1. **INTRODUCTION**

A Student bot project is built using artificial algorithms that analyzes user’s queries and understand user’s message. This System is a web application which provides answer to the query of the student. Students just have to query through the bot which is used for chatting. Students can chat using any format there is no specific format the user has to follow. The System uses built in artificial intelligence to answer the query. The answers are appropriate what the user queries. The User can query any college related activities through the system. The user does not have to personally go to the college for enquiry. The System analyzes the question and than answers to the user. The system answers to the query as if it is answered by the person. With the help of artificial intelligence, the system answers the query asked by the students. The system replies using an effective Graphical user interface which implies that as if a real person is talking to the user. The user just has to register himself to the system and has to login to the system. After login user can access to the various helping pages. Various helping pages has the bot through which the user can chat by asking queries related to college activities. The system replies to the user with the help of effective graphical user interface. The user can query about the college related activities through online with the help of this web application. The user can query college related activities such as date and timing of annual day, sports day, and other cultural activities. This system helps the student to be updated about the college activities.

Changes in Information Technology (IT) allows schools to utilize database and application such as student information system (SIS) thus, making the accessing of records. One of the changes that came about is this system is a web application which provides answer to the query of student. Students just have to query through the bot which used for chatting. The proposed system is a web application to manage student’s details and keeping them updated about latest events in college. The application will be used by teachers, students and parents. The utilities provided by the applications are- the user can query any college related activities through the system. The user does not have to personally go to the college for enquiry. The system analyzes the question and then answers to user. The user just has to register himself to the system and has to login to the system . After login user can access to the various helping pages. This system helps the student to be updated about the college activities.

A chat-bots aims to make a conversation between both human and machine. The machine has been embedded knowledge to identify the sentences and making a decision itself as a response to answer a question. Chat-bots will be completely based on a text-based user interface, allowing the user to type commands and receive text as well as text to speech response. Chat-bots are usually stateful services, remembering previous commands in order to provide functionality. It can be utilized securely by an even larger audience when chat-bots technology is integrated with popular web services. The college inquiry chat-bots will be built using artificial algorithms that analyze user's queries and understand user's message. The response principle is matching the input sentence from a user. The User can ask the question any college-related activities through the chat-bot without physically available to the college for inquiry. The System analyses the question and then answers to the user. With the help of artificial intelligence, the system answers the query asked by the students. The system replies using an effective Graphical User Interface as if a real person is talking to the user. The user just has to register himself to the system and has to login to the system. The chat-bots consists of core and interface that is accessing the core in (MYSQL).Natural language processing technologies are used for parsing, tokenizing, stemming and filtering the content of the complaint.

**2. LITERATURE SURVEY**

* **Human-Computer Speech interaction**

Speech recognition is one of the most natural and sought after techniques in computer and networked device interaction has only recently become possible (last two decades) with the advent of fast computing.

* **Natural Language Toolkit (NLTK)**

In order to deal with and manipulate the text resulting from speech recognition and speech to text conversion, specific toolkits are needed to organise the text into sentences then split them into words, to facilitate semantic and meaning extraction. One of these toolkits is the widely used NLTK which is a free plugin for Python.

* **Chatbot strategies**

To give suitable answers to keywords or phrases extracted from speech and to keep conversation continuous, there is a need to build a dialogue system (programme) called a Chatbot (Chatter-Bot). Chatbots can assist in human computer interaction and they have the ability to examine and influence the behaviour of the user [8] by asking questions and responding to the user's questions. The Chatbot is a computer programme that mimics intelligent conversation. The input to this programme is natural language text, and the application should give an answer that is the best intelligent response to the input sentence. This process is repeated as the conversation continues [9] and the response is either text or speech.

* **Chatbot Fundamental Design Techniques and approaches**

To design any Chatbot, the designer must be familiar with a number of techniques:

**1) Parsing:** this technique includes analysing the input text and manipulating it by using a number of NLP functions; for example, trees in Python NLTK.

**2) Pattern matching:** it is the technique that is used in most Chatbots and it is quite common in question-answer systems depending on matching types, such as natural language enquiries, simple statements, or semantic meaning of enquiries [12].

**3) AIML:** it is one of the core techniques that are used in common Chatbot design. More details about this technique and the language used are explained in section 2.5 below.

**4) Chat Script:** is the technique that helps when no matches occur in AIML. It concentrates on the best syntax to build a sensible default answer. It gives a set of functionalities such as variable concepts, facts, and logical and/or.

**5) SQL and relational database:** is a technique used recently in Chatbot design in order to make the Chatbot remember previous conversations. More details and explanation are provided in section 2.6 below.

**6) Markov Chain:** is used in Chatbots to build responses that are more applicable probabilistically and, consequently, are more correct. The idea of Markov Chains is that there is a fixed probability of occurrences for each letter or word in the same textual data set [13].

**7) Language tricks:** these are sentences, phrases, or even paragraphs available in Chatbots in order to add variety to the knowledge base and make it more convincing.

The types of language tricks are:

• Canned responses.

• Typing errors and simulating key strokes.

• Model of personal history.

• Non Sequitur (not a logical conclusion)

Each of these language tricks is used to satisfy a specific purpose and to provide alternative answers to questions.

**3. REPORT ON PRESENT INVESTIGATION**

**3.1 SYSTEM ANALYSIS:**

**3.1.1 EXISTING SYSTEM:**

Traditionally, the chat bot system is not known to people who are not more into the technology. Even if there exist a chat bot system, it is not much accurate in proving the answer or solutions. Students need to manually visit to the college to get their queries answered by the college help desk. This process consumes lot of time as well as money as the students needed to visit college if its miles away from home. Also, this process may lead to communication gap between student and college.

**3.1.2 DISADVANTAGES OF EXISTING SYSTEM:**

* Lack of accuracy and error prone.
* Overall efficiency is less.
* Non-secure.
* No perfect maintenance of records.
* No method to trace details.
* Human errors.
* The manual system is too slow.
* Searching is more time consuming.

**3.1.3 PROPOSED SYSTEM:**

The proposed system will be a web based system. So the entire project will be hosted on a cloud platform. The users can access this system from any place and at any time. The response time to the queries of the user will depend upon the internet speed of the user. If user has a decent internet connection, he/she will get the answers to his/her queries in the usual time. The usual reply time will be around 3-5 seconds as the process involves fetching the keywords from the users query, searching it in the knowledge base and then showing the output. This process will take some time, which is estimated to be 4 seconds approximately. If the user has a bad internet connection, it will take some more time for him to get the output. But even in the worst case, the response time will not exceed 15 seconds.

**3.1.4 ADVANATAGES OF PROPOSED SYSTEM:**

* User does not have to go personally to college office for the enquiry.
* This application enables the students to be updated with college cultural activities.
* This application saves time for the student as well as teaching and non teaching staffs.

**3.1.5 FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

**TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**3.1.6 SYSTEM REQUIREMENTS**

Hardware Requirements

* System : Pentium IV 2.4 GHz.
* Hard Disk : 40 GB.
* Floppy Drive : 1.44 Mb.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 512 Mb.

Software Requirements

* Operating system : Windows 10.
* Coding Language : JAVA/J2EE
* Tool : Tomcat 7.0
* Database : MYSQL
* Client Side Technologies : HTML, CSS, JavaScript

**4. SYSTEM DESIGN**

**4.1 INTRODUCTION**

System design is transition from a user oriented document to programmers or data base personnel. The design is a solution, how to approach to the creation of a new system. This is composed of several steps. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Designing goes through logical and physical stages of development, logical design reviews the present physical system, prepare input and output specification, details of implementation plan and prepare a logical design walkthrough.

The database tables are designed by analyzing functions involved in the system and format of the fields is also designed. The fields in the database tables should define their role in the system. The unnecessary fields should be avoided because it affects the storage areas of the system. Then in the input and output screen design, the design should be made user friendly. The menu should be precise and compact.

**SOFTWARE DESIGN**

In designing the software following principles are followed:

**1.Modularity and partitioning:** software is designed such that, each system should consists of hierarchy of modules and serve to partition into separate function.

**2. Coupling:** modules should have little dependence on other modules of a system.

3. **Cohesion:** modules should carry out in a single processing function.

4. **Shared use:** avoid duplication by allowing a single module be called by other that need the function it provides

**4.2 SYSTEM ARCHITECTURE**

**Fig 4.2 System Architecture**

**4.3 DATA FLOW DIAGRAM**



**Fig 4.3 Data flow diagram**

**4.4 UML DIAGRAMS**

UML Concepts

The Unified Modelling Language (UML) is a standard language for writing software blue prints. The UML is a language for

* Visualizing
* Specifying
* Constructing
* Documenting the artefacts of a software intensive system.

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modelling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modelling yields an understanding of a system.

Building Blocks of the UML:

The vocabulary of the UML encompasses three kinds of building blocks:

* Things
* Relationships
* Diagrams

Things are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.

Things in the UML:

There are four kinds of things in the UML:

* Structural things
* Behavioral things
* Grouping things
* Annotational things

**Structural things** are the nouns of UML models. The structural things used in the project design are:

First, a **class** is a description of a set of objects that share the same attributes, operations, relationships and semantics.

|  |
| --- |
| Window |
| origin  size |
| open()  close()  move()  display() |

##### Fig: Classes

Second, a **use case** is a description of set of sequence of actions that a system performs that yields an observable result of value to particular actor.



