**INTRODUCTION**

**1. INTRODUCTION**

**1.1. OVERVIEW**

In the modern world, technology is everything. And technology is growing faster than anything else. It is the age of information and people need to be updated about the technology used. The project entitled “HACKERZDOM” is an online tutorial site for all the users who would like to learn computer science and get to know about latest technology. There are many sites giving tutorials but each sitehas their own disadvantage. To overcome some of the disadvantages of other tutorial sitesand give a different approach our team has created a new tutorial site for all students, who are both non computer science users who like to know about the technology and for computer science students who would like to learn the subject deeper. People who like to know about computer does not have to search anywhere, they can easily get access to all the resources faster in their fingertips.

**1.2. OBJECTIVE**

Education should always be free of cost. Awareness about the free resources available should be given to all the students. This project provides quality classes and material at a low cost to users of different learning level.

**1.3. SCOPE**

This project is targeted at all the students seeking knowledge. The area of topic is computer science. In the golden age of information, knowledge has been made into a profitable business. This website serves to three kinds of users,

Level 1: For users who would like to learn new technologies and has little to no knowledge on computer science topics.

Level 2: For computer science students.

Level 3: For students who wish to learn deeply about the subject.

**1.4. METHODOLOGY**

The methodology used here is “waterfall model”.

The waterfall model is a software development methodology strategy which relies on following a set sequence of actions from top to bottom in the form of a waterfall. Its stages are closely aligned with the software development life cycle (SDLC) from which it evolved.

This includes the following phases:

* Requirement gathering and analysis
* Design
* Coding
* Testing
* Implementation
* Maintenance

The outcome of one phase acts as the input for the next phase sequentially. This means that any phase in the development process begins only if the previous phase is complete.

**Advantages**

* This model is simple and easy to understand & use.
* It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
* In this model phases are processed and completed one at a time. Phases do not overlap.
* Waterfall model works well for smaller projects where requirements are clearly defined and very well understood.
* This model goes through each of the SDLC phases completely hence also allows a new developer to know better about each of the SDLC phases.
* There are other models that allow flexibility during project development but since the requirements are fixed there is no big need of flexibility.
* Hence this model stands as a better SDLC model to implement our project.

**1.4.1. SDLC**

An SDLC model is a conceptual framework describing all activities in a software development project from planning to maintenance.

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares. The SDLC aims to produce high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

Some of the SDLC models are:

Iterative Model

Spiral Model

Waterfall model

Agile Model

**1.4.2. ITERATIVE MODEL**

This model leads the software development process in iterations. It projects the process of development in cyclic manner repeating every step after every cycle of SDLC process.

In the Iterative model, iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed.

An iterative life cycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which is then reviewed to identify further requirements. This process is then repeated, producing a new version of the software at the end of each iteration of the model.

In this model we have to revisit each phase in the subsequent iterations and rework is to be done in these phases. This model is used for getting early output and useful for large projects. Since these are not very necessary for the project we are doing this model was not chose.

**1.4.3. SPIRAL MODEL**

Spiral model is a combination of iterative development process model and sequential linear development model i.e. waterfall model with very high emphasis on risk analysis. It allows for incremental releases of the product, or incremental refinement through each iteration around the spiral.

This model is usually used when requirements are unclear and complex. It is not suitable for smaller projects and costs a lot. Hence this method was not chosen for this project.

**1.4.4. AGILE MODEL**

By breaking the product into cycles, the [Agile model](http://istqbexamcertification.com/wp-content/uploads/2012/01/Agile-model.jpg) quickly delivers a working product and is considered a very realistic development approach. The model produces ongoing releases, each with small, incremental changes from the previous release. At each iteration, the product is tested.

This method is also not suitable for this project as it is not normally used by first time developers. It is usually used by experienced senior developers. Final result keeps on changing as per requirements, but the requirements are fixed in this project.

**1.4.5. PROTOTYPE MODEL**

Prototype methodology is defined as a Software Development model in which a prototype is built, tested, and then reworked when needed until an acceptable prototype is achieved. It also creates a base to produce the final system. It is an iterative, trial, and error method which take place between the developer and the client. This model allows the user to interact and try out with a working model.

The prototype model is applied when detailed information related to input and output requirements of the system is not available. It is usually used when a system does not exist or when a system is large and complex. However our project already knows the requirements and is fixed.

**1.5 MISSION OF PROJECT**

To provide a tutorial website which can be used to learn a topic based on a student’s learning level. It will be a platform where anybody with qualification can post tutorials which will be reviewed by a pool of instructors.

**1.6 COMPANY PROFILE**

**2. REQUIREMENT GATHERING AND SYSTEM ANALYSIS**

Every software project is initiated with a thorough system analysis. System analysis is a general term that refers to a structured process for identifying and solving problems. Analysis implies the process of breaking something down into parts so that the whole may be understood. The definition of system analysis includes not only the process of analysis; it is the application of the systems approach to problem solving using computers. The ingredients are system elements, processes, and technology. This means that to do systems work, one needs to understand the systems work, one needs to understand the systems concept and how organisations operate as a system, and then design appropriate computer-based systems that will meet an organisation’s requirements. It is actually a customised approach to the use of the computer science for problem solving.

In other words Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system.

System is a way of thinking about organisations and their problems. It also involves a set of techniques that helps in solving problems.

The Analyst who carries out the system analysis must know what information to gather, where to find it, how to collect it, and what to make of it. The proper use of tools for gathering information is the key to successful analysis. The tools used are the traditional

1. Interview
2. Questionnaire
3. On-site observations
4. Visiting similar sites
5. Literature review

**Interview**

Interview is a face to face interpersonal role situation in which a person called the interviewer asks a person being interviewed questions designed to gather information about a problem area. This is the oldest and most often used device for gathering information in systems work. It can be used for two main purposes

* as an exploratory device to identify relations or verify information and
* to capture information as it exists.

The method chose for this project is interview because we need the information from users on what all features can be included in a tutorial website, what all information is unnecessary and what all is lacking in the currently available tutorial websites.

The people interviewed for this project are students who have visited these websites, content developers who have probably worked on similar projects and teachers.

**Questionnaire**

The method of questionnaire was excluded because the amount of people available to us who have used tutorial websites were less and an interview with them was possible (questionnaire is used when number of people from whom the data is to be collected is large). Also validity of questionnaire is less compared to interview.

**On-site observation**

We are not including the method of on-site observation as it difficult to go and observe users using such websites.

**Visiting similar sites**

We also studied about features required for a tutorial website by visiting other similar websites and also by analysing the reviews about few websites by its users. The websites whose reviews by users are taken into account for this project include:

* Byju's learning app
* Unacademy's website (www. unacademy.com)
* W3Schools (www.w3schools.com)
* Khan Academy ([www.khanacademy.org](http://www.khanacademy.org))
* Cybery (app.cybrary.it)
* Guru99 ([www.guru99.com](http://www.guru99.com))
* Code Academy (ww.codecademy.com)
* Tutorials point ([www.tutorialspoint.com](http://www.tutorialspoint.com)).

**Literature review**

A literature review surveys books, scholarly articles, and any other sources relevant to a particular issue, area of research, or theory, and by doing so, provides a description, summary, and critical evaluation of these works in relation to the research problem being investigated. However there is no literature easily available on tutorial website building.

**2.1. EXISTING SYSTEM**

There are a number of tutorial websites available at present

1. The current systems include tutorial sites with huge payment.
2. Some sites uses complicated language, difficult accent.
3. Quality of videos are controlled automatically.
4. Provides demo videos for users.
5. Some sites allow users to provide video classes to admin for money.
6. Current systems provide interactive interface, provide tests.

**2.2 DRAWBACKS OF EXISTING SYSTEM**

1. Most systems do not take feedback.
2. Some sites do not allow users to control video quality, playback.
3. Poor quality of content, not user friendly.
4. Some websites use lot of data.
5. Classes in native language is not available
6. Most websites are not compatible with certain browsers and interface is same in different devices.
7. Some sites provide either only content or only videos.

**2.3. SOFTWARE REQUIREMENT SPECIFICATION**

Users of internet who are trying to learn subjects through tutorial websites are looking for sites that are simple and explain things to them easily and properly. Some of the key features users are interested in or looking for are:

1. Less data consuming: most sites require a high speed internet and big data plans as they use up lot of internet data of the user.
2. Video quality and playback options: many websites and apps like Byju's learning app do not allow the user to control video quality and playback . This usually result in videos loading at poor quality as the video quality changes frequently based on the internet speeds.
3. Search bar should be provided to search the contents of the website.
4. Should not be overcrowded with advertisements.
5. Provide switches to go from bottom to top, links to other websites to read more and links to give info about keywords.
6. Provide options to convert language, provide feedback, post complaints, ask questions and view most-viewed or most-attended topics/courses.
7. Provide facilities for online tests.
8. Provide attractive interface to users (use flip, dark mode etc.).
9. Create website that is compatible with most devices and browsers.
10. Should load faster, should be able to make notes or video offline.
11. Admin should be able to add videos and content, view users, view videos and content accepted by inspector and accept payments and view payment details.
12. Anybody with qualification can contribute contents to admin.
13. If content developers with qualification are ready to provide quality content , there should be provision for them to register and provide content.
14. Inspector should be able to view contents provided by admin and make sure it is relevant and if necessary should be able to add to website and else should be able to remove those.
15. Students should be able to view courses (and demo classes) see their progress, attend tests and request for contents not available in this site. To create interest in students badges can be made given based on their progress.
16. Log sessions of course for student: It is very useful to the student if whatever the student has learned until then are recorded and shown to the user in his dashboard for this, each time the user logs in and learns something it should be recorded and progress of a user in his course should be shown.
17. Login easily using Google or Facebook.
18. Provide details on what all you can do in this site (welcome note/guide).
19. Live projects -we can work on sample projects, materials available in pdf, inbuilt text editors (we can use online editors without downloading software).

