1. **INTRODUCTION**

**1 .1 PROBLEM DESCRIPTION**

Today’s many of the modern business, houses and almost all firms and factories are working on computer system. Computer provides accurate and fast results.

The purpose of the project titled “E-Learning System” is to provide distance learning through internet. “E-Learning System” is an online web application. A tutor from any part of the world can provide lectures to any student/s on any part of the world. This is a web based application which can be hosted in the website of any training institute. The students will be able to register online, read the course material and write online exam. Certificates also will be issued online.

The new system is developed in ASP.net as front end and MySQL as back end which provides security to the data by preventing unauthorized access.

**1.2 EXISTING SYSTEM**

Shows educators and trainers how to adapt teaching methods and materials to make use of the Internet. Including a framework showing how to apply Internet technology progressively as skills and confidence grow, the project demonstrates the route from adapting materials to developing a virtual course.

Nowadays, when people are not having time to visit an institute. Therefore, the software is designed to provide the education through Internet. The project “E – LEARNING” helps the common world in any field they are to get the knowledge what they want even sitting at there places. This helps them to spare time in their busy schedule and save their time during transportation..

**1.2.1 LIMITATIONS OF THE EXISTING SYSTEM**

• Usage of papers in the payment of fees process leads to less efficiency, less accuracy and less productivity.

• Increasing expenditure for papers shuffling and storage.

• Increasing labors and hence errors.

• Less control of funds.

• Persons who are present in different part of the world cannot exchange their knowledge efficiently.

• Time consuming in the travelling.

• The cost is high for the infrastructure of classroom

**1.3 PROJECT SCOPE**

Education through the internet, network and a computer is E-learning. This helps in network enabled transfer of skills and knowledge. E-learning refers to use of electronic applications and processes to learn. This is a web based application which can be hosted in the website of any training institute. The students will be able to register online, read the course material and write online exam. Certificates also will be issued online.

The objectives of this project are   
• To provide distance learning  
• To make it convenient for people who have other commitments  
• Cost reduction  
• Reduced paper work  
• Computer evaluated tests

**2. SYSTEM ANALYSIS**

Software Engineering is the analysis, design, construction, verification and management of technical or social entities. To engineer software accurately, a software engineering process must be defined.

System analysis is a detailed study of the various operations performed by the system and their relationship within and module of the system. It is a structured method for solving the problems related to the development of a new system. The detailed investigation of the present system is the focal point of system analysis. This phase involves the study of parent system and identification of system objectives. Information has to be collected from all people who are affected by or who use the system. During analysis, data are collected on the variable files, decision point and transactions handled by the present system. The main aim of system is to provide the efficient and user-friendly automation. So the system analysis process should be performed with extreme precision, so that an accurate picture of existing system, its disadvantages and the requirements of the new system can be obtained.

System analysis involves gathering the necessary information and using the structured tool for analysis. This includes the studying existing system and its drawback, designing a new system and conducting cost benefit analysis. System analysis is a problem solving activity that requires intensive communication between the system users and system developers. The system is studied to the minutest detail and analyzed. The system is viewed as a whole and the inputs to the system are identified. The outputs from the organization are traced through various phases of processing of inputs.

There are a number of different approaches to system analysis. When a computer-based information system is developed, systems analysis (according to the Waterfall model) would constitute the following steps:

* The development of a feasibility study, involving determining whether a project is economically, socially, technologically and organizationally feasible.
* Conducting fact-finding measures, designed to ascertain the requirements of the system's end-users. These typically span interviews, questionnaires, or visual observations of work on the existing system.
* Gauging how the end-users would operate the system (in terms of general experience in using computer hardware or software), what the system would be used for and so on.

Techniques such as interviews, questionnaires etc can be used for the detailed study of these processes. The data collected by these sources must be scrutinized to arrive at a conclusion.

The conclusion is an understanding of how the system functions. This system is called the ***Existing System***. The Existing system is then subjected to close observation and the problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as a proposal which is the ***Proposed System.*** The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is then presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is a loop that ends as soon as the user is satisfied with the proposal.

**2.1 FUNCTIONAL SPECIFICATIONS**

A functional specification in systems engineering and software development is the documentation that describes the requested behavior of an engineering system. The documentation typically describes what is needed by the system user as well as requested properties of inputs and outputs. Thus it picks up the results of the requirements analysis stage. On more complex systems multiple levels of functional specifications will typically nest to each other, e.g. on the system level, on the module level and on the level of technical details.

In systems engineering a functional specification is a document that clearly and accurately describes the essential technical requirements for items, materials, or services including the procedures by which it can be determined that the requirements have been met. Specifications help avoid duplication and inconsistencies, allow for accurate estimates of necessary work and resources, act as a negotiation and reference document for engineering changes, provide documentation of configuration, and allow for consistent communication among those responsible for the eight primary functions of Systems Engineering. They provide a precise idea of the problem to be solved so that they can efficiently design the system and estimate the cost of design alternatives. They provide guidance to testers for verification (qualification) of each technical requirement.

The FS is a formal statement of the functions required to be implemented in a specific physical form or technology for a business proposal. There are several elements common to the design process. The first is conceptual (logical or essential) planning of requirements: sometimes known as conceptual design, but more usefully referred to as Analysis-the detailed conceptual planning that is documented in the Business Requirements Specification. These two elements are the domain of the Functional Specification. Accordingly, the Functional Specification is an intermediate document between the BRS and the Technical Specification.

The users of the project are,

* **Admin**
* **Teacher**
* **Student**

The modules in this project are

* **Registration module**
* **Student module**
* **Teacher module**
* **Online Exam module**

**2.1.1 Registration module**

This module consist of registering the participant details inorder to login.The exsisting parcipant can directly login along with userid and password and view the course details.If it is new participant they should register their details with userid and password.The details of participant will be maintained by the database. Admin will select the tutor and provide username and password for them. Tutor can login and accessing this system through username and password . The details of tutor will be maintained by the database.