**Fig: Use Cases**

Third, a node is a physical element that exists at runtime and represents a computational resource, generally having at least some memory and often processing capability.

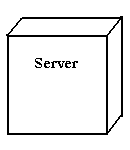


Fig: Nodes

**Behavioural things** are the dynamic parts of UML models. The behavioural thing used is:

**Interaction:**

An interaction is a behaviour that comprises a set of messages exchanged among a set of objects within a particular context to accomplish a specific purpose. An interaction involves a number of other elements, including messages, action sequences (the behaviour invoked by a message, and links (the connection between objects).



**Fig: Messages**

**4.4.1 Relationships in the UML:**

There are four kinds of relationships in the UML:

* Dependency
* Association
* Generalization
* Realization

A **dependency** is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing (the dependent thing).



**Fig: Dependencies**

An **association** is a structural relationship that describes a set links, a link being a connection among objects. Aggregation is a special kind of association, representing a structural relationship between a whole and its parts.



**Fig: Association**

A **generalization** is a specialization/ generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent).



**Fig: Generalization**

A **realization** is a semantic relationship between classifiers, where in one classifier specifies a contract that another classifier guarantees to carry out.



**Fig: Realization**

## Sequence Diagrams:

UML sequence diagrams are used to represent the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram.

Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task or scenario. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behaviour which can be difficult to extract from static diagrams or specifications.

**Actor**

Represents an external person or entity that interacts with the system

**Sequence diagram actor element**

**Fig: Actor**

**Object**

Represents an object in the system or one of its components

**Sequence diagram object element** **Fig: Object**

**Unit**

Represents a subsystem, component, unit, or other logical entity in the system (may or may not be implemented by objects)

**Sequence diagram unit element**

**Fig: Unit**

**Separator**

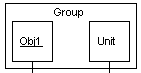
Represents an interface or boundary between subsystems, components or units (e.g., air interface, Internet, network)

**Sequence diagram separator element**

**Fig: Separator**

**Group**

Groups related header elements into subsystems or component

****

**Fig: Group**

Sequence Diagram Body Elements

**Action**

Represents an action taken by an actor, object or unit

**Sequence diagram action element**

**Fig: Action**

**Asynchronous Message**

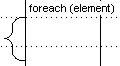
An asynchronous message between header elements

**Sequence diagram asynchronous message element**

**Fig: Asynchronous Message**

**Block**

A block representing a loop or conditional for a particular header element

****

**Fig: Block**

**Call Message**

A call (procedure) message between header elements

**Sequence diagram call message element Fig: Call Message**

**Create Message**

A "create" message that creates a header element (represented by lifeline going from dashed to solid pattern)

**Sequence diagram create message element**

**Fig: Create Message**

**Diagram Link**

Represents a portion of a diagram being treated as a functional block. Similar to a procedure or function call that abstracts functionality or details not shown at this level. Can optionally be linked to another diagram for elaboration.

**Sequence diagram diagram link element** **Fig: Link**

Else Block Represents an "else" block portion of a diagram block

**Sequence diagram else block element**

**Fig: Else**

**Message**

A simple message between header elements

**Sequence diagram message element**

**Fig: Message**

**Return Message**

A return message between header elements

**Sequence diagram return message element**

**Fig: Return**

4.4.2 USE CASE DIAGRAMS

A use case diagram is a graph of actors set of use cases enclosed by a system boundary, communication associations between the actors and users and generalization among use cases. The use case model defines the outside(actors) and inside(use case) of the system’s behavior.

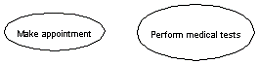
use case diagram is quite simple in nature and depicts two types of elements: one representing the business roles and the other representing the business processes.

**Figure : actor**

To identify an actor, search in the problem statement for business terms that portray roles in the system. For example, in the statement "patients visit the doctor in the clinic for medical tests," "doctor" and "patients" are the business roles and can be easily identified as actors in the system.

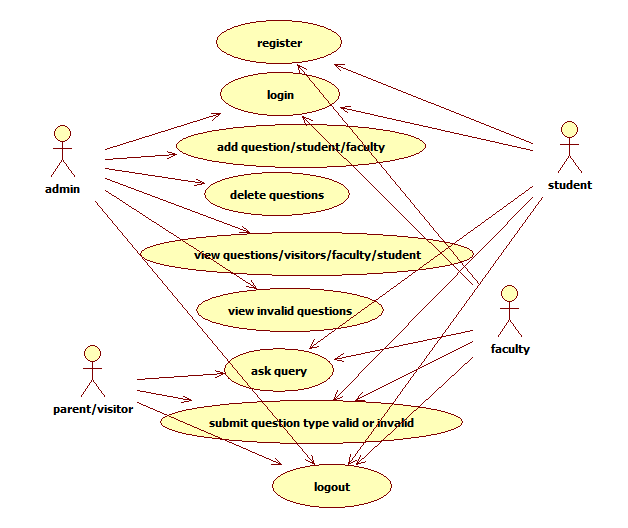
**Use case:** A use case in a use case diagram is a visual representation of a distinct business functionality in a system. The key term here is "distinct business functionality." To choose a business process as a likely candidate for modelling as a use case, you need to ensure that the business process is discrete in nature.

As the first step in identifying use cases, you should list the discrete business functions in your problem statement. Each of these business functions can be classified as a potential use case. Remember that identifying use cases is a discovery rather than a creation. As business functionality becomes clearer, the underlying use cases become more easily evident. A use case is shown as an ellipse in a use case diagram (see Figure ).



**Figure : use case**

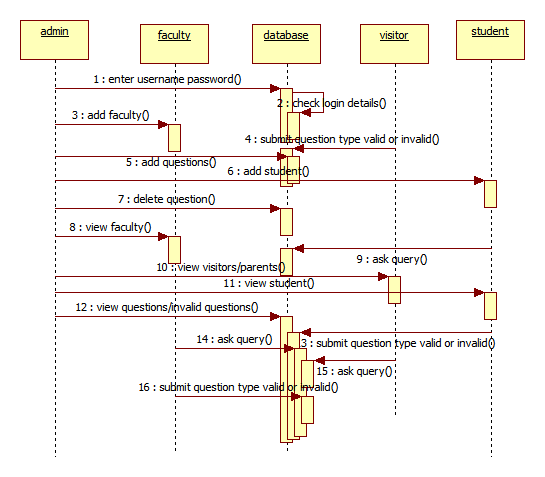
Figure shows two uses cases: "Make appointment" and "Perform medical tests" in the use case diagram of a clinic system. As another example, consider that a business process such as "manage patient records" can in turn have sub-processes like "manage patient's personal information" and "manage patient's medical information." Discovering such implicit use cases is possible only with a thorough understanding of all the business processes of the system through discussions with potential users of the system and relevant domain knowledge.



**Fig 4.4.2 Use Case (Admin,student,faculty,parent)**

4.4.3 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams** or **event scenarios**.



**Fig 4.4.3 Sequence Diagram**

4.4.4 ACTIVITY DIAGRAM:

Activity diagrams represent the business and operational workflows of a system. An Activity diagram is a dynamic diagram that shows the activity and the event that causes the object to be in the particular state.

So, what is the importance of an Activity diagram, as opposed to a State diagram? A State diagram shows the different states an object is in during the lifecycle of its existence in the system, and the transitions in the states of the objects. These transitions depict the activities causing these transitions, shown by arrows.

An Activity diagram talks more about these transitions and activities causing the changes in the object states.

**Activity Diagram:**

#### Defining an Activity diagram

Let us take a look at the building blocks of an Activity diagram.

#### Elements of an Activity diagram

An Activity diagram consists of the following behavioural elements:

**Initial Activity:** This shows the starting point or first activity of the flow. Denoted by a solid circle. This is similar to the notation used for Initial State.

http://www.developer.com/img/articles/2003/08/11/UML07T1.gif

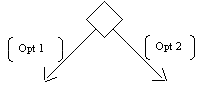
**Fig: Initial State**

**Activity:** Represented by a rectangle with rounded (almost oval) edges.

. http://www.developer.com/img/articles/2003/08/11/UML07T2.gif

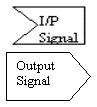
**Fig: Action State**

**Decisions:** Similar to flowcharts, a logic where a decision is to be made is depicted by a diamond, with the options written on either sides of the arrows emerging from the diamond, within box brackets.



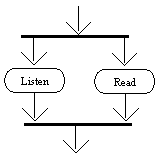
**Fig: Decision**

**Signal:** When an activity sends or receives a message, that activity is called a signal. Signals are of two types: Input signal (Message receiving activity) shown by a concave polygon and Output signal (Message sending activity) shown by a convex polygon.



**Fig: Signal**

**Concurrent Activities:** Some activities occur simultaneously or in parallel. Such activities are called concurrent activities. For example, listening to the lecturer and looking at the blackboard is a parallel activity. This is represented by a horizontal split (thick dark line) and the two concurrent activities next to each other, and the horizontal line again to show the end of the parallel activity.



**Fig: Horizontal**



**Fig 4.4.4 Activity Diagram**

**4.4.5 Class Diagram:**

An object is any person, place, thing, concept, event, screen, or report applicable to your system. Objects both know things (they have attributes) and they do things (they have methods).

A class is a representation of an object and, in many ways; it is simply a template from which objects are created. Classes form the main building blocks of an object-oriented application.