**2.4 PROPOSED SYSTEM**

|  |
| --- |
| **Requirements being implemented in this tutorial site** |

1. Video quality and playback options: users are able to change the quality of videos as per their convenience.
2. Search bar to search for topics and contents in the website.
3. Three levels: Tutorials of this website serves to three kinds of users
   * Level-1: For users who would like to know about technology and has little to no knowledge on computer science topics.
   * Level-2: For computer science students.
   * Level-3: For students who would like to learn deeply about the subject.
4. Provides no advertisements.
5. Provide switches to go from bottom to top, links to other websites to read more and links to give info about keywords.
6. Provides feedback and comments to ask doubt.
7. Provide facilities for online tests, provide badges.
8. Provide attractive interface to users (use flip, dark mode etc.)
9. Admin should be able to add videos and content, view users, view videos & content accepted by inspector and accept payments and view payment details.
10. Inspector should be able to view contents provided by admin and make sure it is relevant and if necessary should be able to add to website and else should be able to remove those.
11. Students should be able to view courses (and demo classes) see their progress, attend tests and request for contents not available in this site. To create interest in students badges can be made given based on their progress.
12. Log sessions of course for student: It is very useful to the student if whatever the student has learned until then are recorded and shown to the user in his dashboard for this, each time the user logs in and learns something it should be recorded and progress of a user in his course should be shown.
13. Provide details on what all users can do in this site (welcome note/guide).
14. Anybody can contribute contents to admin.

**Advantage of proposed system**

1. Anybody with qualification can contribute contents to admin.
2. Students can also earn money by making content.
3. Compatible with most devices and browsers.
4. Less advertisement in the site and can clear doubts.
5. Video quality and playback options for users.
6. Easy, efficient and less time consuming.
7. High speed response to users.
8. Tutorials of this website serves to three kinds of users so they can learn based on their learning level.

**2.5 Feasibility study**

A feasibility study is an assessment of the practicality of a proposed project or system. A feasibility study aims to objectively and rationally uncover the strengths and weaknesses of an existing business or proposed venture

**Economic feasibility**

Economic feasibility also known as cost/benefit analysis is a procedure to determine the benefits and savings that are expected from a candidate system and compare them with cost. If benefits outweigh costs, then the decision is made to design and implement the system.

The cost of the resources needed for this project are less and the softwares used for developing this tutorial site are available free. Only cost would be to host the site, maintenance is also less. Thus development of this website is economically feasible.

Technical feasibility

It involves determining whether or not a system can actually constructed to solve the problem at hand. If there is an existing system it centres around the existing system and to what extend it can support the proposed system.

The proposed system is technically feasible. Since the necessary technology exist and the required resources for the development and maintenance is easily available as well.

**Behavioral feasibility**

People are inherently resistant to change, and computers have been known to facilitate change. An estimate should be made on how strong a reaction the user is likely to have to the system.

In this project there is minimum behavioural feasibility as the users are already very well aware of tutorial sites and normally skilled enough to use a computer properly.

**Legal feasibility**

Legal feasibility is the study to know if the proposed project conform the legal and ethical requirements. Content we distribute should be our own, it should not be copied from any other source, if done we should reference it .

This project abides by the software license of all softwares used for development of the project. The contents used also follows copyright rules if any. This is not a work copied from any other source.

**2.6. COST ESTIMATION AND SCHEDULING**

Software cost estimation is the process of predicting the effort required to develop a software system.

**2.7. SYSTEM SPECIFICATIONS**

It includes two phases: Hardware requirements & Software requirements

HARDWARE REQUIREMENTS

Input device : mouse, keyboard

Output device : monitor

Memory : 2 gb ram(minimum)

Processor: Intel pentium (minimum)

SOFTWARE REQUIREMNTS

Operating system: windows 7 , 8 or 10

Frontend: PHP with Bootstrap framework

Backend: MYSQL

**2.8. SOFTWARE ENVIRONMENT**

2.7.1 PHP

PHP (recursive acronym for PHP: Hypertext pre-processor) is a widely used open source general purpose scripting language that is especially suited for web development and can be embedded into HTML.

What distinguishes PHP from some other client side JAVASCRIPT is that the code executed on the server, generates HTML which is send to the client, the client would receive the results obtained after running that script but would not know what the underlying code was. The best uses of PHP is that it is extremely simple for newcomers but offers many advanced features for professional programmers as well. Although PHP’s development is focused on server side scripting, you can do much with it.

2.7.2 XAMP

XAMP server is a web development platform on windows that allows you to create dynamic web development platform on windows that allows you to create dynamic web application with apache server and MYSQL database by automatically installing everything you need to intuitively develop web application.

2.7.3SUBLIME TEXT

Sublime text is a superfast and feature packed text and development editor. If you are going to be coding regularly you want to try this amazing editor (IDE). Following some of the great features that make Sublime Text stand out from other code editors:

Multiple cursors: Once you have discovered multiple cursors you won’t want to work without them anymore. As the name suggests they let you write or edit in multiple places in a document at the same time.

Vintage mode: vim keyboard shortcuts will work just like in the original vim editor. To use them, all you need to do is to enable vintage mode.

Lightning fast: This is a fast and lightweight code editor.

Command pallet: A great feature that allows you to reach about all functions of the editor via the keyboard. You will hardly use your mouse and thus code more efficiently.

Package control: This add-on let you install plug-in within directly from the editor.

**STAR UML**

StarUML is an open source software modeling tool that supports the UML (Unified Modeling Language) framework for system and software modeling.ThisUML diagramming tool by MKLab was licensed under a modified version of [GNU](https://en.wikipedia.org/wiki/GNU_GPL) GPL.

StarUML is built as a modular and open tool. It provides frameworks for extending the functionality of the tool.

StarUML supports the following diagram types

* Use Case Diagram
* Class Diagram
* Sequence Diagram
* Collaboration Diagram
* State chart Diagram
* Activity Diagram
* Component Diagram
* Deployment Diagram
* Composite Structure Diagram

It also provides features to generate code from class diagrams.

The class diagram, use case diagram and ER diagram of this project where made using STAR UML, subsequent code generation was also done.

**LUCIDCHART**

Lucidchart is a web-based proprietary platform that is used to allow users to collaborate on drawing, revising and sharing charts and diagrams. Lucidchart runs on browsers that support HTML5. This means it does not require updates of third party software like Adobe Flash. It is basically an online diagramming and drawing software for all sorts of diagram , though not free it can be used to an extend with its trial version. The DFD of this project was drawn using lucid chart.

3. SYSTEM DESIGN

3.1INTRODUCTION

System design transforms a logical representation of what a given system is required to do into physical specifications. The specifications are converted into physical reality during development. The design forms a blue print of the system and how the components relate to each other. The design phase proceeds according to an orderly sequence of steps beginning with review and assignment of task and ending with package design. During the first stage, output design, and analyst determines what data the application produces and how to organize and present that data.

System design is the process of planning of system or to replace or to complement an existing system but before this planning should be done. It must be thoroughly understood about the old system and determine how computers can make its operations more effective. The importance of system design can with a single word and that is quality. design is the phase of where quality is fostered in the software development. Design is the only way to transforms a customer’s requirement into software development and software support steps that follow. Without design there is a risk of building an unstable system that will fail when small changes are made. The most creative and challenging phase of software life cycle is system design. The term design describes a final system and a process by which it is developed. The design may define as the “The process of applying various techniques and principles for the purpose of defining a device, a process or a system with sufficient details to permit its physical realization”.

The designer’s goal is how the output is to be produced and in what format. Samples of output and input also presented. Second input files and database files have to be designed to meet the requirements of proposed output. The processing phases are handled through the program construction and testing. Finally details related to justification of the system and an estimate of the impact of the candidate system on the user and the organization are documented and evaluated by management as a step towards implementation.

3.2 INPUT DESIGN

Input design is the process of converting the user originated input into a computer based format. The design for handling input specifies hoe the data are accepted for computer processing. Input design is a part of overall system design that needs careful attention and if includes Specifying the means by which actions are taken. A system user interacting through a workstation must be able to tell the system whether to accept input produce a report or end processing. The collection of input data is considered to be the most expensive part of the system design. Since the input has to be planned in such a manner as to get relevant information taken extreme care is taken to be obtained from the information. If data going in to the system is then the processing and output will be magnify these errors.

Input design is the process of converting user oriented description of the inputs to a computer based business system into a programmer oriented specification. Inaccurate input based system into a programmer oriented specification. Inaccurate input is the most common cause of data processing error. If the input design is poor, particularly where operators must enter data from source permits had data to enter a computer system. The main objective of the system is to specify how the information is put into a form that is acceptable to the computer. The system also needs to include appropriate message which ensure that the user can understood the context. The input data is a validated to minimize the errors in the data entry. User is never left in a state of confusion as to what is happened.

3.3 OUTPUT DESIGN

The output design is an ongoing activity almost from the beginning of the project and follows the principles of forms designing. Compute output is the most important and direct source of information to the user. Efficient, intelligible, and well defined output design improves the relationship of the system and the user. The objective of the output design is to define is the format of all printed documents and reports screens that will be produce by the system. Computer output is the most important and direct source of information to the user.