**2.1.2. Student module**

After registration student can login and view the course details in order to build their carrier. They can select any course and enrolled the course by registering their details. The Course enrolled can be viewed by the student and they can edit. Student can view the course material and learn through online and appear for exam. The results of the exams are also declared just after taking the test

**2.1.3 Teacher module**

In this module, teacher can view the student registration, course enrolled status. The student can clear the doubts by sending email to their allotted teachers. The teacher view the student doubt and sends reply according to the queries. The Questions will be hosted for the student in order to take exams. Teacher will host question paper according to the domain and send to the participant.

**2.1.4. Online Exam module**

In this module, participants will appear for taking exam according to the course would they enrolled. The participant should clearly read the instruction once before they appear for exam. Click on start exam button to start the exam. Questions are basically of Multiple choice type i.e., only one answer is correct, these answers  are represented by four Radio Buttons. After clicking start exam button the exam starts. Once the participant completed their exam they should enter the particular details what they given. Then the result will be viewed along your particular details.

**2.2 SYSTEM REQUIREMENTS**

**2.2.1 Software Specification**

This project is built upon the latest technology software.

**2.2.1.1 Introduction PHP**

PHP is a server-side scripting language designed for Web development but also used as a general-purpose programming language. PHP is now installed on more than 20 million Web sites and 1 million Web servers. While PHP originally stood for Personal Home Page, it is now said to stand for PHP: Hypertext Preprocessor, a recursive acronym.

PHP code is interpreted by a Web server with a PHP processor module which generates the resulting Web page: PHP commands can be embedded directly into an HTML source document rather than calling an external file to process data..PHP includes free and open source libraries with the core build. PHP is a fundamentally Internet-aware system with modules built in for accessing File Transfer Protocol (FTP) servers, many database servers, embedded SQL libraries such as embedded PostgreSQL, MySQL, Microsoft SQL Server and SQLite, LDAP servers, and others. PHP is commonly used as the P in this bundle alongside Linux, Apache and MySQL, although the P may also refer to Python, Perl, or some mix of the three. Similar packages are also available for Windows and OS X, then called WAMP and MAMP, with the first letter standing for the respective operating system.

2.2.1.2. Dream Viewer

Dreamweaver features an integrated browser for previewing developed webpages in the program's own preview pane in addition to allowing content to be open in locally installed web browsers and manipulate the layout of HTML elements.

Dreamweaver, like other HTML editors, edits files locally then uploads them to the remote web server using FTP, SFTP, or WebDAV. Dreamweaver CS4 now supports the Subversion (SVN) version control system.Dreamweaver is a website editor WYSIWYG for Microsoft Windows and Mac OS X created in 1997, marketed by Macromedia and Adobe Systems on End User License .Dreamweaver was one of the first publishers HTML type "such a display, such a result", but also one of the first to integrate a site manager (Adobe Dreamweaver is a web design and development application that provides a visual WYSIWYG editor (colloquially referred to as the Design view) and a code editor with standard features such as syntax highlighting, code completion, and code collapsing as well as more sophisticated features such as real-time syntax checking and code introspection for generating code hints to assist the user in writing code.[4] The Design view facilitates rapid layout design and code generation as it allows users to quickly create CyberStudio GoLive being the first). These innovations quickly imposed it as a leading site web , both used by the novice as professional.

**2.2.1.5 Back End: MySQL Database**

A database is a collection of information that’s related to a particular subject or purpose, such as tracking client orders or maintaining a list of project details. If the database isn’t stored on a computer, or only part of it are one may be tracking information from a variety of sources that one is having to co-ordinate and organize himself using MySQL Server, one can manage all information from a single database file.Within the file, data is divided into separate storage containers called tables; view, add and update data by using online forms; find and retrieve just the data wanted for reports. MySQL Server allows the user to view, update or analyze the database from the Internet of an intranet by creating data access pages. MySQL Server as a relational database stores data in many related tables. A table is a

collection of data about a specific topic such as projects or clients. Using a separate table for each topic means that, store that data only once. This makes the database more efficient and reduces data-entry errors.

Tables organize data into columns (called fields) and rows (called records). A common field relates two tables so that MySQL Server can bring together the data from the two tables for viewing, editing, or printing. In table Design view one can create an entire table from scratch or add, delete or customize the fields in an existing table. The user can also display records from tables that are related to the current table by displaying sub datasheets within the main datasheet. With some restrictions, the user can work with the data in sub datasheets in many of the same ways that they work with data in the main datasheet.

To store data, create one table for each type of information that is to be tracked. To bring the data from multiple tables together in a query, form, report, or data MySQL Server page, define relationships between the tables.

**2.2.1.6 Operating system**

Operating System is defined as a program that manages the computer hardware. An operating system can be viewed as a scheduler, where it has resources for which it has charge. Resources include CPU, memory, I/O device and disk space. In another view, the operating system is a new machine. The third view is that

operating system is a multiplexer which allows sharing of resources provides protection from interference and provides a level of cooperation between users.

This project is developed using Windows 7 as the operating system and supports its later versions. Windows 7 is an operating system produced by Microsoft for use on

personal computers, including home and business desktops, laptops, net books, tablet PCs, and media center PCs .It was released to manufacturing on July 22, 2009, and became generally available retail worldwide on October 22, 2009. Unlike Windows Vista's many new features, Windows 7 was an incremental upgrade designed to work with Vista-compatible applications and hardware. Presentations given by Microsoft in 2008 focused on multi-touch support, an updated Windows shell with a new taskbar,

referred to internally as the Super bar, a home networking system called Home Group, and performance improvements. Some standard applications that have been included with prior releases of Microsoft Windows, including Windows Calendar, Windows Mail, Windows Movie Maker, and Windows Photo Gallery, are not included in Windows 7 most are instead offered separately at no charge as part of the Windows Essentials suite.