**Class diagram of chat bot:**



**Fig 4.4.5 Class Diagram**

## Object diagram

An **object diagram** in the Unified Modeling Language (UML), is a diagram that shows a complete or partial view of the structure of a modeled system at a specific time.

An Object diagram focuses on some particular set of object instances and attributes, and the links between the instances. A correlated set of object diagrams provides insight into how an arbitrary view of a system is expected to evolve over time. Object diagrams are more concrete than class diagrams, and are often used to provide examples, or act as test cases for the class diagrams. Only those aspects of a model that are of current interest need be shown on an object diagram.

## Deployment Diagram

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.

So deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

The purpose of deployment diagrams can be described as:

The name *Deployment* itself describes the purpose of the diagram. Deployment diagrams are used for describing the hardware components where software components are deployed.

* Visualize hardware topology of a system.
* Describe the hardware components used to deploy software components.
* Describe runtime processing nodes.
* Deployment Diagram

**Component Diagram**

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another. This illustrates the service consumer - service provider relation between the two components.

**Entity relationship diagram:**

An entity-relationship (ER) diagram is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems.

**4.5 INPUT AND OUTPUT DESIGN**

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES**

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

### 5. IMPLEMENTATION

**5.1 INTRODUCTION**

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigation of the existing system and it’s constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

**5.2 MODULES**

After careful analysis the system has been identified to have the following modules:

* Admin module
* Student module
* Faculty module
* Parent/Visitor module

**5.2.1 ADMIN MODULE**

Admin: Admin has to login to the system to access various pages given below.

* Add Student: - Can add new student with personal details into the system.
* Add / Manage Course: - Can add a new course details or can manage the pre-added course.
* Upload Time-Table Image: - Can upload an image of the time-table of various standards/Classes.
* Upload Syllabus Image: - Can upload image of syllabus for various academics.
* Upload General Notice: - Here, admin can upload a general notice in form of image/pdf.
* Upload Exam Time Table Image: - Can upload image of exam time-table of every class.
* Upload Exam Notice Image: - Can upload an image file of Exam Notice. o Manage Chat Bot: - System allows admin to view and edit chat bot.
* View Invalid Answer: - Can view all the invalid answers marked by the users.

**5.2.2 STUDENT MODULE**

Chat Bot + Forum:

* User can chat with the bot it implies as if enquiring to the college person about college related activities.
* User has full authority to mark the answer as invalid if the answer doesn’t match the question.
* This will notify the admin which will be analyzed and then corrected.
* Using forum, user can ask question to the admin regarding the college activities or any other.
* View Time- Table: - All the registered user can view the uploaded image of time-table by admin.
* View Syllabus: - Users can view the syllabus of their academics in the form of image.
* View General Notice: - Can view the notice or can download the pdf file.
* View Exam Time Table: - Can view the exam time-table which is uploaded by the admin
* Exam Notice: - Can view exam notice which is uploaded by the admin.

**5.2.3. FACULTY MODULE**

The faculty needs to login to his account by entering the credentials received like password through email. He/She can add the questions related to academic like notes upload, marks sheet upload ,attendance sheet upload but other than this he cannot add other related category questions. He can make enquiry related to college events and notify as notification for invalid questions. At last logout after completion.

**5.2.4. PARENT/VISITOR MODULE**

Parents cannot come to college frequently to get updates related to college so, they can be updated with this application. They can make queries and get answers related to them. Their is no login for this module

**5.3 SAMPLE CODE:**

**LOGIN.jsp:**

<!DOCTYPE HTML>

<html>

<head>

<title></title>

<meta name=*"description"* content=*"website description"* />

<meta name=*"keywords"* content=*"website keywords, website keywords"* />

<meta http-equiv=*"content-type"*

content=*"text/html; charset=windows-1252"* />

<link rel=*"stylesheet"* type=*"text/css"*

href=*"http://fonts.googleapis.com/css?family=Tangerine&amp;v1"* />

<link rel=*"stylesheet"* type=*"text/css"*

href=*"http://fonts.googleapis.com/css?family=Yanone+Kaffeesatz"* />

<link rel=*"stylesheet"* type=*"text/css"* href=*"style/style.css"* />

<script>

**function** check() {

**var** a = document.ff.user.value;

**var** b = document.ff.pass.value;

**if** (a == 0) {

alert('Please Enter UserId');

**return** **false**;

document.getElementById("name").focus();

}

**if** (b == 0) {

alert('Please Enter Password');

**return** **false**;

document.getElementById("pass").focus();

}

}

</script>

</head>

<body>

<div id=*"main"*>

<div id=*"header"*>

<div id=*"logo"*>

<h1>

</h1>

<div class=*"slogan"*>Educational Enquiry Chat bot</div>

</div>

<div id=*"menubar"*>

<ul id=*"menu"*>

<!-- put class="current" in the li tag for the selected page - to highlight which page you're on -->

<li><a class=*"current"* href=*"index.jsp"*>Home</a></li>

<li class=*"hvr-sweep-to-bottom"*><a href=*"login.jsp"*>Login</a></li>

</ul>

</div>

</div>

<div id=*"site\_content"*>

<div id=*"content"*>

<!-- insert the page content here -->

<%**if** (request.getParameter("status") != **null**)

{

%>

<h1>Invalid User Name and Password</h1>

<%

} **else** {

%>

<h1>Login Here</h1>

<%

}

%>

<form action=*"LoginServlet"* name=*"ff"* method=*"get"*

onsubmit="return check()">

<div class=*"form\_settings"*>

<p><span>User Id</span><input class=*"contact"* type=*"text"* name=*"user"* id=*"user"* /></p>

<p><span>Password</span><input class=*"contact"* type=*"password"* name=*"pass"* id=*"pass"* /></p>

<p style="padding-top: *15px*"><span>&nbsp;</span><input class=*"submit"* type=*"submit"*

name=*"contact\_submitted"* value=*"Login"* /></p>

</div>

</form>

</div>

</div>

</div>

<div id=*"footer"*>

<p>Copyright &copy;<a href=*"http://www.technologies.com"*>info</a></p>

</div>

</body>

</html>

**Add question.jsp**

<!DOCTYPE HTML>

<html>

<head>

<title></title>

<meta name=*"description"* content=*"website description"* />

<meta name=*"keywords"* content=*"website keywords, website keywords"* />

<meta http-equiv=*"content-type"*content=*"text/html; charset=windows-1252"* />

<link rel=*"stylesheet"*type=*"text/css"*href=*"http://fonts.googleapis.com/css?family=Tangerine&amp;v1"* />

<link rel=*"stylesheet"* type=*"text/css"*href=*"http://fonts.googleapis.com/css?family=Yanone+Kaffeesatz"* />

<link rel=*"stylesheet"* type=*"text/css"* href=*"style/style.css"* />

</head>

<body>

<div id=*"main"*>

<div id=*"header"*>

<div id=*"logo"*>

<h1>

</h1>

<div class=*"slogan"*>Educational Enquiry Chat bot</div>

</div>

<div>

<ul id=*"menu"*>

<!-- put class="current" in the li tag for the selected page - to highlight which page you're on -->

<li><a class=*"current"* href=*"adminhome.jsp"*>Home</a></li>

<li><a class=*"current"* href=*"viewenquires.jsp"*>Enquires</a></li>

<li><a class=*"current"* href=*"addquestions.jsp"*>Add Questions</a></li>

<li><a class=*"current"* href=*"addstudents.jsp"*>Add Students</a></li>

<li><a class=*"current"* href=*"addfacultys.jsp"*>Add Faculty</a></li>

<li><a class=*"current"* href=*"viewfeedbacks.jsp"*>View Feedbacks</a></li>

<li><a class=*"current"* href=*"visitorquerys.jsp"*>Visitors</a></li>

<li class=*"hvr-sweep-to-bottom"*><a href=*"logout.jsp"*>Logout</a></li>

</ul>

</div>

</div>

<div id=*"site\_content"*>

<div id=*"content"*>

<!-- insert the page content here -->

<%

**if** (request.getParameter("status") != **null**) {

%>

<h1>Success<%=request.getParameter("status")%></h1>

<%

} **else** {

%>

<h1>Add Question</h1>

<%

}

%>

<form action=*"AddQuestionServlet"* name=*"ff"* method=*"post"* enctype=*"multipart/form-data"*onsubmit="return check()">

<div class=*"form\_settings"*>

<p>

<span>Department</span>

<select name=*"department"*>

<option value=*"cse"*>CSE</option>

<option value=*"ece"*>ECE</option>

<option value=*"mech"*>MECH</option>

<option value=*"eee"*>EEE</option>

<option value=*"it"*>IT</option>

</select>

</p>

<p>

<span>Year</span>

<select name=*"year"*>

<option value=*"1"*>1</option>

<option value=*"2"*>2</option>

<option value=*"3"*>3</option>

<option value=*"4"*>4</option>

</select>

</p>

<p>

<span>Semester</span>

<select name=*"semester"*>

<option value=*"1"*>1</option>

<option value=*"2"*>2</option>

</select>

</p>

<p>

<span>Section</span>

<select name=*"section"*>

<option value=*"a"*>A</option>

<option value=*"b"*>B</option>

<option value=*"c"*>C</option>

<option value=*"d"*>D</option>

</select>

</p>

<p>

<span>Category</span>

<select name=*"category"*>

<option value=*"academic"*>academic</option>

<option value=*"admission"*>admission</option>

<option value=*"sports"*>sports</option>

<option value=*"other"*>other</option>

</select>

</p>

<p>

<span>Question</span><input class=*"contact"* type=*"text"* name=*"question"* id=*"pass"* />

