE))  .add(0, 4, Short.MAX\_VALUE))  );  logoPanel.setBackground(new java.awt.Color(255, 0, 51));  logoPanel.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)));  logoPanel.setPreferredSize(new java.awt.Dimension(127, 54));  jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/my/sms/technicise.logo.png"))); // NOI18N  jLabel2.setText(bundle.getString("SmsGUI.jLabel2.text")); // NOI18N  jLabel2.setMaximumSize(new java.awt.Dimension(127, 54));  org.jdesktop.layout.GroupLayout logoPanelLayout = new org.jdesktop.layout.GroupLayout(logoPanel);  logoPanel.setLayout(logoPanelLayout);  logoPanelLayout.setHorizontalGroup(  logoPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(logoPanelLayout.createSequentialGroup()  .add(jLabel2, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 126, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .add(0, 0, Short.MAX\_VALUE))  );  logoPanelLayout.setVerticalGroup(  logoPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(jLabel2, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  );  stdtBtn.setText(bundle.getString("SmsGUI.stdtBtn.text")); // NOI18N  stdtBtn.addActionListener(new java.awt.event.ActionListener() {  @Override  public void actionPerformed(java.awt.event.ActionEvent evt) {  stdtBtnActionPerformed(evt);  }  });  fclBtn.setText(bundle.getString("SmsGUI.fclBtn.text")); // NOI18N  fclBtn.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  fclBtnActionPerformed(evt);  }  });  crsBtn.setText(bundle.getString("SmsGUI.crsBtn.text")); // NOI18N  crsBtn.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  crsBtnActionPerformed(evt);  }  });  exitBtn.setText(bundle.getString("SmsGUI.exitBtn.text")); // NOI18N  exitBtn.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  exitBtnActionPerformed(evt);  }  });  admBtn.setText(bundle.getString("SmsGUI.admBtn.text")); // NOI18N  admBtn.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  admBtnActionPerformed(evt);  }  });  libBtn.setText(bundle.getString("SmsGUI.libBtn.text")); // NOI18N  libBtn.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  libBtnActionPerformed(evt);  }  });  lveBtn.setText(bundle.getString("SmsGUI.lveBtn.text")); // NOI18N  lveBtn.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  lveBtnActionPerformed(evt);  }  });  accBtn.setText(bundle.getString("SmsGUI.accBtn.text")); // NOI18N  accBtn.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  accBtnActionPerformed(evt);  }  });  org.jdesktop.layout.GroupLayout activityPanelLayout = new org.jdesktop.layout.GroupLayout(activityPanel);  activityPanel.setLayout(activityPanelLayout);  activityPanelLayout.setHorizontalGroup(  activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(activityPanelLayout.createSequentialGroup()  .addContainerGap()  .add(activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(activityPanelLayout.createSequentialGroup()  .add(libBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 99, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .add(accBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 99, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE))  .add(activityPanelLayout.createSequentialGroup()  .add(admBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 99, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .add(lveBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 99, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE))))  .add(org.jdesktop.layout.GroupLayout.TRAILING, activityPanelLayout.createSequentialGroup()  .add(121, 121, 121)  .add(activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)  .add(stdtBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 99, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .add(crsBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 99, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE))  .add(18, 18, Short.MAX\_VALUE)  .add(activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)  .add(exitBtn, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, 99, Short.MAX\_VALUE)  .add(fclBtn, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  .add(111, 111, 111))  );  activityPanelLayout.setVerticalGroup(  activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(activityPanelLayout.createSequentialGroup()  .addContainerGap()  .add(activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)  .add(stdtBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 40, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .add(fclBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 40, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE))  .addPreferredGap(org.jdesktop.layout.LayoutStyle.UNRELATED)  .add(activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)  .add(admBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 40, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .add(lveBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 40, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE))  .add(18, 18, 18)  .add(activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)  .add(libBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 40, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .add(accBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 40, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE))  .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .add(activityPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)  .add(crsBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 40, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .add(exitBtn, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, 40, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)))  );  org.jdesktop.layout.GroupLayout layout = new org.jdesktop.layout.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(org.jdesktop.layout.GroupLayout.TRAILING, layout.createSequentialGroup()  .addContainerGap()  .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)  .add(activityPanel, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .add(layout.createSequentialGroup()  .add(logoPanel, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)  .add(loginPanel, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .add(4, 4, 4)))  .addContainerGap())  );  layout.setVerticalGroup(  layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)  .add(layout.createSequentialGroup()  .addContainerGap()  .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)  .add(loginPanel, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .add(logoPanel, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, 56, Short.MAX\_VALUE))  .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)  .add(activityPanel, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .addContainerGap())  );  pack();  }// </editor-fold>  private void stdtBtnActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  final JDialog bwin = new JDialog();    StudentWindow stdWindow = new StudentWindow();    bwin.add(stdWindow);  bwin.pack();  bwin.setVisible(true);  }  private void crsBtnActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  final JDialog bwin = new JDialog();    CourseWindow crsWindow = new CourseWindow();    bwin.add(crsWindow);  bwin.pack();  bwin.setVisible(true);  }  private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {  System.exit(1);  }  private void admBtnActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  final JDialog bwin = new JDialog();    AdmissionWindow admsnWindow = new AdmissionWindow();    bwin.add(admsnWindow);  bwin.pack();  bwin.setVisible(true);    }  private void libBtnActionPerformed(java.awt.event.ActionEvent evt) {    JOptionPane optionPane = new JOptionPane("Is this what you need?", JOptionPane.QUESTION\_MESSAGE,JOptionPane.YES\_NO\_OPTION);  JDialog dialog = optionPane.createDialog("Dialog");  dialog.setVisible(true);  }  private void lveBtnActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  final JDialog bwin = new JDialog();    LeaveWindow lveWindow = new LeaveWindow();    bwin.add(lveWindow);  bwin.pack();  bwin.setVisible(true);  }  private void accBtnActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  final JDialog bwin = new JDialog();    AccountWindow accWindow = new AccountWindow();    bwin.add(accWindow);  bwin.pack();  bwin.setVisible(true);  }  private void fclBtnActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  final JDialog bwin = new JDialog();    FacultyWindow fclWindow = new FacultyWindow();    bwin.add(fclWindow);  bwin.pack();  bwin.setVisible(true);  }  private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  }  /\*\*  \* @param args the command line arguments  \*/  public static void main(String args[]) {  /\* Set the Nimbus look and feel \*/  //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">  /\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.  \* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html  \*/  GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();  GraphicsDevice gd = ge.getDefaultScreenDevice();    if (!gd.isWindowTranslucencySupported(TRANSLUCENT)) {  System.err.println("Translucency is not supported");  System.exit(0);  }    try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException | InstantiationException | IllegalAccessException | javax.swing.UnsupportedLookAndFeelException ex) {  java.util.logging.Logger.getLogger(SmsGUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  //</editor-fold>  /\* Create and display the form \*/  java.awt.EventQueue.invokeLater(new Runnable() {  @Override  public void run() {  // new SmsGUI().setVisible(true);  SmsGUI tw = new SmsGUI();  tw.setOpacity(0.80f);  tw.setVisible(true);  }  });  }  // Variables declaration - do not modify  private javax.swing.JButton accBtn;  private javax.swing.JPanel activityPanel;  private javax.swing.JButton admBtn;  private javax.swing.JButton crsBtn;  private javax.swing.JButton exitBtn;  private javax.swing.JButton fclBtn;  private javax.swing.JButton jButton1;  private javax.swing.JLabel jLabel1;  private javax.swing.JLabel jLabel2;  private javax.swing.JLabel jLabel3;  private javax.swing.JLabel jLabel4;  private javax.swing.JPasswordField jPasswordField1;  private javax.swing.JTextField jTextField1;  private javax.swing.JButton libBtn;  private javax.swing.JPanel loginPanel;  private javax.swing.JPanel logoPanel;  private javax.swing.JButton lveBtn;  private javax.swing.JButton stdtBtn;  // End of variables declaration  } |
| AdmissionWindow.java |
| /\*  \* To change this template, choose Tools | Templates  \* and open the template in the editor.  \*/  package my.sms;  /\*\*  \*  \* @author PoYeS  \*/  public class AdmissionWindow extends javax.swing.JPanel {  /\*\*  \* Creates new form AdmissionWindow  \*/  public AdmissionWindow() {  initComponents();  }  /\*\*  \* This method is called from within the constructor to initialize the form.  \* WARNING: Do NOT modify this code. The content of this method is always  \* regenerated by the Form Editor.  \*/  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  private void initComponents() {  jLabel3 = new javax.swing.JLabel();  jLabel4 = new javax.swing.JLabel();  jLabel5 = new javax.swing.JLabel();  jLabel6 = new javax.swing.JLabel();  jTextField1 = new javax.swing.JTextField();  jTextField2 = new javax.swing.JTextField();  jTextField3 = new javax.swing.JTextField();  jTextField5 = new javax.swing.JTextField();  jTextField6 = new javax.swing.JTextField();  jTextField4 = new javax.swing.JTextField();  jButton1 = new javax.swing.JButton();  jButton2 = new javax.swing.JButton();  jLabel7 = new javax.swing.JLabel();  jLabel8 = new javax.swing.JLabel();  jLabel9 = new javax.swing.JLabel();  jLabel10 = new javax.swing.JLabel();  jLabel11 = new javax.swing.JLabel();  jLabel12 = new javax.swing.JLabel();  jComboBox1 = new javax.swing.JComboBox();  jComboBox2 = new javax.swing.JComboBox();  jTextField7 = new javax.swing.JTextField();  jTextField8 = new javax.swing.JTextField();  jTextField9 = new javax.swing.JTextField();  jComboBox3 = new javax.swing.JComboBox();  setBackground(new java.awt.Color(0, 204, 204));  setBorder(javax.swing.BorderFactory.createTitledBorder("Admission"));  setForeground(new java.awt.Color(204, 0, 255));  setToolTipText("");  setAlignmentX(0.5);  setAlignmentY(0.5);  setMaximumSize(new java.awt.Dimension(250, 50));  setOpaque(false);  jLabel3.setBackground(new java.awt.Color(0, 204, 204));  jLabel3.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel3.setForeground(new java.awt.Color(204, 0, 255));  jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel3.setText("Address");  jLabel3.setToolTipText("");  jLabel3.setAlignmentX(0.5);  jLabel3.setAlignmentY(0.5);  jLabel3.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel3.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel3.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel3.setPreferredSize(new java.awt.Dimension(300, 40));  jLabel4.setBackground(new java.awt.Color(0, 204, 204));  jLabel4.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel4.setForeground(new java.awt.Color(204, 0, 255));  jLabel4.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel4.setText("Contact Number");  jLabel4.setToolTipText("");  jLabel4.setAlignmentX(0.5);  jLabel4.setAlignmentY(0.5);  jLabel4.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel4.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel4.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel4.setPreferredSize(new java.awt.Dimension(300, 40));  jLabel5.setBackground(new java.awt.Color(0, 204, 204));  jLabel5.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel5.setForeground(new java.awt.Color(204, 0, 255));  jLabel5.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel5.setText("Father's Name");  jLabel5.setToolTipText("");  jLabel5.setAlignmentX(0.5);  jLabel5.setAlignmentY(0.5);  jLabel5.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel5.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel5.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel5.setPreferredSize(new java.awt.Dimension(300, 40));  jLabel6.setBackground(new java.awt.Color(0, 204, 204));  jLabel6.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel6.setForeground(new java.awt.Color(204, 0, 255));  jLabel6.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel6.setText("Mother's Name");  jLabel6.setToolTipText("");  jLabel6.setAlignmentX(0.5);  jLabel6.setAlignmentY(0.5);  jLabel6.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel6.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel6.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel6.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField1.setBackground(new java.awt.Color(0, 204, 204));  jTextField1.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField1.setForeground(new java.awt.Color(204, 0, 255));  jTextField1.setText("Enter Your Name");  jTextField1.setMaximumSize(new java.awt.Dimension(300, 40));  jTextField1.setMinimumSize(new java.awt.Dimension(300, 40));  jTextField1.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField1.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  jTextField1ActionPerformed(evt);  }  });  jTextField2.setBackground(new java.awt.Color(0, 204, 204));  jTextField2.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField2.setForeground(new java.awt.Color(204, 0, 255));  jTextField2.setText("Enter Guardian Name");  jTextField2.setToolTipText("");  jTextField2.setMaximumSize(new java.awt.Dimension(300, 50));  jTextField2.setMinimumSize(new java.awt.Dimension(300, 50));  jTextField2.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField3.setBackground(new java.awt.Color(0, 204, 204));  jTextField3.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField3.setForeground(new java.awt.Color(204, 0, 255));  jTextField3.setText("Provide Parmanent Address");  jTextField3.setToolTipText("");  jTextField3.setMaximumSize(new java.awt.Dimension(300, 50));  jTextField3.setMinimumSize(new java.awt.Dimension(300, 50));  jTextField3.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField5.setBackground(new java.awt.Color(0, 204, 204));  jTextField5.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField5.setForeground(new java.awt.Color(204, 0, 255));  jTextField5.setText("Last Examination Passed");  jTextField5.setToolTipText("");  jTextField5.setMaximumSize(new java.awt.Dimension(300, 50));  jTextField5.setMinimumSize(new java.awt.Dimension(300, 50));  jTextField5.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField6.setBackground(new java.awt.Color(0, 204, 204));  jTextField6.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField6.setForeground(new java.awt.Color(204, 0, 255));  jTextField6.setText("Other Details");  jTextField6.setToolTipText("");  jTextField6.setMaximumSize(new java.awt.Dimension(300, 50));  jTextField6.setMinimumSize(new java.awt.Dimension(300, 50));  jTextField6.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField4.setBackground(new java.awt.Color(0, 204, 204));  jTextField4.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField4.setForeground(new java.awt.Color(204, 0, 255));  jTextField4.setText("Contact Number");  jTextField4.setToolTipText("");  jTextField4.setMaximumSize(new java.awt.Dimension(300, 50));  jTextField4.setMinimumSize(new java.awt.Dimension(300, 50));  jTextField4.setPreferredSize(new java.awt.Dimension(300, 40));  jButton1.setBackground(new java.awt.Color(51, 51, 255));  jButton1.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jButton1.setForeground(new java.awt.Color(99, 51, 125));  jButton1.setText("Submit");  jButton2.setBackground(new java.awt.Color(51, 51, 255));  jButton2.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jButton2.setForeground(new java.awt.Color(99, 51, 125));  jButton2.setText("Reset");  jButton2.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  jButton2ActionPerformed(evt);  }  });  jLabel7.setBackground(new java.awt.Color(0, 204, 204));  jLabel7.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel7.setForeground(new java.awt.Color(204, 0, 255));  jLabel7.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel7.setText("Parent's Income Per Annum");  jLabel7.setToolTipText("");  jLabel7.setAlignmentX(0.5);  jLabel7.setAlignmentY(0.5);  jLabel7.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel7.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel7.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel7.setPreferredSize(new java.awt.Dimension(300, 40));  jLabel8.setBackground(new java.awt.Color(0, 204, 204));  jLabel8.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel8.setForeground(new java.awt.Color(204, 0, 255));  jLabel8.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel8.setText("Initial Amount for Registration ");  jLabel8.setToolTipText("");  jLabel8.setAlignmentX(0.5);  jLabel8.setAlignmentY(0.5);  jLabel8.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel8.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel8.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel8.setPreferredSize(new java.awt.Dimension(300, 40));  jLabel9.setBackground(new java.awt.Color(0, 204, 204));  jLabel9.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel9.setForeground(new java.awt.Color(204, 0, 255));  jLabel9.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel9.setText("Parent's Qualification");  jLabel9.setToolTipText("");  jLabel9.setAlignmentX(0.5);  jLabel9.setAlignmentY(0.5);  jLabel9.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel9.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel9.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel9.setPreferredSize(new java.awt.Dimension(300, 40));  jLabel10.setBackground(new java.awt.Color(0, 204, 204));  jLabel10.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel10.setForeground(new java.awt.Color(204, 0, 255));  jLabel10.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel10.setText("Form Submission Date");  jLabel10.setToolTipText("");  jLabel10.setAlignmentX(0.5);  jLabel10.setAlignmentY(0.5);  jLabel10.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel10.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel10.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel10.setPreferredSize(new java.awt.Dimension(300, 40));  jLabel11.setBackground(new java.awt.Color(0, 204, 204));  jLabel11.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel11.setForeground(new java.awt.Color(204, 0, 255));  jLabel11.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel11.setText("Student Assigned to Class");  jLabel11.setToolTipText("");  jLabel11.setAlignmentX(0.5);  jLabel11.setAlignmentY(0.5);  jLabel11.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel11.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel11.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel11.setPreferredSize(new java.awt.Dimension(300, 40));  jLabel12.setBackground(new java.awt.Color(0, 204, 204));  jLabel12.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jLabel12.setForeground(new java.awt.Color(204, 0, 255));  jLabel12.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel12.setText("Student DOB");  jLabel12.setToolTipText("");  jLabel12.setAlignmentX(0.5);  jLabel12.setAlignmentY(0.5);  jLabel12.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  jLabel12.setMaximumSize(new java.awt.Dimension(300, 40));  jLabel12.setMinimumSize(new java.awt.Dimension(300, 40));  jLabel12.setPreferredSize(new java.awt.Dimension(300, 40));  jComboBox1.setBackground(new java.awt.Color(0, 204, 204));  jComboBox1.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jComboBox1.setForeground(new java.awt.Color(204, 0, 255));  jComboBox1.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Below 2,00,000", "Below 3,00,000", "Below 4,00,000", "Below 5,00,000", "Below 6,00,000", "Below 7,00,000", "Below 8,00,000", "Below 9,00,000", "Above 10,00,000" }));  jComboBox1.setToolTipText("");  jComboBox1.setMaximumSize(new java.awt.Dimension(300, 50));  jComboBox1.setMinimumSize(new java.awt.Dimension(300, 50));  jComboBox1.setPreferredSize(new java.awt.Dimension(300, 40));  jComboBox2.setBackground(new java.awt.Color(0, 204, 204));  jComboBox2.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jComboBox2.setForeground(new java.awt.Color(204, 0, 255));  jComboBox2.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Under Graduate", "Graduate", "MA", "M.SC", "M.COM", "MCA", "MBA", "B.Tech", "M.Tech" }));  jComboBox2.setToolTipText("");  jComboBox2.setMaximumSize(new java.awt.Dimension(300, 50));  jComboBox2.setMinimumSize(new java.awt.Dimension(300, 50));  jComboBox2.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField7.setBackground(new java.awt.Color(0, 204, 204));  jTextField7.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField7.setForeground(new java.awt.Color(204, 0, 255));  jTextField7.setText("Other Details");  jTextField7.setToolTipText("");  jTextField7.setMaximumSize(new java.awt.Dimension(300, 50));  jTextField7.setMinimumSize(new java.awt.Dimension(300, 50));  jTextField7.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField8.setBackground(new java.awt.Color(0, 204, 204));  jTextField8.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField8.setForeground(new java.awt.Color(204, 0, 255));  jTextField8.setText("Other Details");  jTextField8.setToolTipText("");  jTextField8.setMaximumSize(new java.awt.Dimension(300, 50));  jTextField8.setMinimumSize(new java.awt.Dimension(300, 50));  jTextField8.setPreferredSize(new java.awt.Dimension(300, 40));  jTextField9.setBackground(new java.awt.Color(0, 204, 204));  jTextField9.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jTextField9.setForeground(new java.awt.Color(204, 0, 255));  jTextField9.setText("Other Details");  jTextField9.setToolTipText("");  jTextField9.setMaximumSize(new java.awt.Dimension(300, 50));  jTextField9.setMinimumSize(new java.awt.Dimension(300, 50));  jTextField9.setPreferredSize(new java.awt.Dimension(300, 40));  jComboBox3.setBackground(new java.awt.Color(0, 204, 204));  jComboBox3.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  jComboBox3.setForeground(new java.awt.Color(204, 0, 255));  jComboBox3.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "I", "II", "III", "IV", "V", "VI", "VII", "VIII", "IX", "X", "XI", "XII" }));  jComboBox3.setToolTipText("");  jComboBox3.setMaximumSize(new java.awt.Dimension(300, 50));  jComboBox3.setMinimumSize(new java.awt.Dimension(300, 50));  jComboBox3.setPreferredSize(new java.awt.Dimension(300, 40));  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);  this.setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addGap(1206, 1206, 1206)  .addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGroup(layout.createSequentialGroup()  .addGap(932, 932, 932)  .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))  .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField5, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField6, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED\_SIZE, 300, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED\_SIZE, 300, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jLabel8, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jLabel9, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jLabel10, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jLabel12, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jComboBox1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jComboBox2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField7, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField8, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jTextField9, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, 0)  .addComponent(jComboBox3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField5, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField6, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jLabel8, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jLabel9, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jLabel10, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jLabel12, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jComboBox1, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jComboBox2, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField7, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField8, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTextField9, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jComboBox3, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGroup(layout.createSequentialGroup()  .addContainerGap()  .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(89, 89, 89)  .addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, 984, javax.swing.GroupLayout.PREFERRED\_SIZE)))  .addGap(0, 0, Short.MAX\_VALUE))  );  }// </editor-fold>//GEN-END:initComponents  private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jTextField1ActionPerformed  // TODO add your handling code here:  }//GEN-LAST:event\_jTextField1ActionPerformed  private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton2ActionPerformed  // TODO add your handling code here:  }//GEN-LAST:event\_jButton2ActionPerformed  // Variables declaration - do not modify//GEN-BEGIN:variables  private javax.swing.JButton jButton1;  private javax.swing.JButton jButton2;  private javax.swing.JComboBox jComboBox1;  private javax.swing.JComboBox jComboBox2;  private javax.swing.JComboBox jComboBox3;  private javax.swing.JLabel jLabel10;  private javax.swing.JLabel jLabel11;  private javax.swing.JLabel jLabel12;  private javax.swing.JLabel jLabel3;  private javax.swing.JLabel jLabel4;  private javax.swing.JLabel jLabel5;  private javax.swing.JLabel jLabel6;  private javax.swing.JLabel jLabel7;  private javax.swing.JLabel jLabel8;  private javax.swing.JLabel jLabel9;  private javax.swing.JTextField jTextField1;  private javax.swing.JTextField jTextField2;  private javax.swing.JTextField jTextField3;  private javax.swing.JTextField jTextField4;  private javax.swing.JTextField jTextField5;  private javax.swing.JTextField jTextField6;  private javax.swing.JTextField jTextField7;  private javax.swing.JTextField jTextField8;  private javax.swing.JTextField jTextField9;  // End of variables declaration//GEN-END:variables  private void setAlignmentX(double d) {  throw new UnsupportedOperationException("Not yet implemented");  }  private void setAlignmentY(double d) {  throw new UnsupportedOperationException("Not yet implemented");  }  } |