Among Windows 7's new features are advances in touch and handwriting recognition, support for virtual hard disks, improved performance on multi-core processors, improved boot performance, Direct Access, and kernel improvements Many new items have been added to the Control Panel, including Clear Type Text Tuner Display Color Calibration Wizard ,Gadgets, Recovery, Troubleshooting, Workspaces Center and so on.

**2.2.2 Hardware requirements**

Hardware Requirements for Development are as listed below.

* Processor: Pentium 4 or above
* RAM :2 GB or above
* Hard disk :40 GB or above

**3. SYSTEM DESIGN**

The system design is the most creative and challenging phase of system development life cycle. It is an approach for the creation of proposed system, in which the logic and details structure of the proposed system is designed, which will help the system coding. The most creative and challenging phase of the system development process is design phase it is a solution, how to approach to the creation of the proposed system. Design is the first step in the development of the engineered product is initiated only after a clear exposition of expected product is available. System Design is vital for efficient database management. It provides the understanding of procedural details necessary for implementing the system .A number of sub- systems is to be identified which constitute the whole system.

**3.1 SYSTEM ARCHITECTURE**

**System architecture** or **systems architecture** is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures of the system,

System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture; collectively these are called architecture description languages (ADLs).

The system architecture can best be thought of as a set of representations of an existing (or to be created) system. It is used to convey the informational content of the elements comprising a system, the relationships among those elements, and the rules governing those relationships. The architectural components and set of relationships between these components that architecture describes may consist of hardware, software, documentation, facilities, manual procedures, or roles played by

organizations or people. System architecture is primarily concerned with the internal interfaces among the system's components or subsystems, and the interface between the system and its external environment, especially the user.

The structural design reduces complexity, facilitates change and result in easier implementation by encouraging parallel development of different parts of the system. The procedural design transforms structural elements of program architecture into a procedural description of software components. The architectural design considers architecture as the most important functional requirement. The system is based on the three-tier architecture.

The first level is the user interface (presentation logic), which displays controls, receives and validates user input. The second level is the business layer (business logic) where the application specific logic takes place. The third level is the data layer where the application information is stored in files or database. It contains logic about to retrieve and update data. The important feature about the three-tier design is that information only travels from one level to an adjacent level.

**3.2 MODULE DESIGN**

Modular programming is a software design technique that emphasizes separating the functionality of a program into independent, interchangeable modules, such that each contains everything necessary to execute only one aspect of the desired functionality. Conceptually, modules represent a separation of concerns, and improve maintainability by enforcing logical boundaries between components. Modules are typically incorporated into the program through interfaces. A module interface expresses the elements that are provided and required by the module. The elements defined in the interface are detectable by other modules.

When creating a modular system, instead of creating a monolithic application (where the smallest component is the whole), several smaller modules are built (and usually compiled) separately so that, when composed together, they construct the executable application program. A just-in-time compiler may perform some of this construction "on-the-fly" at run time. This makes modular designed systems, if built

correctly, far more reusable than a traditional monolithic design, since all (or many) of these modules may then be reused (without change) in other projects. This also facilitates the "breaking down" of projects into several smaller projects. Theoretically, a modularized software project will be more easily assembled by large teams, since no team members are creating the whole system, or even need to know about the system as a whole. They can focus just on the assigned smaller task.

This Project, the online Flower Mart is also built upon the modular programming concept. The module design is done such that each module is perfectly distinguishable and distinct with its own unique functionalities. The project consists of six modules: Registration Module, Product Module, Decoration Order Module, Sales Module, Report Module. This breakdown of the functionalities into modules made the task of programming easier. The modular design reduces the complexity, facilitates change and results in easier implementation by encouraging parallel development of different parts of the system.

**3.3 DATABASE DESIGN**

A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make information access easy, quick, inexpensive and flexible for the users. The general theme behind a database is to integrate all information. Database design is recognized as a standard of management information system and is available virtually for every computer system. In database design several specific objectives are considered:

* Ease of learning and use
* Controlled redundancy
* Data independence
* More information at low cost
* Accuracy and integrity
* Recovery from failure
* Privacy and security

A database is an integrated collection of data and provides centralized access to the data. Usually the centralized data managing the software is called RDBMS. The main significant difference between RDBMS and other DBMS is the separation of data as seen by the program and data has in direct access to stores device. This is the difference between logical and physical data.

**3.3.1 Normalization**

Designing a database is complete task and the normalization theory is a useful aid in the design process. The process of normalization is concerned with transformation of conceptual schema into computer representation form. There will be need for most databases to grow by adding new attributes and new relations. The data will be used in new ways. Tuples will be added and deleted. Information stored may undergo updating also. New association may also be added. In such situations the performance of a database is entirely depend upon its design. A bad database design may lead to certain undesirable things like:

1. Repetition of information

2. Inability to represent certain information

3. Loss of information

To minimize these anomalies, Normalization may be used. If the database is in a normalized form, the data can be growing without, in most cases, forcing the rewriting application programs. This is important because of the excessive and growing cost of maintaining an organization’s application programs and its data from the disrupting effects of database growth. As the quality of application programs increases, the cost of maintaining the without normalization will rise to prohibitive levels. A normalized database can also encompass many related activities of an organization thereby minimizing the need for rewriting the applications of programs. Thus, normalization helps one attain a good database design and there by ensures continued efficiency of database.

Normalization theory is built around the concept of normal forms. A relation is said to be in normal form if it satisfies a certain specified set of constraints. For

example, a relation is said to be in first normal form (1NF) if it satisfies the constraint that it contains atomic values only. Thus every normalized relation is in 1NF.Numerous normal forms have been defined. Cod defined the first three normal forms.