Output design phase of the system is concerned with the convergence of information to the end user friendly manner. The output design should be efficient, intelligible so that system relationship with the end user is improved and thereby enhancing the process of decision making.

3.4 DETAILED SYSTEM DESIGN

Detailed system design is data flow based methodology. The approach begins with the system specification that identifies input and output aspects of the system. The system specification is used as a basis for the graphic representation of the data flow and process. This design phase partitions a program into small independent modules. They are arranged in hierarchy form. It is an attempt to minimize the complexity and make a problem manageable by subdividing in to smaller segments.

USECASE DIAGRAM

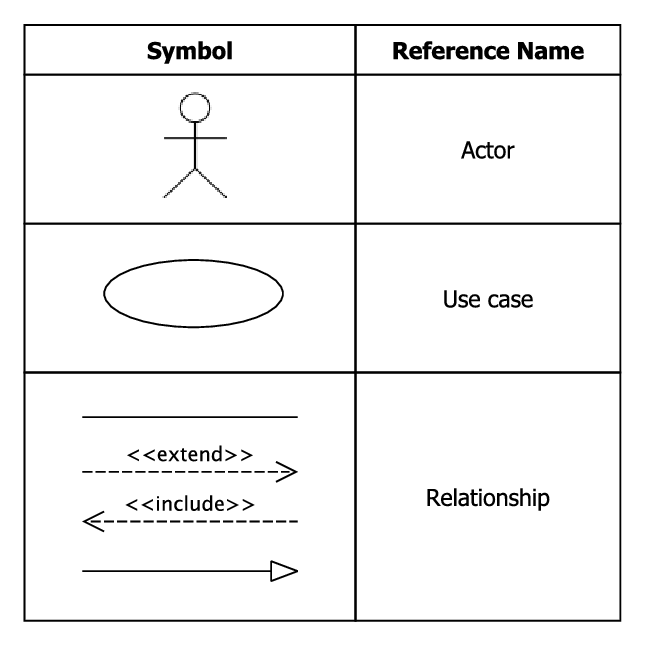
The purpose of a use case diagram in UML is to demonstrate the different ways that a user might interact with a system.

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

* Scenarios in which your system or application interacts with people, organizations, or external systems
* Goals that your system or application helps those entities (known as actors) achieve
* The scope of your system

**symbols and notation**

* **Use cases:** Horizontally shaped ovals that represent the different uses that a user might have.
* **Actors:** Stick figures that represent the people actually employing the use cases.
* **Packages:** A UML shape that allows you to put different elements into groups. Just as with component diagrams, these groupings are represented as file folders.
* **Associations:** A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.
* **Extend:** It is used when a use case conditionally adds steps to another first class use case.
* **Include:** It is used to extract use case fragments that are duplicated in multiple use cases. The included use case cannot stand alone and the original use case is not complete without the included one.



**3.4.1 DATA FLOW DIAGRAM (DFD)**

A data-flow diagram is a way of representing a flow of a data of a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops.

The dataflow diagram is used for classifying system requirements to major transactions that will become programs in system design. This is the starting point of design phase that functionally decompose the required specification down to the lower level of details.

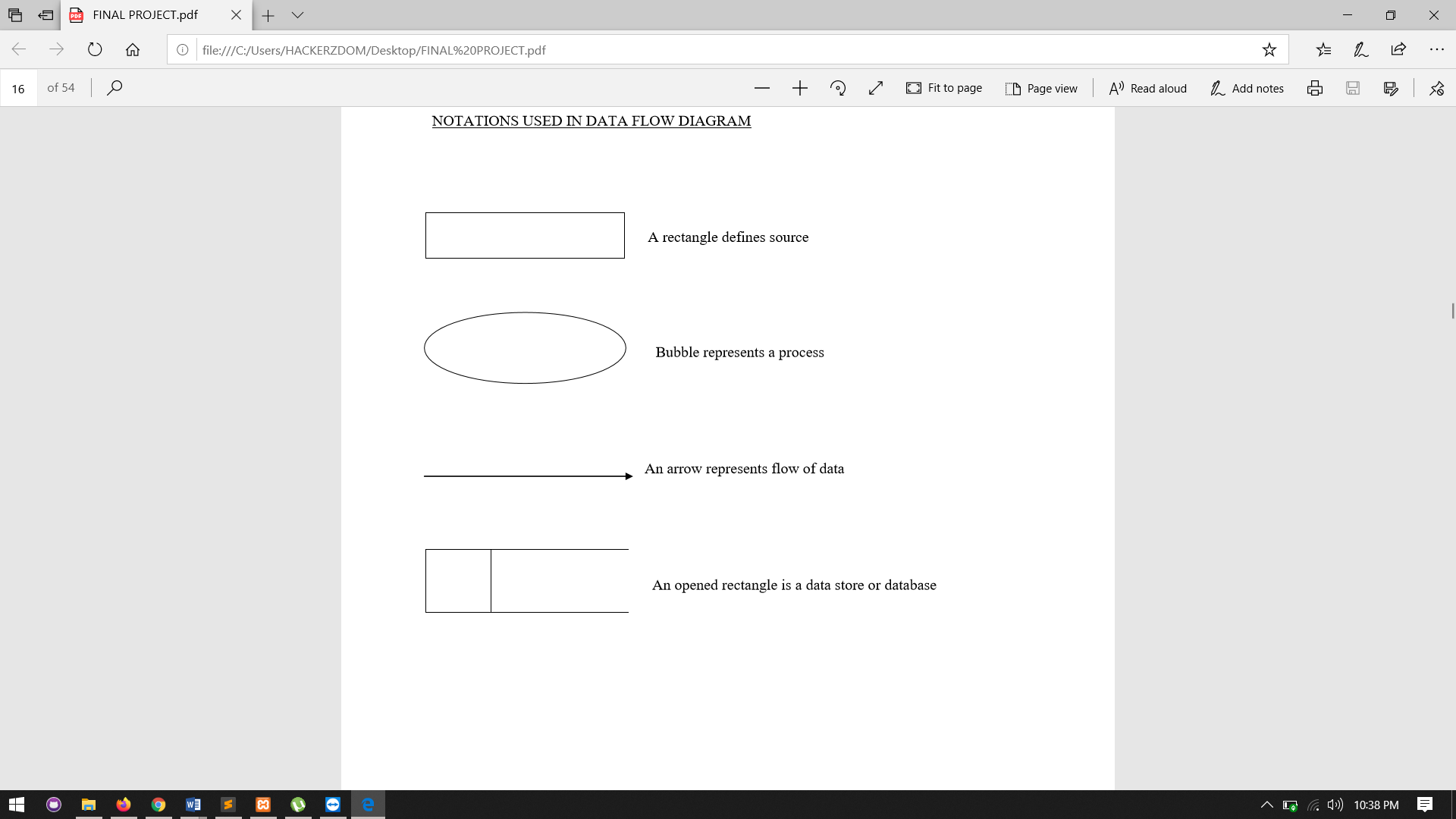
Symbols and notations used:

* **Entity:** an outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system or a business system. They are also known as terminators, sources and sinks or actors. They are typically drawn on the edges of the diagram.
* **Process:**any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules. A short label is used to describe the process, such as “Submit payment.”
* **Data store:** files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label, such as “Orders.”
* **Data flow:** the route that data takes between the external entities, processes and data stores. It portrays the interface between the other components and is shown with arrows, typically labeled with a short data name, like “Billing details.”

A rectangle defines Entity

Circle/Bubble represents a process

An arrow represents flow of data



An opened rectangle is a data store or database

The DFD of this Project is given below:

3.5 ENTITY RELATIONSHIP DIAGRAM

The ER diagram is diagrams that depict a set of real world entities and the logical relationship among them. An ER diagram comprises data objects and entities data attributes relationship, cardinality and modality. An entity is a data object that stores information about the system in the database. Data attributes describes the properties of a data object.

The components and features of an ER diagram

### Entity: A definable thing—such as a person, object, concept or event—that can have data stored about it. It could represent a single table or a group of tables in the database.

**Entity keys:** Refers to an attribute that uniquely defines an entity in an entity set.

### Relationship:

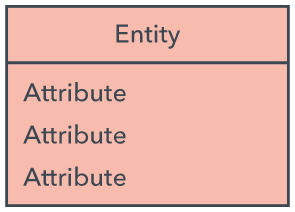
How entities act upon each other or are associated with each other. The link or connection between data objects or entities with each other is known as relationship.

Cardinality defines the numerical attributes of the relationship between two entities or entity sets. The three main cardinal relationships are one-to-one, one-to-many, and many-many.

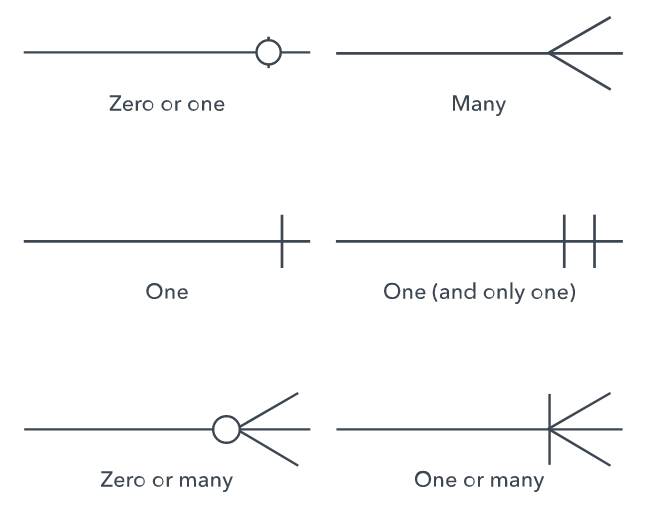
There are several notation systems, which are similar but vary in a few specifics.