### Mobile Application Coding

|  |
| --- |
| Index.xml |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>SMSmob</title>  <link href="s40-theme/css/s40-theme.css" rel="stylesheet" type="text/css" />  <script language="javascript" type="text/javascript" src="s40-theme/js/screensize.js"></script>  <script type="text/javascript">  function refreshPageContent() {  // Add code for refreshing the page here...  }  </script>  <link rel="stylesheet" type="text/css" href="s40-theme/css/s40-theme\_labeled\_text\_field.css" />  <link rel="stylesheet" type="text/css" href="s40-theme/css/s40-theme\_button.css" />  <link rel="stylesheet" type="text/css" href="s40-theme/css/s40-theme\_text\_field\_with\_button.css" />  <link rel="stylesheet" type="text/css" href="s40-theme/css/s40-theme\_category\_list.css" />  <link rel="stylesheet" type="text/css" href="s40-theme/css/s40-theme\_category\_list\_item.css" />  <link rel="stylesheet" type="text/css" href="s40-theme/css/s40-theme\_list.css" />  </head>  <body>  <div class="ui-page">  <!-- header -->  <div class="ui-header">  <div class="ui-title inline"><h5>School Management System</h5></div><div class="refresh-icon inline">  <a onclick="refreshPageContent();"><img alt="icon" src="s40-theme/images/refresh\_40x40.png"/></a></div>  </div>  <div class="ui-content">  <div class="ui-tab-control inline">  <div class="ui-tab ui-tab-selected inline" id="tab\_1" onclick="mwl.setGroupTarget('#tab\_control\_content', '#tab\_1\_content', 'ui-show', 'ui-hide');mwl.switchClass('#tab\_1', 'ui-tab-not-selected', 'ui-tab-selected'); mwl.switchClass('#tab\_2', 'ui-tab-selected', 'ui-tab-not-selected');mwl.switchClass('#tab\_3', 'ui-tab-selected', 'ui-tab-not-selected');">Admission  </div><div class="ui-tab ui-tab-not-selected inline" id="tab\_2" onclick="mwl.setGroupTarget('#tab\_control\_content', '#tab\_2\_content', 'ui-show', 'ui-hide');mwl.switchClass('#tab\_1', 'ui-tab-selected', 'ui-tab-not-selected');mwl.switchClass('#tab\_2', 'ui-tab-not-selected', 'ui-tab-selected');mwl.switchClass('#tab\_3', 'ui-tab-selected', 'ui-tab-not-selected');">Student  </div><div class="ui-tab ui-tab-not-selected inline" id="tab\_3" onclick="mwl.setGroupTarget('#tab\_control\_content', '#tab\_3\_content', 'ui-show', 'ui-hide');mwl.switchClass('#tab\_1', 'ui-tab-selected', 'ui-tab-not-selected');mwl.switchClass('#tab\_2', 'ui-tab-selected', 'ui-tab-not-selected');mwl.switchClass('#tab\_3', 'ui-tab-not-selected', 'ui-tab-selected');">Course</div>  </div>  <div class="ui-tab-content-row" id="tab\_control\_content">  <div id="tab\_1\_content" class="ui-show">  <!-- label-with-text-field -->  <div class="ui-fieldset\_S40labeledtextfield">  <div class="ui-legend\_S40labeledtextfield">First Name:</div>  <input type="text" name="name" class="ui-text-input\_S40labeledtextfield"/>  <div class="ui-legend\_S40labeledtextfield">Last Name:</div>  <input type="text" name="name" class="ui-text-input\_S40labeledtextfield"/>  <div class="ui-legend\_S40labeledtextfield">Address:</div>  <input type="text" name="name" class="ui-text-input\_S40labeledtextfield"/>  <div class="ui-legend\_S40labeledtextfield">Std:</div>  <input type="text" name="name" class="ui-text-input\_S40labeledtextfield"/>  <br /><br /><button name="button1" class="ui-button\_S40button">Submit</button>  </div>  </div>  <div id="tab\_2\_content" class="ui-hide">  <!-- label-with-text-field -->  <div class="ui-fieldset\_S40labeledtextfield">  <div class="ui-legend\_S40labeledtextfield">User Name:</div>  <input type="text" name="name" class="ui-text-input\_S40labeledtextfield"/>  <div class="ui-legend\_S40labeledtextfield">Password</div>  <input type="text" name="name" class="ui-text-input\_S40labeledtextfield"/>  <button name="submit" class="ui-button\_S40button">Enter</button>  </div>  <!-- category-list -->  <div class="ui-category-list\_S40categorylist">  <!-- list-item-1 -->  <div class="ui-category-list-item\_S40categorylist">  <div class="ui-category-list-item-title\_S40categorylist ui-open\_S40categorylist" id="category\_title\_1\_1" onclick="mwl.toggleClass('#category\_items\_1\_1', 'ui-hide\_S40categorylist');mwl.toggleClass('#category\_items\_1\_1', 'ui-show\_S40categorylist');mwl.toggleClass('#category\_title\_1\_1', 'ui-open\_S40categorylist');mwl.toggleClass('#category\_title\_1\_1', 'ui-close\_S40categorylist');">Student Details</div>  <div class="ui-category-list-item-body\_S40categorylist ui-show\_S40categorylist" id="category\_items\_1\_1">  <!-- list-items -->  <div class="ui-list\_S40list">  <div class="ui-list-item\_S40list">Name:Joysankar Sengupta</div>  <div class="ui-list-item\_S40list">Class:X</div>  <div class="ui-list-item\_S40list">Roll No:32</div>  <div class="ui-list-item\_S40list">Attendance:72%</div>  </div>  </div>  </div>  <!-- list-item-2 -->  <div class="ui-category-list-item\_S40categorylist">  <div class="ui-category-list-item-title\_S40categorylist ui-close\_S40categorylist" id="category\_title\_2\_1" onclick="mwl.toggleClass('#category\_items\_2\_1', 'ui-show\_S40categorylist');mwl.toggleClass('#category\_items\_2\_1', 'ui-hide\_S40categorylist');mwl.toggleClass('#category\_title\_2\_1', 'ui-close\_S40categorylist');mwl.toggleClass('#category\_title\_2\_1', 'ui-open\_S40categorylist');">Result</div>  <div class="ui-category-list-item-body\_S40categorylist ui-hide\_S40categorylist" id="category\_items\_2\_1">  <!-- list-items -->  <div class="ui-list\_S40list">  <div class="ui-list-item\_S40list">Total Marks Obtained:756</div>  <div class="ui-list-item\_S40list">Division:1st</div>  <div class="ui-list-item\_S40list">Status:Promoted</div>  </div>  </div>  </div>  </div>  </div>  <div id="tab\_3\_content" class="ui-hide">  <!-- category-list-item -->  <div class="ui-category-list-item\_S40categorylistitem">  <div class="ui-category-list-item-title\_S40categorylistitem ui-open\_S40categorylistitem" id="category\_title\_3\_2" onclick="mwl.toggleClass('#category\_items\_3\_2', 'ui-hide\_S40categorylistitem');mwl.toggleClass('#category\_items\_3\_2', 'ui-show\_S40categorylistitem');mwl.toggleClass('#category\_title\_3\_2', 'ui-open\_S40categorylistitem');mwl.toggleClass('#category\_title\_3\_2', 'ui-close\_S40categorylistitem');">Regular Course</div>  <div class="ui-category-list-item-body\_S40categorylistitem ui-show\_S40categorylistitem" id="category\_items\_3\_2">  <!-- list-items -->  <div class="ui-list\_S40list">  <div class="ui-list-item\_S40list">Science(Class XI-XII)</div>  <div class="ui-list-item\_S40list">Arts(Class XI-XII)</div>  <div class="ui-list-item\_S40list">Class V-X</div>  </div>  </div>  </div>  <!-- category-list-item -->  <div class="ui-category-list-item\_S40categorylistitem">  <div class="ui-category-list-item-title\_S40categorylistitem ui-open\_S40categorylistitem" id="category\_title\_3\_3" onclick="mwl.toggleClass('#category\_items\_3\_3', 'ui-hide\_S40categorylistitem');mwl.toggleClass('#category\_items\_3\_3', 'ui-show\_S40categorylistitem');mwl.toggleClass('#category\_title\_3\_3', 'ui-open\_S40categorylistitem');mwl.toggleClass('#category\_title\_3\_3', 'ui-close\_S40categorylistitem');">Vocational</div>  <div class="ui-category-list-item-body\_S40categorylistitem ui-show\_S40categorylistitem" id="category\_items\_3\_3">  <!-- list-items -->  <div class="ui-list\_S40list">  <div class="ui-list-item\_S40list">Basic Computer</div>  <div class="ui-list-item\_S40list">Drawing</div>  <div class="ui-list-item\_S40list">Others</div>  </div>  </div>  </div>  </div>  </div>  </div>    </div>    </body>  </html> |

## COMMENTS AND DESCRIPTION OF CODING SEGMENTS

#### Code Commenting

* All comments have been written in the same language, be grammatically correct, and contain appropriate punctuation.
* Used // or /// but never /\* … \*/
* Did not “flowerbox” comment blocks.

Example:

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Comment block

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Always Used inline-comments to explain assumptions, known issues, and algorithm insights.
* Never used inline-comments to explain obvious code. Well written code is self documenting.
* Only used comments for bad code to say “fix this code” – otherwise remove, or rewrite the code!
* Included comments using Task-List keyword flags to allow comment-filtering.

Example:

// TODO: Place Database Code Here

// UNDONE: Removed P\Invoke Call due to errors

// HACK: Temporary fix until able to refactor

* Always applied C# comment-blocks (///) to public, protected, and internal declarations.
* Only used C# comment-blocks for documenting the API.
* Always included <summary> comments. Include <param>, <return>, and <exception> comment
* sections where applicable.
* Included <see cref=””/> and <seeAlso cref=””/> where possible.
* Always added CDATA tags to comments containing code and other embedded markup in order to avoid encoding issues.

Example:

/// <example>

/// Add the following key to the “appSettings” section of your config:

/// <code><![CDATA[

/// <configuration>

/// <appSettings>

/// <add key=”mySetting” value=”myValue”/>

/// </appSettings>

/// </configuration>

/// ]]></code> >

/// </example>

## Standardization of the coding

Coding style causes the most inconsistency and controversy between developers. Each developer has a preference, and

rarely are two the same. However, consistent layout, format, and organization are key to creating maintainable code.

The following sections describe the preferred way to implement C# source code in order to create readable, clear, and

consistent code that is easy to understand and maintain.

#### Formatting

* Never declared more than 1 namespace per file.
* Avoided putting multiple classes in a single file.
* Always placed curly braSMS ({ and }) on a new line.
* Always used curly braSMS ({ and }) in conditional statements.
* Always used a Tab & Indention size of 4.
* Declared each variable independently – not in the same statement.
* Placed namespace “using” statements together at the top of file. Group .NET namespaSMS above custom namespaSMS.
* Grouped internal class implementation by type in the following order:

1. Member variables.
2. Constructors & Finalizers.
3. Nested Enums, Structs, and Classes.
4. Properties
5. Methods

* Sequence declarations within type groups based upon acSMSs modifier and visibility:

1. Public
2. Protected
3. Internal
4. Private

* Segregate interface Implementation by using #region statements.
* Append folder-name to namespace for source files within sub-folders.
* Recursively indent all code blocks contained within braSMS.
* Use white space (CR/LF, Tabs, etc) liberally to separate and organize code.
* Only declare related attribute declarations on a single line, otherwise stack each attribute as a separate declaration.

Example:

// Bad!

[Attrbute1, Attrbute2, Attrbute3]

public class MyClass

{…}

// Good!

[Attrbute1, RelatedAttribute2]

[Attrbute3]

[Attrbute4]

public class MyClass

{…}

* Place Assembly scope attribute declarations on a separate line.
* Place Type scope attribute declarations on a separate line.
* Place Method scope attribute declarations on a separate line.
* Place Member scope attribute declarations on a separate line.
* Place Parameter attribute declarations inline with the parameter.
* If in doubt, always err on the side of clarity and consistency.

## 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 unneSMSsary 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

The C# language's exception handling features help us to deal with any unexpected or exceptional situations that occur when a program is running. Exception handling uses the **try**, **catch**, and **finally** keywords to try actions that may not succeed, to handle failures when you decide that it is reasonable to do so, and to clean up resourSMS afterward. Exceptions can be generated by the common language runtime (CLR), by the .NET Framework or any third-party libraries, or by application code. Exceptions are created by using the **throw** keyword.

In many cases, an exception may be thrown not by a method that your code has called directly, but by another method further down in the call stack. When this happens, the CLR will unwind the stack, looking for a method with a **catch** block for the specific exception type, and it will execute the first such **catch** block that if finds. If it finds no appropriate **catch** block anywhere in the call stack, it will terminate the proSMSs and display a message to the user.

[Exceptions Overview](javascript:void(0))

Exceptions have the following properties:

Exceptions are types that all ultimately derive from **System.Exception**.

Use a **try** block around the statements that might throw exceptions.

Once an exception occurs in the **try** block, the flow of control jumps to the first associated exception handler that is present anywhere in the call stack. In C#, the**catch** keyword is used to define an exception handler.

If no exception handler for a given exception is present, the program stops executing with an error message.

Do not catch an exception unless you can handle it and leave the application in a known state. If you catch **System.Exception**, rethrow it using the **throw** keyword at the end of the **catch** block.

If a **catch** block defines an exception variable, you can use it to obtain more information about the type of exception that occurred.

Exceptions can be explicitly generated by a program by using the **throw** keyword.

Exception objects contain detailed information about the error, such as the state of the call stack and a text description of the error.

Code in a **finally** block is executed even if an exception is thrown. Use a **finally** block to release resourSMS, for example to close any streams or files that were opened in the **try** block.

Managed exceptions in the .NET Framework are implemented on top of the Win32 structured exception handling mechanism.

## Parameters calling/passing

## Validation checks

We have performed following data validation checks on available data:

#### Allowed character checks

Checks that ascertain that only expected characters are present in a field. For example a numeric field may only allow the digits 0-9, the decimal point and perhaps a minus sign or commas. A text field such as a personal name might disallow characters such as < and >, as they could be evidence of a markup-based security attack. An e-mail address might require exactly one @ sign and various other structural details. Regular expressions are effective ways of implementing such checks. (See also data type checks below)

#### Batch totals

Checks for missing records. Numerical fields may be added together for all records in a batch. The batch total is entered and the computer checks that the total is correct, e.g., add the 'Total Cost' field of a number of transactions together.

