



**NEW HORIZON
COLLEGE OF ENGINEERING**



“ONLINE EXAMINATION”

A MINI PROJECT REPORT

Submitted by

PRAVEEN VISHWANATH HEGDE [1NH19IS405]

Under the guidance of,

Mrs. D.Jessica Prathyusha

Sr. Assistant Professor, ISE, NHCE

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION SCIENCE AND ENGINEERING

FOR

COURSE NAME: MINI PROJECT

COURSE CODE: 20ISE59



NEW HORIZON
COLLEGE OF ENGINEERING



CERTIFICATE

Certified that the project work entitled Online Examination carried out by Mr. PRAVEEN VISHWANATH HEGDE, bearing USN 1NH19IS405, a bonafide student of 5th semester in partial fulfillment for the award of Bachelor of Engineering in Information Science & Engineering of the Visveswaraiah Technological University, Belagavi during the year 2020-21. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated. The project report has been approved as it satisfies the academic requirements in respect of Mini Project work prescribed for the said Degree.

Name & Signature of Guide

Mrs. D.Jessica prathyusha

Name & Signature of HOD

Dr. Anandhii R J

Name & Signature of Principal

Dr. Manjunatha

Examiners:

Name

Signature

1.

.....

2.

.....

ABSTRACT

Online examinations are an important method of evaluating the success potential of students. This research effort the individuals under consideration were students who would be enrolling in computer courses or Technologies Registrations. A prototype of a web-based placement examination system is described from the standpoint of the research effort, end user, and software development.

The administrators, instructor, Students who are attending for online examination can communicate with the system through this project, thus facilitating effective implementation and monitoring of various activities of Online Examinations like conducting Exams as per scheduled basis and delivering result to that particular use or student. And the details of students who attempted Online Examination are maintained at administrator.

ACKNOWLEDGEMENT

Any project is a task of great enormity and it cannot be accomplished by an individual without support and guidance. I am grateful to a number of individuals whose professional guidance and encouragement has made this project completion a reality.

I have a great pleasure in expressing my deep sense of gratitude to the beloved Chairman **Dr. Mohan Manghnani** for having provided me with a great infrastructure and well-furnished labs.

I take this opportunity to express my profound gratitude to the Principal **Dr. Manjunatha** for his constant support and management.

I am grateful to **Dr. R J Anandhi**, Professor and Head of Department of ISE, New Horizon College of Engineering, Bengaluru for his strong enforcement on perfection and quality during the course of my project work.

I would like to express my thanks to the guide Mrs. **D.Jessica Prathyusha**, Senior Assistant Professor, Department of ISE, New Horizon College of Engineering, Bengaluru who has always guided me in detailed technical aspects throughout my project.

I would like to mention special thanks to all the Teaching and Non-Teaching staff members of Information Science and Engineering Department, New Horizon College of Engineering, Bengaluru for their invaluable support and guidance.

PRAVEEN VISHWANATH HEGDE
1NH19IS405

TABLE OF CONTENTS

<u>S. N</u>	<u>CONTENTS</u>
1.	INTRODUCTION
2.	ANALYSIS
2.1	SYSTEM ANALYSIS
2.2	SYSTEM SPECIFICATIONS
3.	DESIGN APPROACH
3.1	INTRODUCTION TO DESIGN
3.2	UML DIAGRAMS
3.3	USECASE DIAGRAM
3.4	SEQUENCE DIAGRAM
3.5	COLLABORATION DIAGRAM
3.6	CLASS DIAGRAM
3.7	STATE CHART DIAGRAM
3.8	DATA FLOW DIAGRAMS
3.9	E-R DIAGRAMS
4.	IMPLEMENTATION
4.1	CONCEPTS AND TECHNIQUES

4.2 TESTING

4.2.1 TEST CASES

5. CONCLUSION

6. FUTURE ENHANCEMENTS

7. BIBLIOGRAPHY

CHAPTER 1**1. INTRODUCTION:**

on-line examinations contents providers to focus on creating effective assessment questions and focusing on exam's feedback delivery to students. In the paper we present techniques that are pertinent to the elements of assessment process: answers submission, computerized grading, and feedback after submission.

As the modern organizations are automated and computers are working as per the instructions, it becomes essential for the coordination of human beings, commodity and computers in a modern organization.

The administrators, instructor, Students who are attending for online examination can communicate with the system through this project, thus facilitating effective implementation and monitoring of various activities of Online Examinations like conducting Exams as per scheduled basis and delivering result to that particular user or student. And the details of students who attempted Online Examination are maintained at administrator.

CHAPTER 2

2.1 SYSTEM ANALYSIS:

1. Existing System

Existing system is a manual one in which users are maintaining books to store the information like Student Details, Instructor Details, Schedule Details and feedbacks about students who attempted exam as per schedule. It is very difficult to maintain historical data.

DISADVANTAGES:

The following drawbacks of existing system emphasize the need for computerization:

1. A lot of copies of question papers have to be made
2. A lot of correction work hence delay in giving the results
3. A lot of tabulation work for each subject results

2. Proposed System

This application is used to conduct online examination. The students can sit at individual terminals and login to write the exam in the given duration. . The questions have to be given to the students. This application will perform correction, display the result immediately and also store it in database. This application provides the administrator with a facility to add new exams. This application provides the Instructor add questions to the exam, modify questions in the exam in a particular exam. This application takes care of authentication of the administrator, Instructor as well as the student.

3. Objective of the System

The objective of the Online Examination Tool is to provide better information for the users of this system for better results for their maintenance in student examination schedule details and grading details.

2.2 SYSTEM SPECIFICATIONS:

Hardware Requirements:-

- Pentium-IV(Processor).
- 256 MB Ram
- 512 KB Cache Memory
- Hard disk 10 GB
- Microsoft Compatible 101 or more Key Board

Software Requirements: -

- **Operating System :** Windows
- **Web-Technology:** PHP
- **Front-End:** HTML,CSS,JAVASCRIPT
- **Back-End:** MySQL
- **Web Server:** Apache SERVER.