### The main styles are Chen notation style and Crow’s Foot/Martin/Information Engineering style

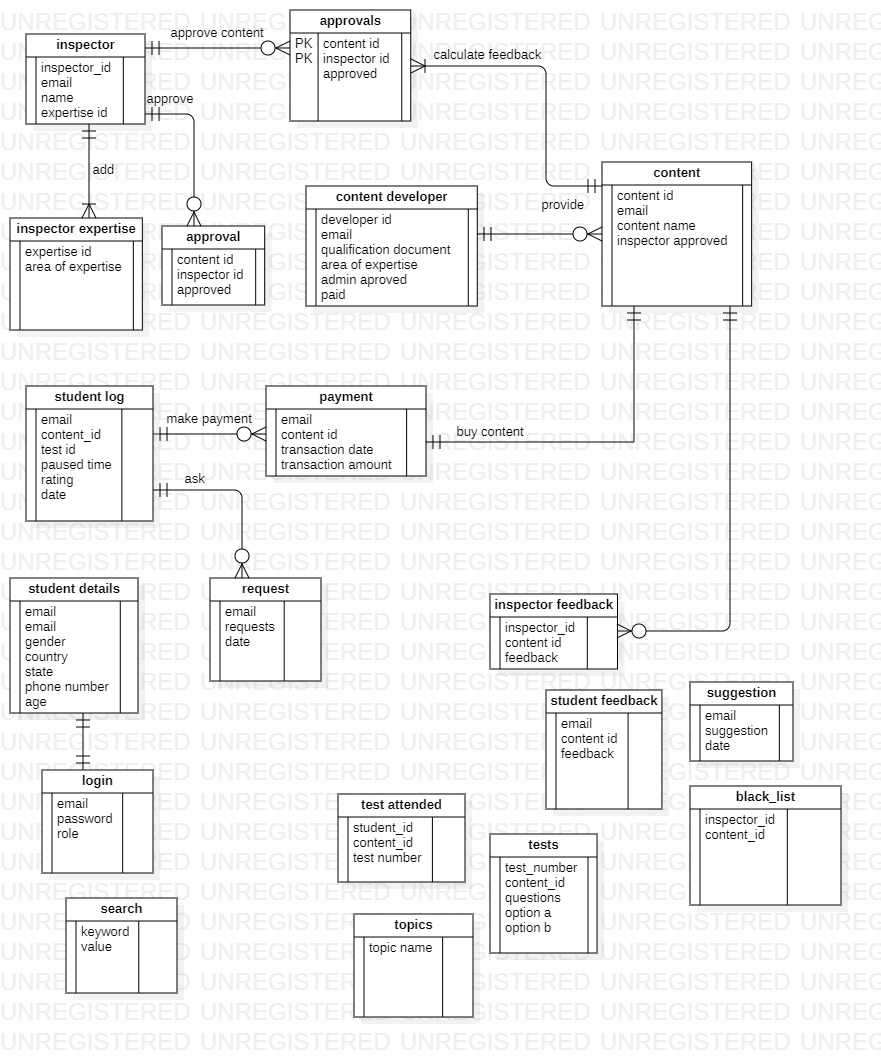
Notations used are:



**Cardinality**



The ER Diagram of this project is given below:



Class Diagram

The [Unified Modeling Language](https://www.lucidchart.com/pages/what-is-UML-unified-modeling-language) (UML) can help you model systems in various ways. One of the more popular types in UML is the class diagram. Popular among software engineers to document software architecture, class diagrams are a type of structure diagram because they describe what must be present in the system being modelled.

UML was set up as a standardized model to describe an object-oriented programming approach. Since classes are the building block of objects, class diagrams are the building blocks of UML. The various components in a class diagram can represent the classes that will actually be programmed, the main objects, or the interactions between classes and objects.

Class diagrams offer a number of benefits for any organization. Use UML class diagrams to:

* Illustrate data models for information systems, no matter how simple or complex.
* Better understand the general overview of the schematics of an application.
* Visually express any specific needs of a system and disseminate that information throughout the business.
* Create detailed charts that highlight any specific code needed to be programmed and implemented to the described structure.
* Provide an implementation-independent description of types used in a system that are later passed between its components.

## Basic components of a class diagram:

* **Upper section:**Contains the name of the class. This section is always required, whether you are talking about the classifier or an object.
* **Middle section:**Contains the attributes of the class. Use this section to describe the qualities of the class. This is only required when describing a specific instance of a class.
* **Bottom section:**Includes class operations (methods). Displayed in list format, each operation takes up its own line. The operations describe how a class interacts with data.

### Member access modifiers:

All classes have different access levels depending on the access modifier (visibility). Here are the access levels with their corresponding symbols:

* Public (+)
* Private (-)
* Protected (#)
* Package (~)
* Derived (/)
* Static (underlined)

### Additional class diagram components

**Classes:** A template for creating objects and implementing behavior in a system. In UML, a class represents an object or a set of objects that share a common structure and behavior. They're represented by a rectangle that includes rows of the class name, its attributes, and its operations. When you draw a class in a class diagram, you're only required to fill out the top row—the others are optional if you'd like to provide more detail.

**Name:** The first row in a class shape.

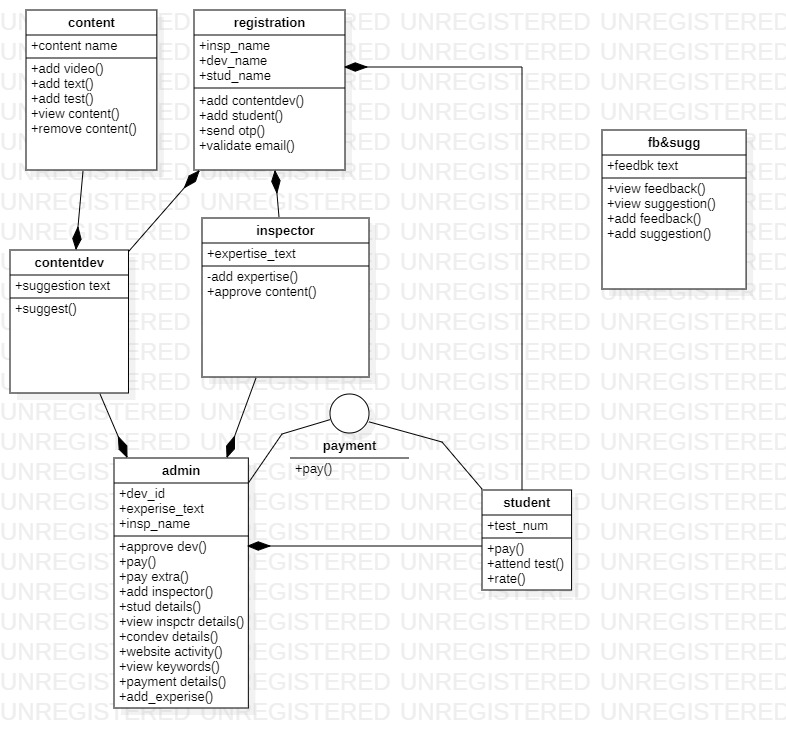
**Attributes:** The second row in a class shape. Each attribute of the class is displayed on a separate line.

**Methods:** The third row in a class shape. Also known as operations, methods are displayed in list format with each operation on its own line.

### Interactions

Interactions like inheritance can also be shown in this diagram

The class diagram of this project is given below:



3.6 DATABASE DESIGN

Database design is the process of producing a detailed data model of a database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity. Database design is one of the important parts in developing software. It is a process of conceptual design of data. It minimizes the need of using separate files. It is definition of entire information content of organization and it specifies the relationship between the data. A relational database consist of collection of tables having attributes, are assigned a unique name. A row in the table represents a relationship among set of values. The database level is made up of two levels.

3.6.1 CONCEPTUAL MODEL

The conceptual level represents the major data and relationship between them. Conceptual level describes the essential features of system data. It uses symbols from a modelling called entity relationship analysis. It is an early phase of the design process, in which the broad outlines of function and form of something are articulated. It includes the design of interactions, experiences, processes and strategies. It involves an understanding of people's needs and how to meet them with products, services & processes. Common artefacts of conceptual design are concept sketches and models.

3.6.2 NORMALIZATION

The normalization of data refers to the way data item are grouped into record structures. Normalization is used to overcome drawbacks likes repetition of data (redundancy), loss of information and inconsistency. In other words neutralization is a technique of separating redundant fields and breaking up large table into smaller one. In our design all tables have been normalized up to the third normal form. The different normal forms applied during the database design are given below:

3.6.3 FIRST NORMAL FORM (1NF)

A relation is said to be in 1NF if it satisfies the constraints :

* Each column contain atomic values.
* In each column value stored should be of same datatype.
* Each column should have unique name.

3.6.4 SECOND NORMAL FORM (2NF)

A relation is said to be in 1NF if it satisfies the constraints:

* Should be in 1NF.
* Should not have any partial dependencies.

3.6.5 THIRD NORMAL FORM

A relation is said to be in 1NF if it satisfies the constraints:

* Should be in 2NF.
* Should not have transitive dependency.

3.6.6. BOYCE-CODD NORMAL FORM (3.5NF)

It is called 3.5NF or BCNF. A relation is said to be in 1NF if it satisfies the constraints:

* Should be in 3NF.
* For any dependency A => B , A should be a super key

3.6.7. FOURTH NORMAL FORM (4NF)

A relation is said to be in 1NF if it satisfies the constraints:

* Should satisfy BCNF
* No multivalued dependency should be there. ( A=>B is multivalued dependency if a single value of A has more than one value of B in a table [for multivalued dependency there should be min. 3 columns] then B and other columns should not depend on each other.