#### Cardinality check

Checks that record has a valid number of related records. For example if Contact record classified as a Customer it must have at least one associated Order (Cardinality > 0). If order does not exist for a "customer" record then it must be either changed to "seed" or the order must be created. This type of rule can be complicated by additional conditions. For example if contact record in Payroll database is marked as "former employee", then this record must not have any associated salary payments after the date on which employee left organization (Cardinality = 0).

#### Check digits

Used for numerical data. An extra digit is added to a number which is calculated from the digits. The computer checks this calculation when data are entered. For example the last digit of an ISBN for a book is a check digit calculated modulus 10.[3]

#### Consistency checks

Checks fields to ensure data in these fields corresponds, e.g., If Title = "Mr.", then Gender = "M".

#### Control totals

This is a total done on one or more numeric fields which appears in every record. This is a meaningful total, e.g., add the total payment for a number of Customers.

#### Cross-system consistency checks

Compares data in different systems to ensure it is consistent, e.g., The address for the customer with the same id is the same in both systems. The data may be represented differently in different systems and may need to be transformed to a common format to be compared, e.g., one system may store customer name in a single Name field as 'Doe, John Q', while another in three different fields: First\_Name (John), Last\_Name (Doe) and Middle\_Name (Quality); to compare the two, the validation engine would have to transform data from the second system to match the data from the first, for example, using SQL: Last\_Name || ', ' || First\_Name || substr(Middle\_Name, 1, 1) would convert the data from the second system to look like the data from the first 'Doe, John Q'

#### Data type checks

Checks the data type of the input and give an error message if the input data does not match with the chosen data type, e.g., In an input box accepting numeric data, if the letter 'O' was typed instead of the number zero, an error message would appear.

#### File existence check

Checks that a file with a specified name exists. This check is essential for programs that use file handling.

#### Format or picture check

Checks that the data is in a specified format (template), e.g., dates have to be in the format DD/MM/YYYY.

Regular expressions should be considered for this type of validation.

#### Hash totals

This is just a batch total done on one or more numeric fields which appears in every record. This is a meaningless total, e.g., add the Telephone Numbers together for a number of Customers.

#### Limit check

Unlike range checks, data are checked for one limit only, upper OR lower, e.g., data should not be greater than 2 (<=2).

#### Logic check

Checks that an input does not yield a logical error, e.g., an input value should not be 0 when there will be a number that divides it somewhere in a program.

#### Presence check

Checks that important data are actually present and have not been missed out, e.g., customers may be required to have their telephone numbers listed.

#### Range check

Checks that the data lie within a specified range of values, e.g., the month of a person's date of birth should lie between 1 and 12.

#### Referential integrity

In modern Relational database values in two tables can be linked through foreign key and primary key. If values in the primary key field are not constrained by database internal mechanism,[4] then they should be validated. Validation of the foreign key field checks that referencing table must always refer to a valid row in the referenced table.[5]

#### Spelling and grammar check

Looks for spelling and grammatical errors.

#### Uniqueness check

Checks that each value is unique. This can be applied to several fields (i.e. Address, First Name, Last Name).

#### Table Look Up Check

A table look up check takes the entered data item and compares it to a valid list of entries that are stored in a database table.

# 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:

### 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. |

### 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. |

### 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. |

### 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.

### 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

### Creation of Test Plan

Early in the deployment planning phase, the testing team creates a test plan. The *test plan* defines the objectives and scope of the testing effort, and identifies the methodology that our team will use to conduct tests. It also identifies the hardware, software, and tools required for testing and the features and functions that will be tested. A well-rounded test plan notes any risk factors that jeopardize testing and includes a testing schedule.

If our testing team is divided into technology subteams, each subteam should develop a test plan for that team’s specific technology area. For example, the networking team would write a test plan for testing networking features. All members of each subteam should review and approve its team’s test plan before it is integrated into the general test plan.

Figure 2.3 illustrates the tasks we performed to create the test plan.

**Figure 2.3   Creating a Test Plan**



### Testing Scope and Objectives

In the scope and objectives section of the test plan, the testing team described specifically what we want our testing to accomplish. Also, we needed to define the scope of our testing by identifying what we will test and what we will not. We might limit our testing of client computer hardware to the minimum supported configurations or to the standard configurations.

### Testing Methodology

The general methodology that our team used for testing to testing schema changes might be to configure an isolated domain in the test lab where schema changes can be applied without affecting other lab tests. This section of the test plan addressed the following:

* The domain architecture used for testing
* The tools and techniques used to conduct the tests or to measure results
* Automated techniques we plan to use during testing

### Features and Functions to Test

Included tests that verify or address:

* The functionality of each feature and service that you will implement.
* Interoperability with existing components and systems in the production environment, both during the phase-in period, when there is a mix of old functionality and new Windows Server 2003 functionality, and after the Windows Server 2003 environment has been rolled out.
* Hardware and driver compatibility for every type of client computer that will be running Windows XP Professional.
* Application compatibility for every application that will run on Windows XP Professional.
* Application compatibility for every server application that will run on Windows Server 2003.
* Optimization of configurations, such as those for standardized desktops on client computers.

### Risk Factors

We described the risk factors that could prevent the sucSMSsful completion of all required tests. We found that the test lab is behind schedule, or that required hardware or software is unavailable, or that testers are working on other projects or need additional training. After we have identified the risk factors, decide what we will do to avoid or mitigate each risk.

### Testing Schedule

We drafted a preliminary schedule that includes each test listed in the test plan. The schedule can help us coordinate test lab use among sub teams. Assign a team member, ideally the test lab manager, if our team has one, to maintain and update the lab schedule. Having an up-to-date schedule is critical when unscheduled lab requests are submitted.

## TESTING REPORTS

### UNIT TEST CASES

|  |  |  |
| --- | --- | --- |
| Test Case Id | Comments | Status |
| SMS – 001  SMS – 002  SMS – 003  SMS – 004  SMS – 005  SMS – 006  SMS – 007 | NA  NA  NA  NA  NA  NA  NA | PASS  PASS  PASS  PASS  PASS  PASS  PASS |

### SYSTEM TEST CASES

|  |  |  |
| --- | --- | --- |
| Test Case Id | Comments | Status |
| SMS – 001 | NA | PASS |
| SMS – 002  SMS – 003  SMS – 004  SMS – 005  SMS – 006  SMS – 007  SMS – 008  SMS – 009  SMS – 010  SMS – 011  SMS – 012  SMS – 013  SMS – 014  SMS – 015  SMS – 016  SMS – 017  SMS – 018  SMS – 019  SMS – 020  SMS – 021  SMS – 022  SMS – 023  SMS – 024  SMS – 025  SMS – 026  SMS – 027  SMS – 028  SMS – 029  SMS – 030  SMS – 031  SMS – 032  SMS – 033  SMS – 034  SMS – 035  SMS – 036 | NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA | PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS  PASS |

## DEBUGGING AND CODE IMPROVEMENT

The steps in the bellow section demonstrate how to create a console application that uses the **Debug** class to provide information about the program execution.   
  
When the program is run, we can use methods of the **Debug** class to produce messages that help we to monitor the program execution sequence, to detect malfunctions, or to provide performance measurement information. By default, the messages that the **Debug** class produSMS appear in the Output window of the Visual Studio Integrated Development Environment (IDE).   
  
The sample code uses the **WriteLine** method to produce a message that is followed by a line terminator. When we use this method to produce a message, each message appears on a separate line in the Output window.   
  
When we use the **Assert** method of the **Debug** class, the Output window displays a message only if a specified condition evaluates to false. The message also appears in a modal dialog box to the user. The dialog box includes the message, the project name, and the **Debug.Assert** statement number. The dialog box also includes the following three command buttons: 

* **Abort:** The application stops running.
* **Retry:** The application enters debug mode.
* **Ignore:** The application proceeds.

The user must click one of these buttons before the application can continue.   
  
We can also direct output from the **Debug** class to destinations other than the Output window. The **Debug** class has a collection named **Listeners** that includes **Listener** objects.  
  
Each **Listener** object monitors **Debug** output and directs the output to a specified target.   
  
Each **Listener** in the **Listener** collection receives any output that the **Debug** class generates. Use the **TextWriterTraceListener**class to define **Listener** objects. We can specify the target for a **TextWriterTraceListener** class through its constructor.   
  
Some possible output targets include the following:

* The Console window by using the **System.Console.Out** property.
* A text (.txt) file by using the **System.IO.File.CreateText("FileName.txt")** statement.

After we create a **TextWriterTraceListener** object, we must add the object to the **Debug.Listeners** collection to receive Debug output. 

### Create a Sample with the Debug Class

1. Start Visual Studio or Visual C# Express Edition.
2. Create a new Visual C# Console Application project named **conInfo**. Class1 is created in Visual Studio .NET. Program.cs is created in Visual Studio 2005.
3. Add the following namespace at top in Class1 or Program.cs.

using System.Diagnostics;

1. To initialize variables to contain information about a product, add the following declaration statements to **Main** method:
2. string sProdName = "Widget";
3. int iUnitQty = 100;

double dUnitCost = 1.03;

1. Specify the message that the class produSMS as the first input parameter of the **WriteLine** method. Press the CTRL+ALT+O key combination to make sure that the Output window is visible.

Debug.WriteLine("Debug Information-Product Starting ");

1. For readability, use the **Indent** method to indent subsequent messages in the Output window:

Debug.Indent();

1. To display the content of selected variables, use the **WriteLine** method as follows:
2. Debug.WriteLine("The product name is " + sProdName);
3. Debug.WriteLine("The available units on hand are" + iUnitQty.ToString());

Debug.WriteLine("The per unit cost is " + dUnitCost.ToString());

1. We can also use the **WriteLine** method to display the namespace and the class name for an existent object. For example, the following code displays the **System.Xml.XmlDocument** namespace in the Output window:
2. System.Xml.XmlDocument oxml = new System.Xml.XmlDocument();

Debug.WriteLine(oxml);

1. To organize the output, we can include a category as an optional, second input parameter of the **WriteLine** method. If we specify a category, the format of the Output window message is "category: message." For example, the first line of the following code displays "Field: The product name is Widget" in the Output window:
2. Debug.WriteLine("The product name is " + sProdName,"Field");
3. Debug.WriteLine("The units on hand are" + iUnitQty,"Field");
4. Debug.WriteLine("The per unit cost is" + dUnitCost.ToString(),"Field");

Debug.WriteLine("Total Cost is " + (iUnitQty \* dUnitCost),"Calc");

1. The Output window can display messages only if a designated condition evaluates to true by using the **WriteLineIf**method of the **Debug** class. The condition to be evaluated is the first input parameter of the **WriteLineIf** method. The second parameter of **WriteLineIf** is the message that appears only if the condition in the first parameter evaluates to true.
2. Debug.WriteLineIf(iUnitQty > 50, "This message WILL appear");
3. Debug.WriteLineIf(iUnitQty < 50, "This message will NOT appear");
4. Use the **Assert** method of the **Debug** class so that the Output window displays the message only if a specified condition evaluates to false:
5. Debug.Assert(dUnitCost > 1, "Message will NOT appear");
6. Debug.Assert(dUnitCost < 1, "Message will appear since dUnitcost < 1 is false");
7. Create the **TextWriterTraceListener** objects for the Console window (tr1) and for a text file named Output.txt (tr2), and then add each object to the **Debug Listeners** collection:
8. TextWriterTraceListener tr1 = new TextWriterTraceListener(System.Console.Out);
9. Debug.Listeners.Add(tr1);
11. TextWriterTraceListener tr2 = new TextWriterTraceListener(System.IO.File.CreateText("Output.txt"));

Debug.Listeners.Add(tr2);

1. For readability, use the **Unindent** method to remove the indentation for subsequent messages that the **Debug** class generates. When we use the **Indent** and the **Unindent** methods together, the reader can distinguish the output as group.
2. Debug.Unindent();

Debug.WriteLine("Debug Information-Product Ending");

1. To make sure that each **Listener** object receives all its output, call the **Flush** method for the **Debug** class buffers:

Debug.Flush();

### Using the Trace Class

We can also use the **Trace** class to produce messages that monitor the execution of an application. The **Trace** and **Debug**classes share most of the same methods to produce output, including the following:

* **WriteLine**
* **WriteLineIf**
* **Indent**
* **Unindent**
* **Assert**
* **Flush**

We can use the **Trace** and the **Debug** classes separately or together in the same application. In a Debug Solution Configuration project, both **Trace** and **Debug** output are active. The project generates output from both of these classes to all**Listener** objects. However, a Release Solution Configuration project only generates output from a **Trace** class. The Release Solution Configuration project ignores any **Debug** class method invocations.

Trace.WriteLine("Trace Information-Product Starting ");

Trace.Indent();

Trace.WriteLine("The product name is "+sProdName);

Trace.WriteLine("The product name is"+sProdName,"Field" );

Trace.WriteLineIf(iUnitQty > 50, "This message WILL appear");

Trace.Assert(dUnitCost > 1, "Message will NOT appear");

Trace.Unindent();

Trace.WriteLine("Trace Information-Product Ending");

Trace.Flush();

Console.ReadLine();

### Verify That It Works

1. Make sure that **Debug** is the current solution configuration.
2. If the **Solution Explorer** window is not visible, press the CTRL+ALT+L key combination to display this window.
3. Right-click **conInfo**, and then click **Properties**.
4. In the left pane of the conInfo property page, under the **Configuration** folder, make sure that the arrow points to**Debugging**.  
     
   **Note** In Visual C# 2005 and in Visual C# 2005 Express Edition, click **Debug** in the **conInfo** page.
5. Above the **Configuration** folder, in the **Configuration** drop-down list box, click **Active (Debug)** or **Debug**, and then click **OK**. In Visual C# 2005 and in Visual C# 2005 Express Edition, click **Active (Debug)** or **Debug** in the **Configuration**drop-down list box in the **Debug** page, and then click **Save** on the **File** menu.
6. Press CTRL+ALT+O to display the Output window.
7. Press the F5 key to run the code. When the **Assertion Failed** dialog box appears, click **Ignore**.
8. In the Console window, press ENTER. The program should finish, and the Output window should display the output that resembles the following
9. Debug Information-Product Starting
10. The product name is Widget
11. The available units on hand are100
12. The per unit cost is 1.03
13. System.Xml.XmlDocument
14. Field: The product name is Widget
15. Field: The units on hand are100
16. Field: The per unit cost is1.03
17. Calc: Total Cost is 103
18. This message WILL appear
19. ---- DEBUG ASSERTION FAILED ----
20. ---- Assert Short Message ----
21. Message will appear since dUnitcost < 1 is false
22. ---- Assert Long Message ----
23. at Class1.Main(String[] args) <%Path%>\class1.cs(34)
24. The product name is Widget
25. The available units on hand are100
26. The per unit cost is 1.03
27. Debug Information-Product Ending
28. Trace Information-Product Starting
29. The product name is Widget
30. Field: The product name isWidget
31. This message WILL appear
32. Trace Information-Product Ending
34. The Console window and the Output.txt file should display the following output:
35. The product name is Widget
36. The available units on hand are 100
37. The per unit cost is 1.03
38. Debug Information-Product Ending
39. Trace Information-Product Starting
40. The product name is Widget
41. Field: The product name is Widget
42. This message WILL appear
43. Trace Information-Product Ending

**Note** The Output.txt file is located in the same directory as the conInfo executable (conInfo.exe). Typically, this is the \bin folder where the project source is stored. By default, this is C:\Documents and Settings\User login\My Documents\Visual Studio Projects\conInfo\bin. In Visual C# 2005 and in Visual C# 2005 Express Edition, the Output.txt file is located in the following folder:

C:\Documents and Settings\User login\My Documents\Visual Studio 2005\Projects\conInfo\conInfo\bin\Debug

### Complete Code Listing

using System;

using System.Diagnostics;

class Class1

{

[STAThread]

static void Main(string[] args)

{

string sProdName = "Widget";

int iUnitQty = 100;

double dUnitCost = 1.03;

Debug.WriteLine("Debug Information-Product Starting ");

Debug.Indent();

Debug.WriteLine("The product name is "+sProdName);

Debug.WriteLine("The available units on hand are"+iUnitQty.ToString());

Debug.WriteLine("The per unit cost is "+ dUnitCost.ToString());

System.Xml.XmlDocument oxml = new System.Xml.XmlDocument();

Debug.WriteLine(oxml);

Debug.WriteLine("The product name is "+sProdName,"Field");

Debug.WriteLine("The units on hand are"+iUnitQty,"Field");

Debug.WriteLine("The per unit cost is"+dUnitCost.ToString(),"Field");

Debug.WriteLine("Total Cost is "+(iUnitQty \* dUnitCost),"Calc");

Debug.WriteLineIf(iUnitQty > 50, "This message WILL appear");

Debug.WriteLineIf(iUnitQty < 50, "This message will NOT appear");

Debug.Assert(dUnitCost > 1, "Message will NOT appear");

Debug.Assert(dUnitCost < 1, "Message will appear since dUnitcost < 1 is false");

TextWriterTraceListener tr1 = new TextWriterTraceListener(System.Console.Out);

Debug.Listeners.Add(tr1);

TextWriterTraceListener tr2 = new TextWriterTraceListener(System.IO.File.CreateText("Output.txt"));

Debug.Listeners.Add(tr2);

Debug.WriteLine("The product name is "+sProdName);

Debug.WriteLine("The available units on hand are"+iUnitQty);

Debug.WriteLine("The per unit cost is "+dUnitCost);

Debug.Unindent();

Debug.WriteLine("Debug Information-Product Ending");

Debug.Flush();

Trace.WriteLine("Trace Information-Product Starting ");

Trace.Indent();

Trace.WriteLine("The product name is "+sProdName);

Trace.WriteLine("The product name is"+sProdName,"Field" );

Trace.WriteLineIf(iUnitQty > 50, "This message WILL appear");

Trace.Assert(dUnitCost > 1, "Message will NOT appear");

Trace.Unindent();

Trace.WriteLine("Trace Information-Product Ending");

Trace.Flush();

Console.ReadLine();

}

}

### Troubleshoot

* If the solution configuration type is **Release**, the **Debug** class output is ignored.
* After we create a **TextWriterTraceListener** class for a particular target, **TextWriterTraceListener** receives output from the **Trace** and the **Debug** classes. This occurs regardless of whether we use the **Add** method of the **Trace** or the**Debug** class to add **TextWriterTraceListener** to the **Listeners** class.
* If we add a **Listeners** object for the same target in the **Trace** and the **Debug** classes, each line of output is duplicated, regardless of whether **Debug** or **Trace** generates the output.
* TextWriterTraceListener myWriter = new TextWriterTraceListener(System.Console.Out);
* Debug.Listeners.Add(myWriter);
* TextWriterTraceListener myCreator = new TextWriterTraceListener(System.Console.Out);

Trace.Listeners.Add(myCreator);

# SYSTEM SECURITY MEASURES

* School Management System is password protected software. It will be developed such a way that the admin will have complete control on the school’s data.
* Admin can create account with various permission levels, like clerk, librarian, teacher, admin etc. so that the users can see relevant data only.

