**[Type the company name]**

11

**Online-Exam System Documentation**

**Lamiya Mohmmed El\_Saedi 220093158**

Table of Contents

[Chapter 1 3](#_Toc313887676)

[Project Plane 3](#_Toc313887677)

[Introduction: 3](#_Toc313887678)

[Objectives and concentrations: 4](#_Toc313887679)

[Scope and limitations: 4](#_Toc313887680)

[Project Organization (The team): 5](#_Toc313887681)

[Risk analysis and risk planning: 5](#_Toc313887682)

[Hardware and software Requirements: 6](#_Toc313887683)

[Work break down: 7](#_Toc313887684)

[Schedule: 8](#_Toc313887685)

[Monitoring and reporting mechanisms: 8](#_Toc313887686)

[Project management approach: 9](#_Toc313887687)

[Chapter 2 11](#_Toc313887688)

[Software Requirement Specification 11](#_Toc313887689)

[(1) Preface: 11](#_Toc313887690)

[(2) Introduction: 12](#_Toc313887691)

[(3) Glossary: 12](#_Toc313887692)

[(4) User Requirements Definition: 12](#_Toc313887693)

[(4.1)The products and process features: 13](#_Toc313887694)

[(5) System Architecture: 13](#_Toc313887695)

[(6) System Requirement Specification: 14](#_Toc313887696)

[(6.1) Functional System Requirement: 14](#_Toc313887697)

[(6.2) Non-Functional System Requirements: 15](#_Toc313887698)

[6.4) Software Quality Attributes 16](#_Toc313887699)

[(6.3) System Interfaces: 17](#_Toc313887700)

[(7) System Models: 18](#_Toc313887701)

[(8) System Evolution: 18](#_Toc313887702)

[(9) Appendices: 19](#_Toc313887703)

[Chapter (3) 21](#_Toc313887704)

[System Design 21](#_Toc313887705)

[Introduction: 21](#_Toc313887706)

[Context Diagram: 22](#_Toc313887707)

[Models: 23](#_Toc313887708)

[3.1) Interaction model: 23](#_Toc313887709)

[System Architecture: 28](#_Toc313887710)

[Principal system objects: 29](#_Toc313887712)

[Develop design model: 31](#_Toc313887713)

[Object interface: 32](#_Toc313887714)

[Chapter (4) 33](#_Toc313887715)

[COCOMO 33](#_Toc313887716)

[COCOMOO II: 34](#_Toc313887717)

[The early design model: 34](#_Toc313887718)

# Chapter 1

# Project Plane

## 1.1) Introduction:

This document will propose all features and procedures to develop the system.

This document specially containing details about objectives, scope limitation, process model, primary requirements, team development, possible project risks, project schedule, and finally monitoring and reporting mechanisms.

On-line Exam System is very useful for Educational Institute to prepare an exam, safe the time that will take to check the paper and prepare mark sheets. It will help the Institute to testing of students and develop their skills. But the disadvantages for this system, it takes a lot of times when you prepare the exam at the first time for usage. And we are needs number of computers with the same number of students.

The effective use of "On-line Exam System", any Educational Institute or training centers can be use it to develop their strategy for putting the exams, and for getting better results in less time.

### Objectives and concentrations:

* Corporate between the data stored in the server of the Institution and our On-line Exam system. To deal with On-line System in an easy way and an efficient mannered. (connection process)
* Create strong and secrete data base that allow for any connection in a secret way, to prevent any outside or inside attacks.
* Specify a privilege for each person to allow each person use this system to create his own exam. And have a complete control on his exam.
* Allow each person to create more than one exam with different way to create variant questions.

### Scope and limitations:

* On-line Exam system is designed for Educational Institutes (like schools, universities, training centers).
* The system handles all the operations, and generates reports as soon as the test is finish, that includes name, mark, time spent to solve the exam.
* Allow students to see or display his answers after the exam is finish.
* The type of questions is only multiple choice or true and false.

## 1.2) Project Organization (The team):

|  |  |  |
| --- | --- | --- |
|  | **Job Title** | **Description** |
| **1** | Project Manager | * To manage all processes in the project |
| **2** | SW Designer | * To design the models and diagrams that helps the programmer in implementation phase. |
| **3** | Two Testers | * One from outside the team and the other from the inside the project team. |
| **4** | Two programmers | * Professional in ASP.NET and SQL * To programming the processes of the project. |
| **5** | SW Analyst | * To analyze the requirements of On-Line Exam System. |
| **6** | Writer | * Collects drafts from each member. * Rewrite and reformate the documents come from each member. * Have good print skills. * Have a good skill to correct grammars of statements. |