All normalized relations are in 1NF, some 1NF relations are also in 2NF and some 2NF relations are also in 3NF.2NF relations are more desirable than 1Nf and 3NF are more desirable than 2NF. That is, the database designer should prefer 3NF than 1NF or 2NF.Normalization procedure states that a relation that is in some given normal form can be converted into a set of relations in a more desirable form. We can define this procedure as the successive reduction of a given collection of relations to some more desirable form. This procedure is reversible. That is, it is always possible to take the output from the procedure and convert them back into input. In this process, no information is lost. So it is also called “no loss decomposition”.

**First Normal Form**

A relation is in first normal form (1NF) if and all its attributes are based on single domain. The objective of normalizing a table is to remove its repeating groups and ensure that all entries of the resulting table have at most single value.

**Second Normal Form**

A table is said to be second Normal Form (2NF), when it is in 1NF and every attribute in the record is functionally dependent upon the whole key, and not just a part of the key.

**Third Normal Form**

A table is in third Normal Form (3NF), when it is in 2NF and every non-key attribute is functionally dependent on just the primary key.

**3.3.2 Table Structure**

Table is a collection of complete details about a particular subject. These data are saved in rows and Columns. The data of each Row are different units. Hence, rows are called RECORDS and Columns of each row are called FIELDS.

Data is stored in tables, which is available in the backend. The items and data, which are entered in the input, form id directly stored in this table using linking of database. We can link more than one table to input forms. We can collect the details from the different tables to display on the output.

There are mainly 12 tables in our project. They are,

1. tbl\_admin
2. tbl\_department
3. tbl\_course
4. tbl\_subject
5. tbl\_teacher
6. tbl\_student
7. tbl\_teacher\_subject
8. tbl\_class
9. tbl\_file
10. tbl\_question
11. tbl\_exam
12. tbl\_account

Table Name: **tbl\_admin**

Description: Storing login details of admin

Primary Key: id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| User\_id | Int | User\_id, Primary Key , Auto Increment |
| Username | Varchar(10) | Username |
| Password | Varchar(10) | Password |
| First name | Varchar(10) | Firstname |
| Last name | Varchar(10) | Last name |

Table Name**: tbl\_department**

Description: Storing details of department

Primary Key: dept\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Dept\_id | Int | Department id, Primary Key, Auto Increment |
| inCharge | Varchar(10) | inCharge |
| Title | Varchar(10) | Title |
| Department | Varchar(10) | Department name |

Table Name: **tbl\_course**

Description: Storing details of course

Primary Key: Course\_id

Foreign Key: Department\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Course\_id | Varchar(20) | Course id, Primary Key |
| Course\_title | Varchar(10) | Course title |
| Department\_id | Int | Department id, Foreign key |
| Fee structure | Varchar(10) | Fee structure |
| Course duration | Varchar(10) | Course duration |

Table Name: **tbl\_subject**

Primary Key: subject\_id

Foreign Key: course\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Subject\_id | Int | Subject id, Primary key,Auto increment |
| Course\_id | Int | Course id,Foreign Key |
| Subject\_code | Varchar(10) | Subject code |
| Subject\_title | Varchar(10) | Subject title |

Table Name: **tbl\_teacher**

Description: Storing details of teacher

Primary Key: teacher\_id

Foreign Key: Department\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Teacher\_id | Int | Teacher id, Primary key,Auto Increment |
| Username | Varchar(10) | Username |
| Password | Varchar(10) | Password |
| Email | Varchar(10) | Email |
| Firstname | Varchar(10) | First name |
| Last name | Varchar(10) | Last name |
| Middle\_name | Varchar(10) | Middle name |
| Dept\_id | Int | Department id,Foreign key |
| Image | Varchar(10) | Image |

Table Name: **tbl\_student**

Description: Storing details of student

Primary Key: student\_id

Foreign Key: dept\_id, course\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Stud\_id | Int | Student id, Primary key, Auto Increment |
| fullname | Varchar(10) | Fullname |
| Department\_id | Int | Department id, Foreign key |
| Course\_id | Int | Course id, Foreign key |
| Username | Varchar(10) | Username |
| Password | Varchar(10) | Password |
| Image | Varchar(10) | Image |
| Gender | Varchar(10) | Gender |
| Email | Varchar(10) | Email |
| Contact\_no | Int(10) | Contact no |
| City | Varchar(20) | City |
| Due | Varchar(20) | Due |

Table Name: **tbl\_teacher\_subject**

Description: Storing subjects of teacher

Primary Key: teacher\_subject\_id

Foreign Key: teacher\_id, course\_id, subject\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Teacher\_subject\_id | Int | Teacher subject id, primary key, auto increment |
| Teacher\_id | Int | Teacher id, foreign key |
| Course\_id | Varchar(20) | Course id, foreign key |
| Subject\_id | Int | Subject id, foreign key |

Table Name: **tbl\_class**

Description: Storing classes of students

Primary Key: class\_id

Foreign Key: stud\_id,course\_id, teacher\_id,subject\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Class\_id | Int | Class id, primary key, auto increment |
| Stud\_id | Int | Student id, foreign key |
| Course\_id | Varchar(20) | Course id, foreign key |
| Subject\_id | Int | Subject id, foreign key |
| Teacher\_id | Int | Teacher id, foreign key |

Table Name: **tbl\_file**

Description: Storing details of study materials

Primary Key: file\_id

Foreign Key: teacher\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| File\_id | Int,PK,AI | File id |
| Location | Varchar(10),FK | Location |
| Fdatein | Varchar(10) | File uploaded date |
| Fdescription | Varchar(100) | File description |
| Teacher\_id | Int,FK | Teacher id |
| Fname | Varchar(10) | File name |