## DATABASE / DATA SECURITY

* The data of the school will be stored in the database with an encrypted format so even if someone hacks the database somehow still he can make no real harm.
* The software will provide a backup and restore feature in case of loss of data.

## CREATION OF USER PROFILES AND ACCESS RIGHTS

* The software asks for a predefined user-type, username and password to use its feature.
* All the data are not available for all types of user, for example, only an admin can use all the fields of the application. On the other hand, a clerk can only enter data and see data from some selected fields. A librarian can use library related data only.

# COST ESTIMATION

## 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)b2months

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 SMS 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 SMS semidetached software:

Our Effort = 3.0 \* (4)1.12 PM

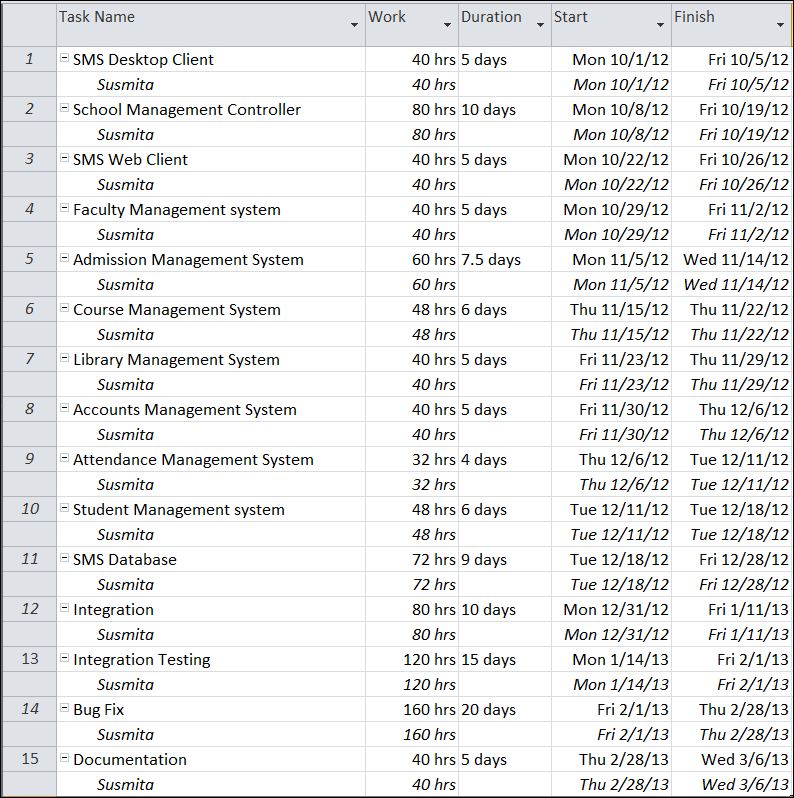
= 14 PM

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

= 6 months

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

= Rs. 90,000



# REPORTS

List of reports that are likely to be generated in this software are given below:

* Results & Grade report can be generated
* Class toppers will be given a certificate as a token of appreciation
* List of students can be generated
* List of teachers can be generated
* List of courses can be generated
* Course details can be generated
* Fund details can be generated
* Yearly donation report can be generated
* Salary slips can be created

# FUTURE SCOPE AND FURTHER ENHANCEMENT

* Mobile application could be developed for students’ guardians for querying about various details.
* Support for Linux operating system could be added.
* Online result checking and fees payment feature could be added.

# BIBLIOGRAPHY

## Websites

* <http://en.wikipedia.org>
* http://en.wikipedia.org/wiki/Windows\_Presentation\_Foundation
* <http://msdn.microsoft.com/en-us/>
* http://www.c-sharpcorner.com/beginners/
* <http://www.microsoft.com/en-us/default.aspx>
* <http://www.codeplex.com/>
* http://searchitchannel.techtarget.com/
* <http://stackoverflow.com/>
* <http://www.codeguru.com/>
* http://www.roseindia.net/
* http://www.csharpcourse.com/
* <http://www.w3schools.com>
* <http://blogs.technicise.com/>
* http://connect.technicise.com/
* <http://www.mysql.com/>
* http://www.mysql.com/support/
* <http://dev.mysql.com/>
* http://dev.mysql.com/support/
* <http://www.homeandlearn.co.uk/csharp/csharp.html>
* <http://www.wpftutorial.net/Home.html>
* *http://netbeans.org/*
* https://www.facebook.com/
* <https://developers.facebook.com/>
* <https://developers.facebook.com/apps>
* <https://www.facebook.com/creatormyapp>
* https://www.facebook.com/help/
* https://twitter.com/#
* <https://dev.twitter.com/>
* <https://dev.twitter.com/discussions/3477>
* https://support.twitter.com/
* <https://developers.google.com/>
* <https://maps.google.co.in/>
* http://support.google.com/maps/?hl=en&rd=1
* <http://www.youtube.com/?gl=IN>
* <http://blendinsider.com/>
* https://github.com/
* <https://github.com/anirban-nandy>
* [https://github.com/anirban-nandy/DailyNoteBook](https://github.com/%20anirban-nandy%20/DailyNoteBook)
* <http://learn.github.com/p/intro.html>
* <http://www.vogella.com/articles/Git/article.html>
* <http://try.github.com/levels/1/challenges/1>
* <https://enterprise.github.com/support>
* <https://support.enterprise.github.com/home>

## Books

* **Programming Java** - E. R. Balaguruswamy.
* Fundamentals of software engineering by Rajib Mall.
* Pro C# 2010 and the .NET 4.0 Platform by Andrew Troselen.
* C# Programming by Rob Miles.

# APPENDICES

## IDE:

### Visual Studio 2010

visual_studio_logo

Microsoft Visual Studio is a powerful IDE that ensures quality code throughout the entire application lifecycle, from design to deployment. Whether we are developing applications for SharePoint, the web, Windows, Windows Phone, and beyond, Visual Studio is the ultimate all-in-one solution. Visual Studio includes a [code editor](http://en.wikipedia.org/wiki/Code_editor) supporting [IntelliSense](http://en.wikipedia.org/wiki/IntelliSense) as well as [code refactoring](http://en.wikipedia.org/wiki/Code_refactoring). The integrated [debugger](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio_Debugger) works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building [GUI](http://en.wikipedia.org/wiki/GUI) applications, web designer, [class](http://en.wikipedia.org/wiki/Class_(computing)) designer, and [database schema](http://en.wikipedia.org/wiki/Database_schema) designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for [source-control](http://en.wikipedia.org/wiki/Source_control) systems (like [Subversion](http://en.wikipedia.org/wiki/Subversion_(software)) and [Visual SourceSafe](http://en.wikipedia.org/wiki/Visual_SourceSafe)) and adding new toolsets like editors and visual designers for [domain-specific languages](http://en.wikipedia.org/wiki/Domain-specific_language) or toolsets for other aspects of the [software development lifecycle](http://en.wikipedia.org/wiki/Software_development_lifecycle) (like the [Team Foundation Server](http://en.wikipedia.org/wiki/Team_Foundation_Server) client: Team Explorer).

#### Standout features

* User interface built on Windows Presentation Foundation (WPF)
* Improved Start page
* Improved code editor
* Improved IntelliSense
* Call Hierarchy Viewer

#### What problems does it solve?

The newly designed user experience is refreshing for an application showing its age. The user interface is built on WPF and no longer relies on the limited MDI interface in previous versions; this allows for better multi-monitor support with fly-out windows. The first thing you might notice when opening Visual Studio 2010 is the new Start page. As an xaml file, this page is completely customizable and includes the ability to remove and pin project files in the Recent Projects section.

The code editor has a number of enhancements. You can scale the font by holding down [Ctrl] while scrolling the mouse wheel. In previous versions of Visual Studio, users had to set the font size through a dialog and exit to see if the changes were correct.

In Visual Studio 2010, Box Selection is enhanced to allow for zero-length boxes and improved pasting.

The feature that will see the most use (by accident if not design) is Highlight References. By selecting any symbol, such as a variable or a property, all references to the symbol are highlighted. The symbols can then be navigated by holding down [Ctrl][Shift] and pressing the up/down keys.

IntelliSense has been improved to allow for acronyms based on Pascal casing. For example, typing *String.INOE* and then a non-alphanumeric character will convert the call to*String.IsNullOrEmpty*. This still doesn’t prevent IntelliSense from interfering when you’re writing code that doesn’t exist, as you would with a unit test.

The Suggestion Completion mode allows you to type freely without IntelliSense changing the text you typed. You can toggle between Standard and Suggestion Completion modes by pressing [Ctrl][Alt]space.

IntelliSense for JavaScript has seen the most improvement, as it is now able to determine the correct structure of a variable even after the structure is changed.

In the past, I would use .NET Reflector or another tool to analyze a user’s call hierarchy; now that functionality is built-in. Right-click the user and choose View Call Hierarchy, and calls to and from the user will be available for browsing.

### Netbeans IDE 7.3

#### Description



|  |
| --- |
| The NetBeans Platform is a generic framework for Swing applications. It provides the "plumbing" that, before, every developer had to write themselves—saving state, connecting actions to menu items, toolbar items and keyboard shortcuts; window management, and so on.  The NetBeans Platform provides all of these out of the box. You don't need to manually code these or other basic features, yourself, anymore. The platform does not add a lot of overhead to your application — but it can save a huge amount of time and work. |

The NetBeans Platform provides reliable and flexible application architecture. It can save you years of development time. The NetBeans Platform gives you a time-tested architecture for free. An architecture that encourages sustainable development practices. Because the NetBeans Platform architecture is modular, it's easy to create applications that are robust and extensible.

#### Features

The main reusable features and components comprising the NetBeans Platform are outlined below.

##### Module System

The modular nature of a NetBeans Platform application gives you the power to meet complex requirements by combining several small, simple, and easily tested modules encapsulating coarsely-grained application features.

Powerful versioning support helps give you confidence that your modules will work together, while strict control over the public APIs your modules expose will help you create a more flexible application that's easier to maintain.

Since your application can use either standard NetBeans Platform modules or OSGi bundles, you'll be able to integrate third-party modules or develop your own.

##### Lifecycle Management

Just as application servers, such as GlassFish or WebLogic, provide lifecycle services to web applications, the NetBeans runtime container provide lifecycle services to Java desktop applications.

Application servers understand how to compose web modules, EJB modules, and related artifacts, into a single web application. In a comparable manner, the NetBeans runtime container understands how to compose NetBeans modules into a single Java desktop application.

There is no need to write a main method for your application because the NetBeans Platform already contains one. Also, support is provided for persisting user settings across restart of the application, such as, by default, the size and positions of the windows in the application.

##### Pluggability, Service Infrastructure, and File System

End users of the application benefit from pluggable applications because these enable them to install modules into their running applications.

NetBeans modules can be installed, uninstalled, activated, and deactivated at runtime, thanks to the runtime container.

The NetBeans Platform provides an infrastructure for registering and retrieving service implementations, enabling you to minimize direct dependencies between individual modules and enabling a loosely coupled architecture (high cohesion and low coupling).

The NetBeans Platform provides a virtual file system, which is a hierarhical registry for storing user settings, comparable to the Windows Registry on Microsoft Windows systems. It also includes a unified API providing stream-oriented access to flat and hierarchical structures, such as disk-based files on local or remote servers, memory-based files, and even XML documents.

##### Window System, Standardized UI Toolkit, and Advanced Data-Oriented Components

Most serious applications need more than one window. Coding good interaction between multiple windows is not a trivial task. The NetBeans window system lets you maximize/minimize, dock/undock, and drag-and-drop windows, without you providing any code at all.

Swing and JavaFX are the standard UI toolkits on the Java desktop and can be used throughout the NetBeans Platform. Related benefits include the ability to change the look and feel easily via "Look and Feel" support in Swing and CSS integration in JavaFX, as well as the portability of GUI components across all operating systems and the easy incorporation of many free and commercial third-party Swing and JavaFX components.

With the NetBeans Platform you're not constrained by one of the typical pain points in Swing: the JTree model is completely different to the JList model, even though they present the same data. Switching between them means rewriting the model. The NetBeans Nodes API provides a generic model for presenting your data. The NetBeans Explorer & Property Sheet API provides several advanced Swing components for displaying nodes.

In addition to a window system, the NetBeans Platform provides many other UI-related components, such as a property sheet, a palette, complex Swing components for presenting data, a Plugin Manager, and an Output window.

##### Miscellaneous Features, Documentation, and Tooling Support

The NetBeans IDE, which is the software development kit (SDK) of the NewtBeans Platform, provides many templates and tools, such as the award winning Matisse GUI Builder that enables you to very easily design your application's layout.

#### Advantages of NetBeans:

##### Best support for latest java technologies:

NetBeans IDE provides first-class comprehensive support for the newest Java technologies and latest Java enhancements before other IDEs. It is the first IDE providing support for JDK 7, Java EE 6, and JavaFX 2.   
  
With its constantly improving Java Editor, many rich features and an extensive range of tools, templates and samples, NetBeans IDE sets the standard for developing with cutting edge technologies out of the box.

##### Fast & smart code editing

An IDE is much more than a text editor. The NetBeans Editor indents lines, matches words and brackets, and highlights source code syntactically and semantically. It also provides code templates, coding tips, and refactoring tools.   
  
The editor supports many languages from Java, C/C++, XML and HTML, to PHP, Groovy, Javadoc, JavaScript and JSP. Because the editor is extensible, you can plug in support for many other languages.

##### Easy & efficient project management

Keeping a clear overview of large applications, with thousands of folders and files, and millions of lines of code, is a daunting task. NetBeans IDE provides different views of your data, from multiple project windows to helpful tools for setting up your applications and managing them efficiently, letting you drill down into your data quickly and easily, while giving you versioning tools via Subversion, Mercurial, and Git integration out of the box.  
  
When new developers join your project, they can understand the structure of your application because your code is well-organized.

##### Rapid user interface development

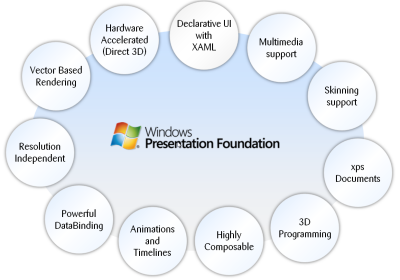
Design GUIs for Java EE, Java SE, and Java ME applications quickly and smoothly by dragging and positioning GUI components from a palette into the NetBeans Editor.   
  
For Java SE applications, the NetBeans GUI Builder automatically takes care of correct spacing and alignment, while supporting in-place editing, as well. The GUI builder is so intuitive that it has been used to prototype GUIs at customer presentations.

##### Write bug free code

The cost of buggy code increases the longer it remains unfixed. NetBeans provides static analysis tools, especially integration with the widely used FindBugs tool, for identifying and fixing common problems in Java code. In addition, the NetBeans Debugger lets you place breakpoints in your source code, add field watches, step through your code, run into methods, take snapshots and monitor execution as it occurs.  
  
The NetBeans Profiler provides expert assistance for optimizing your application's speed and memory usage, and makes it easier to build reliable and scalable Java SE, JavaFX and Java EE applications. NetBeans IDE includes a visual debugger for Java SE applications, letting you debug user interfaces without looking into source code. Take GUI snapshots of your applications and click on user interface elements to jump back into the related source code.

## Front End

### WPF (Windows Presentation Framework)



Windows Presentation Foundation (WPF) is a next-generation presentation system for building Windows client applications with visually stunning user experiences. With WPF, you can create a wide range of both standalone and browser-hosted applications.

Windows Presentation Foundation (WPF) provides developers with a unified programming model for building rich Windows smart client user experiences that incorporate UI, media, and documents. Windows Presentation Foundation (WPF) is a next-generation presentation system for building Windows client applications with visually stunning user experiences. With WPF, you can create a wide range of both standalone and browser-hosted applications. The core of WPF is a resolution-independent and vector-based rendering engine that is built to take advantage of modern graphics hardware. WPF extends the core with a comprehensive set of application-development features that include Extensible Application Markup Language (XAML), controls, data binding, layout, 2-D and 3-D graphics, animation, styles, templates, documents, media, text, and typography. WPF is included in the Microsoft .NET Framework, so you can build applications that incorporate other elements of the .NET Framework class library.

The core of WPF is a resolution-independent and vector-based rendering engine that is built to take advantage of modern graphics hardware. WPF extends the core with a comprehensive set of application-development features that include Extensible Application Markup Language (XAML), controls, data binding, layout, 2-D and 3-D graphics, animation, styles, templates, documents, media, text, and typography. WPF is included in the Microsoft .NET Framework, so you can build applications that incorporate other elements of the .NET Framework class library.

#### Programming with wpf

WPF exists as a subset of .NET Framework types that are for the most part located in the [System.Windows](http://msdn.microsoft.com/en-IN/library/system.windows.aspx) namespace. If you have previously built applications with .NET Framework using managed technologies like ASP.NET and Windows Forms, the fundamental WPF programming experience should be familiar; you instantiate classes, set properties, call methods, and handle events, all using your favorite .NET Framework programming language, such as C# or Visual Basic.

#### Markup & code-behind

WPF offers additional programming enhancements for Windows client application development. One obvious enhancement is the ability to develop an application using both *markup* and *code-behind*, an experience that ASP.NET developers should be familiar with. You generally use Extensible Application Markup Language (XAML) markup to implement the appearance of an application while using managed programming languages (code-behind) to implement its behavior.

#### security

Because XBAPs are hosted in a browser, security is important. In particular, a partial-trust security sandbox is used by XBAPs to enforce restrictions that are less than or equal to the restrictions imposed on HTML-based applications. Furthermore, each HTML feature that is safe to run from XBAPs in partial trust has been tested using a comprehensive security process.

#### controls

The user experiences that are delivered by the application model are constructed controls. In WPF, "control" is an umbrella term that applies to a category of WPF classes that are hosted in either a window or a page, have a user interface (UI), and implement some behavior.

#### Wpf controls by function

The built-in WPF controls are listed here.