</p>

<p>

<span>Answer</span><input class=*"contact"* type=*"text"* name=*"answer"* id=*"user"* />

</p>

<p>

<span></span><input class=*"contact"* type=*"file"* name=*"file"* id=*"pass"* />

</p>

<p>

<span>Keyword 1</span><input class=*"contact"* type=*"text"* name=*"keyword1"* id=*"user"* />

</p>

<p>

<span>Keyword 2</span><input class=*"contact"* type=*"text"* name=*"keyword2"* id=*"user"* />

</p>

<p>

<span>Keyword 3</span><input class=*"contact"* type=*"text"* name=*"keyword3"* id=*"user"* />

</p>

<p>

<span>Keyword 4</span><input class=*"contact"* type=*"text"* name=*"keyword4"* id=*"user"* />

</p>

<p>

<span>Keyword 5</span><input class=*"contact"* type=*"text"* name=*"keyword5"* id=*"user"* />

</p>

<p style="padding-top: *15px*">

<span>&nbsp;</span><input class=*"submit"* type=*"submit"* name=*"contact\_submitted"* value=*"Add Question"* />

</p>

</div>

</form>

</div>

</div>

</div>

<div id=*"footer"*>

<p>Copyright &copy; | <a href=*"http://www.technologies.com"*>info</a></p>

</div>

</body>

</html>

**Add students.jsp**

<!DOCTYPE HTML>

<html>

<head>

<title></title>

<meta name=*"description"* content=*"website description"* />

<meta name=*"keywords"* content=*"website keywords, website keywords"* />

<meta http-equiv=*"content-type"* content=*"text/html; charset=windows-1252"* />

<link rel=*"stylesheet"* type=*"text/css"* href=*"http://fonts.googleapis.com/css?family=Tangerine&amp;v1"* />

<link rel=*"stylesheet"* type=*"text/css"*

href=*"http://fonts.googleapis.com/css?family=Yanone+Kaffeesatz"* />

<link rel=*"stylesheet"* type=*"text/css"* href=*"style/style.css"* />

</head>

<body>

<div id=*"main"*>

<div id=*"header"*>

<div id=*"logo"*>

<h1>

</h1>

<div class=*"slogan"*>Educational Enquiry Chat bot</div>

</div>

<div>

<ul id=*"menu"*>

<!-- put class="current" in the li tag for the selected page - to highlight which page you're on -->

<li><a class=*"current"* href=*"adminhome.jsp"*>Home</a></li>

<li><a class=*"current"* href=*"viewenquires.jsp"*>Enquires</a></li>

<li><a class=*"current"* href=*"addquestions.jsp"*>Add Questions</a></li>

<li><a class=*"current"* href=*"addstudents.jsp"*>Add Students</a></li>

<li><a class=*"current"* href=*"addfacultys.jsp"*>Add Faculty</a></li>

<li><a class=*"current"* href=*"viewfeedbacks.jsp"*>View Feedbacks</a></li>

<li><a class=*"current"* href=*"visitorquerys.jsp"*>Visitors</a></li>

<li class=*"hvr-sweep-to-bottom"*><a href=*"logout.jsp"*>Logout</a></li>

</ul>

</div>

</div>

<div id=*"site\_content"*>

<div id=*"content"*>

<!-- insert the page content here -->

<%

**if** (request.getParameter("status") != **null**) {

%>

<h1>success</h1>

<%

} **else** {

%>

<h1>Upload Students</h1>

<%

}

%>

<form action=*"AddStudentServlet"* name=*"ff"* method=*"post"* enctype=*"multipart/form-data"* onsubmit="return check()">

<div class=*"form\_settings"*>

<p>

<span></span><input class=*"contact"* type=*"file"* name=*"file"* id=*"pass"* /></p>

<p style="padding-top: *15px*">

<span>&nbsp;</span><input class=*"submit"* type=*"submit"* name=*"contact\_submitted"* value=*"Upload"* />

</p>

</div>

</form>

</div>

</div>

</div>

<div id=*"footer"*>

<p>Copyright &copy; | <a href=*"http://www.technologies.com"*>info</a>

</p>

</div>

</body>

</html>

**6. SYSTEM TESTING**

**6.1 INTRODUCTION:**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**6.2 TYPES OF TESTS**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page

### Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

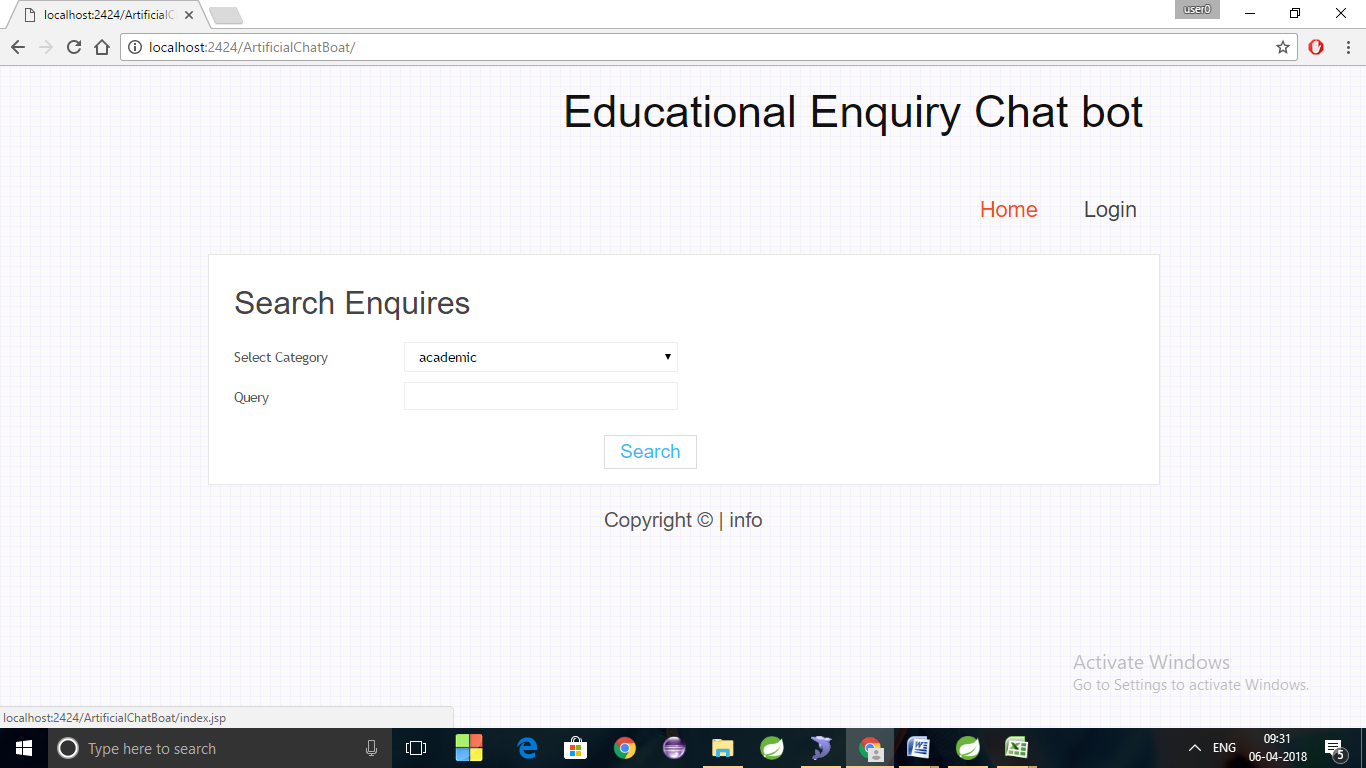
**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