CHAPTER 3

3.1 INTRODUCTION TO DESIGN:

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization.

Once the software requirements have been analyzed and specified the software design involves three technical activities - design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer's requirements into finished software or a system.

Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

3.2 UML DIAGRAMS:

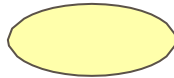
Actor:

A coherent set of roles that users of use cases play when interacting with the use `cases.



Use case:

A description of sequence of actions, including variants, that a system performs that yields an observable result of value of an actor.



UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and documenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

There are various kinds of methods in software design:

They are as follows:

- Use case Diagram
- Sequence Diagram
- Collaboration Diagram
- Activity Diagram
- State chat Diagram

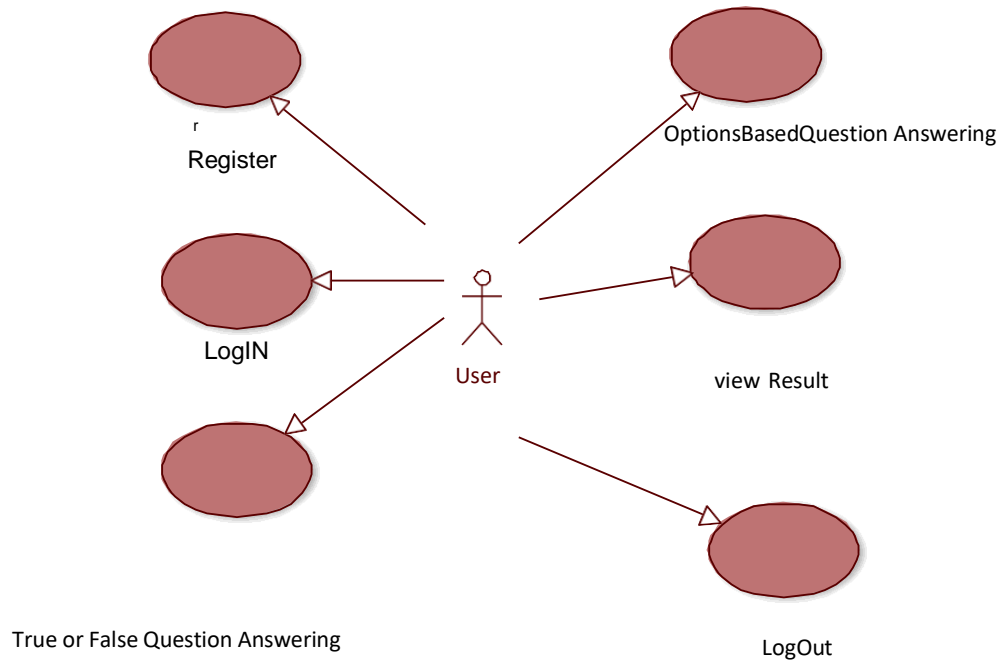
3.3 USECASE DIAGRAMS:

Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what's called an actor.

Use case diagram can be useful for getting an overall view of the system and clarifying who can do and more importantly what they can't do. Use case diagram consists of use cases and actors and shows the interaction between the use case and actors.

- The purpose is to show the interactions between the use case and actor.
- To represent the system requirements from user's perspective.
- An actor could be the end-user of the system or an external system.

A Use case is a description of set of sequence of actions. Graphically it is rendered as an ellipse with solid line including only its name. Use case diagram is a behavioral diagram that shows a set of use cases and actors and their relationship. It is an association between the use cases and actors. An actor represents a real-world object. Primary Actor – Sender, Secondary Actor Receiver.

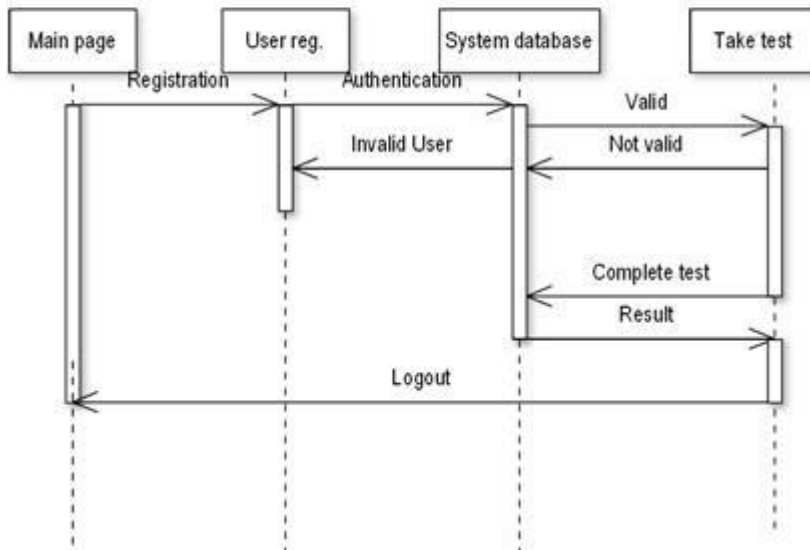


3.4 SEQUENCE DIAGRAM:

Sequence diagram and collaboration diagram are called INTERACTION DIAGRAMS. An interaction diagram shows an interaction, consisting of set of objects and their relationship including the messages that may be dispatched among them.