* **Buttons**: [Button](http://msdn.microsoft.com/en-IN/library/system.windows.controls.button.aspx) and [RepeatButton](http://msdn.microsoft.com/en-IN/library/system.windows.controls.primitives.repeatbutton.aspx).
* **Data Display**: [DataGrid](http://msdn.microsoft.com/en-IN/library/system.windows.controls.datagrid.aspx), [ListView](http://msdn.microsoft.com/en-IN/library/system.windows.controls.listview.aspx),and [TreeView](http://msdn.microsoft.com/en-IN/library/system.windows.controls.treeview.aspx).
* **Date Display and Selection**: [Calendar](http://msdn.microsoft.com/en-IN/library/system.windows.controls.calendar.aspx) and [DatePicker](http://msdn.microsoft.com/en-IN/library/system.windows.controls.datepicker.aspx).
* **Dialog Boxes**: [OpenFileDialog](http://msdn.microsoft.com/en-IN/library/microsoft.win32.openfiledialog.aspx), [PrintDialog](http://msdn.microsoft.com/en-IN/library/system.windows.controls.printdialog.aspx), and [SaveFileDialog](http://msdn.microsoft.com/en-IN/library/microsoft.win32.savefiledialog.aspx).
* **Digital Ink**: [InkCanvas](http://msdn.microsoft.com/en-IN/library/system.windows.controls.inkcanvas.aspx) and [InkPresenter](http://msdn.microsoft.com/en-IN/library/system.windows.controls.inkpresenter.aspx).
* **Documents**: [DocumentViewer](http://msdn.microsoft.com/en-IN/library/system.windows.controls.documentviewer.aspx), [FlowDocumentPageViewer](http://msdn.microsoft.com/en-IN/library/system.windows.controls.flowdocumentpageviewer.aspx), [FlowDocumentReader](http://msdn.microsoft.com/en-IN/library/system.windows.controls.flowdocumentreader.aspx), [FlowDocumentScrollViewer](http://msdn.microsoft.com/en-IN/library/system.windows.controls.flowdocumentscrollviewer.aspx), and[StickyNoteControl](http://msdn.microsoft.com/en-IN/library/system.windows.controls.stickynotecontrol.aspx).
* **Input**: [TextBox](http://msdn.microsoft.com/en-IN/library/system.windows.controls.textbox.aspx), [RichTextBox](http://msdn.microsoft.com/en-IN/library/system.windows.controls.richtextbox.aspx), and [PasswordBox](http://msdn.microsoft.com/en-IN/library/system.windows.controls.passwordbox.aspx).
* **Layout**: [Border](http://msdn.microsoft.com/en-IN/library/system.windows.controls.border.aspx), [BulletDecorator](http://msdn.microsoft.com/en-IN/library/system.windows.controls.primitives.bulletdecorator.aspx), [Canvas](http://msdn.microsoft.com/en-IN/library/system.windows.controls.canvas.aspx), [DockPanel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.dockpanel.aspx), [Expander](http://msdn.microsoft.com/en-IN/library/system.windows.controls.expander.aspx), [Grid](http://msdn.microsoft.com/en-IN/library/system.windows.controls.grid.aspx), [GridView](http://msdn.microsoft.com/en-IN/library/system.windows.controls.gridview.aspx), [GridSplitter](http://msdn.microsoft.com/en-IN/library/system.windows.controls.gridsplitter.aspx), [GroupBox](http://msdn.microsoft.com/en-IN/library/system.windows.controls.groupbox.aspx), [Panel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.panel.aspx),[ResizeGrip](http://msdn.microsoft.com/en-IN/library/system.windows.controls.primitives.resizegrip.aspx), [Separator](http://msdn.microsoft.com/en-IN/library/system.windows.controls.separator.aspx), [ScrollBar](http://msdn.microsoft.com/en-IN/library/system.windows.controls.primitives.scrollbar.aspx), [ScrollViewer](http://msdn.microsoft.com/en-IN/library/system.windows.controls.scrollviewer.aspx), [StackPanel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.stackpanel.aspx), [Thumb](http://msdn.microsoft.com/en-IN/library/system.windows.controls.primitives.thumb.aspx), [Viewbox](http://msdn.microsoft.com/en-IN/library/system.windows.controls.viewbox.aspx), [VirtualizingStackPanel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.virtualizingstackpanel.aspx), [Window](http://msdn.microsoft.com/en-IN/library/system.windows.window.aspx), and[WrapPanel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.wrappanel.aspx).
* **Media**: [Image](http://msdn.microsoft.com/en-IN/library/system.windows.controls.image.aspx), [MediaElement](http://msdn.microsoft.com/en-IN/library/system.windows.controls.mediaelement.aspx), and [SoundPlayerAction](http://msdn.microsoft.com/en-IN/library/system.windows.controls.soundplayeraction.aspx).
* **Menus**: [ContextMenu](http://msdn.microsoft.com/en-IN/library/system.windows.controls.contextmenu.aspx), [Menu](http://msdn.microsoft.com/en-IN/library/system.windows.controls.menu.aspx), and [ToolBar](http://msdn.microsoft.com/en-IN/library/system.windows.controls.toolbar.aspx).
* **Navigation**: [Frame](http://msdn.microsoft.com/en-IN/library/system.windows.controls.frame.aspx), [Hyperlink](http://msdn.microsoft.com/en-IN/library/system.windows.documents.hyperlink.aspx), [Page](http://msdn.microsoft.com/en-IN/library/system.windows.controls.page.aspx), [NavigationWindow](http://msdn.microsoft.com/en-IN/library/system.windows.navigation.navigationwindow.aspx), and [TabControl](http://msdn.microsoft.com/en-IN/library/system.windows.controls.tabcontrol.aspx).
* **Selection**: [CheckBox](http://msdn.microsoft.com/en-IN/library/system.windows.controls.checkbox.aspx), [ComboBox](http://msdn.microsoft.com/en-IN/library/system.windows.controls.combobox.aspx), [ListBox](http://msdn.microsoft.com/en-IN/library/system.windows.controls.listbox.aspx), [RadioButton](http://msdn.microsoft.com/en-IN/library/system.windows.controls.radiobutton.aspx), and [Slider](http://msdn.microsoft.com/en-IN/library/system.windows.controls.slider.aspx).
* **User Information**: [AccessText](http://msdn.microsoft.com/en-IN/library/system.windows.controls.accesstext.aspx), [Label](http://msdn.microsoft.com/en-IN/library/system.windows.controls.label.aspx), [Popup](http://msdn.microsoft.com/en-IN/library/system.windows.controls.primitives.popup.aspx), [ProgressBar](http://msdn.microsoft.com/en-IN/library/system.windows.controls.progressbar.aspx), [StatusBar](http://msdn.microsoft.com/en-IN/library/system.windows.controls.primitives.statusbar.aspx), [TextBlock](http://msdn.microsoft.com/en-IN/library/system.windows.controls.textblock.aspx), and [ToolTip](http://msdn.microsoft.com/en-IN/library/system.windows.controls.tooltip.aspx).

#### layout

When you create a UI, you arrange your controls by location and size to form a layout. A key requirement of any layout is to adapt to changes in window size and display settings. Rather than forcing you to write the code to adapt a layout in these circumstances, WPF provides a first-class, extensible layout system for you.

The cornerstone of the layout system is relative positioning, which increases the ability to adapt to changing window and display conditions. In addition, the layout system manages the negotiation between controls to determine the layout. The negotiation is a two-step process: first, a control tells its parent what location and size it requires; second, the parent tells the control what space it can have.

The layout system is exposed to child controls through base WPF classes. For common layouts such as grids, stacking, and docking, WPF includes several layout controls:

* [Canvas](http://msdn.microsoft.com/en-IN/library/system.windows.controls.canvas.aspx) : Child controls provide their own layout.
* [DockPanel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.dockpanel.aspx) : Child controls are aligned to the edges of the panel.
* [Grid](http://msdn.microsoft.com/en-IN/library/system.windows.controls.grid.aspx) : Child controls are positioned by rows and columns.
* [StackPanel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.stackpanel.aspx) : Child controls are stacked either vertically or horizontally.
* [VirtualizingStackPanel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.virtualizingstackpanel.aspx) : Child controls are virtualized and arranged on a single line that is either horizontally or vertically oriented.
* [WrapPanel](http://msdn.microsoft.com/en-IN/library/system.windows.controls.wrappanel.aspx) : Child controls are positioned in left-to-right order and wrapped to the next line when there are more controls on the current line than space allows.

#### graphics

WPF introduces an extensive, scalable, and flexible set of graphics features that have the following benefits:

* **Resolution-independent and device-independent graphics**. The basic unit of measurement in the WPF graphics system is the device independent pixel, which is 1/96th of an inch, regardless of actual screen resolution, and provides the foundation for resolution-independent and device-independent rendering. Each device-independent pixel automatically scales to match the dots-per-inch (dpi) setting of the system it renders on.
* **Improved precision**. The WPF coordinate system is measured with double-precision floating-point numbers rather than single-precision. Transformations and opacity values are also expressed as double-precision. WPF also supports a wide color gamut (scRGB) and provides integrated support for managing inputs from different color spaces.
* **Advanced graphics and animation support**. WPF simplifies graphics programming by managing animation scenes for you; there is no need to worry about scene processing, rendering loops, and bilinear interpolation. Additionally, WPF provides hit-testing support and full alpha-compositing support.
* **Hardware acceleration**. The WPF graphics system takes advantage of graphics hardware to minimize CPU usage.

### Extensible application Markup Language (XaML)



XAML stands for Extensible Application Markup Language. Its a simple language based on XML to create and initialize .NET objects with hierarchical relations. Although it was originally invented for WPF it can by used to create any kind of object trees.

Today XAML is used to create user interfaces in WPF, Silverlight, declare workflows in WF and for electronic paper in the XPS standard.

All classes in WPF have parameter less constructors and make excessive usage of properties. That is done to make it perfectly fit for XML languages like XAML.

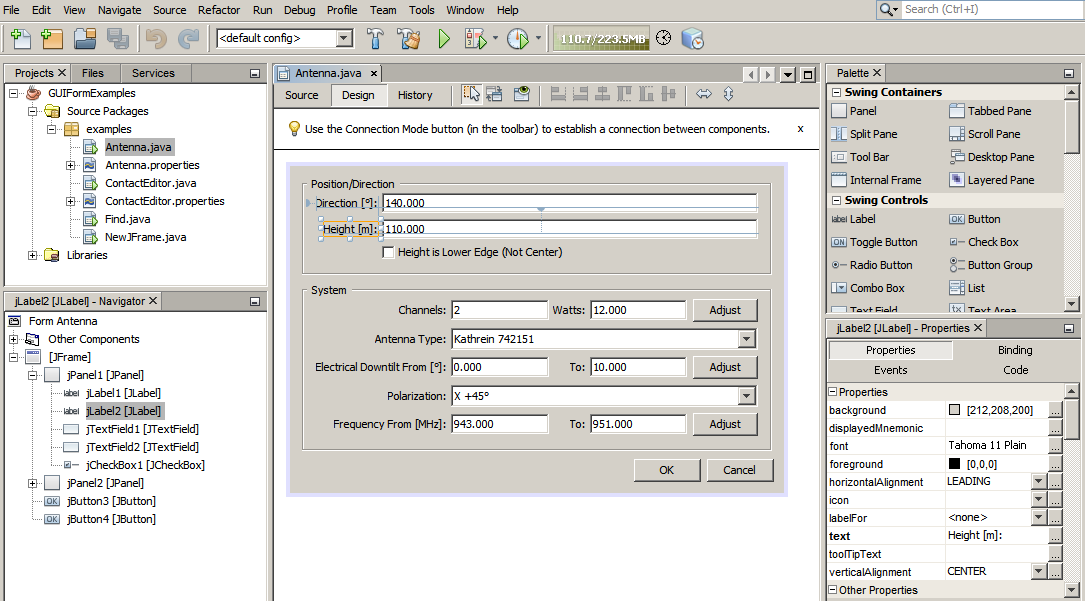
All you can do in XAML can also be done in code. XAML ist just another way to create and initialize objects. You can use WPF without using XAML. It's up to you if you want to declare it in XAML or write it in code. Declare your UI in XAML has some advantages:

* XAML code is short and clear to read
* Separation of designer code and logic
* Graphical design tools like Expression Blend require XAML as source.
* The separation of XAML and UI logic allows it to clearly separate the roles of designer and developer.

### Swing

Swing is the primary Java GUI widget toolkit. It is part of Oracle's Java Foundation Classes (JFC) — an API for providing a graphical user interface (GUI) for Java programs. Swing was developed to provide a more sophisticated set of GUI components than the earlier Abstract Window Toolkit (AWT). Swing provides a native look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

Unlike AWT components, Swing components are not implemented by platform-specific code. Instead they are written entirely in Java and therefore are platform-independent. The term "lightweight" is used to describe such an element.

The Java Swing provides the multiple platform independent APIs interfaces for interacting between the users and GUIs components. All Java Swing classes imports form the import javax.swing.\*; package.  Java provides an interactive features for design the **GUI**s toolkit or components like: labels, buttons, text boxes, checkboxes, combo boxes, panels and sliders etc. All AWT flexible components can be handled by the Java Swing. The Java Swing supports the plugging between the look and feel features. The look and feel that means the dramatically changing in the component like JFrame, JWindow, JDialog etc. for viewing it into the several types of window.

Here the following APIs interfaces and classes are available:

The following interfaces and it's descriptions to be used by the Java swing.

|  |  |
| --- | --- |
| Interfaces | Descriptions |
| Action | This interface performed the action with the ActionListenerwhere the multiple controls are used for same purposes. |
| BoundedRangeModel | This interface defines the data model of components like: sliders and progressBars. |
| ButtonModel | It defines the state model for the buttons like: radio buttons, check boxes etc. |
| CellEditor | This interface used by the developer for creating the new editor and it has the new components implement interfaces. TheCellEditor implements the wrapper based approach. |
| ComboBoxEditor | In this interface, the editor component used to JComboBoxcomponents. |
| ComboBoxModel | This interface represents the data model in a list model with the selected items. |
| DesktopManager | This interface has JDesktopPane object. The JInternalFrameimplements in the JDesktopPane with the help of DesktopManager. |
| Icon | This interface used to graphical representation of the components. It has fixed size picture. |
| JComboBox.KeySelectionManager | This interface has KeySelectionManager and used for the combo box data model. |
| ListCellRenderer | This interface used for paint the cell in the list with the help of "rubber stamps" . |
| ListModel | This interface used for JList components method. It gets the value of each cell of list. |
| ListSelectionModel | This interface indicates the components, which are stable or not. |
| MenuElement | This interface used where the any components are implements in the menu. |
| MutableComboBoxModel | This interface extends from the ComboBoxModel. It is a mutable version of ComboBoxModel. |
| Renderer | It defines the requirements of an object for displaying the values. |
| RootPaneContainer | This interface uses the RootPane properties and it has the components like: JFrame, JInternalFrame and JWindow etc. |
| Scrollable | This interface provides the scrolling to show the large amount of data with the help of JScrollPane. |
| ScrollPaneConstants | This interface used for JScrollPane components. |
| SingleSelectionModel | This interface used to select the one index in a model. |
| SwingConstants | You can set the components on the screen to own requirements. |
| UIDefaults.ActiveValue | It constructs the DefaultListCellRenderer. |
| UIDefaults.LazyValue | This enables one to store an entry in the default table. The entered value is not constructed until first time is a real value is created through it using LazyValue.createValue() method. |
| WindowConstants | This interface has two methods setDefaultCloseOperation and getDefaultCloseOperation and provides the window close opration. |

The following classes and it's descriptions to  be used by the Java swing.