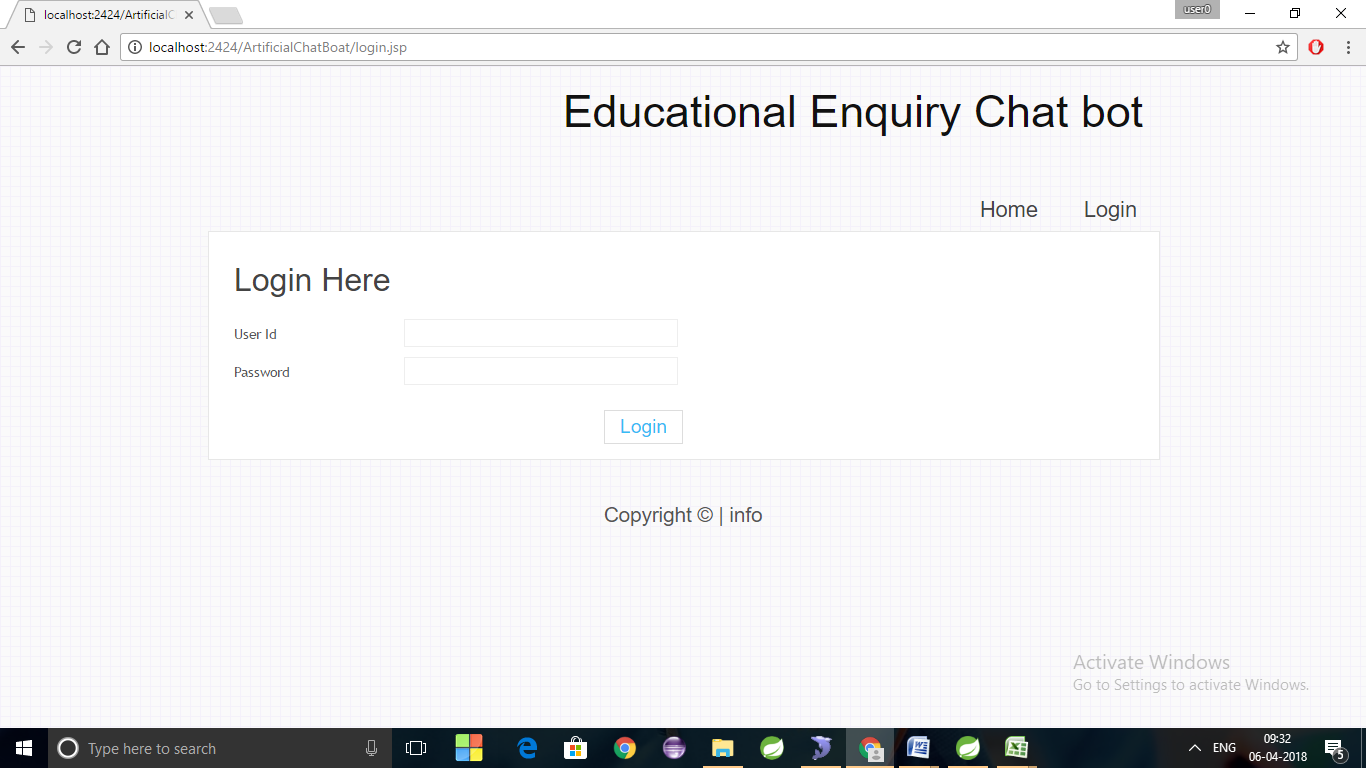
**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**7. RESULTS AND DISCUSSION**



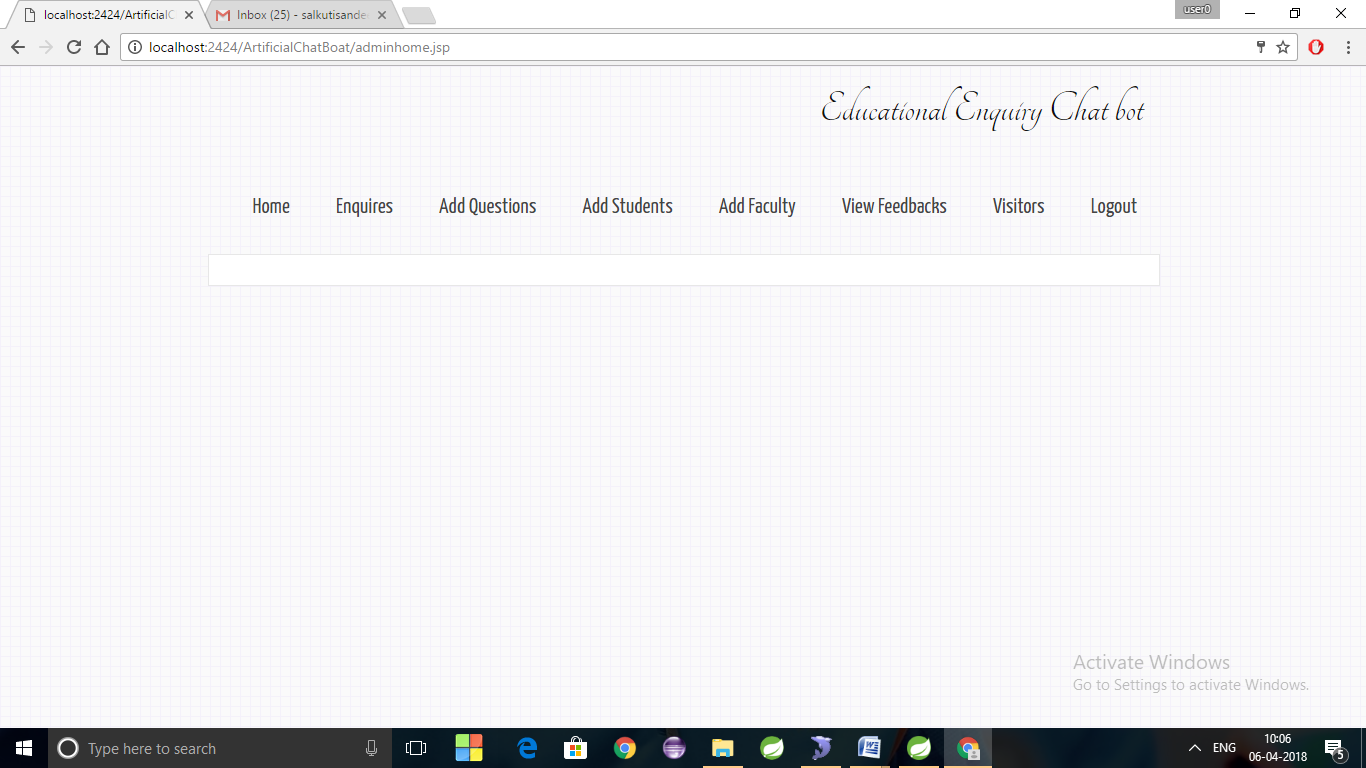
**Fig 7.1 Home page**

This is the home page of chat bot application where visitor can directly enquiry without any registration and other users namely students, faculty and admin can login here.

****

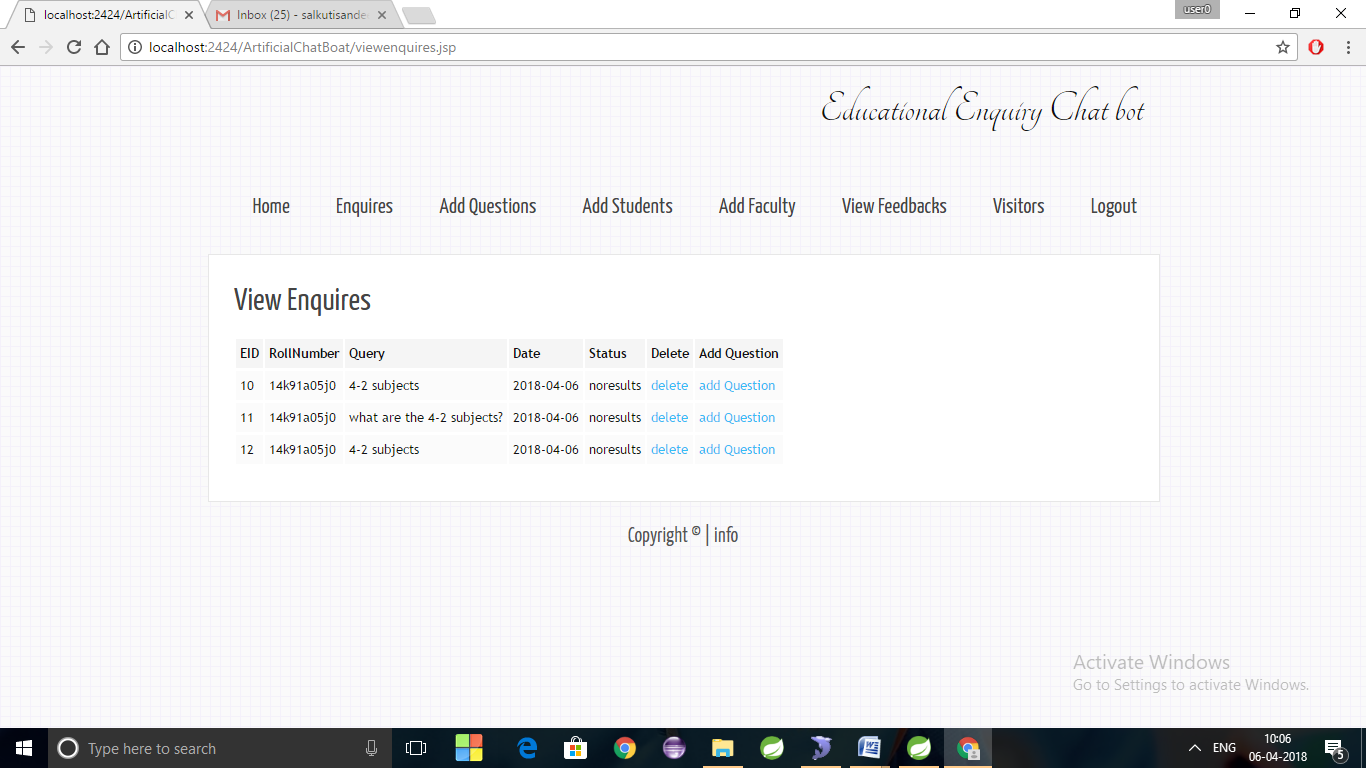
**Fig 7.1.1 Home page**

This is the login page of admin, student and faculty.