A sequence diagram is an introduction that empathizes the time ordering of messages. Graphically a sequence diagram is a table that shows objects arranged along the X-axis and messages ordered in increasing time along the Y-axis

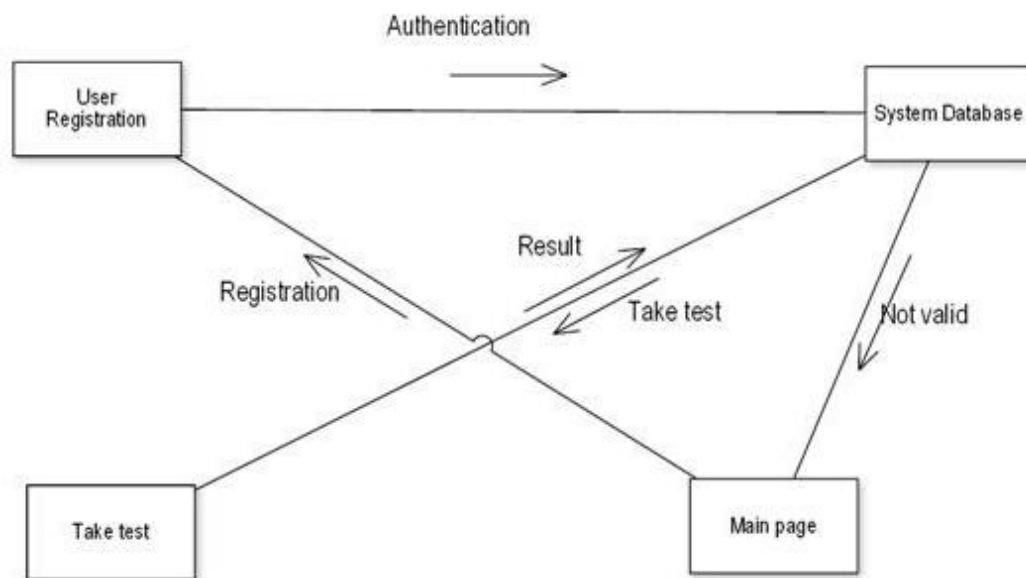
Sequence Diagram



3.5 COLLABORATION DIAGRAM:

A collaboration diagram is an introduction diagram that emphasizes the structural organization of the objects that send and receive messages. Graphically a collaboration diagram is a collection of vertices and arcs.

Collaboration Diagram



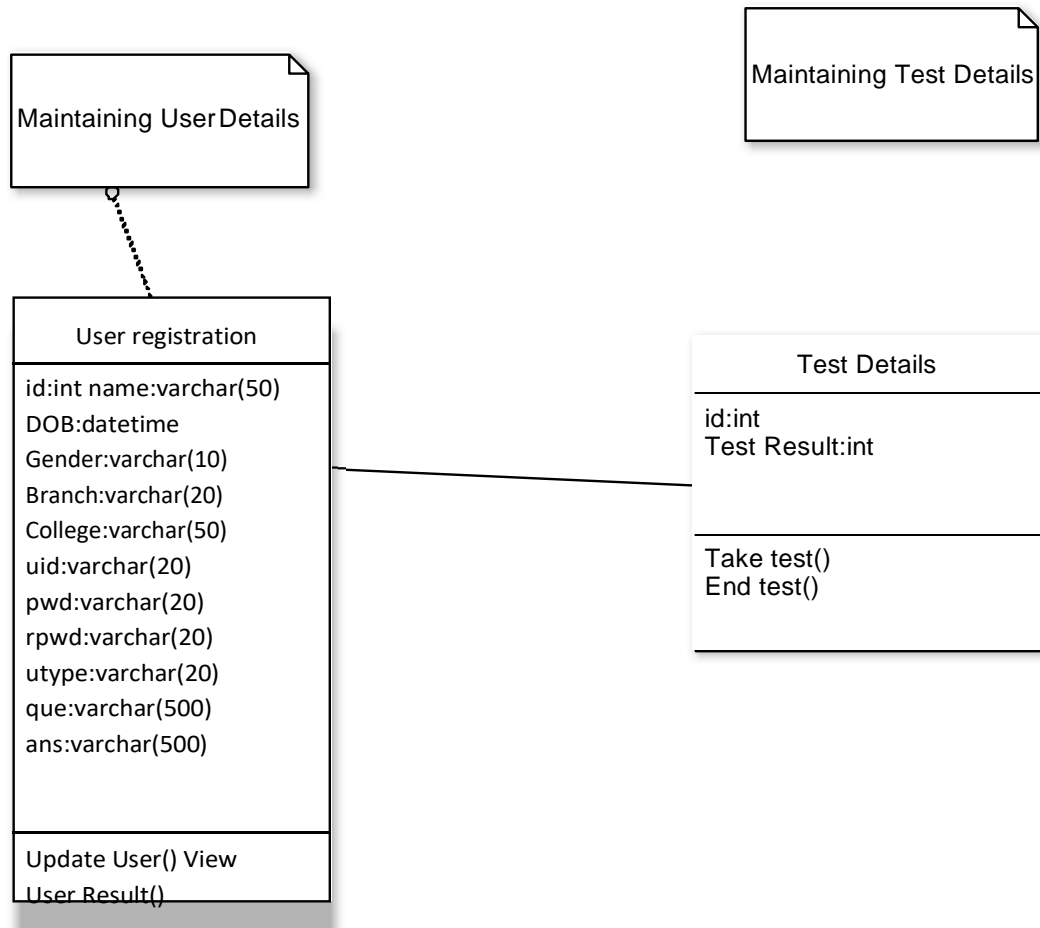
3.6 CLASS DIAGRAM:

Class is nothing but a structure that contains both variables and methods. The Class Diagram shows a set of classes, interfaces, and collaborations and their relating ships. There is most common diagram in modeling the object-oriented systems and are used to give the static view of a system. It shows the dependency between the classes that can be used in our system.