|  |  |
| --- | --- |
| Classes | Descriptions |
| AbstractAction | This class handles the any types of action and provides JFC Action interface. |
| AbstractButton | This class defines the nature of buttons and menu items. |
| AbstractCellEditor | It provides a list and contents of the data model. |
| AbstractListModel | This class defines the data model which provides the list with its contents. |
| ActionMap | This class works with InputMap and performs any action when the key is pressed. |
| BorderFactory | This class extends from Object and creates the Border instance in the factory. |
| Box | It provides the fixed spaces between two components and uses the BoxLayout object of the layout manager. |
| Box.Filler | This class participates in the Layout and uses the lightweight components. |
| BoxLayout | This class uses the arranging the multiple components either horizontally or vertically. The Box container uses this class. |
| ButtonGroup | This class used to create the multiple buttons in aButtonGroup object. |
| CellRandererPane | This class used to insert the components like: JList, JTable and JTree. |
| ComponentInputMap | This class has ComponentInputMap constructor and creates the components with the help of InpuMap. |
| DebugGraphics | It extends from the Graphics and used to debug the graphics |
| DefaultBoundedRangeModel | This class provides the implementation of default BoundedRangeModel. |
| DefaultButtonModel | This class implements the generic ButtonModel. |
| DefaultCellEditor | It implements the TableCellEditor and TreeCellEditor for the table and tree cells. |
| DefaultComboBoxModel | It provides the default model for combo boxes. |
| DefaultDesktopManager | It implements the DesktopManager. The DesktopManager has the JInternalFrame for creating the internal fame in a frame. |
| DefaultFocusManager | It provides the implementing the FocusManager. |
| DefaultListCellRanderer | It implements the default ListCellRanderer. |
| DefaultListCellRanderer.UIResource | This extends the DefaultListCellRanderer and implementing in the UIResource. |
| DefaultListModel | It extends the AbstractListModel and implementing thejava.util.Vector. |
| DefaultListSelectionModel | This class used for select the list in a data model. |
| DefaultSingleSelectionModel | This class provides the default SingleSelectionModel. |
| FocusManager | It handles all focus like: gainedFocus and lostFocus. |
| GrayFilter | It extends the RGBImageFilter and used for disabling the image through the button. |
| ImageIcon | This class implements the Icon and paints the icons from the images. |
| InputMap | This class uses the ActionMap to performed the action when you press any key of keyboard. It bounds data between the input event and an object. |
| InputVerifier | This class helps you when you works with the text fields through the focus. |
| JApplet | This class extends the Applet and implements the Accessibleand RootPaneContainer. |
| JButton | This class extends the AbstractButton and you can create the new button. |
| JCheckBox | This class extends the JToggleButton and implements the check box in which buttons are selected or deselected. |
| JCheckBoxMenuItem | It extends the JMenuItem and determines the items which is selected or deselected. |
| JColorChooser | It extends the JComponent and implementing the Accessable. Here, you choose and manipulate the colors. |
| JComboBox | This class extends the JComboBox. It provides the drop-down list where user select only one item or value at a time. But combo box is a combination of multiple text or buttons etc. |
| JComponent | In java swing, All components are used the JComponent except the top-level containers like: JFrame, JDialog etc. |
| JDesktopPane | This class extends the JLayeredPane and when you create the object of JInternalFrame to be maintained in the JDesktopPane. The JDesktopPane has DesktopManager. |
| JDialog | It extends the Dialog. This class used to create the dialog window and when you want to create the custom dialog window with the help of JOptionPane method. |
| JEditorPane | This class extends the JTextComponent. It edits the component by the EditorKit. |
| JFileChooser | This class provides the facility to choosing the file. |
| JFrame | It extends the Frame and supports the swing components architecture. |
| JInternalFrame | This class extends from the JComponent and provides the facility to dragging, closing, resizing and menu bar of the internal frame. The JInternalFrame added into the JDesktopPane. |
| JInternalFrame.JDesktopIcon | It displays the desktop icon and create the instance of JInternalFrame and iconify. |
| JLabel | This class used to show the small text and image. |
| JLayeredPane | It has JFC/Swing container that can be used to overlap the components to each other. |
| JList | This class used to create a list where you select the one or more than objects. |
| JMenu | This class used to create a new menu where you add the JMenuItems. When you select the item then shows the popup menu items in the JMenuBar. |
| JMenuBar | It used to create a new menu bar where the JMenu objects are added. |
| JMenuItem | This class used to create new menu items in the mebus. |
| JOptionPane | It used to create some different types of dialog box like: message dialog box, error dialog box etc. |
| JPanel | It extends the JComponent and used to create a new panel. |
| JPassworkField | It provides the single line text editing. Here, don't available the original characters but view type indication characters are available. |
| JPopupMenu | This class used to create a popup menu. It provides small window where the various types of choices are available. |
| JPopupMenu.Separator | Here the popup menu and the separator are available. |
| JProgressBar | It shows the integer types values in percent within a bounded range to determine the working process. |
| JRadioButton | It implements the radio button and shows the state of an item selected or deselected. |
| JRadioButtonMenuItem | It extends the JMenuItem and implements the radio button menu item |
| JRootPane | This class provides the component behind the scenes by JFrame, JWindow, JDialog etc. for providing the task-orientation and functionality. |
| JScrollBar | This class used to create a scroll bar. It provides the view content area where you show the content to scroll this. |
| JScrollPane | It provides the scrollable view components. |
| JSeparator | This class use the separator among the components. |
| JSlider | This class provides a control to represent a numeric value by dragging the slider. |
| JSplitPane | This class used to divides the two components graphically like: top and button, left and right. |
| JTabbedPane | This class provides the tab component through which you can switch from one component to another component regarding to the specific tab button by clicking on that. |
| JTable | It provides the user interface component and represents the two dimensional data. |
| JTextArea | It provides the multi line plain text area. |
| JTextField | It provides the facility to editing the text in a single line. |
| JTextPane | This class provides the component like JTexArea for multiple lines text with more capabalities. |
| JToggleButton | It implements two state button that means selected or deselected. |
| JToggleButton.ToggleButtonModel | It extends the DefaultButtonModel and provides theToggleButton model. |
| JToolBar | It provides set of command buttons icons that performs the different actions or controls. |
| JToolBar.Separator | It provides the tool bar separator. |
| JToolTip | It shows the tool tips related to it's components. |
| JTree | It shows the data in a hierarchical way. |
| JTree.DynamicUtilTreeNode | This extends the DefaultMutableTreeNode and create children nodes. |
| JTree.EmptySelectionModel | It does not allows the any selection. |
| JViewPort | It gives you about the underlying information. |
| JWindow | It extends window and shows the any location or area on the desktop. It couldn't any title bar and window management buttons. |
| KeyStroke | This class controls the key events on the keyboard for the equivalent device. |
| LayoutFocusTraversalPolicy | This class conducts the sorting objects according to their size, type, position or orientation. |
| LookAndFeel | It provides the dramatically changes in the component like frame related to the graphics for the application. Through this the application can be done very efficient and easier. |
| MenuSelectionManager | It has menu selection hierarchy. |
| OverlayLayout | The layout manager arrange the components. |
| ProgressMonitor | This class is used to monitoring the progress of any operation to be done. |
| ProgressMonitorInputStream | This class creates a progress monitor to monitor the progress of reading input from the input stream. It cleanups all the rights when the stream is closed. |
| RepaintManager | This class manage and override the repaint requests. |
| ScrollPaneLayout | It implements the LayoutManager and manage the components like: scroll bar, row header, column header etc. |
| ScrollPaneLayout.UIResource | It extends the ScrollPaneLayout and implements theUIResource. |
| SizeRequirements | It calculates the size and positions of components. |
| SizeSequence | It represents the order list of size and it's positions. |
| SwingUtilities | This class has utilities methods for swing. |
| Timer | Actions perform the predefined rate. |
| ToolTipManager | It manages the all tool tips. |
| UIDefaults | It extends the Hashtable and you set/get the value with the help of UIManager. |
| UIDefaults.LazyInputMap | This class creates a Input Map through it's createValue() method. The array of key after binding is passed to the constructor of this. Example of binding of key is array of pressing key information (e.g. ctrl + c or alt + f). |
| UIDefaults.ProxyLazyValue | This class is used to create a lazy value which is used to delay loading of the class to create instance for that. |
| UIManager | This class has track of the current look and feel details. |
| UIManager.LookAndFeelInfo | This is the nested class of UIManager class i.e. used for getting information about all the look and feels installed with the software development kit. |
| ViewportLayout | It implements the LayoutManager and defines the policy for the layout. |

The following Exceptions and it's description to be used by the Java swing.

|  |  |
| --- | --- |
| Exception | Descriptions |
| UnsupportedLookAndFeelException | This exception occurred when the look and feel classes are not supported to user's system. |

## Programming Framework

### .NET 4.5



The .NET Framework is a development platform for building apps for Windows, Windows Phone, Windows Server, and Windows Azure. It consists of the common language runtime (CLR) and the .NET Framework class library, which includes classes, interfaces, and value types that support an extensive range of technologies. The .NET Framework provides a managed execution environment, simplified development and deployment, and integration with a variety of programming languages, including Visual Basic and Visual C#.

#### .net framework class libraries

The .NET Framework class library is a library of classes, interfaces, and value types that provide access to system functionality. It is the foundation on which .NET Framework applications, components, and controls are built. The namespaces and namespace categories in the class library are listed in the following table and documented in detail in this reference. The namespaces and categories are listed by usage, with the most frequently used namespaces appearing first.

|  |  |
| --- | --- |
| **Namespace** | **Description** |
| [System](http://msdn.microsoft.com/en-us/library/system.aspx) | The [System](http://msdn.microsoft.com/en-us/library/system.aspx) namespace contains fundamental classes and base classes that define commonly-used value and reference data types, events and event handlers, interfaces, attributes, and processing exceptions. |
| [System.Activities](http://msdn.microsoft.com/en-us/library/gg145022.aspx) | The System.Activities namespaces contain all the classes necessary to create and work with activities in Window Workflow Foundation. |
| [System.AddIn](http://msdn.microsoft.com/en-us/library/gg145020.aspx) | The System.AddIn namespaces contain types used to identify, register, activate, and control add-ins, and to allow add-ins to communicate with a host application. |
| [System.CodeDom](http://msdn.microsoft.com/en-us/library/gg145034.aspx) | The System.CodeDom namespaces contain classes that represent the elements of a source code document and that support the generation and compilation of source code in supported programming languages. |
| [System.Collections](http://msdn.microsoft.com/en-us/library/gg145035.aspx) | The System.Collections namespaces contain types that define various standard, specialized, and generic collection objects. |
| [System.ComponentModel](http://msdn.microsoft.com/en-us/library/gg145042.aspx) | The System.ComponentModel namespaces contain types that implement the run-time and design-time behavior of components and controls. Child namespaces support the Managed Extensibility Framework (MEF), provide attribute classes that define metadata for ASP.NET Dynamic Data controls, and contain types that let you define the design-time behavior of components and their user interfaces. |
| [System.Configuration](http://msdn.microsoft.com/en-us/library/gg145027.aspx) | The System.Configuration namespaces contain types for handling configuration data, such as data in machine or application configuration files. Child namespaces contain types that are used to configure an assembly, to write custom installers for components, and to support a pluggable model for adding functionality to, or removing functionality from, both client and server applications. |
| [System.Data](http://msdn.microsoft.com/en-us/library/gg145028.aspx) | The System.Data namespaces contain classes for accessing and managing data from diverse sources. The top-level namespace and a number of the child namespaces together form the ADO.NET architecture and ADO.NET data providers. For example, providers are available for SQL Server, Oracle, ODBC, and OleDB. Other child namespaces contain classes used by the ADO.NET Entity Data Model (EDM) and by WCF Data Services. |
| [System.Deployment](http://msdn.microsoft.com/en-us/library/gg145029.aspx) | The System.Deployment namespaces contain types that support deployment of ClickOnce applications. |
| [System.Device.Location](http://msdn.microsoft.com/en-us/library/system.device.location.aspx) | The [System.Device.Location](http://msdn.microsoft.com/en-us/library/system.device.location.aspx) namespace allows application developers to easily access the computer's location by using a single API. Location information may come from multiple providers, such as GPS, Wi-Fi triangulation, and cell phone tower triangulation. The [System.Device.Location](http://msdn.microsoft.com/en-us/library/system.device.location.aspx) classes provide a single API to encapsulate the multiple location providers on a computer and support seamless prioritization and transitioning between them. As a result, application developers who use this API do not need to tailor applications to specific hardware configurations. |
| [System.Diagnostics](http://msdn.microsoft.com/en-us/library/gg145030.aspx) | The System.Diagnostics namespaces contain types that enable you to interact with system processes, event logs, and performance counters. Child namespaces contain types to interact with code analysis tools, to support contracts, to extend design-time support for application monitoring and instrumentation, to log event data using the Event Tracing for Windows (ETW) tracing subsystem, to read to and write from event logs and collect performance data, and to read and write debug symbol information. |
| [System.DirectoryServices](http://msdn.microsoft.com/en-us/library/gg145037.aspx) | The System.DirectoryServices namespaces contain types that provide access to Active Directory from managed code. |
| [System.Drawing](http://msdn.microsoft.com/en-us/library/gg145023.aspx) | The System.Drawing parent namespace contains types that support basic GDI+ graphics functionality. Child namespaces support advanced two-dimensional and vector graphics functionality, advanced imaging functionality, and print-related and typographical services. A child namespace also contains types that extend design-time user-interface logic and drawing. |
| [System.Dynamic](http://msdn.microsoft.com/en-us/library/system.dynamic.aspx) | The [System.Dynamic](http://msdn.microsoft.com/en-us/library/system.dynamic.aspx) namespace provides classes and interfaces that support Dynamic Language Runtime. |
| [System.EnterpriseServices](http://msdn.microsoft.com/en-us/library/gg145047.aspx) | The System.EnterpriseServices namespaces contain types that define the COM+ services architecture, which provides an infrastructure for enterprise applications. A child namespace supports Compensating Resource Manager (CRM), a COM+ service that enables non-transactional objects to be included in Microsoft Distributed Transaction Coordinator (DTC) transactions. Child namespaces are described briefly in the following table and documented in detail in this reference. |
| [System.Globalization](http://msdn.microsoft.com/en-us/library/system.globalization.aspx) | The [System.Globalization](http://msdn.microsoft.com/en-us/library/system.globalization.aspx) namespace contains classes that define culture-related information, including language, country/region, calendars in use, format patterns for dates, currency, and numbers, and sort order for strings. These classes are useful for writing globalized (internationalized) applications. Classes such as [StringInfo](http://msdn.microsoft.com/en-us/library/system.globalization.stringinfo.aspx) and[TextInfo](http://msdn.microsoft.com/en-us/library/system.globalization.textinfo.aspx) provide advanced globalization functionalities, including surrogate support and text element processing. |
| [System.IdentityModel](http://msdn.microsoft.com/en-us/library/gg145031.aspx) | The System.IdentityModel namespaces contain types that are used to provide authentication and authorization for .NET applications. |
| [System.IO](http://msdn.microsoft.com/en-us/library/gg145019.aspx) | The System.IO namespaces contain types that support input and output, including the ability to read and write data to streams either synchronously or asynchronously, to compress data in streams, to create and use isolated stores, to map files to an application's logical address space, to store multiple data objects in a single container, to communicate using anonymous or named pipes, to implement custom logging, and to handle the flow of data to and from serial ports. |
| [System.Linq](http://msdn.microsoft.com/en-us/library/gg145016.aspx) | The System.Linq namespaces contain types that support queries that use Language-Integrated Query (LINQ). This includes types that represent queries as objects in expression trees. |
| [System.Management](http://msdn.microsoft.com/en-us/library/gg145024.aspx) | The System.Management namespaces contain types that provide access to management information and management events about the system, devices, and applications instrumented to the Windows Management Instrumentation (WMI) infrastructure. These namespaces also contain types necessary for instrumenting applications so that they expose their management information and events through WMI to potential customers. |
| [System.Media](http://msdn.microsoft.com/en-us/library/system.media.aspx) | The [System.Media](http://msdn.microsoft.com/en-us/library/system.media.aspx) namespace contains classes for playing sound files and accessing sounds provided by the system. |
| [System.Messaging](http://msdn.microsoft.com/en-us/library/gg145046.aspx) | The System.Messaging namespaces contain types that enable you to connect to, monitor, and administer message queues on the network and to send, receive, or peek messages. A child namespace contains classes that can be used to extend design-time support for messaging classes. |
| [System.Net](http://msdn.microsoft.com/en-us/library/gg145039.aspx) | The System.Net namespaces contain classes that provide a simple programming interface for a number of network protocols, programmatically access and update configuration settings for the System.Net namespaces, define cache policies for web resources, compose and send e-mail, represent Multipurpose Internet Mail Exchange (MIME) headers, access network traffic data and network address information, and access peer-to-peer networking functionality. Additional child namespaces provide a managed implementation of the Windows Sockets (Winsock) interface and provide access to network streams for secure communications between hosts. |
| [System.Numerics](http://msdn.microsoft.com/en-us/library/system.numerics.aspx) | The [System.Numerics](http://msdn.microsoft.com/en-us/library/system.numerics.aspx) namespace contains numeric types that complement the numeric primitives, such as [Byte](http://msdn.microsoft.com/en-us/library/system.byte.aspx), [Double](http://msdn.microsoft.com/en-us/library/system.double.aspx), and [Int32](http://msdn.microsoft.com/en-us/library/system.int32.aspx), that are defined by the .NET Framework. |
| [System.Printing](http://msdn.microsoft.com/en-us/library/gg145044.aspx) | The System.Printing namespaces contain types that support printing, that provide access to the properties of print system objects and enable rapid copying of their property settings to another object of the same type, and that support the interconversion of managed System.PrintTicket objects and unmanaged GDI DEVMODE structures. |
| [System.Reflection](http://msdn.microsoft.com/en-us/library/gg145033.aspx) | The System.Reflection namespaces contain types that provide a managed view of loaded types, methods, and fields, and that can dynamically create and invoke types. A child namespace contains types that enable a compiler or other tool to emit metadata and Microsoft intermediate language (MSIL). |
| [System.Resources](http://msdn.microsoft.com/en-us/library/gg145043.aspx) | The System.Resources namespaces contain types that enable developers to create, store, and manage an application's culture-specific resources. |
| [System.Runtime](http://msdn.microsoft.com/en-us/library/gg145017.aspx) | The System.Runtime namespaces contain types that support an application's interaction with the common language runtime, and types that enable features such as application data caching, advanced exception handling, application activation within application domains, COM interop, distributed applications, serialization and deserialization, and versioning. Additional namespaces enable compiler writers to specify attributes that affect the run-time behavior of the common language runtime, define a contract for reliability between a set of code and other code that takes a dependency on it, and implement a persistence provider for Windows Communication Foundation (WCF). |
| [System.Security](http://msdn.microsoft.com/en-us/library/gg145025.aspx) | The System.Security namespaces contain classes that represent the .NET Framework security system and permissions. Child namespaces provide types that control access to and audit securable objects, allow authentication, provide crytographic services, control access to operations and resources based on policy, and support rights management of application-created content. |
| [System.ServiceModel](http://msdn.microsoft.com/en-us/library/gg145010.aspx) | The System.ServiceModel namespaces contain the types necessary to build Windows Communication Foundation (WCF) service and client applications. |
| [System.ServiceProcess](http://msdn.microsoft.com/en-us/library/gg145038.aspx) | The System.ServiceProcess namespaces contain types that enable you to implement, install, and control Windows service applications and extend design-time support for Windows service applications. |
| [System.Speech](http://msdn.microsoft.com/en-us/library/gg145021.aspx) | The System.Speech namespaces contain types that support speech recognition. |
| [System.Text](http://msdn.microsoft.com/en-us/library/gg145012.aspx) | The System.Text namespaces contain types for character encoding and string manipulation. A child namespace enables you to process text using regular expressions. |
| [System.Threading](http://msdn.microsoft.com/en-us/library/gg145014.aspx) | The System.Threading namespaces contain types that enable multithreaded programming. A child namespace provides types that simplify the work of writing concurrent and asynchronous code. |
| [System.Timers](http://msdn.microsoft.com/en-us/library/system.timers.aspx) | The [System.Timers](http://msdn.microsoft.com/en-us/library/system.timers.aspx) namespace provides the [Timer](http://msdn.microsoft.com/en-us/library/system.timers.timer.aspx) component, which allows you to raise an event on a specified interval. |
| [System.Transactions](http://msdn.microsoft.com/en-us/library/gg145032.aspx) | The System.Transactions namespaces contain types that support transactions with multiple, distributed participants, multiple phase notifications, and durable enlistments. A child namespace contains types that describe the configuration options used by the System.Transactions types. |
| [System.Web](http://msdn.microsoft.com/en-us/library/gg145018.aspx) | The System.Web namespaces contain types that enable browser/server communication. Child namespaces include types that support ASP.NET forms authentication, application services, data caching on the server, ASP.NET application configuration, dynamic data, HTTP handlers, JSON serialization, incorporating AJAX functionality into ASP.NET, ASP.NET security, and web services. |
| [System.Windows](http://msdn.microsoft.com/en-us/library/gg145013.aspx) | The System.Windows namespaces contain types used in Windows Presentation Foundation (WPF) applications, including animation clients, user interface controls, data binding, and type conversion. System.Windows.Forms and its child namespaces are used for developing Windows Forms applications. |
| [System.Workflow](http://msdn.microsoft.com/en-us/library/gg145026.aspx) | The System.Workflow namespaces contain types used to develop applications that use Windows Workflow Foundation. These types provide design time and run-time support for rules and activities, to configure, control, host, and debug the workflow runtime engine. |
| [System.Xaml](http://msdn.microsoft.com/en-us/library/gg145048.aspx) | The System.Xaml namespaces contain types that support parsing and processing the Extensible Application Markup Language (XAML). |
| [System.Xml](http://msdn.microsoft.com/en-us/library/gg145036.aspx) | The System.Xml namespaces contain types for processing XML. Child namespaces support serialization of XML documents or streams, XSD schemas, XQuery 1.0 and XPath 2.0, and LINQ to XML, which is an in-memory XML programming interface that enables easy modification of XML documents. |
| [Accessibility](http://msdn.microsoft.com/en-us/library/accessibility.aspx) | The [Accessibility](http://msdn.microsoft.com/en-us/library/accessibility.aspx) and all of its exposed members are part of a managed wrapper for the Component Object Model (COM) accessibility interface. |
| [Microsoft.Activities](http://msdn.microsoft.com/en-us/library/hh135392.aspx) | The Microsoft.Activities namespaces contain types that support MSBuild and debugger extensions for Windows Workflow Foundation applications. |
| [Microsoft.Aspnet.Snapin](http://msdn.microsoft.com/en-us/library/microsoft.aspnet.snapin.aspx) | The [Microsoft.Aspnet.Snapin](http://msdn.microsoft.com/en-us/library/microsoft.aspnet.snapin.aspx) namespace defines the types necessary for the ASP.NET management console application to interact with Microsoft Management Console (MMC). For more information, see "MMC Programmer's Guide" in the [MSDN Library](http://go.microsoft.com/fwlink/?linkid=37118). |
| [Microsoft.Build](http://msdn.microsoft.com/en-us/library/gg145008.aspx) | The Microsoft.Build namespaces contain types that provide programmatic access to, and control of, the MSBuild engine. |
| [Microsoft.CSharp](http://msdn.microsoft.com/en-us/library/gg145015.aspx) | The Microsoft.CSharp namespaces contain types that support compilation and code generation of source code written in the C# language, and types that support interoperation betwen the dynamic language runtime (DLR) and C#. |
| [Microsoft.Data.Entity.Build.Tasks](http://msdn.microsoft.com/en-us/library/microsoft.data.entity.build.tasks.aspx) | The [Microsoft.Data.Entity.Build.Tasks](http://msdn.microsoft.com/en-us/library/microsoft.data.entity.build.tasks.aspx) namespace contains two MSBuild tasks that are used by the ADO.NET Entity Data Model Designer (Entity Designer). |
| [Microsoft.JScript](http://msdn.microsoft.com/en-us/library/gg145041.aspx) | The Microsoft.JScript namespaces contain classes that support compilation and code generation using the JScript language. |
| [Microsoft.SqlServer.Server](http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.server.aspx) | The [Microsoft.SqlServer.Server](http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.server.aspx) namespace contains classes, interfaces, and enumerations that are specific to the integration of the Microsoft .NET Framework common language runtime (CLR) into Microsoft SQL Server, and the SQL Server database engine process execution environment. |
| [Microsoft.VisualBasic](http://msdn.microsoft.com/en-us/library/gg145009.aspx) | The Microsoft.VisualBasic namespaces contain classes that support compilation and code generation using the Visual Basic language. Child namespaces contain types that provide services to the Visual Basic compiler and types that include support for the Visual Basic application model, the My namespace, lambda expressions, and code conversion. |
| [Microsoft.VisualC](http://msdn.microsoft.com/en-us/library/gg145040.aspx) | The Microsoft.VisualC namespaces contain types that support the Visual C++ compiler and types that implement the STL/CLR Library and the generic interface to the STL/CLR Library. |
| [Microsoft.Win32](http://msdn.microsoft.com/en-us/library/gg145011.aspx) | The Microsoft.Win32 namespaces provide types that handle events raised by the operating system, that manipulate the system registry, and that represent file and operating system handles. |
| [Microsoft.Windows](http://msdn.microsoft.com/en-us/library/hh135393.aspx) | The Microsoft.Windows namespaces contain types that support themes and preview in Windows Presentation Framework (WPF) applications. |
| [UIAutomationClientsideProviders](http://msdn.microsoft.com/en-us/library/uiautomationclientsideproviders.aspx) | Contains a single type that maps client automation providers. |
| [XamlGeneratedNamespace](http://msdn.microsoft.com/en-us/library/xamlgeneratednamespace.aspx) | Contains compiler-generated types that are not intended to be used directly from your code. |