## 1.3) Risk analysis and risk planning:

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Risks:** | | | |
| **Risk** | **Probability** | **Effects** | **Risk planning strategy** |
| The experience staff in the team leave the project before it is finish, or someone was ill | low | serious | Use more than one staff for each section, which might minimize this risk. Also, manager tries to increase salary for him. |
| The methodology to solve the problem can't work in a proper manner. | high | serious | Must be study more than one methodology to minimize this risk. |
| Budget does not enough or there is no budget. | low | catastrophic | Put a condition in the contract if there any more expenses, the funded side must be pay it. To avoid this risk. |
| HW requirement can't come in the time. | moderate | serious | See if there is any more time to delay the project or not. If there is no more time work by the team computers, to minimize this risk. |
| **Product Risks:** | | | |
| **Risk** | **Probability** | **Effects** | **Risk planning strategy** |
| Packages and Development tools does not enough. | high | serious | Put a condition in the contract to increase the time of project delivery depends on the problem occur. To avoid this risk. |
| Can't found the suitable components. | high | tolerable | Programmer must have professional programming skills to write a new code, which minimize this risk. |
| **Business Risks:** | | | |
| **Risk** | **Probability** | **Effects** | **Risk planning strategy** |
| Can't found the suitable place for meeting the team. | moderate | tolerable | Monitoring the work by E-mail every day. To avoid this risk. |
| Damage the electricity generator. | high | serious | There is a spare generator to avoid this risk. |
| Marketing the product system. | low | catastrophic | Distribution of advertisements, which minimize this risk. |

## 1.4) Hardware and software Requirements:

|  |  |  |  |
| --- | --- | --- | --- |
| **Hardware Requirements:** | | | |
| **Item** | | **Item Count** | **Item price** |
| Computers (laptop) resent version | | 4 | 600$ for each one |
| ADSL Internet provider | | - | 50$ in month |
| Electricity Generator | | 2 | 300$ for each one |
| Office | | - | 200$ in month |
| External HD | | 2 | 100$ for each one |
| Team salary | | 6 | 500$ per month (5500$) |
| **Software Requirements:** | | | |
| **Item** | | **Item Count** | **Item price** |
| MS project | | 5 | 100$ |
| Office 2007 | | 5 | 100$ |
| ASP.NET | | 2 | 100$ |
| Monitors program | | 1 | 100$ |
| Upload services | | - | 72$ in year |
| Node Anti-virus (the correct version) | | 5 | 30$ |
| **Another Requirements:** | | | |
| Foods and drinks for ( breakfast, lunch and dinner) | 6 | | 10$ for each person in a day (3600$) |
| **Total** | | | **13,302$** |

## Work break down:

1. Project manager contracts with the user who demands the system and write a project plan. (three days)
2. Deliver the draft of project plan documentation to writer to rewrite the documentation and rewrite the document. (three days)
3. Then gives documentation of project plan to SW analyzer to do more analysis to verify the SRS document requirements. Then delivers SRS document to writer. (twenty-six days)
4. SW designer gives the SRS document and start to design the diagrams and models that helps the programmer to implement the project. Then delivers the draft design document to writer. (forty-seven days)
5. The two programmers take a partition of the project to start an implementation. (sixty days)
6. Throw the implementation the inner tester make validate the system and delivers his report to writer (sixteen days)
7. After finish the project and throw the implementation phase the outside tester validate the system and write his document then deliver to writer. (sixteen days)
8. The final report is ready now. (nine days)

## 1.6) Schedule:

## 1.7) Monitoring and reporting mechanisms:

The manager should monitor all activities in the project via minimize, avoid the risks or via management control as follows:

* + - 1. Put a table for all SW requirements and print in percentage how much finish.
      2. Using software programming to monitor programmer's progress.
      3. Using spyware profile to monitor the team.
      4. Using software that calculate how many lines written per hour.
      5. monitoring the risks as follows:
         1. Change the probability and effect.
         2. Delete risks or add a new one depends on the working on project.

## 1.8) Project management approach:

**Software Process Model:**

To solve an actual problems in an industry, software developer or a team of developers must integrate with a development strategy that include the process, methods and tools layer and generic phases. This strategy is often referred to a process model or a software developing paradigm. []

Our project follows the **waterfall model.**

**The steps of waterfall model are:**

* Requirement Definition
* System and Software Design
* Implementation
* Integration and System Testing
* Operation and Maintenance