****

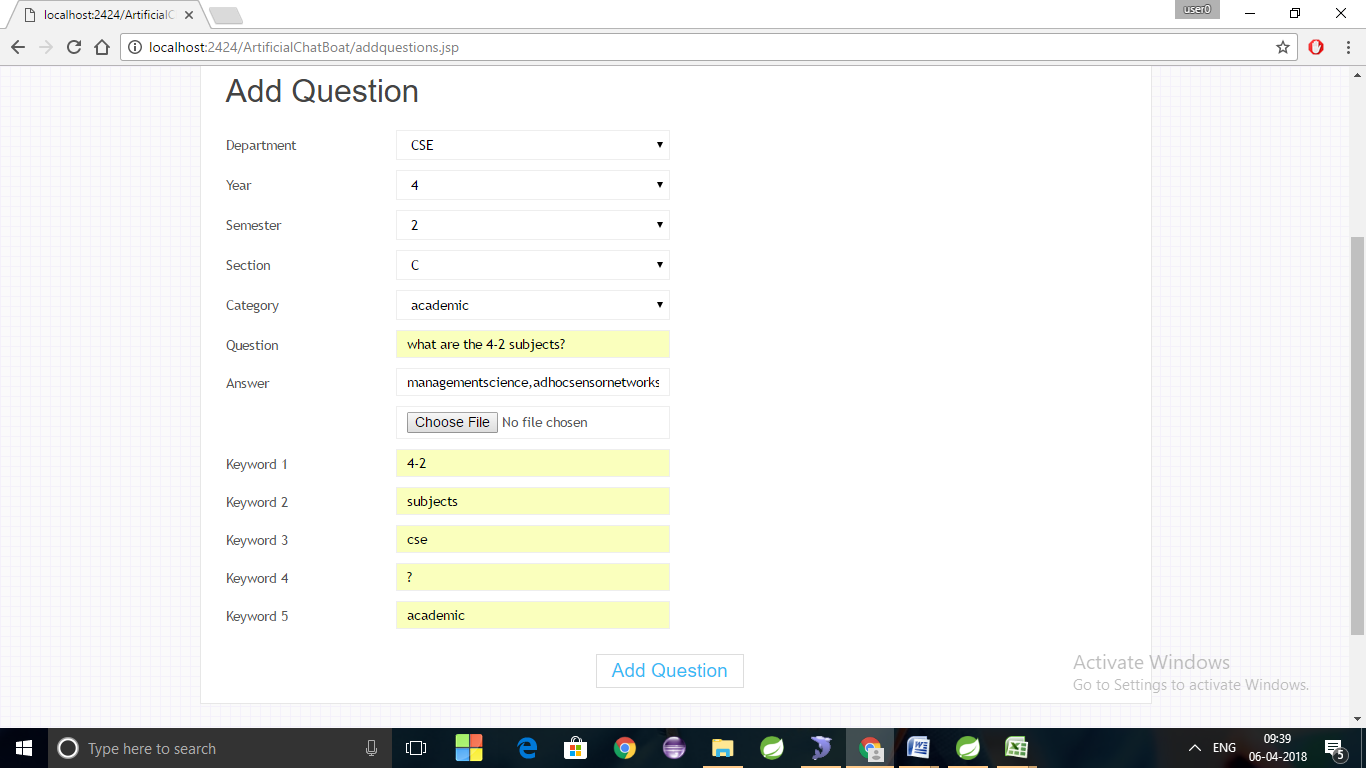
**Fig 7.2 Admin page**

This is the admin page where he can see the enquires made by students, faculties and visitors, he can add questions, he can add students by uploading excel sheets, he can view feedbacks.

****

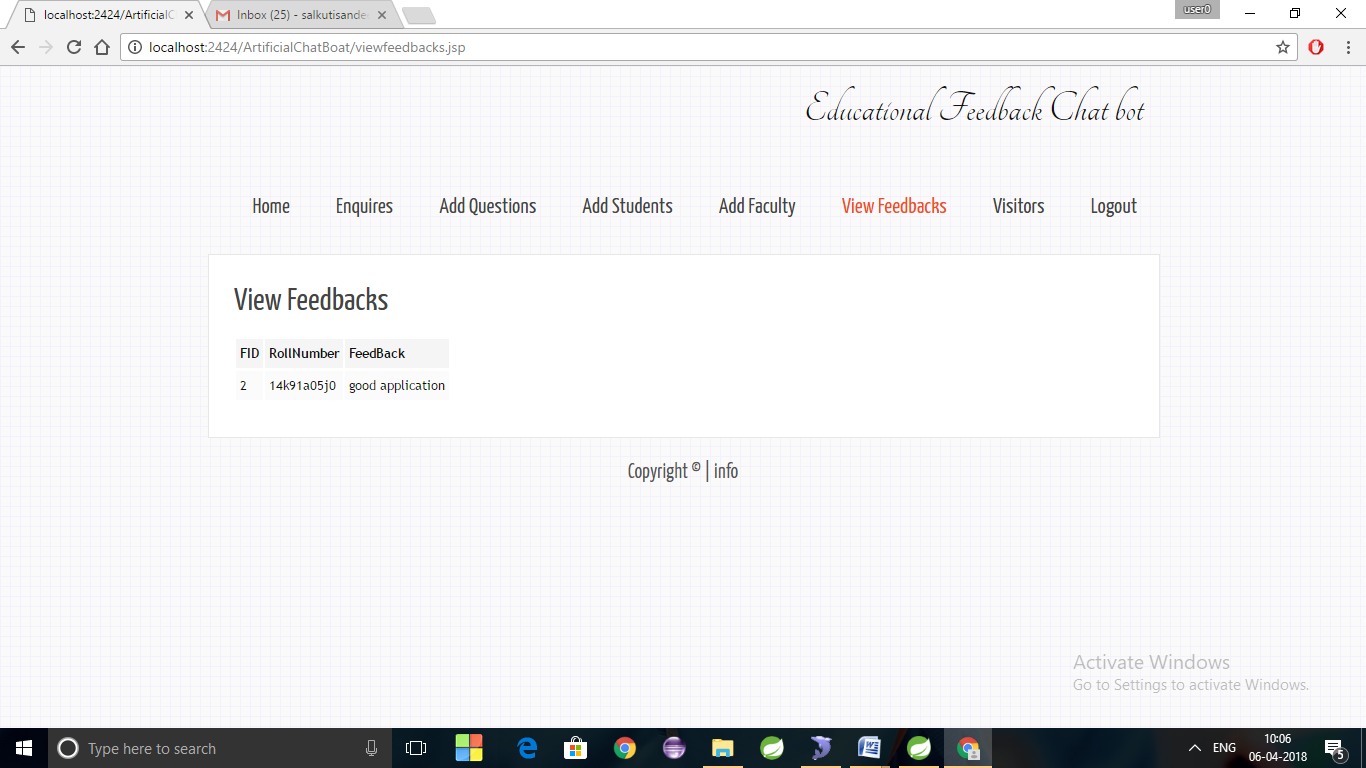
**Fig 7.2.1 view enquiry page**

In this admin can view queries made by students with their roll numbers and either he can delete it or add for no results found questions.

****

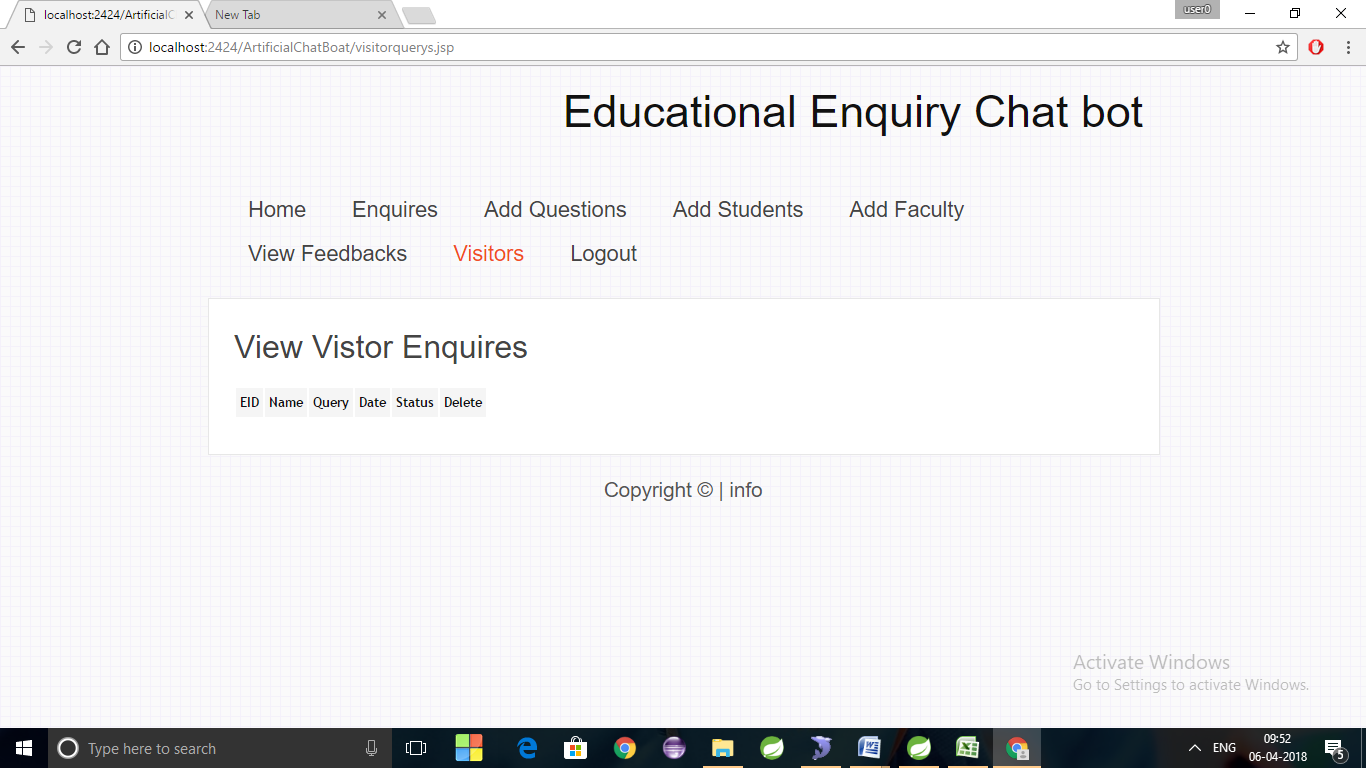
**Fig 7.2.2 Add question page**

This is add question page where admin adds question and write answers or uploads images according to it, he also adds keywords to questions.