Table Name: **tbl\_question**

Description: Storing questions and options

Primary Key: qnid

Foreign Key: course\_id, Student\_id, Teacher\_id, Subject\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Qnid | Int,PK,AI | Question id |
| Qn\_paper\_code | Int,FK | Student id |
| Course\_id | Varchar(20) | Course id, foreign key |
| Student\_id | Int | Student id, foreign key |
| Teacher\_id | Int | Teacher id, foreign key |
| Subject\_id | Int | Subject id, foreign key |
| Qns | Varchar(10) | question |
| Option1 | Varchar(10) | Option 1 |
| Option2 | Varchar(10) | Option 2 |
| Option3 | Varchar(10) | Option 3 |
| Option4 | Varchar(10) | Option 4 |
| Correct\_ans | Varchar(10) | Correct answer |

Table Name: **tbl\_exam**

Description: Storing result of examination

Primary Key: Exam\_id

Foreign Key: Student\_id, Subject\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Exam\_id | Int | Question id |
| Student\_id | Int | Student id |
| Subject\_id | Varchar(20) | Course id, foreign key |
| Exam\_date | Int | Student id, foreign key |
| Max\_mark | Float | Maximum mark |
| Mark\_obtained | Float | Mark obtained |

Table Name: **tbl\_account**

Description: Storing details of an account

Primary Key: Acc\_id

|  |  |  |
| --- | --- | --- |
| **FIELD** | **TYPE** | **DESCRIPTION** |
| Acc\_id | Varchar(20) | Account number, primary key |
| Name | Varchar(20) | Name |
| Email | Varchar(10) | Email Id |
| Phno | Date | Phone number |
| Amount | Float | Amount |
| Pin | Int | Pin |

**3.3.3 Data Flow Diagram**

A Data Flow Diagram is a network that describes the flow of data and processes that change, or transform, data throughout the system. This network is constructed by use a set of symbols that do not imply a physical implementation. It is a graphical tool for structured analysis of the system requirements. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates, output-data-flows which go to other processes or external entities or files. Data in files may also flow to processes as inputs.

There are various symbols used in a DFD. Bubbles represent the processes. Named arrows indicate the data flow. External entities are represented by rectangles. Entities supplying data are known as sources and those that consume data are called sinks. Data are stored in a data store by a process in the system. Each component in a DFD is labeled with a descriptive name. Process names are further identified with a number.

The Data Flow Diagram shows the logical flow of a system and defines the boundaries of the system. For a candidate system, it describes the input (source), outputs (destination), database (files) and procedures (data flow), all in a format that meet the user’s requirements.

The main merit of DFD is that it can provide an overview of system requirements, what data a system would process, what transformations of data are done, what files are used, and where the results flow.

**3.3.3.1 Rules for constructing a Data Flow Diagram**

1. Arrows should not cross each other

2. Squares, circles and files must bear names.

3. Choose meaningful names for data flow

4. Draw all data flows around the outside of the diagram

5. Decomposed dataflow squares and circles can have same time

**3.3.3.2 Basic Data Flow Diagram Symbols**

A **data flow** is a route, which enables packets of data to travel from one point to another. Data may flow from a source to a process and from data store or process. An arrow line depicts the flow, with arrow head pointing in the direction of the flow.

**Circles** stands for process that converts data in to information. A process represents transformation where incoming data flows are changed into outgoing data flows.

A **data store** is a repository of data that is to be stored for use by a one or more process may be as simple as

buffer or queue or sophisticated as relational database. They should have clear names. If a process merely uses the content of store and does not alter it, the arrowhead goes only from the store to the process. If a process alters the details in the store then a double-headed arrow is used.

A **source or sink** is a person or part of an organization, which enters or receives information from the system, but is considered to be outside the contest of data flow model.

Fig 3.1 data flow diagram Symbols

**3.3.3.3 Data Flow Diagrams**

Each component in a DFD is labeled with a descriptive name. Process name are further identified with number. Context level DFD is draw first. Then the process is decomposed into several elementary levels and is represented in the order of importance. A DFD describes what data flow (logical) rather than how they are processed, so it does not depend on hardware, software, and data structure or file organization.

A DFD methodology is quite effective; especially when the required design is clear and the analyst need a notation language for communication. The DFD is easy to understand after a brief orientation.

Context Level



Fig 3.2 context level

First Level DFD for E-Learning System



Fig 3.3 First Level DFD for friends clone

Second Level DFD: Registration



Fig3.4 Second Level DFD for registration

Second Level DFD: Student



Fig3.5 Second Level DFD for Product

Second Level DFD: Teacher



Fig3.6 Second Level DFD for Decoration

Second Level DFD: Online Exam



Fig3.7 Second Level DFD for Sales

**3.4 INTERFACE DESIGN**

Interface design deals with the process of developing a method for two (or more) modules in a system to connect and communicate. These modules can apply to hardware, software or the interface between a user and a machine.[1][2][3] An example of a user interface could include a GUI, a control panel for a nuclear power plant, or even the cockpit of an aircraft. n systems engineering, all the inputs and outputs of a system, subsystem, and its components are listed in an interface control document often as part of the requirements of the engineering project. The development of a user interface is a unique field.

**3.4.1 Input Design**

The user interface design is very important for any application. The interface design describes how the software communicates within itself, to system that interpreted with it and with humans who use it. The input design is the process of converting the user-oriented inputs into the computer based format. The data is fed into the system using simple inactive forms. The forms have been supplied with messages so that the user can enter data without facing any difficulty. They data is validated wherever it requires in the project. This ensures that only the correct data have been incorporated into system. The goal of designing input data is to make the automation as easy and free from errors as possible. For providing a good input design for the application easy data input and selection features are adopted. The input design requirements such as user friendliness, consistent format and interactive dialogue for giving the right messages and help for the user at right are also considered for development for this project.