3.6.6 CONTROL REDUNDANCY

Redundant occupies space and therefore, is wasteful. If versions of the data are in different phases of updating the system often gives conflicting information. A unique aspect of database design is storing only ones, which controls redundancy and improves system performance.

3.6.7 ENTITY RELATIONSHIP MODEL

The Entity Relationship data model is based on a perception of a real world, among these objects. An entity is an object they exist and is distinguishable from other objects/entities is an object as a concept meaningful to the organization. An entity set is a set of entities of the same type. A primary key is an attribute which when taken, allow us to identify uniquely an entity in the entity set.

3.7 DESIGN PROCESS

3.7.1 TABLE DESIGN

|  |  |  |  |
| --- | --- | --- | --- |
| Login | | |  |
| column name | data type | constraint |  |
| login\_id | integer | primary key |  |
| email | varchar(18) |  |  |
| password | varchar(20) |  |  |
| role | Varchar(20 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| student details | | | | |
| column name | data type | | constraint | |
| login\_id | integer | |  | |
| student\_id | integer | | primary key | |
| name | varchar(15) | |  | |
| gender | varchar(10) | |  | |
| country | varchar(10) | |  | |
| state | varchar(10) | |  | |
| phone number | integer | |  | |
| age | integer | |  | |
| last login | date | |  | |
|  |  | |  | |
| **content** | | | | |
| column name | | data type | | constraint |
| content\_id | | integer | | primary key |
| content name | | varchar(18) | |  |
| inspector approved | | boolean | |  |
| content depth | | integer | |  |
| people viewed | | integer | |  |
| developer id | | integer | |  |
| content length | | integer | |  |
| paid | | boolean | |  |

|  |  |  |
| --- | --- | --- |
| **content developer** | | |
| column name | data type | constraint |
| developer id | integer | primary key |
| name | varchar(18) |  |
| quali\_doc | varchar(18) |  |
| expertise | varchar(10) |  |
| admin\_aproved | boolean |  |
| login\_id | integer |  |

|  |  |  |
| --- | --- | --- |
| **inspector** | | |
| column name | data type | constraint |
| inspector\_id | integer | primary key |
| email | varchar(18) |  |
| name | varchar(18) |  |
| login\_id | integer |  |

|  |  |  |
| --- | --- | --- |
| **inspector expertise** | | |
| column name | data type | constraint |
| expertise id | integer | primary key |
| inspector\_id | integer |  |
| expertise | varchar(10) |  |

|  |  |  |
| --- | --- | --- |
| **search** | | |
| column name | data type | constraint |
| keyword | varchar(15) | primary key |
| value | integer |  |

|  |  |  |
| --- | --- | --- |
| **approvals** | | |
| column name | data type | constraint |
| content id | integer | composite key |
| inspector id | integer |
| approved | boolean |  |

|  |  |  |
| --- | --- | --- |
| **inspector feedback** | | |
| column name | data type | constraint |
| inspector\_id | integer | composite key |
| content id | integer |
| feedback | varchar(100) |  |

|  |  |  |
| --- | --- | --- |
| **student log** | | |
| column name | data type | constraint |
| student\_id | integer | composite key |
| content\_id | integer |
| paused time | time |  |
| rating | integer |  |
| date(paid for course on) | date |  |
| watch time | time |  |

|  |  |  |
| --- | --- | --- |
| **request** | | |
| column name | data type | constraint |
| student\_id | integer | primary key |
| requests | varchar(100) |  |
| date | integer |  |

|  |  |  |
| --- | --- | --- |
| **tests** | | |
| test number | varchar(6) | composite key |
| content id | integer |
| question\_num | integer |
| question | varchar(100) |  |
| option a | varchar(50) |  |
| option b | varchar(50) |  |
| option c | varchar(50) |  |
| option d | varchar(50) |  |
| answer option | char(1) |  |
| dificulty level | integer |  |

|  |  |  |
| --- | --- | --- |
| **payment** | | |
| column name | data type | constraint |
| trans\_num | varchar(18) | primary key |
| email | varchar(18) |  |
| content id | integer |  |
| transaction date | varchar(10) |  |
| transaction amount | integer |  |

|  |  |  |
| --- | --- | --- |
| **content keyword** | | |
| column name | data type | constraint |
| keyword id | integer | primary key |
| content id | integer |  |
| keyword | varchar(15) |  |

|  |  |  |
| --- | --- | --- |
| **blacklist** | | |
| column name | data type | constraint |
| inspector id | integer | primary key |
| content id | integer |  |

|  |  |  |
| --- | --- | --- |
| **test attended** | | |
| column name | data type | constraint |
| student\_id | integer | composite key |
| content id | integer |
| test number | integer |  |
| badges | small int |  |

|  |  |  |
| --- | --- | --- |
| **suggestion** | | |
| column name | data type | constraint |
| suggestion id |  |  |
| developer\_id | integer | primary key |
| suggestion | varchar(100) |  |
| date | integer |  |

|  |  |  |
| --- | --- | --- |
| **stud\_feedback** | | |
| column name | data type | constraint |
| student\_id | integer | composite primary key |
| content id | integer |
| feedback | varchar(100) |  |

|  |  |  |
| --- | --- | --- |
| **topics** | | |
| column name | data type | constraint |
| topic name | varchar(30) | primary key |

4. SYSTEM CODING

4.1 INTRODUCTION

The goal of coding is to translate the design of the system protected during the design phase into code in a programming language, which can be executed by a computer. When considered as a step in software engineering process. The coding translate a detailed design representation of software into a programming language realization which can be executed by the computer and which computation specified by the design. The quality of source code can be improved by the use of structured coding techniques, good coding style and readable, consistent source code.

The goal of coding phase is to produce simple and clear programs. It should be constructed in a way that are easy to read and understand. Maintenance phase in any software development takes lot of time.

4.2 CODING DESIGN

The design must be translate into a machine readable form. The coding steps perform the task. If the design is performed inn detailed manner, coding can be accomplished mechanically. The goal of coding phase is to be translating the design of the systeminto code in given programming language.

4.3 PROGRAMMING PARADIGM

All design contains hierarchies as crating a hierarchy is a natural way to manage complexity. Most design methodology for software also produces hierarchy. The question at coding time is given the hierarchy of modules being built starting from the top level or staring from bottom level.

The programming paradigm used here is OOP.

5. TESTING AND IMPLEMENTATION

5.1 INTRODUCTION

Testing is a set of activity that can be planned in advance and conducted. Systematically, this is aimed at ensuring that the systems work accurately and efficiently before live operation commences.

Testing is the process of correcting a program with intend of finding an error.

A good test case is one that has high probability of finding a yet undiscovered error.

A successful test is one that uncovers a yet undiscovered error.

5.2 TESTING OBJECTIVES

There are several rules that can serve as testing objectives

Testing is a process of executing a program with the intent of finding an error.

A good test case is one that has high probability of finding an undiscovered error. Testing is vital to the success of the system to the success of the system. System testing makes a logical assumption that if all parts of the system are subject to variety of tests online response, volume, stress, recovery and security and usability tests. A series of tests are performed before the system is ready for user acceptance testing.

5.3 TESTING AND STRATERGIES

White Box Testing

Black Box Testing

5.3.1 WHITE BOX TESTING

White box testing is also known as code testing. The code checking strategy checks for the correctness of the every statement in the program. To follow this strategy, there should be cases that result in execution of every instruction in the program or module, which is every path in the program, is tested. The test cases should be guaranteed that independent paths within modules are executed once.

Exercise all logical decision on their true or false sides.

Execute all loops at their boundaries and within their operational bounds.

This testing strategy, on the face of it, sounds exhaustive. If every statement in the program is checked for its validity, there does not seem to be much scope of error.

5.3.2 BLACK BOX TESTING

Black box testing is also known as specification testing. To perform black box testing, the analyst examine the result, the analyst can examine specification taking what the program or module should do and how it should perform on the various condition and submitted for processing . By examine the result, the analyst can examine whether the program performs according to the specified requirements.

5.4 TYPES OF TESTING

Different types of testing are,

Unit Testing

Integration Testing

Validation Testing

Output Testing

5.4.1 UNIT TESTING

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 module of the system is tested separately. The testing is carried out during programming stage itself. In this testing step each module is found to be 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 to debug the system.

5.4.2 INTEGRATION TESTING

Data can be lost across an interface; one module can have an adverse effect 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 successful for this sample data. The need for integrated test is to find the overall system performance.

5.4.3 VALIDATION TESTING

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

The function of performance characteristics confirm to specification and are accepted.

A deviation from specification is uncovered and a deficiency list is created.

5.4.4 OUTPUT TESTING

After performing the validation testing the next test is output testing of the proposed system since no system could be useful if it does not produce the required data in the specific format. The output displayed or generated by the system under consideration is tested by, asking the user about the format displayed. The output format on the screen is found to be correct as the format was designed in the system is according to the user needs. Hence the output is testing doesn‟t any correction in the system.

5.4.5 USER ACCEPTANCE TESTING

User acceptance of the system is the key factor for the success of a system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing and making change wherever required. This is done with regard to the following words.