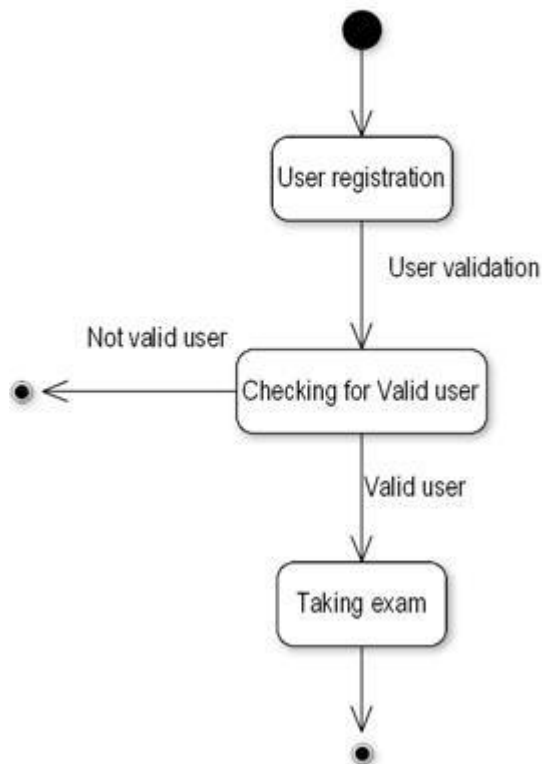
The interactions between the modules or classes of our projects are shown below. Each block contains Class Name, Variables and Methods.

CLASS:

A description of set of objects that share the same attributes, operations, relationships, and semantics



3.7 STATE CHART DIAGRAM:

Statechart Diagram

3.8 DATA FLOW DIAGRAMS:

The DFD takes an input-process-output view of a system i.e. data objects flow into the software, are transformed by processing elements, and resultant data objects flow out of the software.

Data objects represented by labeled arrows and transformation are represented by circles also called as bubbles. DFD is presented in a hierarchical fashion i.e. the first data flow model represents the system as a whole. Subsequent DFD refine the context diagram (level 0 DFD), providing increasing details with each subsequent level.

The DFD enables the software engineer to develop models of the information domain & functional domain at the same time. As the DFD is refined into greater levels of details, the analyst performs an implicit functional decomposition of the system. At the same time, the DFD refinement results in a corresponding refinement of the data as it moves through the process that embody the applications.

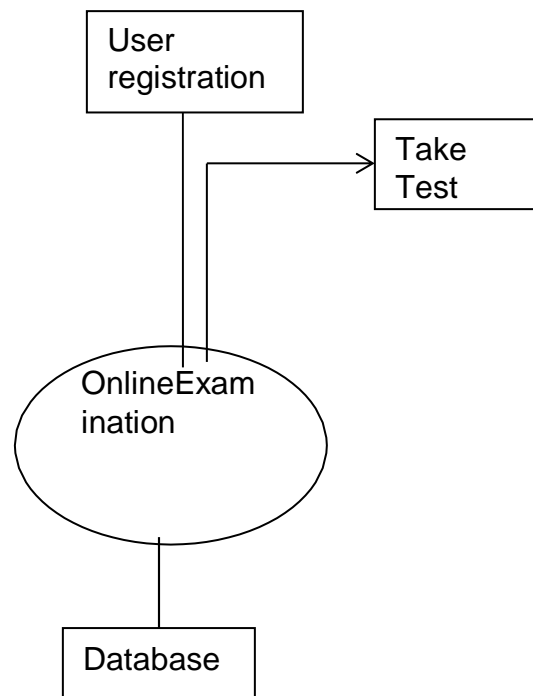
A context-level DFD for the system the primary external entities produce information for use by the system and consume information generated by the system. The labeled arrow represents data objects or object hierarchy.

RULES FOR DFD:

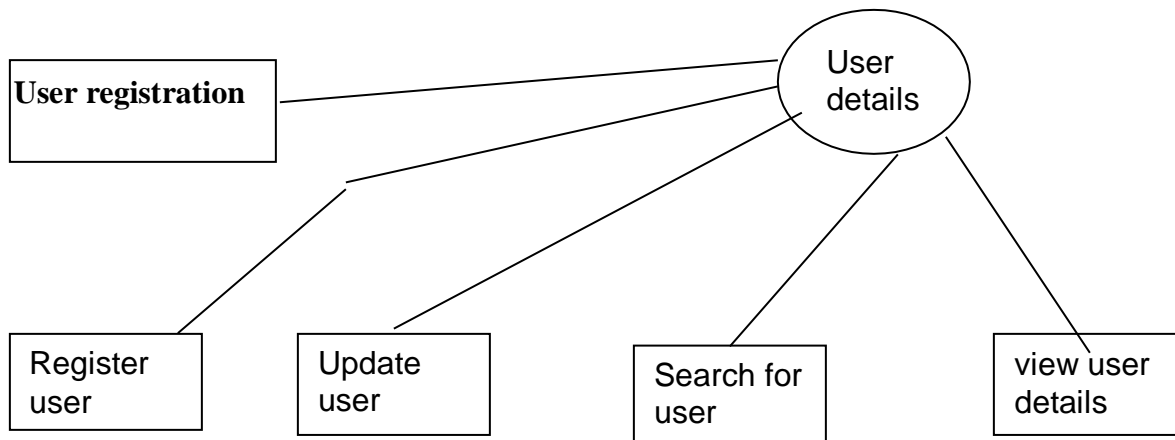
- Fix the scope of the system by means of context diagrams.
- Organize the DFD so that the main sequence of the actions
- Reads left to right and top to bottom.
- Identify all inputs and outputs.
- Identify and label each process internal to the system with Rounded circles.
- A process is required for all the data transformation and Transfers. Therefore, never connect a data store to a data Source or the destinations or another data store with just a Data flow arrow.
- Do not indicate hardware and ignore control information.
- Make sure the names of the processes accurately convey everything the process is done.
- There must not be unnamed process.
- Indicate external sources and destinations of the data, with Squares.
- Number each occurrence of repeated external entities.
- Identify all data flows for each process step, except simple Record retrievals.
- Label data flow on each arrow.
- Use details flow on each arrow.
- Use the details flow arrow to indicate data movements.