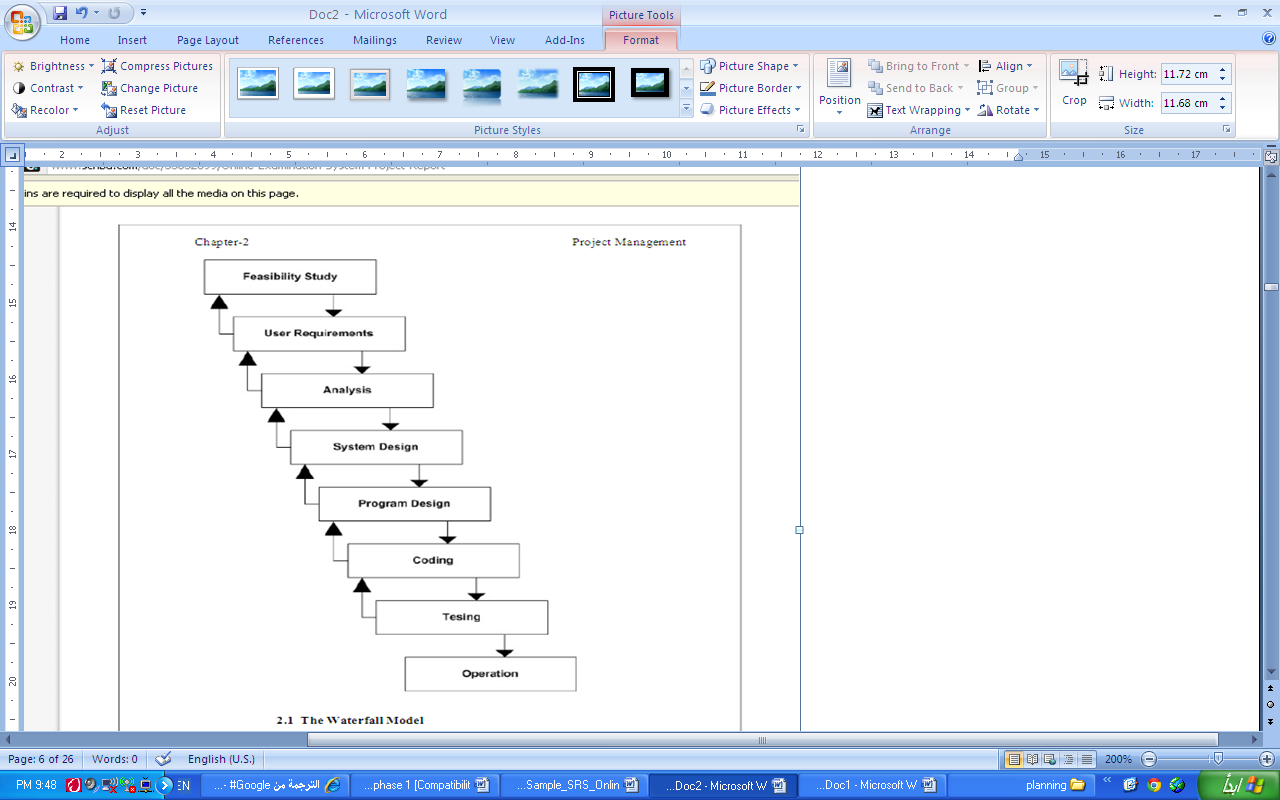


Figure (2.1): Waterfall model

# Chapter 2

# Software Requirement Specification

## (1) Preface:

This document has been written to apply a new version of SRS Software Requirements Specification depends on IEEE-STD-830-1998 standard. So, you must compare this document with this standard.

This is the first version for ***On-Line Exam*** system.

This document is the basic intended for any individual user, developer, tester, project manager or documentation writer that needs to understand the basic system architecture and its specifications. [1]

## (2) Introduction:

The purpose of this SRS document is to write the functional and non functional user or system requirements that represent the characteristics of ***On-Line Exam System***.

The scope and limitation of this system is:

* The on-line exam system design to educational institutes.
* Hold all operation and generate reports to student, teachers and administrator.
* Support multiple choices questions.
* Allow the student to prochoice the answer and to see his mark.
* Verify a security, authority and safty.

## (3) Glossary:

|  |  |  |
| --- | --- | --- |
|  | Short name | description |
| 1 | OES | On-line Exam System |
| 2 | On-line Exam | An exam written on a web site and solves the questions, also on the same web site from any place by entered user name and password. |
| 3 | Administrator | Who is responsible to create a new course, delete course, add member or delete it, i.e.: the person who control the system |
| 4 | Faculty member | A teacher in the faculty |

## (4) User Requirements Definition:

The user requirement for this system is to make the system fast, flexible, less prone to error, reduce expenses and save the time.

* Time can be saved by scheduling the exams, if it is available a question bank to store questions for different subjects.
* A system can be given a mark by checking the students answers, and give the result as soon as students finish his exam.
* A facility to generate a result chart as pre required without manual interface.
* The system should have records of students and faculty that can be access to the system which can be used only for the authorized person.
* The system should be more secure for management user records and more reliable to work at any conditions.