Output screen design

Input screen designed

Menu driven system

5.5 IMPLEMENTATION

Implementation is the stag of the project when theoretical design is turned into a working system. Most crucial stage is achieving a successful system and confidence that the new system will be work effectively. It involves careful planning investigation of the manual system and to new system. Implementation means converting a new or revised system design into an operational one. The implementation includes all though activity that takes place to convert from old system to new one. There are several activities involves while implementing a project:

Careful planning

Investigating the current system and its constraints on implementation

Design of methods to achieve the changeover

Training of the staff in the changeover procedure and evaluation of change over method

Implementation is the final stage and it is an important phase the first task in implementation was implementation planning that is deciding on methods to be adopted. After the system was implemented successfully training of the user was one of the most important tasks of the developer. For this purpose the user of system manual where prepared and handed over to the user to operate the developed system. So change over plays a vital role, which checks the developed tool for the following requirements and then only the user accept the developed tool. The change over took place only when he system had been proved to the satisfaction of the system analysis and other implementation have been completed.

6. MAINTENANCE

6.1 INTRODUCTION

Software maintenance is the various modification activities that occur following the product release. The project is coded in an efficient manner that facilitates the easy understanding and there by easy maintenance. Modification is made to enhance, adapt, and correct errors in software product. Maintenance includes all the activities after the installation of the software that is performed to keep the system operational. Two forms of maintenance are being used. They are adaptive maintenance and corrective maintenance.

6.2 CORRECTIVE MAINTENANCE

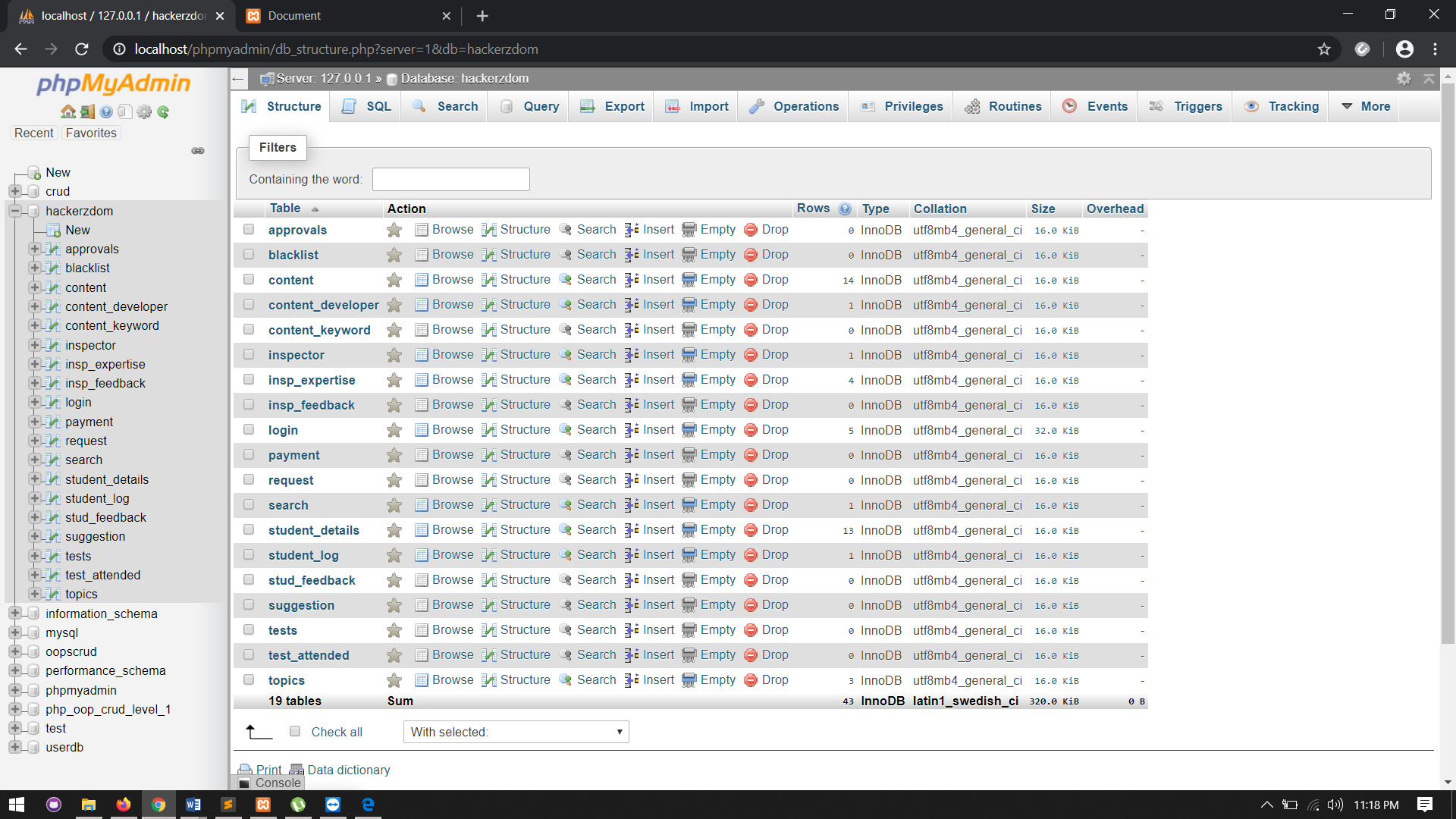
It is generally agreed that for large systems, removing all the faults before delivery is extremely difficult. Maintenance activities related to the fixing of errors falls under the corrective maintenance.