DATAFLOW DIAGRAMS:

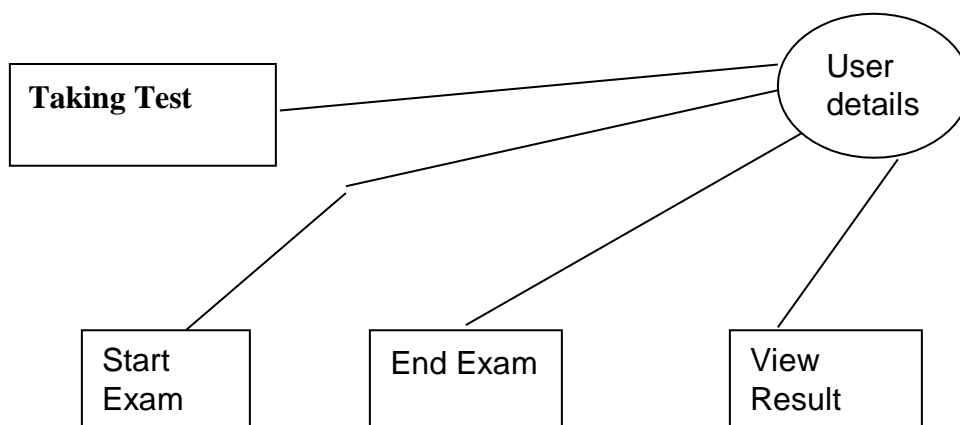
Database:



user registration:



Taking Test:



3.9. E-R DIAGRAMS:

The Entity-Relationship (ER) model was originally proposed by Peter in 1976 [Chen76] as a way to unify the network and relational database views. Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represents data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database design For the database designer, the utility of the ER model is:

- it maps well to the relational model. The constructs used in the ER model can easily be transformed into relational tables.
- it is simple and easy to understand with a minimum of training. Therefore, the model can be used by the database designer to communicate the design to the end user.
- In addition, the model can be used as a design plan by the database developer to implement a data model in a specific database management software.

Connectivity and Cardinality :

The basic types of connectivity for relations are: one-to-one, one-to-many, and many-to-many. A one-to-one (1:1) relationship is when at most one instance of a entity A is associated with one instance of entity B. For example, "employees in the company are each assigned their own office. For each employee there exists a unique office and for each office there exists a unique employee.

A one-to-many (1:N) relationships is when for one instance of entity A, there are zero, one, or many instances of entity B, but for one instance of entity B, there is only one instance of entity A. An example of a 1:N relationships is a department has many employees each employee is assigned to one department.

A many-to-many (M:N) relationship, sometimes called non-specific, is when for one instance of entity A, there are zero, one, or many instances of entity B and for one instance of entity B there are zero, one, or many instances of entity A. The connectivity of a relationship describes the mapping of associated

ER Notation:

There is no standard for representing data objects in ER diagrams. Each modeling methodology uses its own notation. The original notation used by Chen is widely used in academics texts and journals but rarely seen in either CASE tools or publications by non-academics. Today, there are a number of notations used, among the more common are Bachman, crow's foot, and IDEFIX.

All notational styles represent entities as rectangular boxes and relationships as lines connecting boxes. Each style uses a special set of symbols to represent the cardinality of a connection. The notation used in this document is from Martin. The symbols used for the basic ER constructs are:

- **entities** are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.
- **relationships** are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs
- **attributes**, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
- **cardinality** of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.
- **existence** is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional

CHAPTER 4

4.1 OVERVIEW OF TECHNOLOGIES USED

PHP

PHP: Hypertext Preprocessor, is a widely used, general-purpose scripting language that was originally designed for web development, to produce dynamic web pages. It can be embedded into HTML and generally runs on a web server, which needs to be configured to process PHP code and create web page content from it. It can be deployed on most web servers and on almost every operating system and platform free of charge.

PHP was originally created by Rasmus Lerdorf in 1995 and has been in continuous development ever since. The main implementation of PHP is now produced by The PHP Group and serves as the de facto standard for PHP as there is no formal specification. PHP is free software released under the PHP License, which is incompatible with the GNU General Public License (GPL) because of restrictions on the use of the term PHP

PHP has evolved to include a command line interface capability and can also be used in standalone graphical applications.

USAGE

PHP is a general-purpose scripting language that is especially suited for web development. PHP generally runs on a web server. Any PHP code in a requested file is executed by the PHP runtime, usually to create dynamic web page content. It can also be used for command-line scripting and client-side GUI applications. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems. It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.

PHP primarily acts as a filter, taking input from a file or stream containing text and/or PHP instructions and outputs another stream of data; most commonly the output will be HTML. Since PHP 4, the PHP parser compiles input to produce byte code for processing by the Zend Engine, giving improved performance over its interpreter predecessor

Originally designed to create dynamic web pages, PHP now focuses mainly on server-side scripting, and it is similar to other server-side scripting languages that provide dynamic content from a web server to a client, such as Microsoft's Active Server Pages, Sun Microsystems' Java Server Pages and Perl. PHP has also attracted the development of many frameworks that provide building blocks and a design structure to promote rapid application development (RAD).

About HTML

HTML, which stands for **Hyper Text Markup Language**, is the predominant markup language for web pages. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists etc as well as for links, quotes, and other items. It allows images and objects to be embedded and can be used to create interactive forms. It is written in the form of HTML elements consisting of "tags" surrounded by angle brackets within the web page content. It can include or can load scripts in languages such as JavaScript which affect the behavior of HTML processors like Web browsers; and Cascading Style Sheets (CSS) to define the appearance and layout of text and other material. The W3C, maintainer of both HTML and CSS standards, encourages the use of CSS over explicit presentational markup.

Hyper Text Markup Language(HTML) is the encoding scheme used to create and format a web document. A user need not be an expert programmer to make use of HTML for creating hypertext documents that can be put on the internet.

Most graphical e-mail clients allow the use of a subset of HTML (often ill-defined) to provide formatting and semantic markup not available with plain text. This may include typographic information like colored headings, emphasized and quoted text, inline images and diagrams. Many such clients include both a GUI editor for composing HTML e-mail messages and a rendering engine for displaying them. Use of HTML in e-mail is controversial because of compatibility issues, because it can help disguise phishing attacks, because it can confuse spam filters and because the message size is larger than plain text.