### (4.1)The products and process features:

This system must be designed as user required. So, the complete requirement must be found:

* **Quick scheduling:**

The system helps the faculty member to generate an automatic exam instead of using papers. Which save a time for writing, checking and for input marks. Also, student can see the exam when he login as an individual to the system.

* **Immediate results and solutions:**

When the student finishes his exam, the system checks her answers and compared with the correct answer. And the system saves the incorrect and correct answers and calculates the mark of correct answers. Then give the total mark. And send a report for student to see where he is fault.

* **Easy to store and retrieve information:**

Rather to save the information on a papers or in separate sheets. There are a data base management to store and retrieve the information needed by the administrator or Faculty member or student according a report generated by the system.

## (5) System Architecture:

Web Browser

Login

Role checking

Form & Menu Manager

Data Validation

Security Manager

OES Appointment Manager

Data Import & Export

Report Generation

Transaction Management for OES Database

Figure (2.1): system architecture for OES

## (6) System Requirement Specification:

### (6.1) Functional System Requirement:

This section gives a functional requirement that applicable to the On-Line Exam system.

There are three sub modules in this phase.

* Candidate module.
* Examiner module.
* Administrator module.

**The functionality of each module is as follows:**

* **Candidate module**: The candidate will logon to the software and take his examination. He can also check his previous examinations marks and his details. The candidate will get result immediately after the completion of the examination.
* **Examiner module:** The database is prepared & loaded into the software. Selection for examination can be done language wise by the examiner. The results will be displayed immediately after completion of the examination.
* **Administrator module:** The administrator collects all the results after successful completion of the examination and sends to the head quarters as and when required.

**The features that are available to the Administrator are:**

* The administrator has the full fledged rights over the OES.
* Can create/delete an account.
* Can view the accounts.
* Can change the password.
* Can hide any kind of features from the both of users.
* Insert/delete/edit the information of available on OES.
* Can access all the accounts of the faculty members/students.

**The features available to the Students are:**

* Can view the different categories of Test available in their account.
* Can change password.
* Can view their marks.
* Can view the various reading material.
* Can view and modify its profile but can modify it to some limited range.

**The features available to the Examiner are:**

* Can view the different categories of Test conducted by users.
* Can change password.
* Can view their marks.
* Can view and modify Results.

### (6.2) Non-Functional System Requirements:

#### 6.2.1) Performance Requirements

Some Performance requirements identified is listed below:

* The database shall be able to accommodate a minimum of 10,000 records of students.
* The software shall support use of multiple users at a time.
* There are no other specific performance requirements that will affect development.

#### 6.2.2) Safety Requirements

The database may get crashed at any certain time due to virus or operating system failure. Therefore, it is required to take the database backup.

#### 6.2.3) Security Requirements

Some of the factors that are identified to protect the software from accidental or malicious access, use, modification, destruction, or disclosure are described below. Keep specific log or history data sets

* Assign certain functions to different modules
* Restrict communications between some areas of the program
* Check data integrity for critical variables
* Later version of the software will incorporate encryption techniques in the user/license authentication process.

1. Communication needs to be restricted when the application is validating the user or license. (i.e., using https).

### 6.4) Software Quality Attributes

The Quality of the System is maintained in such a way so that it can be very user friendly to all the users.

The software quality attributes are assumed as under:

* Accurate and hence reliable.
* Secured.
* Fast speed.
* Compatibility.

### (6.3) System Interfaces:

This section describes how the software interfaces with other software products or users for input or output.

#### 6.3.1) User Interface

Application will be accessed through a Browser Interface. The interface would be viewed best using 1024 x 768 and 800 x 600 pixels resolution setting. The software would be fully compatible with Microsoft Internet Explorer for version 6 and above. No user would be able to access any part of the application without logging on to the system.

#### 6.3.2) Hardware Interfaces

**Server Side:**

* Operating System: Windows 9x/xp ,Windows ME
* Processor: Pentium 3.0 GHz or higher
* RAM: 256 Mb or more
* Hard Drive: 10 GB or more

**Client side:**

* Operating System: Windows 9x or above, MAC or UNIX.
* Processor: Pentium III or 2.0 GHz or higher.
* RAM: 256 Mb or more

#### 6.3.3) Software Interfaces

* **Client Side:** .HTML, Web Browser, Windows XP/2000/Vista
* **Web Server:** .HTML, Windows XP/2000/Vista

#### 6.3.4) Communications Interfaces

The Customer must connect to the Internet to access the Website:

* Dialup Modem of 52 kbps
* Broadband Internet
* Dialup or Broadband Connection with a Internet Provider.

## (7) System Models:

In this system we are use waterfall model to apply these ideas. Which is help us to separate each step and when we finish a one phase the output of it is the input to the next phase. Also, we can backwards if there is a new requirement or to apply any update.