Input Design is a part of the overall design. The input methods can be broadly classified into batch and online. Internal controls must be established for monitoring the number of inputs and for ensuring that the data are valid. The basic steps involved in input design are:

• Review input requirements.

• Decide how the input data flow will be implemented.

• Decide the source document.

• Prototype on line input screens.

• Design the input screens.

The quality of the system input determines the quality of the system output. Input specifications describe the manner in which data enter the system for processing. Input design features can ensure the reliability of the system and produce results from accurate data. The input design also determines whether the user can interact efficiently with the system.

**3.4.2 Output Design**

A quality output is one, which meets the requirements of end user and presents the information clearly. In any system result of processing are communicated to the user and to the other system through outputs. In the output design it is determined how the information is to be displayed for immediate need.

It is the most important and direct source information is to the user. Efficient and intelligent output design improves the system’s relationships with the user and helps in decision -making. The objective of the output design is to convey the information of all the past activities, current status and to emphasis important events. The output generally refers to the results and information that is generated from the system. Outputs from computers are required primarily to communicate the results of processing to the users.

Output also provides a means of storage by copying the results for later reference in consultation. There is a chance that some of the end users will not actually operate the input data or information through workstations, but will see the output from the system.

Two phases of the output design are:

1.Output Definition

2.Output Specification

Output Definition takes into account the type of output contents, its frequency and its volume, the appropriate output media is determined for output. Once the media is chosen, the detail specification of output documents are carried out. The nature of output required from the proposed system is determined during logical design stage. It takes the outline of the output from the logical design and produces output as specified during the logical design phase.

In a project, when designing the output, the system analyst must accomplish the following:

• Determine the information to present.

• Decide whether to display, print, speak the information and select the output medium.

• Arrange the information in acceptable format.

• Decide how to distribute the output to the intended receipt.

Thus by following the above specifications, a high quality output can be generated.

1. **IMPLEMENTATION**

As the system is tested it starts to move into the implementation phase. Ideally the system should be completed and fully tested implementation gets under way but unless a package is being installed this seldom happens. Normally what happens is that parts of the system which are required for file set-up are completed first and this process gets under way. Conversion programs may also have to be available which allow data from another system to be used in setting up the files. Once this data is set up it must kept up-to-date and thus the first use is made of the new system. This may be followed by a period of parallel running and then a decision is made to drop the old system.

Implementation involved placing the completed and tested system of hardware and software into the actual work environment of the users. When systems personnel check out and put new equipment into use, train user personnel, install the new application, and construct any files of data needs to use it, we say it is implemented. There are both technical-and people-oriented activities during this stage. Examples of technical activities include converting data files, replacing old programs with new ones, and scheduling computer operations. Examples of people-oriented activities include orientation, training and support.

Implementation includes all those activities that take place to convert from the old system to the new one. The new system may be totally new, replacing an external system manual or automated system or it may be a modification to an external system. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert to old system to new one. The most crucial stage is achieving a new successful system and giving confidence in new system that it will work efficiently and effectively. The system is implemented only after through checking is done and if it is found working in according to the specifications.

**4.1. CODING STANDARD**

Coding conventions are a set of guidelines for a specific programming language that recommend programming style, practices and methods for each aspect of a piece program written in this language

**4.1.1 Purpose**

To develop reliable and maintainable applications, you must follow coding standards and best practices. The naming conventions, coding standards and best practices described in this document are compiled from various Microsoft and non Microsoft guidelines. There are several standards that exist in the programming industry. You may follow any standard approach, but it is most important to ensure everyone is following the same standard.

**4.1.2 Major Coding Standards**

**4.1.2.1 Naming Conventions**

1. Use Pascal casing for Class names, method names and file names.
2. Use Camel casing for variables and method parameters
3. Use the prefix “I” with Camel Casing for interfaces.
4. Use Meaningful, descriptive words to name variables. Do not use abbreviations.
5. Do not use single character variable names like i, n, s etc. Use names like index, temp. One exception in this case would be variables used for iterations in loops:
6. Do not use underscores (\_) for local variable names.
7. All member variables must be prefixed with underscore (\_) so that they can be identified from other local variables.
8. Declare all member variables at the top of a class, with static variables at the very top.
9. Do not use variable names that resemble keywords.
10. File name should match with class name.

Table 4.1 List of Control Prefixes

|  |  |
| --- | --- |
| **Control** | **Prefix** |
| Label | lbl |
| TextBox | txt |
| DataGrid | dtg |
| Button | btn |
| ImageButton | imb |
| HyperLink | hlk |
| DropDownList | ddl |
| ListBox | lst |
| DataList | dtl |
| Repeater | rep |
| CheckBox | chk |
| CheckBoxList | cbl |
| RadioButton | rdo |
| RadioButtonList | rbl |
| Image | img |
| Panel | pnl |
| PlaceHolder | phd |
| Validators | val |
| Table | tbl |
| AdRotator | adr |
| BulletedList | blt |
| Calendar | cal |
| FileUpload | flp |
| HiddenField | hdn |
| ImageMap | map |
| LinkButton | lbn |

Table 4.2 List of Variable Prefixes

|  |  |
| --- | --- |
| **Variable** | **Prefix** |
| Array | A |
| Bool | B |
| Char | C |
| Byte | By |
| Double | D |
| Integer | I |
| Short | N |
| String | Str |
| Date & time | Dt |

**4.1.2.2 Indentation and Spacing**

1. Use TAB for indentation. Do not use SPACES.
2. Comments should be in the same level as the code
3. Curly braces ( {} ) should be in the same level as the code outside the braces
4. Use one blank line to separate logical groups of code
5. There should be one and only one single blank line between each method inside the class.
6. The curly braces should be on a separate line and not in the same line as if, for etc
7. Use a single space before and after each operator.