NAMING CONVENTIONS

The most common filename extension for files containing HTML is .html. A common abbreviation of this is .htm, which originated because some early operating systems and file systems, such as DOS and FAT, limited file extensions to three letters.

HTML APPLICATION

An HTML Application is a Microsoft Windows application that uses HTML and Dynamic HTML in a browser to provide the application's graphical interface. A regular HTML file is confined to the security model of the web browser, communicating only to web servers and manipulating only webpage objects and site cookies. An HTA runs as a fully trusted application and therefore has more privileges, like creation/editing/removal of files and Windows Registry entries. Because they operate outside the browser's security model, HTAs cannot be executed via HTTP, but must be downloaded (just like an EXE file) and executed from local file system

ABOUT JAVASCRIPT

JavaScript is an object-oriented scripting language used to enable programmatic access to objects within both the client application and other applications. It is primarily used in the form of client-side JavaScript, implemented as an integrated component of the web browser, allowing the development of enhanced user interfaces and dynamic websites. JavaScript is a dialect of the ECMAScript standard and is characterized as a dynamic, weakly typed, prototype-based language with first-class functions. JavaScript was influenced by many languages and was designed to look like Java, but to be easier for non-programmers to work with.

PROTOTYPE-BASED

JavaScript uses prototypes instead of classes for inheritance. It is possible to simulate many class-based features with prototypes in JavaScript.

Functions double as object constructors along with their typical role. Prefixing a function call with new creates a new object and calls that function with its local this keyword bound to that object for that invocation. The constructor's prototype property determines the object used for the new object's internal prototype. JavaScript's built-in constructors, such as Array, also have prototypes that can be modified.

Unlike many object-oriented languages, there is no distinction between a function definition and a method definition. Rather, the distinction occurs during function calling; a function can be called as a method. When a function is called as a method of an object, the function's local this keyword is bound to that object for that invocation.

ABOUT MySQL

MySQL Introduction

There are a large number of database management systems currently available, some commercial and some free.

Some of them : Oracle, Microsoft Access, MySQL and PostgreSQL.

These database systems are powerful, feature-rich software, capable of organizing and searching millions of records at very high speeds.

Understanding Databases, Records, and Primary Keys

Every Database is composed of one or more tables.

These Tables, which structure data into rows and columns, Impose organization on the data.

The records in a table(below) are not arranged in any particular order.

To make it easy to identify a specific record, therefore, it becomes necessary.

standing Relationships and Foreign Keys(RDBMS)

You already know that a single database can hold multiple tables.

In a Relational database management system(RDBMS), these tables can be linked to each other by one or more common fields, called **foreign keys**.

What is Database administrator(DBA) ?

Database administrator is the super user of database, he has unrestricted rights and privileges to access database, grant permission to other database users.

What is Database user(DBU) ?

Database user is the person who uses the database in a restricted privileges, provided by database administrator.

Download MySQL Database

If you have installed PHP's WAMP or XAMPP server, then MySQL database already exists. if you don't have then download MySQL database from here <http://www.mysql.com>

DATABASE TABLES:

USER REG TABLE

NAME	NULL/NOTNULL	TYPE	KEY
ID	NOTNULL	INT	PRIMARYKEY
NAME	NULL	VARCHAR(50)	
DOB	NULL	DATETIME	
GENDER	NULL	VARCHAR(10)	
BRANCH	NULL	VARCHAR(20)	
COLLEGE	NULL	VARCHAR(50)	
UID	NULL	VARCHAR(50)	
PWD	NULL	VARCHAR(20)	
RPWD	NULL	VARCHAR(20)	
UTYPE	NULL	VARCHAR(20)	
QUE	NULL	VARCHAR(500)	
ANS	NULL	VARCHAR(500)	

True/False Based Question Table

NAME	NULL/NOTNULL	TYPE	KEY
ID	NOTNULL	INT	PRIMARYKEY
QUE	NULL	VARCHAR(500)	
AW	NULL	VARCHAR(500)	

True/False Based Answer Table

NAME	NULL/NOTNULL	TYPE	KEY
ID	NOTNULL	INT	FOREIGNKEY
AW	NULL	VARCHAR(500)	

Options Based Question Table

NAME	NULL/NOTNULL	TYPE	KEY
QID	NOTNULL	INT	PRIMARYKEY
QN	NULL	VARCHAR(500)	
OPTIONS1	NULL	VARCHAR(100)	
OPTIONS2	NULL	VARCHAR(100)	
ANSWER	NULL	VARCHAR(100)	

Options Based Answers

NAME	NULL/NOTNULL	TYPE	KEY
QID	NOTNULL	INT	FOREIGNKEY
ANSWER	NULL	VARCHAR(10)	

All Student Marks

NAME	NULL/NOTNULL	TYPE	KEY
ID	NULL	INT	
MARKS	NULL	INT	

Exam Schedule

NAME	NULL/NOTNULL	TYPE	KEY
ENAME	NULL	VARCHAR(30)	
EDATE	NULL	DATETIME	

FEASIBILITY STUDY:

Feasibility study is conducted once the problem is clearly understood. Feasibility study is a high level capsule version of the entire system analysis and design process. The objective is to determine quickly at a minimum expense how to solve a problem. The purpose of feasibility is not to solve the problem but to determine if the problem is worth solving.

The system has been tested for feasibility in the following points.

1. Technical Feasibility
2. Economical Feasibility
3. Operational Feasibility.

1. Technical Feasibility

The project entitles "Courier Service System" is technically feasibility because of the below mentioned feature. The project was developed in Java which Graphical User Interface.

It provides the high level of reliability, availability and compatibility. All these make Java an appropriate language for this project. Thus the existing software Java is a powerful language.