## Database/backend:

### MySQL



MySQL is the world's most popular open source database software, with over 100 million copies of its software downloaded or distributed throughout its history.

The MySQL Community Edition includes:

* Pluggable Storage Engine Architecture
* Multiple Storage Engines: InnoDB , MyISAM, NDB (MySQL Cluster),Memory ,Merge , Archive, CSV
* MySQL Replication to improve application performance and scalability
* MySQL Partitioning to improve performance and management of large database applications
* Stored Procedures to improve developer productivity

#### Detailed features of mysql

The following list shows the most important properties of MySQL. This section is directed to the reader who already has some knowledge of relational databases. We will use some terminology from the relational database world without defining our terms exactly. On the other hand, the explanations should make it possible for database novices to understand to some extent what we are talking about.

**Relational Database System:** Like almost all other database systems on the market, MySQL is a relational database system.

**Client/Server Architecture:** MySQL is a client/server system. There is a database server (MySQL) and arbitrarily many clients (application programs), which communicate with the server; that is, they query data, save changes, etc. The clients can run on the same computer as the server or on another computer (communication via a local network or the Internet).

|  |  |
| --- | --- |
|  | |
|  |  |
|  | |

Almost all of the familiar large database systems (Oracle, Microsoft SQL Server, etc.) are client/server systems. These are in contrast to the file-server systems, which include Microsoft Access, dBase and FoxPro. The decisive drawback to file-server systems is that when run over a network, they become extremely inefficient as the number of users grows.

**SQL compatibility:** MySQL supports as its database language -- as its name suggests – SQL (Structured Query Language). SQL is a standardized language for querying and updating data and for the administration of a database. There are several SQL dialects (about as many as there are database systems). MySQL adheres to the current SQL standard (at the moment SQL:2003), although with significant restrictions and a large number of extensions.

Through the configuration setting sql-mode you can make the MySQL server behave for the most part compatibly with various database systems. Among these are IBM DB/2 and Oracle. (The setting sql-mode changes some of the syntax conventions, and performs no miracles.

**SubSELECTs:** Since version 4.1, MySQL is capable of processing a query in the form SELECT \* FROM table1 WHERE x IN (SELECT y FROM table2) (There are also numerous syntax variants for subSELECTs.)

**Views:** Put simply, views relate to an SQL query that is viewed as a distinct database object and makes possible a particular view of the database. MySQL has supported views since version 5.0.

**Stored procedures:** Here we are dealing with SQL code that is stored in the database system.

Stored procedures (SPs for short) are generally used to simplify certain steps, such as inserting or deleting a data record. For client programmers this has the advantage that they do not have to process the tables directly, but can rely on SPs. Like views, SPs help in the administration of large database projects. SPs can also increase efficiency. MySQL has supported SPs since version 5.0.

**Triggers:** Triggers are SQL commands that are automatically executed by the server in certain database operations (INSERT, UPDATE, and DELETE). MySQL has supported triggers in a limited form from version 5.0, and additional functionality is promised for version 5.1.

**Unicode:** MySQL has supported all conceivable character sets since version 4.1, including Latin-1, Latin-2, and Unicode (either in the variant UTF8 or UCS2).

**User interface:** There are a number of convenient user interfaces for administering a MySQL server.

**Full-text search:** Full-text search simplifies and accelerates the search for words that are located within a text field. If you employ MySQL for storing text (such as in an Internet discussion group), you can use full-text search to implement simply an efficient search function.

**Replication:** Replication allows the contents of a database to be copied (replicated) onto a number of computers. In practice, this is done for two reasons: to increase protection against system failure (so that if one computer goes down, another can be put into service) and to improve the speed of database queries.

**Transactions:** In the context of a database system, a transaction means the execution of several database operations as a block. The database system ensures that either all of the operations are correctly executed or none of them. This holds even if in the middle of a transaction there is a power failure, the computer crashes, or some other disaster occurs. Thus, for example, it cannot occur that a sum of money is withdrawn from account A but fails to be deposited in account B due to some type of system error.

Transactions also give programmers the possibility of interrupting a series of already executed commands (a sort of revocation). In many situations this leads to a considerable simplification of the programming process. In spite of popular opinion, MySQL has supported transactions for a long time. One should note here that MySQL can store tables in a variety of formats. The default table format is called MyISAM, and this format does not support transactions. But there are a number of additional formats that do support transactions. The most popular of these is InnoDB, which will be described extensively in this book.

**Foreign key constraints:** These are rules that ensure that there are no cross references in linked tables that lead to nowhere. MySQL supports foreign key constraints for InnoDB tables.

**GIS functions:** Since version 4.1, MySQL has supported the storing and processing of two-dimensional geographical data. Thus MySQL is well suited for GIS (geographic information systems) applications.

**Programming languages:** There are quite a number of APIs (application programming interfaces) and libraries for the development of MySQL applications. For client programming you can use, among others, the languages C, C++, Java, Perl, PHP, Python, and Tcl.

**ODBC:** MySQL supports the ODBC interface [Connector/ODBC](http://searchenterpriselinux.techtarget.com/definition/MySQL-Connector-ODBC). This allows MySQL to be addressed by all the usual programming languages that run under Microsoft Windows (Delphi, Visual Basic, etc.). The ODBC interface can also be implemented under Unix, though that is seldom necessary.

Windows programmers who have migrated to Microsoft's new .NET platform can, if they wish, use the ODBC provider or the .NET interface Connector/NET.

**Platform independence:** It is not only client applications that run under a variety of operating systems; MySQL itself (that is, the server) can be executed under a number of operating systems. The most important are Apple Macintosh OS X, Linux, Microsoft Windows, and the countless Unix variants, such as AIX, BSDI, FreeBSD, HP-UX, OpenBSD, Net BSD, SGI Iris, and Sun Solaris.

**Speed:** MySQL is considered a very fast database program. This speed has been backed up by a large number of benchmark.

## ide for Database

### MySQL workbench



MySQL Workbench is a visual database design tool that integrates SQL evelopment,administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system. It is the successor to DBDesigner 4 from fabFORCE.net, and replaces the previous package of software,MySQL GUI Tools Bundle. [MySQL Workbench](http://www.mysql.com/products/workbench/) enables a DBA, developer, or data architect to visually design, generate, and manage all types of databases including Web, OLTP, and data warehouse databases. It includes everything a data modeler needs for creating complex ER models, and also delivers key features for performing difficult change management and documentation tasks that normally require much time and effort. MySQL Workbench is available on Windows, Linux and Mac OS.

#### benefits

* Simplifies database design and maintenance
* Automates time-consuming and error-prone tasks
* Enables data architects to visualize requirements, communicate with stakeholders, and resolve design issues before a major investment of time and resources is made
* Enables model-driven database design—the most efficient methodology for creating valid and well-performing databases—while providing the flexibility to respond to evolving business requirements
* Provides capabilities to forward-engineer physical database designs and reverse-engineer existing databases
* Allows you to import SQL scripts to build models and export models to DDL scripts that can be run at a later time
* Enables you to compare two live databases or a model and a live database, visually see the differences, and perform a synchronization between a model and a live database or vice versa
* Simplifies the documentation of database designs, providing a point-and-click process that delivers documentation in HTML or plain-text format

#### tools

The three main tools of MySQL Workbench are:

* SQL Development
* Data Modelling
* Server Administration

## Programming Language

### C# - c sharp



C# is a type-safe, object-oriented language that is simple yet powerful, allowing programmers to build a breadth of applications. C# is a [multi-paradigm programming language](http://en.wikipedia.org/wiki/Multi-paradigm_programming_language) encompassing [imperative](http://en.wikipedia.org/wiki/Imperative_programming), [declarative](http://en.wikipedia.org/wiki/Declarative_programming), [functional](http://en.wikipedia.org/wiki/Functional_programming), [generic](http://en.wikipedia.org/wiki/Generic_programming), [object-oriented](http://en.wikipedia.org/wiki/Object-oriented_programming)([class-based](http://en.wikipedia.org/wiki/Class_(computer_science))), and [component-oriented](http://en.wikipedia.org/wiki/Component-based_software_engineering) programming disciplines. It was developed by [Microsoft](http://en.wikipedia.org/wiki/Microsoft) within the [.NET](http://en.wikipedia.org/wiki/.NET_Framework) initiative and later approved as a standard by [Ecma](http://en.wikipedia.org/wiki/Ecma_International) (ECMA-334) and [ISO](http://en.wikipedia.org/wiki/International_Organization_for_Standardization) (ISO/IEC 23270). C# is one of the programming languages designed for the [Common Language Infrastructure](http://en.wikipedia.org/wiki/Common_Language_Infrastructure).

C# is intended to be a simple, modern, general-purpose, object-oriented programming language.

C# was developed to bring rapid development to C++ without sacrificing the power and control of C and C++. C# provides various characteristics,which are:  
Simple:  
C# eliminates the use of tedious operators such as -->, and pointers. C# treats inter and Boolean as two different data types, which enable the compiler   
to recognize the use of = in place of = = with if statement.  
  
**Consistent:-**  
C# supports only one integer tyoe and there is no limitation of range.  
**Modern:-**  
C# contains various features necessary to develop web applications. Following are the features of C#:  
It provides automatic garbage collection.  
It provides robust security model.  
It provides decimal data type for financial application.  
It provides modern approach for debugging.  
It provides a rich intrinsic model for error handling.  
**Object Oriented:-**  
C# supports all the features of object oriented language such as encapsulation, inheritance and polymorphism . It treats everything as an object and there are no global   
functions,variables and constants in C#.  
**Type Safe:-**  
C# provides various type safe measures ,which are :  
Dynamically allocated objects and arrays are initialised to zero.  
Products an error message while using an uninitialised variable.  
Checks the range of an array and warns when the access goes out of bound.  
Unsafe casts are not allowed.  
Enforces overflow checking in arithmetic operations.  
**Versionable:-**  
C# supports versioning that enables the existing applications to run on different versions with the help of new and override command.  
Compatible:  
C# contains the .NET specifications and therefore, allows inter operation with other .NET languages.  
**Flexible:-**  
C# does not support pointers but you may use pointers to manipulate the data of certain classes and methods by declaring them unsafe.  
Inter-operability:  
C# enables a program to call out any native API. It also allows the use of COM objects written in different languages.

### java



**Java** is a [general-purpose](http://en.wikipedia.org/wiki/General_purpose_programming_language), [concurrent](http://en.wikipedia.org/wiki/Concurrent_computing), [class-based](http://en.wikipedia.org/wiki/Class-based), [object-oriented](http://en.wikipedia.org/wiki/Object-oriented_programming) [computer programming language](http://en.wikipedia.org/wiki/Computer_programming_language) that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically [compiled](http://en.wikipedia.org/wiki/Compiler) to [bytecode](http://en.wikipedia.org/wiki/Java_bytecode) ([class file](http://en.wikipedia.org/wiki/Class_(file_format))) that can run on any [Java virtual machine](http://en.wikipedia.org/wiki/Java_virtual_machine) (JVM) regardless of [computer architecture](http://en.wikipedia.org/wiki/Computer_architecture). Java is, as of 2012, one of the most popular programming languages in use, particularly for client-server web applications, with a reported 10 million users.[[10]](http://en.wikipedia.org/wiki/Java_(programming_language)#cite_note-10)[[11]](http://en.wikipedia.org/wiki/Java_(programming_language)#cite_note-11) Java was originally developed by [James Gosling](http://en.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) (which has since[merged into Oracle Corporation](http://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle)) and released in 1995 as a core component of Sun Microsystems' [Java platform](http://en.wikipedia.org/wiki/Java_(software_platform)). The language derives much of its [syntax](http://en.wikipedia.org/wiki/Syntax_(programming_languages)) from [C](http://en.wikipedia.org/wiki/C_(programming_language)) and [C++](http://en.wikipedia.org/wiki/C%2B%2B), but it has fewer[low-level](http://en.wikipedia.org/wiki/Low-level_programming_language) facilities than either of them.

The original and [reference implementation](http://en.wikipedia.org/wiki/Reference_implementation_(computing)) Java [compilers](http://en.wikipedia.org/wiki/Compiler), virtual machines, and [class libraries](http://en.wikipedia.org/wiki/Library_(computing)) were developed by Sun from 1991 and first released in 1995. As of May 2007, in compliance with the specifications of the [Java Community Process](http://en.wikipedia.org/wiki/Java_Community_Process), Sun relicensed most of its Java technologies under the [GNU General Public License](http://en.wikipedia.org/wiki/GNU_General_Public_License). Others have also developed alternative implementations of these Sun technologies, such as the [GNU Compiler for Java](http://en.wikipedia.org/wiki/GNU_Compiler_for_Java) and [GNU Classpath](http://en.wikipedia.org/wiki/GNU_Classpath).

#### Java features

* **Platform Independent**The concept of Write-once-run-anywhere (known as the Platform independent) is one of the important key feature of java language that makes java as the most powerful language. Not even a single language is idle to this feature but java is more closer to this feature. The programs written on one platform can run on any platform provided the platform must have the JVM.
* **Simple**There are various features that makes the java as a simple language. Programs are easy to write and debug because java does not use the pointers explicitly. It is much harder to write the java programs that can crash the system but we cannot say about the other programming languages. Java provides the bug free system due to the strong memory management. It also has the automatic memory allocation and deallocation system.
* **Object Oriented**to be an Object Oriented language, any language must follow at least the four characteristics.
* Inheritance   :   It is the process of creating the new classes and using the behavior of the existing classes by extending them just to reuse the existing code and adding the additional features as needed.
* Encapsulation:  It is the mechanism of combining the information and providing the abstraction.
* Polymorphism:  As the name suggest one name multiple form, Polymorphism is the way of providing the different functionality by the   
     functions having the same name based on the signatures of the methods.
* Dynamic binding: Sometimes we don't have the knowledge of objects about their specific types while writing our code. It is the way     of providing the maximum functionality to a program about the specific type at runtime.

As the languages like Objective C, C++ fulfills the above four characteristics yet they  are not fully object oriented languages because they are structured as well as object oriented languages. But in case of java,  it is a fully Object Oriented language because object is at the outer most level of data structure in java. No stand alone methods, constants, and variables are there in java. Everything in java is object even the primitive data types can also be converted into object by using the wrapper class.