****

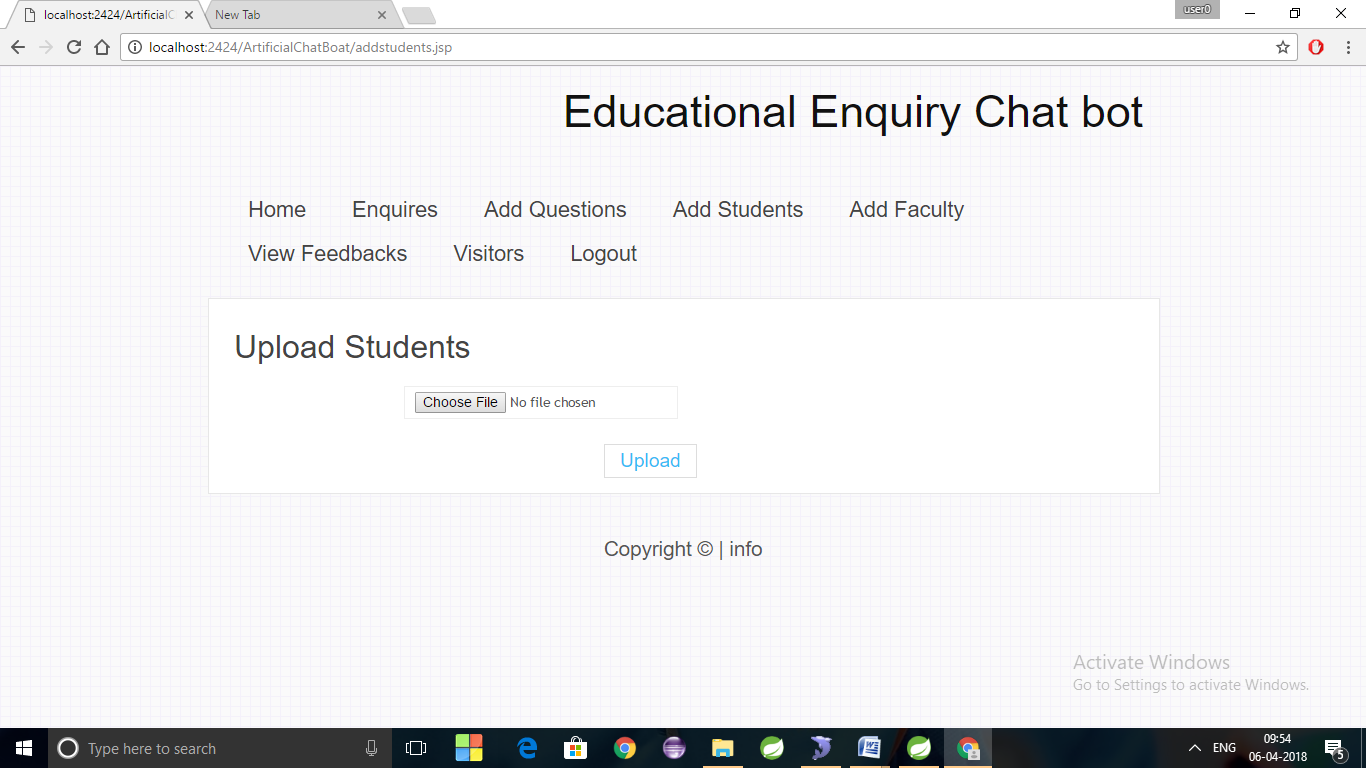
**Fig 7.2.3 view feedbacks page**

This is feedback section where admin sees the feedbacks given by students about application or any other queries.

****

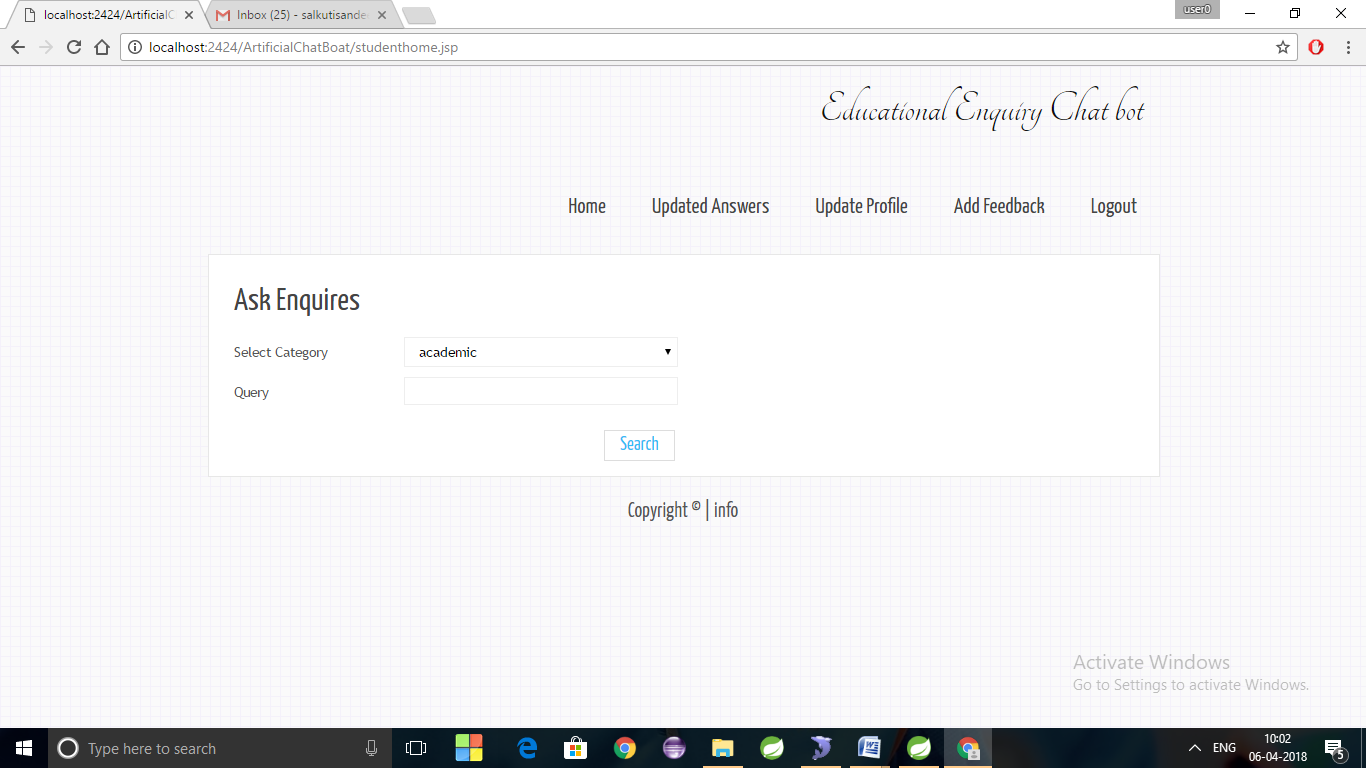
**Fig 7.2.4 View visitor enquires page**

This is visitor made queries section where he can see their name, query, status and he can delete.