## (8) System Evolution:

* **Including image support:**

Allow to adding students, faculty members and administrator images to the system. Which available for student to ensure that exam for his teacher. Also, the teacher can see his student's image.

* **Flags:**

Allow the student to put a symbol near the question that helps the student to return and review the questions and change them accordingly.

* **Enable and disable exam:**

Allow the faculty member to control for enable or disable the exam for his students.

* **Allow to transfer exam from one subject to another:**

So, that saves the time to rewrite the questions for future course.

* **Allow to upload the exam from word or excel file:**

So, that saves the time to enter a question in the on-line system, if the teacher needs not the direct answers.

* **Enhanced the questions to be appear as random for each student:**

Make the order of questions as random, or select random questions from a set of questions.

## (9) Appendices:

**Definition of online examination system:**

**Introduction:**

Online Examination System is a software application which allows a particular company or institute to arrange, conduct and manage any objective examination via online.

**Purpose:**

The purpose of this application is to conduct and process various types of certificate/non-certificate exams at different centers across any country via online.

**Features:**

* Any institute or company can register their various types of certificate/non-certificate programs and conduct an online examination for the same.
* Just register the programs, their fees (if paid) and the centers (where the exam will be conducted) in order to start the examination process.
* Questions and answers would be objective type and the format would be as per the company’s choice.
* User can select the company, its program, exam schedule and pay fees online in order to give his exam at the selected center.

**Advantages:**

* Today, most of the companies or institutes are conducting their exams online to be a part of this fastest growing world.
* Online Examination System covers almost all type of problems faced by a company or institute while conducting online examinations.
* User can give any available exam at any available center as per his/her choice.
* The results of the online exam will help a company or institute to list out the outstanding exam takers all over the country.

# Chapter (3)

# System Design

## 3.1) Introduction:

Design is the abstraction of a solution; it is a general description of the solution to a problem without the details. Design is view patterns seen in the analysis phase to be a pattern in a design phase. After design phase we can reduce the time required to create the implementation.

In this chapter we are introduce context diagram, models, system architecture, principal system object, design model and object interface.

## 3.2) Context Diagram:

This diagram represents what are the bounders and scope of **On-Line Exam** **System** project. It describes the main objective of the system and its entities involved.

**On-Line Exam system**

**Administrator**

**Student**

**Faculty**

Figure (3.2.1): the context diagram of On-line Exam System

**The Administrator can be done the following:**

* Create/delete accounts (add a list of faculty names and list of his student)
* Change password for Faculty/Student
* Create/ delete/update courses (subject).

**The Faculty can be done the following:**

* Change password.
* Insert questions.
* Specify the answers.
* Update mark of questions and answers.

**The Student can be done the following:**

* Change password.
* Choose exam.
* Review answers.
* See his exam mark.
* View other material.

## 3.3) Models:

### 3.3.1) Interaction model:

Is a dynamic model that shows how the system interacts with its environment. We use a data flow diagram.

#### 3.3.1.1) use case diagram:



Administrator

Faculty

Student

# 

Figure (3.3.1.1.1): the basic function for each actor

#### 3.3.1.2) activity diagram:

Administrator

Registration Process

Receiving details

Course details

Faculty details

Subject details

Receive master

Subject master

Course master

Faculty master

Request Report

View Report

(a)

Faculty

Insert Questions

Insert question

Subject master

Question master

Request Report

View Report

Student

Give exam

Register

Receive master

Subject master

Exam master

User master

Request Report

View Report

Control master

(b)

(c)

Figure (3.3.1.2.1): the activity diagram for basic operation in OES. (a) for administrator, (b) for Faculty and (c) for student.

#### 3.3.1.3) Séquence diagram:

Receive Faculty course subject

Administrator

New registration:

Registration process:

Receive master DB:

Subject master DB:

If new

Insert

Faculty master DB:

Course master DB:

Accept/ reject

Insert

Insert

Insert

Figure (3.3.1.3.1): the insert operation done by administrator. The update operation is similar to this sequence diagram but rather than Registration process put Update process.

Enter user name and password

Faculty

Login:

Select subject:

Subject master DB:

Insert question:

Verify

Request subject

Question master DB:

Accept/ reject

Store question

Subject selection

Return subject

Accept/ reject

Accept/ reject

If Accept

Figure (3.3.1.3.2): the insert question operation done by Faculty.

Store result in DB:

Enter user name and password

Student

Login:

Select subject:

Subject master DB:

Select question:

Verify

Request subject

Question master DB:

Invalid subject

Inactive subject

Verify

Verify

If Accept

Unavailable question

Unavailable question

Start exam

If Accept

If Accept

Return result and finish the exam

Figure (3.3.1.3.3): present how student take an exam and give the result.