* **Robust**Java has the strong memory allocation and automatic garbage collection mechanism. It provides the powerful exception handling and type checking mechanism as compare to other programming languages. Compiler checks the program whether there any error and interpreter checks any run time error and makes the system secure from crash. All of the above features make the java language robust.
* **Distributed**The widely used protocols like HTTP and FTP are developed in java. Internet programmers can call functions on these protocols and can get access the files from any remote machine on the internet rather than writing codes on their local system.
* **Portable**The feature Write-once-run-anywhere makes the java language portable provided that the system must have interpreter for the JVM. Java also have the standard data size irrespective of operating system or the processor. These features make the java as a portable language.
* **Dynamic**While executing the java program the user can get the required files dynamically from a local drive or from a computer thousands of miles away from the user just by connecting with the Internet.
* **Secure**Java does not use memory pointers explicitly. All the programs in java are run under an area known as the sand box. Security manager determines the accessibility options of a class like reading and writing a file to the local disk. Java uses the public key encryption system to allow the java applications to transmit over the internet in the secure encrypted form. The bytecode Verifier checks the classes after loading.
* **Performance**Java uses native code usage, and lightweight process called threads. In the beginning interpretation of byte code resulted the performance slow but the advance version of JVM uses the adaptive and just in time compilation technique that improves the performance.
* **Multithreaded**  
  As we all know several features of Java like Secure, Robust, Portable, dynamic etc; you will be more delighted to know another feature of Java which is **Multithreaded.**Java is also a multithreaded programming language. Multithreading means a single program having different threads executing independently at the same time. Multiple threads execute instructions according to the program code in a process or a program. Multithreading works the similar way as multiple processes run on one computer.    
  Multithreading programming is a very interesting concept in Java. In multithreaded programs not even a single thread disturbs the execution of other thread. Threads are obtained from the pool of available ready to run threads and they run on the system CPUs. This is how Multithreading works in Java which you will soon come to know in details in later chapters.
* **Interpreted**We all know that Java is an interpreted language as well. With an interpreted language such as Java, programs run directly from the source code.   
  The interpreter program reads the source code and translates it on the fly into computations. Thus, Java as an interpreted language depends on an interpreter program.  
  The versatility of being **platform independent**makes Java to outshine from other languages. The source code to be written and distributed is platform independent.    
  Another advantage of Java as an interpreted language is its error debugging quality. Due to this any error occurring in the program gets traced. This is how it is different to work with Java.
* **Architecture Neutral**The term architectural neutral seems to be weird, but yes Java is an architectural neutral language as well. The growing popularity of networks makes developers think distributed. In the world of network it is essential that the applications must be able to migrate easily to different computer systems. Not only to computer systems but to a wide variety of hardware architecture and Operating system architectures as well.  The Java compiler does this by generating byte code instructions, to be easily interpreted on any machine and to be easily translated into native machine code on the fly.The compiler generates an architecture-neutral object file format to enable a Java application to execute anywhere on the network and then the compiled code is executed on many processors, given the presence of the Java runtime system.Hence Java was designed to support applications on network. This feature of Java has thrived the programming language.

## Mobile App Development

Quickly and efficiently create and test Java™ applications with the Nokia SDKs for Java and the Series 40 platform SDKs.

The Nokia SDKs for Java provide the development tools for phones containing a Java Runtime for Series 40. Each release of the Java Runtime has a corresponding Nokia SDK.

For earlier platform releases Series 40 platform SDKs offer versions to support specific editions and feature packs.

Within both families of tools, each SDK includes Java APIs, an emulator, documentation, code examples, and emulator based debugging tools. The SDKs can be used with either the NetBeans or Eclipse IDEs to create, compile, and package applications and content. Applications can be tested using the emulator.

### Nokia SDK 2.0 for Java — for Series 40 apps

|  |
| --- |
|  |

Create apps for Series 40 phones with the Java Runtime 2.0.0, including the full-touch UI equipped Nokia Asha 305, Nokia Asha 306, and Nokia Asha 311 using the Nokia SDK 2.0 for Java. Then test your apps in an emulator based on the Nokia Asha 305. In addition to the features of the Nokia SDK 1.1 for Java, the 2.0 SDK offers:

#### LWUIT for Series 40 arrives at 1.0

LUWIT for Series 40 has graduated beta to a full productised release. With a number of new APIs — such as PopUpChoiceGroup, ContextMenu, and NokiaListCellRenderer — the 1.0 release includes significant improvements in performance, particularly in lists, themes loading, and the HTMLComponent. Compatibility with the native full-touch UI has been fine-tuned and many bugs fixed, particularly in command handling and text input. A LWUIT Developer's Library has also been released, providing full technical and design guides. There are many new examples too.  
  
This release is delivered to the Nokia SDK 2.0 for Java through the SDK Manager, while a download for the Nokia SDK 1.1 for Java is available from LWUIT for Series 40 project. 

#### The Nokia SDK 2.0 for Java adds new features

The Nokia SDK 2.0 for Java delivers everything you need to develop apps for the exciting new full-touch UI equipped Nokia Asha 305, Nokia Asha 306, and Nokia Asha 311 phones. And now the SDK has graduated from beta.

The SDK delivers the updated Nokia UI API for advanced touch interaction, the Mobile Sensor API (JSR-256) to take advantage of the orientation sensors on the latest Series 40 phones, and the in-app purchase APIs. Now LWUIT is included as a plug-in, so you can create slick UIs, faster. In addition, the emulator gains features for the simulation of multipoint-touch gestures, such as pinch-to-zoom, and PC keyboard input. Based on the Nokia Asha 305, the emulator also provides improved sensor and location support so you can test more of your app on a PC.

The Nokia IDE for Java ME (Eclipse) has also been enhanced with improved searching in the Device SDK Manager and a tool that lets you pull code examples directly into the IDE. Building on the power of the Eclipse platform for Java development, the Nokia IDE makes delivering your Series 40 Java apps easier with features such as an editor for Nokia specific JAD attributes.

#### Testing your Java apps for Series 40 using Remote Device Access

Testing your Java apps on several Series 40 phones is easy and cheap with remote Device Access. Remote Device Access offers a range of Series 40 phones that you access over the internet free-of-charge. So when you need to test your app you can simply pop-on the internet, book a phone, install your app, and you will be testing in minutes.   
  
Right now, the Nokia Asha 311 is available for you to test your apps.

#### Explore in-app purchasing in your Java apps

Using the Nokia SDK 1.1 for Java or later you can explore adding in-app purchase features to your apps, a feature available on phones with Java Runtime 1.1.0 for Series 40 or later. Now you can generate revenue by offering users digital assets and content as part of the app experience — and allow them to purchase these items without leaving your application.

#### Add a new dimension to location with the Maps API

Leveraging the location information provided by the Location API for J2ME™ (JSR-179) on Series 40 phones, you can add rich maps to your apps with the Maps API for Java ME. With exciting features, such as custom overlays, you can create a unique experience. And with the release of the Nokia SDK 2.0 for Java, you now get the Maps APIs delivered ready for your use without additional downloads. Find out more about creating location aware applications with Java technology ›

### Nokia Web - Tools

#### Series 40 web apps tools

##### Nokia Web Tools 2.3

###### **Description**

Nokia Web Tools provides a set of tools that enable the creation of Series 40 web apps. The tools included are:

* Web Developer Environment (WDE) — enabling web apps to be created, edited, packaged and deployed.
* Web App Simulator (WAS) — enabling web apps to be previewed and debugged on a computer.
* Web Developer Channel (WDC) — included in Web Developer Environment, to deliver information and tools to facilitate web app development.

###### **What's new**

This new version of Nokia Web Tools provides:

* UI Designer in WDE offering drag-and-drop population of the web app’s UI.
* the ability to deploy a web app over a USB connection from a PC running Microsoft Windows.
* automatic reloading of the simulator for locally previewed web apps as code changes are saved.
* additional templates upon which to base new web apps, including web apps for trivia games, shopping, and video browsing among others.
* more sample web apps and snippets to help developers use the platform capabilities easily.

###### **Symbian WRT widget development**

This version of Nokia Web Tools no longer supports Symbian WRT widget development. If you wish to continue using [Nokia Web Tools 1.2](http://www.developer.nokia.com/info/sw.nokia.com/id/054b94d5-4cdf-4b17-b268-bedcfc421ba6/Nokia_Web_Tools_1_2_for_Symbian.html) for Symbian WRT widget development, please refer to the Installation Guide for details on the setup requirements.

###### **App publication**

Web apps created with Nokia Web Tools can be submitted for publication in Nokia Store.

###### **Versions available**

Nokia Web Tools are available in versions for:

* 32- or 64-bit Microsoft Windows XP Service Pack 2, Windows Vista, or Windows 7.
* 32-bit Ubuntu Linux 10.04.
* 64-bit Apple Mac OS X 10.6.

To make your Series 40 web apps development as straightforward as possible Nokia Web Tools, Bluetooth Launcher, and the Nokia Xpress Browser are available in the Series 40 web apps section.

Alternatively, Xpress Web App Builder is an online tool that enables content owners to create web apps from clipped content, RSS feeds, and social media content using a wide selection of formats.

You create Series 40 web apps using Nokia Web Tools. Based on Eclipse, Nokia Web Tools builds on the powerful web development features of the Eclipse Web Tools Platform to create the Web Developer Environment (WDE). WDE includes features to create, edit, validate, test, package, and deploy Series 40 web apps. Testing is supported by the Web Apps Simulator (WAS) that enables web apps to be run and tested on a computer. WAS includes an implementation of Web Inspector, so you can perfom debugging and examine of a web app's content and performance. This getting started guide takes you through installing Nokia Web Tools, creating a web app from a template, testing it on your computer, and running it on a phone, before providing links to the resources you need to build great web apps and deliver them to Nokia Store.

WDE offers a number of templates you can use to create Series 40 web apps easily. These templates range from the Basic web app template, which contains the core web app files with no functionality, through a selection of templates offering basic UI constructions to fully functional web apps, such as the Videos browsing project template that offers a working web app to browse videos. You can work with web app examples or a web app project you have already created as well. For more details on importing web apps, see [Importing a web app or web app project](http://www.developer.nokia.com/Resources/Library/Series_40_web_apps_library/#%21tools-library/creating-a-web-app-project/importing-an-existing-web-app-or-web-app-project.html) in the Series 40 Web App Developer's Library.

During development, transferring a web app onto a phone each time you make code changes isn’t a practical way of previewing and testing your web app. To simplify testing of a web app, WDE integrates with the Web App Simulator (WAS) to enable testing on your computer.

You have two options for running your web app in WAS:

* a server (cloud) based preview, this option provides a simulation that is very close to the experience that will be seen on a phone.
* a local preview, which is useful when you are working offline or want to debug your web app.

Having tested your web app in the simulator the next stage is to run it on a Series 40 phone. You have three ways to do this::

* deploying the web app to a phone from WDE over a Bluetooth connection.
* deploying the web app from WDE to a phone over a USB connection (but only if you are working on a Microsoft Windows PC as [Nokia Suite](http://www.nokia.com/nokiasuite) or [Nokia PC Suite](http://www.nokia.com/global/support/nokia-pc-suite/) is required).
* running the web app by entering a short URL into the Nokia Xpress Browser on a phone.

##### Series 40 Web App Developer’s Library

The Series 40 Web App Developer’s Library describes the Series 40 web apps development environment for Series 40 phones that run the Xpress Browser for Series 40, the tools for developing Series 40 web apps, and the design considerations for Series 40 web apps.

#### Xpress Web App Builder

Xpress Web App Builder is an online tool that guides you through the process of creating rich web apps, with no coding required. Select from a variety of templates, customise your theme, and then add clipped web content, RSS feeds, and social media information. The key features of the tool are:

* layout templates to present content, including single pane, tabbed view, and accordion view, as well as focused templates for news, pictures, and video content.
* a wide range of content widgets for clipped web content; RSS feeds; video from YouTube; pictures from Flickr, Picasa, and other photo sharing sites; and blogs from Tumblr and WordPress.
* the ability to add SMS and call capabilities, static HERE Maps, and in-app advertising from [Nokia Ad Exchange](http://www.developer.nokia.com/NAX).
* the option to customise your app's colour scheme, including header and font colours.
* static and dynamic previews of your app, for all supported screen resolutions.

When you've completed your web app, the tool provides a short URL for testing the app on your phone, and lets you submit the app to Nokia Publish to start the process of publication in Nokia store. However, if you want to customise your web app further, you can download the source code and import it into Nokia Web Tools.

#### Test your Series 40 web apps

If you don't have access to a Series 40 phone, you can test your web content and apps by making use of the Remote Device Access service. This service provides you with access to ten Series 40 phone models, more than 30 phones, over an internet connection. The service is available free to all Nokia Developer members.

#### Mobile Web Components

Make the most of the latest HTML5 feature in Nokia browsers.

Add rich, HTML5 based components to your web pages and web apps for Symbian Anna phones and the Nokia N9 Smartphone. Included are components for collapsible content blocks, scrollable large content item windows, pop-up menus, expandable sliding menus, slideshows, and others.

#### Leverage the power of QtWebKit

Using Qt WebKit technology, Web developer can easily transform web apps and web services into powerful native applications. Qt offers HTML5 and CSS3 support today. The quick and powerful way to use web assets and skills to produce apps for smartphones and mobile computers.

#### Device APIs::API Bridge

APIBridge is a component for Nokia Symbian devices that enables WRT widgets, Adobe Flash Lite content, and Java applications to access device features through a plug-in architecture. The APIBridge package ships with a set of plug-ins and the components to enable the features of the plug-ins to be used. Developers can extend the APIBridge component with their own plug-ins.

## Other technologies

### http://t1.gstatic.com/images?q=tbn:ANd9GcS-CmbHGLD4MH83JH1oNIr_acREqblVhrcFuvQfYZR8HFi1UpaqlgGitHub – repository management tool

GitHub is a web-based hosting service for software development projects that use the Git revision control system. GitHub offers both paid plans for private repositories, and free accounts for open source projects. As of May 2011, GitHub was the most popular open source code repository site.GitHub Inc. was founded in 2008 and is based in San Francisco, California.

#### Description

The site provides social networking functionality such as feeds, followers and the network graph to display how developers work on their versions of a repository.

GitHub also operates other services: a pastebin-style site called Gist that provides wikis for individual repositories and web pages that can be edited through a Git repository, a slide hosting service called Speaker Deck, and a web analytics platform called Gauges.

As of January 2010, GitHub is operated under the name GitHub, Inc.

The software that runs GitHub was written using Ruby on Rails and Erlang by GitHub, Inc. (previously known as Logical Awesome) developers Chris Wanstrath, PJ Hyett, and Tom Preston-Werner.

#### Limitations and constraints

According to the terms of service,if an account's bandwidth usage significantly exceeds the average of other GitHub customers, the account's file hosting service may be immediately disabled or throttled until bandwidth consumption is reduced. In addition, while there is no hard limit, the guideline for the maximum size of a repository is one gigabyte.

### Dia for Diagram Drawing & Modeling

Dia is free and open source general-purpose diagramming software, developed as part of the GNOME project's office suite and was originally created by Alexander Larsson. Dia uses a controlled single document interface (CSDI) similar to GIMP and Sodipodi.

Dia has a modular design with several shape packages available for different needs: flowchart, network diagrams, circuit diagrams, and more. It does not restrict symbols and connectors from various categories from being placed together.

Dia is a gtk+ based diagram creation program released under the GPL license.

Dia is inspired by the commercial Windows program 'Visio', though more geared towards informal diagrams for casual use. It can be used to draw many different kinds of diagrams. It currently has special objects to help draw entity relationship diagrams, UML diagrams, flowcharts, network diagrams, and many other diagrams. It is also possible to add support for new shapes by writing simple XML files, using a subset of SVG to draw the shape.

It can load and save diagrams to a custom XML format (gzipped by default, to save space), can export diagrams to a number of formats, including EPS, SVG, XFIG, WMF and PNG, and can print diagrams (including ones that span multiple pages).

### Cacoo: online drawing tool

 Cacoo is a diagram creation tool that runs in your web browser.Multiple people can work together on the same diagram in real time.Diagrams can be published directly to websites, wikis, and blogs.

#### Creating Diagrams

* Elements can be dragged and drop to easily create diagrams.
* Elements can be linked together with connectors.
* Connectors automatically move when elements are repositioned.
* You can use a text box and put text anywhere you like.
* You can upload images from your PC and include them in Diagrams.
* You can take screenshots of your computer from within Cacoo.
* Smart styles can easily be applied to stencils.
* You can have multiple sheets in a diagram and use them as backgrounds or layers.
* When you move the objects on your canvas, they will be snapped at the objects or grids nearby and align automatically.
* Copying, pasting and other functionality of basic drawing software is also built in to Cacoo.
* All actions are stored so there are unlimited levels of undo.
* You can import an image from the other websites by indicating the URL.
* The imported image can be easily trimmed only using your mouse.
* According to your editing status, tips will be shown on the right bottom corner of the canvas.

#### Collaboration

* You can invite collaborators to work with you in Cacoo.
* Multiple people can edit a diagram in real time.
* There is a chat function in the editor so people can communicate while creating diagrams.
* People can leave comments about the diagrams.
* Each user can set their own user icon.
* When editing with multiple people, users icons appear on selected objects.
* Sharing diagrams become much smoother. Diagrams in the shared folders can be accessible and editable by people who you have shared the folder with.

#### Sharing Diagrams

* If you keep the diagram private then other users can't see it.
* If you make the diagram URL public, then anyone who knows the URL can see it.
* Publishing a diagram to a blog can be useful in various ways.
* You can place code into blogs to create a slideshow
* Published images always display the most recent version.
* Diagrams can be exported to SVG format (Plus Plan users only) and PNG format. (More formats will be available in the future.)
* Diagrams can be posted to Twitter/Facebook/GoogleBuzz
* Diagrams can be displayed in SVG format for printing. (Plus Plan users only. A few browsers are not supported.)

#### Managing Diagrams

* Diagrams can be placed into folders.
* Diagrams can be copied.
* Diagrams can be displayed as thumbnails or as a list.

#### Languages and Time Zones

* All pages and notification e-mails support English and Japanese
* Users can enter text from almost all languages.
* Dates are displayed relative to your local time zone.

#### Security

* Private diagrams can only be seen by users you select.
* URLs which you do not share cannot be found by other users or search engines.
* All editing and management is protected by SSL.
* In order to access information about diagrams a Cacoo ID and password are requited.
* User passwords are encrypted on Cacoo's server.

### API

* You can access Cacoo using the API.
* The Cacoo API supports OAuth and an API Key.

By using the Cacoo API you are able to interact with Cacoo from other services and applications.

Authorization Methods

There are two ways to access the Cacoo API.

#### 1. API Key

The API key allows you make requests to the Cacoo API. You can make an API key here.

#### API Key

Append your API key to requests to the API to return data from your account.(Parameter name "apiKey")

Example: https://cacoo.com/api/v1/diagrams.json?apiKey=abcdefghijklmn

#### OAuth

OAuth 1.0a is supported as an authorization method for Cacoo. You can register applications here.

#### applications

Access Token:https://cacoo.com/oauth/access\_token

Authorize:https://cacoo.com/oauth/authorize

Request Token:https://cacoo.com/oauth/request\_token

### Google Spreadsheet Interface:

*With Google Spreadsheets, we can easily create, share, and edit spreadsheets online. Here are a few specific things we can do:*

* *Import and export these file types: .xls, .csv, .txt and .ods. We can also export data to a PDF or an HTML file.*
* *Format cells and edit formulas so we can calculate results and make data look the way we want it.*
* *Chat in real time with others who are editing our spreadsheet.*
* *Embed a spreadsheet, or a section of a spreadsheet, in our blog or website.*

# GLOSSARY