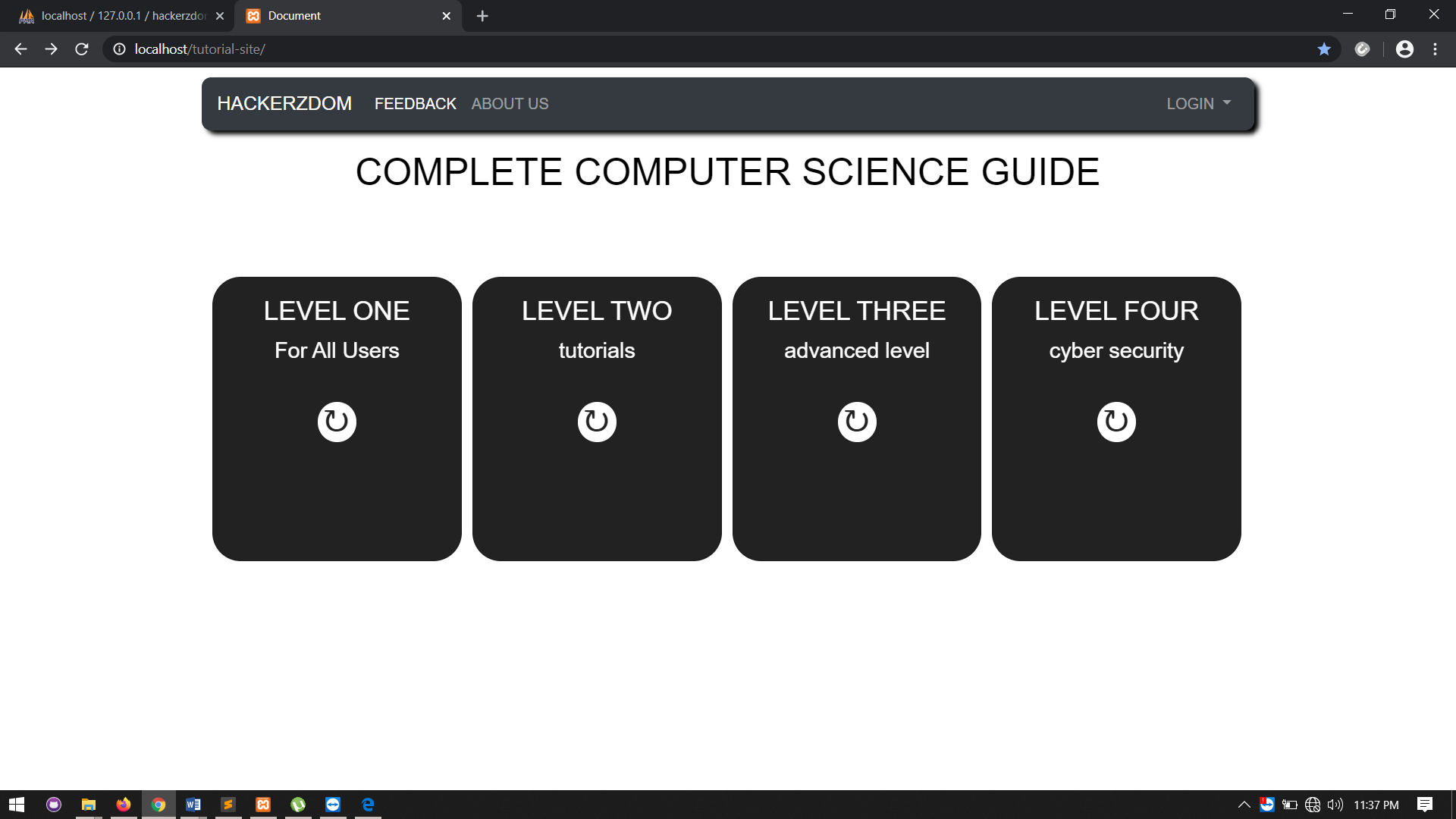
6.3 ADAPTIVE MAINTENANCE

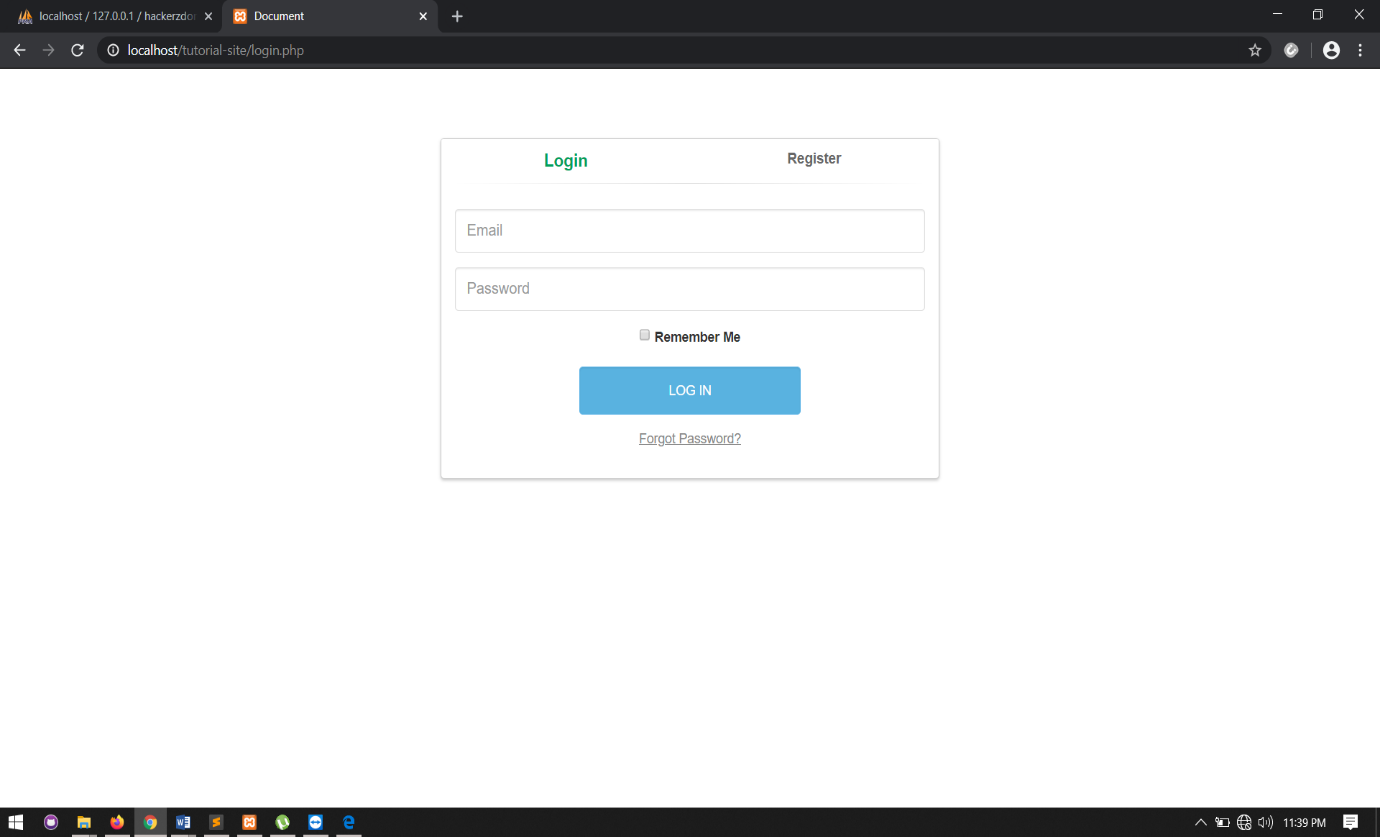
Removing errors is one of the activities of maintenance. Maintenance is also needed due to the change in environment or the requirement of the system. The introduction of the software system affects the maintenance activities related to such modification, falls under adaptive maintenance. 2

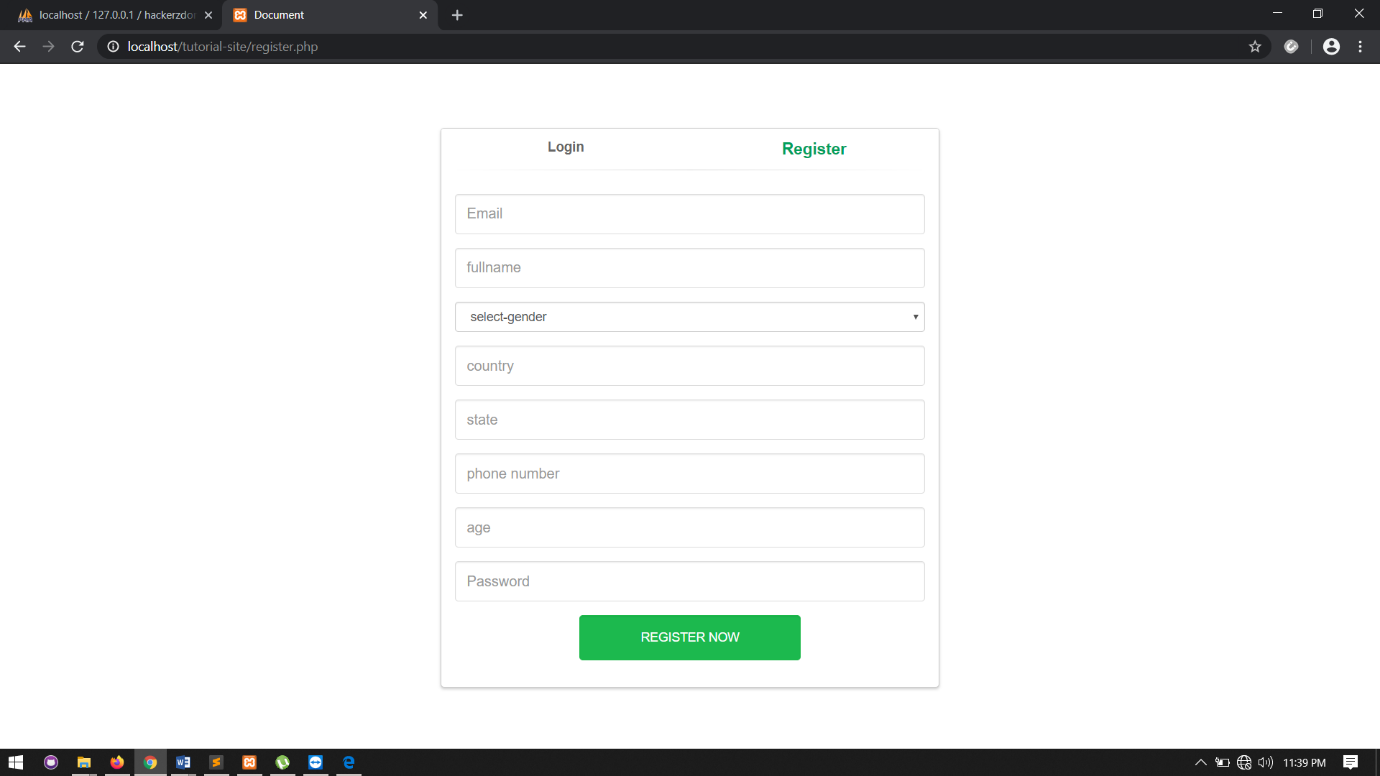
During the modification of the software, the effects of change have to be clearly understood by the maintainer since introducing undesired side effects to the system during the modification is easy. To test whether those aspects of the system that are not supported to be modified will operate as they were before modification, regression testing is done. This test includes old test cases to test that no new errors have been introduced.