**4.1.2.3 Good Programming practices**

1. Convert strings to lowercase or upper case before comparing. This will ensure the string will match even if the string being compared has a different case.
2. Avoid writing very long methods. A method should typically have 1~25 lines of code. If a method has more than 25 lines of code, you must consider refactoring into separate methods.
3. Method name should tell what it does. Do not use mis-leading names. If the method name is obvious, there is no need of documentation explaining what the method does.
4. A method should do only 'one job'. Do not combine more than one job in a single method, even if those jobs are very small.
5. Show short and friendly message to the user. But log the actual error with all possible information. This will help a lot in diagnosing problems.
6. Do not make the member variables public or protected. Keep them private and expose public/protected Properties.
7. Declare variables as close as possible to where it is first used. Use one variable declaration per line.
8. Always watch for unexpected values. For example, if you are using a parameter with 2 possible values, never assume that if one is not matching then the only possibility is the other value.
9. Do not hardcode numbers. Use constants instead. Declare constant in the top of the file and use it in your code. However, using constants are also not recommended. You should use the constants in the config file or database so that you can change it later. Declare them as constants only if you are sure this value will never need to be changed.
10. Convert strings to lowercase or upper case before comparing. This will ensure the string will match even if the string being compared has a different case.

1. The event handler should not contain the code to perform the required action. Rather call another method from the event handler.
2. Never hardcode a path or drive name in code. Get the application path programmatically and use relative path.
3. Avoid having very large files. If a single file has more than 1000 lines of code, it is a good candidate for refactoring. Split them logically into two or more classes.
4. Do not store large objects in session. Storing large objects in session may consume lot of server memory depending on the number of users.
5. Avoid passing too many parameters to a method. If you have more than 4~5 parameters, it is a good candidate to define a class or structure.
6. If you have a method returning a collection, return an empty collection instead of null, if you have no data to return. For example, if you have a method returning an ArrayList, always return a valid ArrayList. If you have no items to return, then return a valid ArrayList with 0 items. This will make it easy for the calling application to just check for the “count” rather than doing an additional check for “null”.
7. Use StringBuilder class instead of String when you have to manipulate string objects in a loop. The String object works in weird way in .NET. Each time you append a string, it is actually discarding the old string object and recreating a new object, which is a relatively expensive operations.
8. To validate form field input received through HTML input controls, query string parameters or cookies, perform validation in server-side code and use the **Regex** class to help constrain text input.

**Exception Handling**

1. Always catch only the specific exception, not generic exception.
2. In case of exceptions, give a friendly message to the user, but log the actual error with all possible details about the error, including the time it occurred, method and class name etc.
3. Do not write very large try-catch blocks. Write your own custom exception classes if required in your application. Do not derive your custom exceptions from the base class System Exception. Instead, inherit from Application Exception.

**4.2 SCREEN SHOTS**

**5. TESTING**

Coding conventions are a set of guidelines for a specific programming language that recommend programming style, practices and methods for each aspect of a piece program written in this language. These conventions usually cover file organization, indentation, comments, declarations, statements, white space, naming conventions, programming practices, programming principles, programming rules of thumb, architectural best practices, etc. These are guidelines for software structural quality. Software programmers are highly recommended to follow these guidelines to help improve the readability of their source code and make software maintenance easier.

**5.1 TEST CASES**

The objective of system testing is to ensure that all individual programs are working as expected, that the programs link together to meet the requirements specified and to ensure that the computer system and the associated clerical and other procedures work together. The initial phase of system testing is the responsibility of the analyst who determines what conditions are to be tested, generates test data, produced a schedule of expected results, runs the tests and compares the computer produced results with the expected results with the expected results. The analyst may also be involved in procedures testing. When the analyst is satisfied that the system is working properly, he hands it over to the users for testing. The importance of system testing by the user must be stressed. Ultimately it is the user must verify the system and give the go-ahead.

During testing, the system is used experimentally to ensure that the software does not fail, i.e., that it will run according to its specifications and in the way users expect it to. Special test data is input for processing (test plan) and the results are examined to locate unexpected results. A limited number of users may also be allowed to use the system so analysts can see whether they try to use it in unexpected ways. It is preferably to find these surprises before the organization implements the system and depends on it. In many organizations, testing is performed by persons other than those who write the original programs. Using persons who do not know

how certain parts were designed or programmed ensures more complete and unbiased testing and more reliable software.

Parallel running is often regarded as the final phase of system testing. Since he parallel operation of two systems is very demanding in terms of user resources it should be embarked on only if the user is satisfied with the results of testing -- it should not be started if problems are known to exist. Testing is the major quality control measure during software development. Its basic function is to detect errors in the software. Thus the goal of testing is to uncover requirement design and coding errors in the program.

Testing is the process of correcting a program with intends of finding an error. Different types of testing are,

1. Unit Testing
2. Integrated Testing
3. Black Box Testing
4. White Box Testing
5. Validation Testing
6. Output Testing
7. User Acceptance Testing

**5.1.1 Unit Testing**

In computer programming, unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. In this testing we test each module individual and integrated the overall system. Unit testing focuses verification efforts on the smaller unit of software design in the module. This is also known as module testing. The modules of the system are tested separately. The testing is carried out during programming stage itself. In this testing step each module is found to working satisfactory as regard to the expected output from the module. There are some validation checks for verifying the data input given by the user which both the formal and validity of the entered. It is very easy to find error debug the system.

**5.1.2 Integration Testing**

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed. Data can be lost across an interface; one module can have an adverse effort on the other sub functions when combined by, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. This testing was done with sample data. The developed system has run success full for this sample data. The need for integrated test is to find the overall system performance.

**5.1.3 Black Box Testing**

Black-box testing is a method of software testing that examines the functionality of an application (e.g. what the software does) without peering into its internal structures or workings. This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher level testing, but can also dominate unit testing as well. In black box testing the structure of the program is not considered. Test cases are decided solely on the basis of the requirements or the specification of the program or module, and the internals of the module or program are not considered for selection of the test cases.