## 3.4) System Architecture:

# 

Web Browser

Login

Role checking

Form & Menu Manager

Data Validation

Security Manager

OES Appointment Manager

Data Import & Export

Report Generation

Transaction Management for OES Database

## 3.5) Principal system objects:

|  |
| --- |
| User Master |
| int User\_ID |
| int ref­\_No |
| Int reg\_No |
| String F\_name |
| String l\_name |
| Int course\_id |
| Int year\_id |
| String username |
| String password |
| String e-mail |
| String gender |
| Date birth-date |
| String education |
| String occupation |
| String address |
| String country |
| String state |
| String city |
| String Zip |
| Int Active |
| Date Current-date |
| Date Modified-date |
| Get-userid() |
| Set-userid() |
| Get-ref-id() |
| Set-ref-id() |
| Get-reg-id() |
| Set-reg-id() |
| Get-name() |
| Set-name() |
| Get-course-id() |
| Set-course-id() |
| Get-year-id() |
| Set-year-id() |
| Get-username() |
| Set-username() |
| Get-password() |
| Set-password() |
| Get-emai() |
| Set-email() |
| Get-gender() |
| Set-gender() |
| Get-birthdate() |
| Set-birthdate() |
| Get-education() |
| Set-education() |

|  |
| --- |
| Receive Master |
| int receive\_ID |
| int ref­\_No |
| Int reg\_No |
| String F\_name |
| String l\_name |
| Int course\_id |
| Int year\_id |
| Date created-date |
| Date modify-date |
| Int active |
|  |
| Get-recive-id() |
| Set-recive-id() |
| Get-ref-id() |
| Set-ref-id() |
| Get-reg-id() |
| Set-reg-id() |
| Get-name() |
| Set-name() |
| Get-course-id() |
| Set-course-id() |
| Get-year-id() |
| Set-year-id() |
| Get-created-date() |
| Set-created-date() |
| Get-modified-date() |
| Set-modified-date() |
| Get-inactive() |
| Set-inactive() |
|  |

|  |
| --- |
| admin Master |
| int Admin\_ID |
| String F\_name |
| String l\_name |
| String username |
| String password |
| Date created-date |
| Date modify-date |
| Int active |
|  |
| Get-radmin-id() |
| Set-admin-id() |
| Get-name() |
| Set-name() |
| Get-username() |
| Set-username() |
| Get-password() |
| Set-password() |
| Get-created-date() |
| Set-created-date() |
| Get-modified-date() |
| Set-modified-date() |
| Get-inactive() |
| Set-inactive() |
|  |

|  |
| --- |
| Faculty Master |
| int Faculty\_ID |
| int ref­\_No |
| String F\_name |
| String l\_name |
| String username |
| String password |
| String e-mail |
| String gender |
| Date birth-date |
| String education |
| String occupation |
| String address |
| String contact -no |
| String main-subject |
| Date created-date |
| Date modify-date |
| Int active |
| Get-facutyid() |
| Set-facultyid() |
| Get-ref-id() |
| Set-ref-id() |
| Get-name() |
| Set-name() |
| Get-username() |
| Set-username() |
| Get-password() |
| Set-password() |
| Get-birthdate() |
| Set-birthdate() |
| Get-education() |
| Set-education() |
| Get-gender() |
| Set-gender() |
| Get-created-modify-day() |
| Set-created-modify-day() |
| Get-inactive() |
| Set-inactive() |

# 

|  |
| --- |
| Course Master |
| int course\_ID |
| String course-name |
| Stirng course-desc |
| String created-by |
| String modified-by |
| Date created-date |
| Date modified-date |
| Get-course-id() |
| Set-course-id() |
| Get-course-name() |
| Set-course-name() |
| Get-course-discrip() |
| Set-course-discrip() |
| Get-created-date() |
| Set-created-date() |
| Get-modified-date() |
| Set-modified-date() |
| Get-created-by() |
| Set-created-by() |
| Get-modified-by() |
| Set-modified-by() |
|  |

|  |
| --- |
| Year Master |
| int year\_ID |
| int course\_ID |
| String year-name |
| int duration |
|  |
| Get-year-id() |
| Set-year-id() |
| Get-course-id() |
| Set-course-id() |
| Get-year-name() |
| Set-year-name() |
| Get-duration() |
| Set-duration() |
|  |