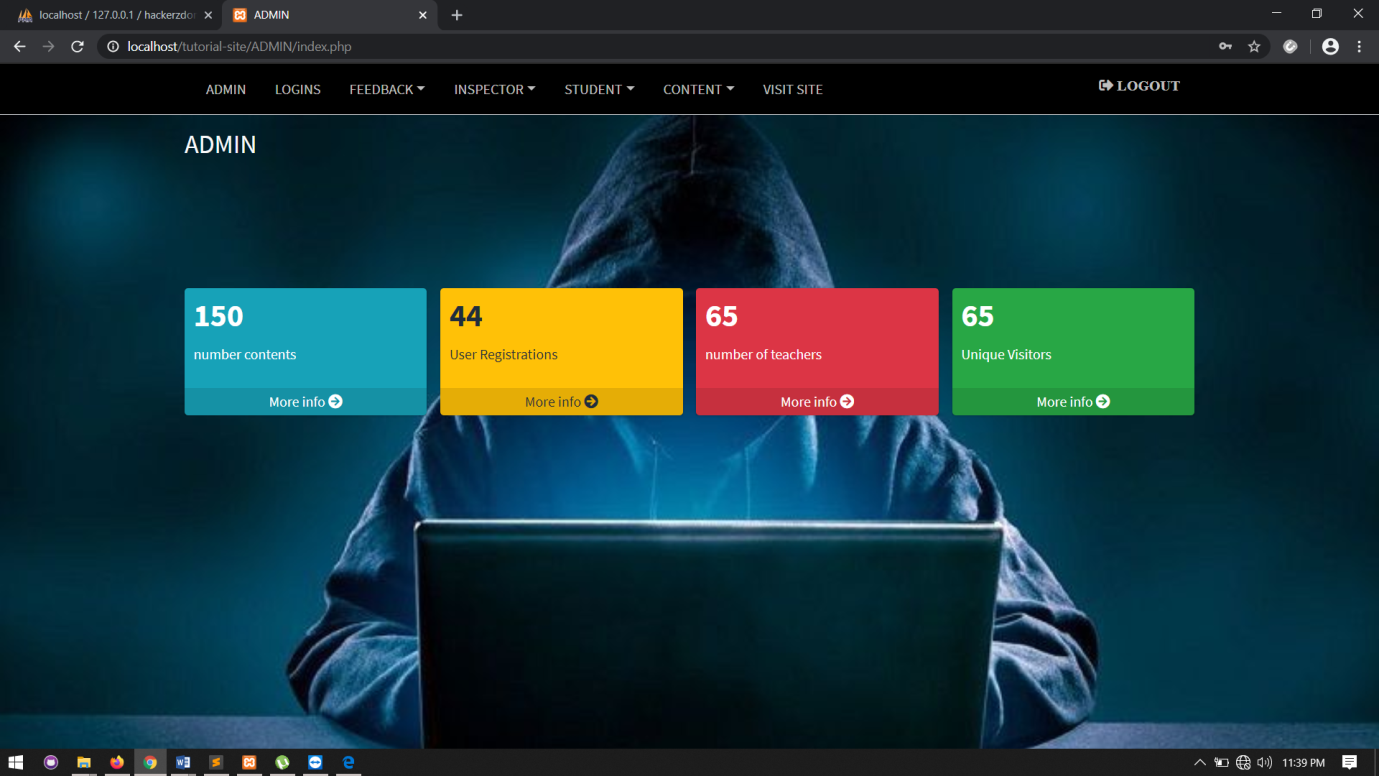
7. SCREENSHOTS

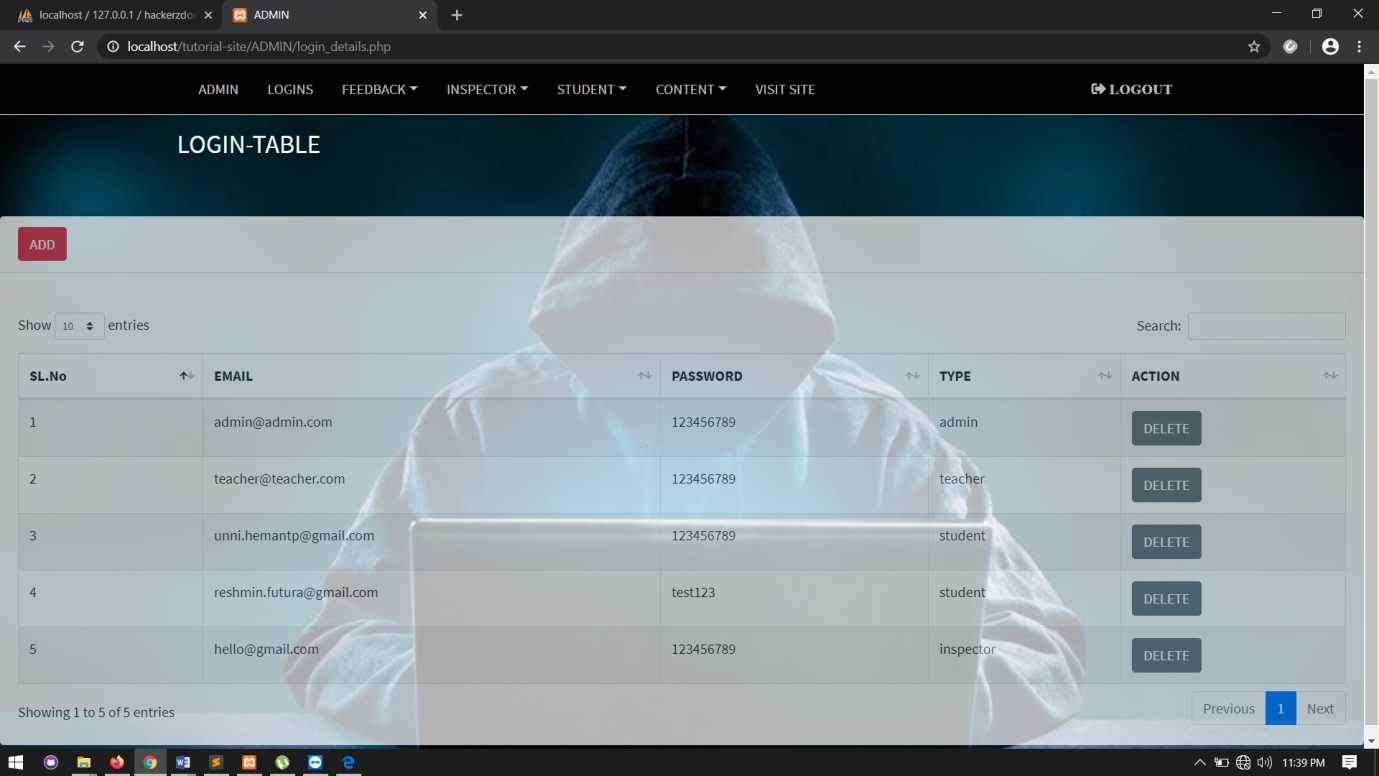


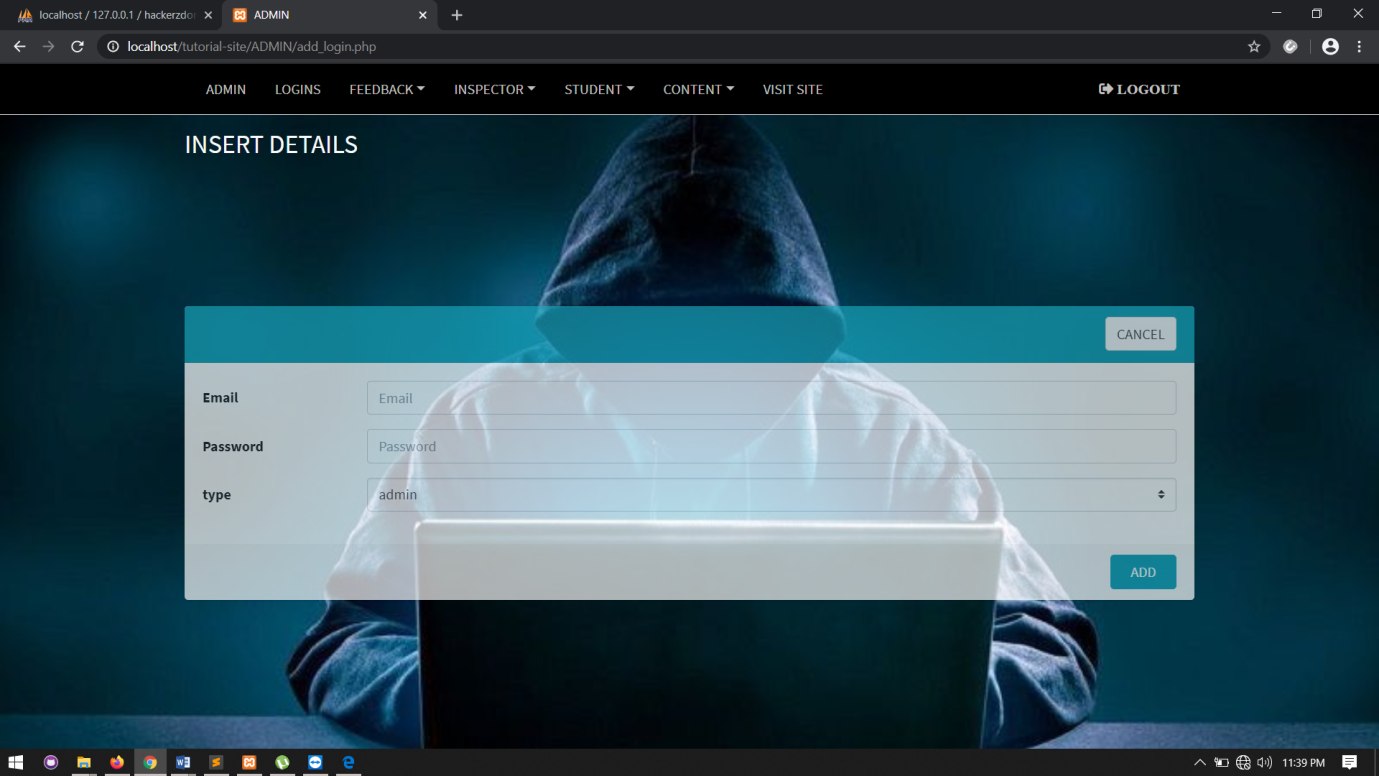












8. CONCLUSION

Any system, which has been in use for a number of years gradually, decays and become less effective because of the change in environment to which it has to be adapted. For a time being, it’s possible to overcome problems by amendments and minor modification to acknowledge the need of fundamental changes. All the suggestions forwarded in the software proposal have successfully been completed and the final threshold of the application has been crossed. Weaving through the system developed, a brief idea can be given as follows:

Comprehending the problem

Studying the existing system

Building up the course of action to reach the goal

Designing the problem

Visualizing the solution

Preparing the screen outputs

Testing the system with test data

Achieving the required results

Documenting the software developed

8.1 FUTURE ENHANCEMENT

The project has a very vast scope towards future. In the future we aim at providing free tutorials through this website. We also plan on implementing the following features:

* Provide tutorials in many native languages.
* Students can be assigned with a guide.
* Less data consuming: most sites require a high speed internet and big data plans as they use up a lot of data, providing quality at low data cost is always an advantage.
* Should load faster, should be able to make materials and video offline.
* Login easily using Google or Facebook.
* Study materials available in pdf also.
* Live projects along with inbuilt text editors (we can use inbuilt online editors without downloading software).

BIBLIOGRAPHY

BOOKS

* Systems Analysis and Design : Elias M. Awad
* Principles of Software Engineering : Rohitkhurana

WEBSITES

[www.w3schools.com](http://www.w3schools.com)

www.github.com

www.bootstrap.com

www.tutorialspoint.com

www.studytonight.com

Coding