**5.1.4 White Box Testing**

White-box testing (also known as clear box testing, glass box testing, transparent box testing and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the

code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).

White-box test design techniques include:

* Control flow testing
* Data flow testing
* Branch testing
* Path testing
* Statement coverage
* Decision coverage

**5.1.5 Validation Testing**

At the culmination of Black Box testing, software is completely assembled as a package, interface errors have been uncovered and corrected and final series of software tests, Validation tests begins. Validation testing can be defined many was but a simple definition is that validation succeeds when the software functions in a manner that can be reasonably accepted by the customer. After validation test has been conducted one of the two possible conditions exists.

1. The function or performance characteristics confirm to specification and are accepted.
2. A derivation from specification uncovered and a deficiency list is created.

**5.1.6 User Acceptance Testing**

Acceptance Testing is a level of the software testing process where a system is tested for acceptability. User Acceptance testing is the software testing process where system tested for acceptability & validates the end to end business flow. Such type of testing executed by client in separate environment & confirms whether system meets the requirements as per requirement specification or not.

UAT is performed after System Testing is done and all or most of the major defects have been fixed. This testing is to be conducted in the final stage of Software Development Life Cycle (SDLC) prior to system being delivered to a live environment. UAT users or end users are concentrating on end to end scenarios & typically involves running a suite of tests on the completed system.

**5.2 TEST REPORTS**

Test Report provides a summary of the results of test performed. A Test report must contain the following details:-

* **Test Summary**

This must include basic information about what was tested and what happened.

* **Test Type**

This must include basic information about what type of testing (unit testing, integration testing, validation testing etc.) was done and what happened.

* **Test Assessment**

It should contain a comprehensive assessment of your interpretation of how adequate the test was in light of how thorough the test plan said it should be? It must also specify What wasn't tested well enough.

* **Test Results**

Summarize the test results. Include a detailed description of any deviations from the original test plan, design, test case, or expected results. Include any issues or bugs discovered during the test.

* **Variances**

Describe any variances between the testing that was planned and the testing that actually occurred. Also, provide an assessment of the manner in which the test

environment may be different from the operational environment and the effect of this difference on the test results.

* **Test Instances**

Provide a brief description of the unexpected results, problems, or defects that occurred during the testing.

## Resolved Test Incidents

Identify all resolved test incidents and summarize their resolutions. Reference may be made to Test Incident Reports that describe in detail the unexpected results, problems, or defects reported during testing, along with their documented resolutions, which may be included as an appendix to test document.

## Unresolved Test Incidents

Identify all unresolved test incidents and provide a plan of action for their resolution.

* **Recommendations**

Describe what actions are suggested upon completion of this test. Provide any recommended improvements in the design, operation, or future testing of the business product that resulted from the testing being reported. A discussion of each recommendation and its impact on the business product may be provided. If there are no recommendations to report, then simply state as such.

**6. CONCLUSION**

**6.1 DESIGN AND IMPLEMENTATION ISSUES**

The relevance of the project is clear, since the implementation of the system is very easy.Even though the computer and its applications are rapidly becoming popular, only a few established organization, companies and educational institutions are automatic the system. The others just continue with the old type of processing, which is the manual system.

The main idea behind this project is for tracking the details of the purchase of products and services on the internet.The Online Shopping system enables vendors to setup online shops, customers to browse through the shops, and a system administrator to approve and reject request for new shops and maintain list of shop categories. Also on the agenda is designing an Online Shopping site to manage the items in the shop and also help customers purchase them online without having to visit the shop physically. My Online Shopping mall will use the internet as the sole method for sealing goods to its consumers. Shopping will be highly personalized and the mall will provide lower prices than most competitors.

**6.2 ADVANTAGES AND LIMITATIONS**

**6.2.1. Advantages of the Project**

* Reduction of time
* Reducing cost
* Increased accuracy
* Increased data security
* It provides better and efficient services.
* Increased Reliability.
* Reduce the storage space
* Redundancy

**6.2.2. Limitations of the Project**

* Internet facility is needed for the successful working of the project.

**6.3 FUTURE ENHANCEMENTS**

This project adds a number of functionalities to the system. But still it have a greater enhancement in future. They are the system can‘t the payment facility connected through the banking system.

* + Easy to generate the Purchase Bill
  + Includes several categories for sale
  + These categories includes several subcategories
  + Includes searching based on popularity

**REFERENCES**

[1] Waman S Jawadekar –“Software Engineering principles & Practice”- 2nd EditionTata

Mc-Graw Hilll Publishing Co. Ltd.

[2] "Android Overview". Open Handset Alliance Retrieved 2012-02-15.

<<http://en.wikipedia.org/wiki/Android_(operating_system)>>

[3] MacDonald, Matthew; Szpuszta, Mario (2005). Pro ASP.NET 2.0 in C# 2005 (1st edition

<<http://en.wikipedia.org/wiki/ASP.NET>>

[4] <http://wwwmsdn.microsoft.com/en-us/library/aa287561(vs.71).aspx>

[5] Agarwal,Bharat Bhushan and Tayal Sumit Prakash, Software Engineering. Laxmi Publication (p) Ltd.-NewDelhi. 2007

[6] Kolawa, Adam and Huizinga, Dorota. Automated Defect Prevention: Best Practices in Software Management. Wiley-IEEE Computer Society Press.2007

[7]Software testing. 21 October 2013. <http:// en.wikipedia.org/ wiki/ Software testing>

[8]Black box testing. 18 July 2013. < http://en.wikipedia.org/ wiki/ Black-box testing>

[9] White box testing.25 September 2013. <http://en.wikipedia.org/wiki/White-box\_testing>

[10]Systems architecture. 14 August 2013. <http://en.wikipedia.org/ wiki/ Systems architecture>