|  |
| --- |
| Exam Master |
| int Exam\_ID |
| int sub\_id |
| Int question-ID |
| String username |
| Int attend |
| String mark |
| String user-answer |
| Date exam-date |
|  |
| Get-exam-id() |
| Set-exam-id() |
| Get-sub-id() |
| Set-sub-id() |
| Get-question-id() |
| Set-question-id() |
| Get-username() |
| Set-username() |
| Get-attend() |
| Set-attend() |
| Get-user-answer() |
| Set-user-answer() |
| Get-marks() |
| Set-marks() |
| Get-exam-date() |
| Set-exam-date() |
|  |

|  |
| --- |
| Subject Master |
| int sub-id |
| int year-id |
| Int course-id |
| String sub-name |
| String sub-discription |
| Int active |
|  |
| Get-sub-id() |
| Set-sub-id() |
| Get-year-id() |
| Set-year-id() |
| Get-course-id() |
| Set-course-id() |
| Get-sub-name() |
| Set-sub-name() |
| Get-sub-discription() |
| Set-sub-discription() |
| Get-inactive() |
| Set-inactive() |
|  |

|  |
| --- |
| Question Master |
| int question\_ID |
| int sub\_id |
| String question |
| String answer1 |
| String answer2 |
| String answer3 |
| String answer4 |
| String correct-answer |
| String created-by |
| String modified-by |
| String main-subject |
| Date created-date |
| Date modify-date |
| Int active |
|  |
| Get-question-id() |
| Set-question-id() |
| Get-sub-id() |
| Set-sub-id() |
| Get-question() |
| Set-question() |
| Get-answer1() |
| Set-answer1() |
| Get-answer2() |
| Set-answer2() |
| Get-answer3() |
| Set-answer3() |
| Get-answer4() |
| Set-answer4() |
| Get-correct-answer() |
| Set-correct-answer() |
| Get-main-subject() |
| Set-main-subject() |
| Get-created-date() |
| Set-created-date() |
| Get-modified-date() |
| Set-modified-date() |
| Get-created-by() |
| Set-created-by() |
| Get-modified-by() |
| Set-modified-by() |
| Get-inactive() |
| Set-inactive() |
|  |

|  |
| --- |
| Result Master |
| int result\_ID |
| int sub\_id |
| String username |
| String marks |
| Date exam-date |
|  |
| Get-result-id() |
| Set-result-id() |
| Get-sub-id() |
| Set-sub-id() |
| Get-username() |
| Set-username() |
| Get-marks() |
| Set-marks() |
| Get-exam-date() |
| Set-exam-date() |
|  |

|  |
| --- |
| Year Master |
| int year\_ID |
| int course\_ID |
| String year-name |
| int duration |
|  |
| Get-year-id() |
| Set-year-id() |
| Get-course-id() |
| Set-course-id() |
| Get-year-name() |
| Set-year-name() |
| Get-duration() |
| Set-duration() |
|  |

|  |
| --- |
| State Master |
| int state\_ID |
| Int country-ID |
| String state-name |
| String state-code |
|  |
| Get-state-id() |
| Set-state-id() |
| Get-country-id() |
| Set-country-id() |
| Get-state-name() |
| Set-state-name() |
| Get-state-code() |
| Set-state-code() |
|  |

|  |
| --- |
| Course Master |
| int course\_ID |
| String course-name |
| Stirng course-desc |
| String created-by |
| String modified-by |
| Date created-date |
| Date modified-date |
| Get-course-id() |
| Set-course-id() |
| Get-course-name() |
| Set-course-name() |
| Get-course-discrip() |
| Set-course-discrip() |
| Get-created-date() |
| Set-created-date() |
| Get-modified-date() |
| Set-modified-date() |
| Get-created-by() |
| Set-created-by() |
| Get-modified-by() |
| Set-modified-by() |
|  |

|  |
| --- |
| City Master |
| int city\_ID |
| int country\_ID |
| Int state-ID |
| String cityname |
|  |
| Get-city-id() |
| Set-city-id() |
| Get-state-id() |
| Set-state-id() |
| Get-country-id() |
| Set-country-id() |
| Get-city-name() |
| Set-city-name() |
|  |

|  |
| --- |
| Country Master |
| int country\_ID |
| String country-name |
| String course-code |
|  |
| Get-country-id() |
| Set-country-id() |
| Get-country-name() |
| Set-country-name() |
| Get-course-code() |
| Set-course-code() |
|  |

## 3.6) Develop design model:

# 

Administrator

User authentication process

Username and password

Verify

Change password

Admin master

Student master

Faculty master

Faculty

Student

Figure (3.6.1): user authentication

## 3.7) Object interface:

|  |
| --- |
| Student Interface |
|  |
| Change-password() |
| Choose-exam() |
| View-material() |
| Registration() |
|  |

|  |
| --- |
| Faculty Interface |
|  |
| Insert-question() |
| Change-password() |
| Add-student() |
| Delete-student() |
| Registration() |
|  |

|  |
| --- |
| Admin Interface |
|  |
| Insert-subject() |
| Update-subject()  Delete-subject()  Delete-course |
| Insert-course() |
| Update-course() |
| Assign-faculty()  Update-faculty() |
| Delete-faculty() |
| Assign-student-to-course() |
| Update-student-course() |
| delete-student-course() |
|  |