2. Economical Feasibility

The computerized system will help in automate the selection leading the profits and details of the organization. With this software, the machine and manpower utilization are expected to go up by 80-90% approximately. The costs incurred of not creating the system are set to be great, because precious time can be wanted by manually.

3. Operational Feasibility

In this project, the management will know the details of each project where he may be presented and the data will be maintained as decentralized and if any inquires for that particular contract can be known as per their requirements and necessities.

Implementation:

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system and in giving confidence on the new system for the users that it will work efficiently and effectively.

The system can be implemented only after thorough testing is done and if it is found to work according to the specification.

It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over and an evaluation of change over methods a part from planning. Two major tasks of preparing the implementation are education and training of the users and testing of the system.

The more complex the system being implemented, the more involved will be the systems analysis and design effort required just for implementation.

The implementation phase comprises of several activities. The required hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued.

TESTING:

The testing phase is an important part of software development. It is the computerized system will help in automate process of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied.

Software testing is carried out in three steps:

1. The first includes unit testing, where in each module is tested to provide its correctness, validity and also determine any missing operations and to verify whether the objectives have been met. Errors are noted down and corrected immediately. Unit testing is the important and major part of the project. So errors are rectified easily in particular module and program clarity is increased. In this project entire system is divided into several modules and is developed individually. So unit testing is conducted to individual modules.
2. The second step includes Integration testing. It need not be the case, the software whose modules when run individually and showing perfect results, will also show perfect results when run as a whole. The individual modules are clipped under this major module and tested again and verified the results. This is due to poor interfacing, which may results in data being lost across an interface. A module can have inadvertent, adverse effect on any other or on the global data structures, causing serious problems
3. The final step involves validation and testing which determines which the software functions as the user expected. Here also some modifications were. In the completion of the project it is satisfied fully by the end user.

Maintenance and environment:

AS the number of computer based systems, grieve libraries of computer software began to expand. In house developed projects produced tones of thousand soft program source statements. Software products purchased from the outside added hundreds of thousands of new statements. A dark cloud appeared on the horizon. All of these programs, all of those source statements-had to be corrected when false were detected, modified as user requirements changed, or adapted to new hardware that was purchased. These activities were collectively called software Maintenance.

The maintenance phase focuses on change that is associated with error correction, adaptations required as the software's environment evolves, and changes due to enhancements brought about by changing customer requirements. Four types of changes are encountered during the maintenance phase.

1. Correction
2. Adaptation
3. Enhancement
4. Prevention

1. Correction:

Even with the best quality assurance activities is lightly that the customer will uncover defects in the software. Corrective maintenance changes the software to correct defects.

Maintenance is a set of software Engineering activities that occur after software has been delivered to the customer and put into operation. Software configuration management is a set of tracking and control activities that began when a software project begins and terminates only when the software is taken out of the operation.

We may define maintenance by describing four activities that are undertaken after a program is released for use:

Corrective Maintenance

Adaptive Maintenance

Perfective Maintenance or Enhancement

Preventive Maintenance or reengineering

Only about 20 percent of all maintenance work are spent "fixing mistakes". The remaining 80 percent are spent adapting existing systems to changes in their external environment, making enhancements requested by users, and reengineering an application for use.

2.ADAPTATION:

Over time, the original environment (E>G., CPU, operating system, business rules, external product characteristics) for which the software was developed is likely to change. Adaptive maintenance results in modification to the software to accommodate change to its external environment.

3.ENHANCEMENT:

As software is used, the customer/user will recognize additional functions that will provide benefit. Perceptive maintenance extends the software beyond its original function requirements.

4.PREVENTION :

Computer software deteriorates due to change, and because of this, preventive maintenance, often called software re engineering, must be conducted to enable the software to serve the needs of its end users. In essence, preventive maintenance makes changes to computer programs so that they can be more easily corrected, adapted, and enhanced. Software configuration management (SCM) is an umbrella activity that is applied throughout the software process.

SOFTWARE METHODOLOGY

The software methodology followed in this project includes the object-oriented methodology and the application system development methodologies. The description of these methodologies is given below.

Application System Development – A Life cycle Approach

Although there are a growing number of applications (such as decision support systems) that should be developed using an experimental process strategy such as prototyping, a significant amount of new development work continue to involve major operational applications of broad scope. The application systems are large highly structured. User task comprehension and developer task proficiency is usually high. These factors suggest a linear or iterative assurance strategy. The most common method for this stage class of problems is a system development life cycle modal in which each stage of development is well defined and has straightforward requirements for deliverables, feedback and sign off. The system development life cycle is described in detail since it continues to be an appropriate methodology for a significant part of new development work.

The basic idea of the system development life cycle is that there is a well-defined process by which an application is conceived and developed and implemented. The life cycle gives structure to a creative process. In order to manage and control the development effort, it is necessary to know what should have been done, what has been done, and what has yet to be accomplished. The phrases in the system development life cycle provide a basis for management and control because they define segments of the flow of work, which can be identified for managerial purposes and specifies the documents or other deliverables to be produced in each phase.

The phases in the life cycle for information system development are described differently by different writers, but the differences are primarily in the amount of necessity and manner of categorization. There is a general agreement on the flow of development steps and the necessity for control procedures at each stage.

The information system development cycle for an application consists of three major stages.

- 1) Definition.
- 2) Design.
- 3) Development.
- 4) Installation and operation.

The first stage of the process, which defines the information requirements for a feasible cost effective system. The requirements are then translated into a physical system of forms, procedures, programs etc., by the system design, computer programming and procedure development. The resulting system is test and put into operation. No system is perfect so there is always a need for maintenance changes. To complete the cycle, there should be a post audit of the system to evaluate how well it performs and how well it meets the cost and performance specifications. The stages of definition, development and installation and operation can therefore be divided into smaller steps or phrases as follows.

1. Definition

Proposed definition : preparation of request for proposed applications.

Feasibility assessment : evaluation of feasibility and cost benefit of proposed system.

Information requirement analysis : determination of information needed.