# Chapter (4)

# COCOMO

## 4.1) COCOMOO II:

### 4.1.1) The early design model:

* Is used once user requirements have been agreed and initial stages of the system design process are underway.
* The estimates produced at this stage are based on the standard formula for algorithmic models, namely:

PM = A \* SizeB \* M where

M = PERS \* RCPX \* RUSE \* PDIF \* PREX \* FCIL \* SCED;

A = 2.94 in initial calibration, Size in KLOC,

B varies from 1.1 to 1.24 depending on novelty of the project, development flexibility, risk management approaches and the process maturity.

* Multipliers reflect the capability of the developers, the non-functional requirements, the familiarity with the development platform, etc.

RCPX - product reliability and complexity; 🡪(3)

RUSE - the reuse required; 🡪(2)

PDIF - platform difficulty; -🡪(1)

PREX - personnel experience; 🡪 (5)

PERS - personnel capability; 🡪 (5)

SCED - required schedule; 🡪(5)

FCIL - the team support facilities. 🡪 (5)

* You estimate values for these attributes using a six-point scale where 1 corresponds to very low values for these multipliers and 6 corresponds to very high values.
* *Function-related metrics*:
  + Related to the **overall functionality** of the delivered software.
  + Productivity is expressed **in terms of the amount of useful functionality** produced in some given time.
  + **Function points and object points** are the best-known metrics of this type.
* You **compute the total number of function points** in a program **by** measuring or estimating the following program **features**:
  + ***External inputs and outputs;***
  + ***User interactions;***
  + ***External interfaces;***
  + ***Files used by the system***.

****

* **Unadjusted function-point count**
* **Weighting factor varies from 3 (for simple external inputs) to 15 for complex internal files.**
* **External input and output:**
* **Only for registration interface:**

For user (student) interface:

* Input: there are 23 inputs. 🡪(7)

For faculty interface:

* Input: 17 inputs. 🡪(7)

For administrator interface:

* Input: 8 inputs. 🡪(7)

**Output: Store in data base file 🡪 (13)**

* **Only for add course interface:**
* Input: 8 inputs 🡪(7)
* Output: Store in data base file 🡪 (13)
* **Only for add subject interface:**
* Input: 7 inputs 🡪(7)
* Output: Store in data base file 🡪 (13)
* **Only for add question interface:**
* Input: 15 inputs 🡪(7)
* Output: Store in data base file 🡪 (13)
* **Only for take result interface:**
* Input: 1 input 🡪(7)
* Output: Store in data base file 🡪 (13)
* Output: 3 outputs 🡪(10)
* **User interaction:**

There are 48 user interactions. 🡪(12)

* **External interface:**

3 main external interfaces. 🡪(13)

* **Files used by the system:**

13 tables used to Store in data base. 🡪 (13)

**UFC=23\*7+17\*7+8\*7+13\*13+8\*7+7\*7+15\*7+7+3\*10+48\*12**

**= 1146**

**M=3\*2\*1\*5\*5\*5\*5**

**=3750**

**PM = A \* SizeB \* M**

**=1.49\*1146^1.2\*3750**

**= 26196247.04 KLOC (1000 Line Of Code)**

**References:**

[1] Software Requirements Specification for project iTest, 2008

[2] http:// [www.scribd.com/doc/33852099/on-line-examiniation-system-project-report](http://www.scribd.com/doc/33852099/on-line-examiniation-system-project-report)

Tu. 21/12/2011.

[3]<http://whatis.techtarget.com/definition/0,,sid9_gci1103696,00.html>, Sat. 29/10/2011.

[4] Software Requirements Specification for Problem Based Learning Module, Souman Mandal, 2010.

[5] Software Design Specification (SDS) Acropolis Course Management System, 2011

[6] IEEE Recommended Practice for Software Requirements Specifications, Software Engineering Standards Committee of the IEEE Computer Society. 1998

[7] Software Requirements Specification for PPDP Contact Management System (CMS)

[8]<http://www.ehow.com/facts_5156877_preface-book.html>**,** Sat. 29/10/2011.

[9]<http://www.sil.org/lingualinks/literacy/referencematerials/glossaryofliteracyterms/WhatIsAPreface.htm>**,** Sat. 29/10/2011.

[10] <http://www.deftinfosystems.com/index.php/application/e-education-system/online-examination-system.html>, Mon. 9/1/2012.

[11] Software Requirement Specifications, Online Examination System.