2. Design

Conceptual design : User-oriented design of application development.

Physical system design : Detailed design of flows and processes in applications processing system and preparation of program specification.

3. Development

Program development : coding and testing of computer programs.

Procedure development : design of procedures and preparation of user instructions.

4. Installation and operation

Conversion: Final system test and conversion.

Operation and maintenance: Month to month operation and maintenance

Post audit: Evaluation of development process, application system and results of use at the completion of each phase, formal approval sign-off is required from the users as well as from the manager of the project development.

4.2 TESTING:

Testing is a process of executing a program with the intent of finding an error. Testing is a crucial element of software quality assurance and presents ultimate review of specification, design and coding.

System Testing is an important phase. Testing represents an interesting anomaly for the software. Thus a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

A good test case is one that has a high probability of finding an as undiscovered error. A successful test is one that uncovers an as undiscovered error.

Testing Objectives:

1. Testing is a process of executing a program with the intent of finding an error
2. A good test case is one that has a probability of finding an as yet undiscovered error
3. A successful test is one that uncovers an undiscovered error

Testing Principles:

1. All tests should be traceable to end user requirements
2. Tests should be planned long before testing begins
3. Testing should begin on a small scale and progress towards testing in large
4. Exhaustive testing is not possible
5. To be most effective testing should be conducted by an independent third party

4.2.1 TEST CASES:

The primary objective for test case design is to derive a set of tests that has the highest likelihood for uncovering defects in software. To accomplish this objective two different categories of test case design techniques are used. They are

White box testing.

Black box testing.

White-box testing:

White box testing focus on the program control structure. Test cases are derived to ensure that all statements in the program have been executed at least once during testing and that all logical conditions have been executed.

Block-box testing:

Black box testing is designed to validate functional requirements without regard to the internal workings of a program. Black box testing mainly focuses on the information domain of the software, deriving test cases by partitioning input and output in a manner that provides through test coverage. Incorrect and missing functions, interface errors, errors in data structures, error in functional logic are the errors falling in this category.

Testing strategies:

A strategy for software testing must accommodate low-level tests that are necessary to verify that all small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

Testing fundamentals:

Testing is a process of executing program with the intent of finding error. A good test case is one that has high probability of finding an undiscovered error. If testing is conducted successfully it uncovers the errors in the software. Testing cannot show the absence of defects, it can only show that software defects present.

Testing Information flow:

Information flow for testing flows the pattern. Two class of input provided to test the process. The software configuration includes a software requirements specification, a design specification and source code.

Test configuration includes test plan and test cases and test tools. Tests are conducted and all the results are evaluated. That is test results are compared with expected results. When erroneous data are uncovered, an error is implied and debugging commences.

Unit testing:

Unit testing is essential for the verification of the code produced during the coding phase and hence the goal is to test the internal logic of the modules. Using the detailed design description as a guide, important paths are tested to uncover errors with in the boundary of the modules. These tests were carried out during the programming stage itself. All units of Vienna SQL were successfully tested.

Integration testing :

Integration testing focuses on unit tested modules and build the program structure that is dictated by the design phase.

System testing:

System testing tests the integration of each module in the system. It also tests to find discrepancies between the system and it's original objective, current specification and system documentation. The primary concern is the compatibility of individual modules. Entire system is working properly or not will be tested here, and specified path ODBC connection will correct or not, and giving output or not are tested here these verifications and validations are done by giving input values to the system and by comparing with expected output. Top-down testing implementing here.

Acceptance Testing:

This testing is done to verify the readiness of the system for the implementation. Acceptance testing begins when the system is complete. Its purpose is to provide the end user with the confidence that the system is ready for use. It involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements.

Tools to special importance during acceptance testing include:

Test coverage Analyzer – records the control paths followed for each test case.

Timing Analyzer – also called a profiler, reports the time spent in various regions of the code are areas to concentrate on to improve system performance.

Coding standards – static analyzers and standard checkers are used to inspect code for deviations from standards and guidelines.

Test Cases:

Test cases are derived to ensure that all statements in the program have been executed at least once during testing and that all logical conditions have been executed.

Using White-Box testing methods, the software engineer can drive test cases that

- Guarantee that logical decisions on their true and false sides.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and with in their operational bounds.
- Exercise internal data structure to assure their validity.

The test case specification for system testing has to be submitted for review before system testing commences.

5.CONCLUSION:

The package was designed in such a way that future modifications can be done easily. The following conclusions can be deduced from the development of the project.

- Automation of the entire system improves the efficiency
- It provides a friendly graphical user interface which proves to be better when compared to the existing system.
- It gives appropriate access to the authorized users depending on their permissions.
- It effectively overcomes the delay in communications.
- Updating of information becomes so easier.
- System security, data security and reliability are the striking features.
- The System has adequate scope for modification in future if it is necessary.

6.FUTURE ENHANCEMENTS:

This application avoids the manual work and the problems concern with it. It is an easy way to obtain The information regarding the different scheduled examinations information that are currently issued.

Well, I and my team members have worked hard in order to present an improved website better than the existing one's regarding the information about the various activities. Still, we found out that project can be done in better way. Particularly when we request information about a particular schedule it just shows the exam date and platform. So, after getting the information we can get access the online exam.

The enhancement that we can add the searching option. We can directly search to the particular student details from the site.

7.BIBLIOGRAPHY:

The following books were referred during the analysis and execution phase of the project

✓ Books Referred:

- BEGINNING PHP 5 ---DAVE MERCER
- BLACK BOOK HTML ---WILEY DREAMTECH
- PHP AND MYSQL WEB DEVELOPMENT --- LUKEWELLING,LAURA
- MICROSOFT SQL SERVER-2000 ---RANKIN, PAUL & JENSEN
- SQL SERVER-2000 ---DUSAN PETKOVIC
- PHP IN A NUTSHELL --- PAUL HUDSON

✓ Websites Referred:

<http://www.phptpoit.com>