****

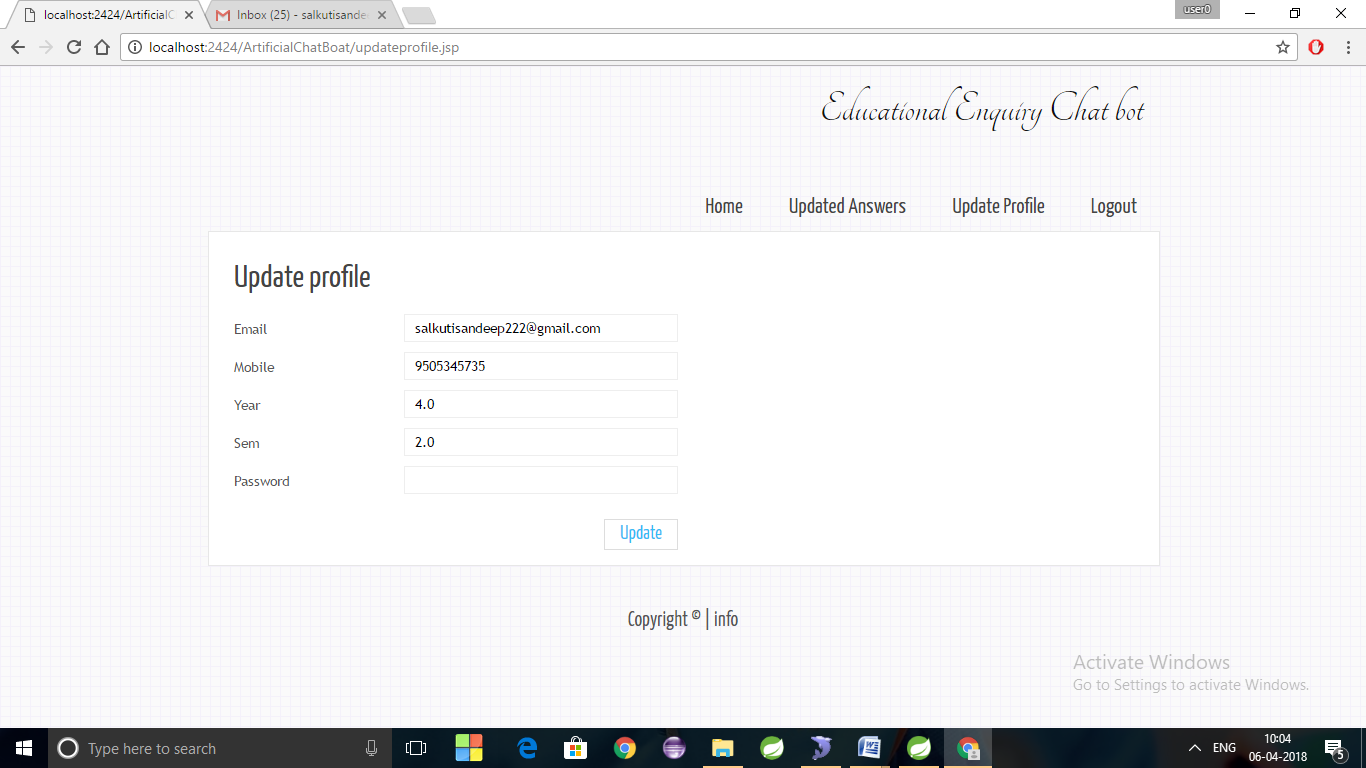
**Fig 7.2.5 Add students page**

This is upload students section where admin uploads excel sheet and the members in sheet get emails with password to login into their account this process is similar for faculty also.

****

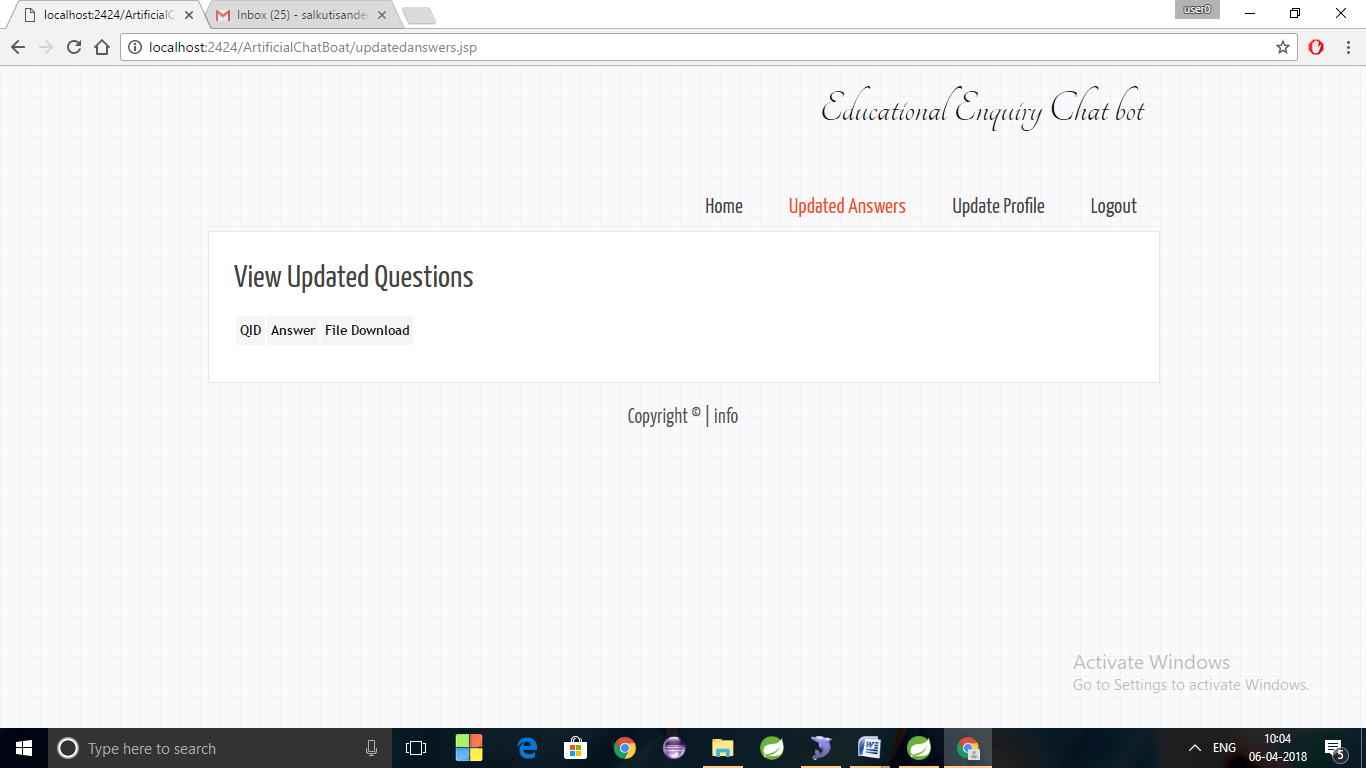
**Fig 7.3 Student page**

This is student home page where he can ask questions and get answers, he can see the updated answers for queries which shown as no result found, he can update his profile, he can give feedback and logout after performing all operations.

****

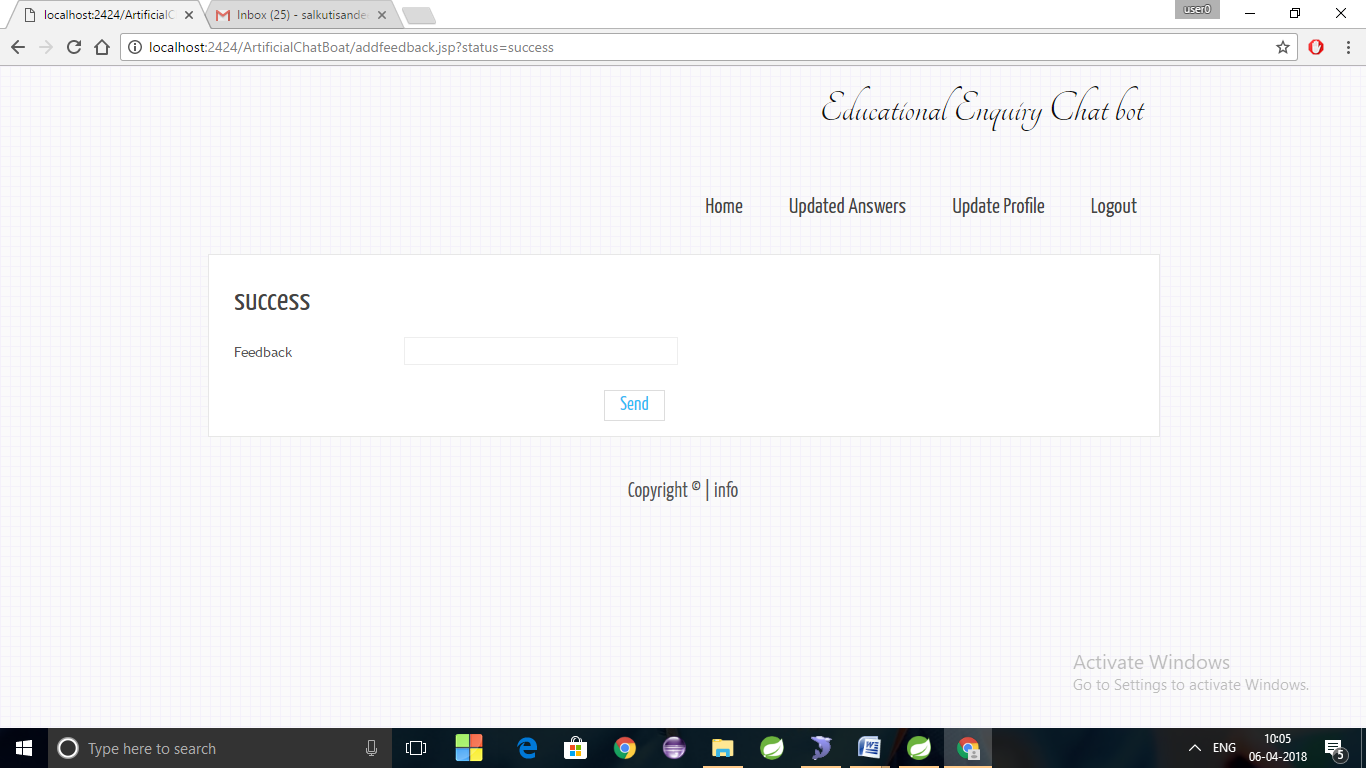
**Fig 7.3.1 Update profile page**

This is update profile page he can change his password and update his other details.

****

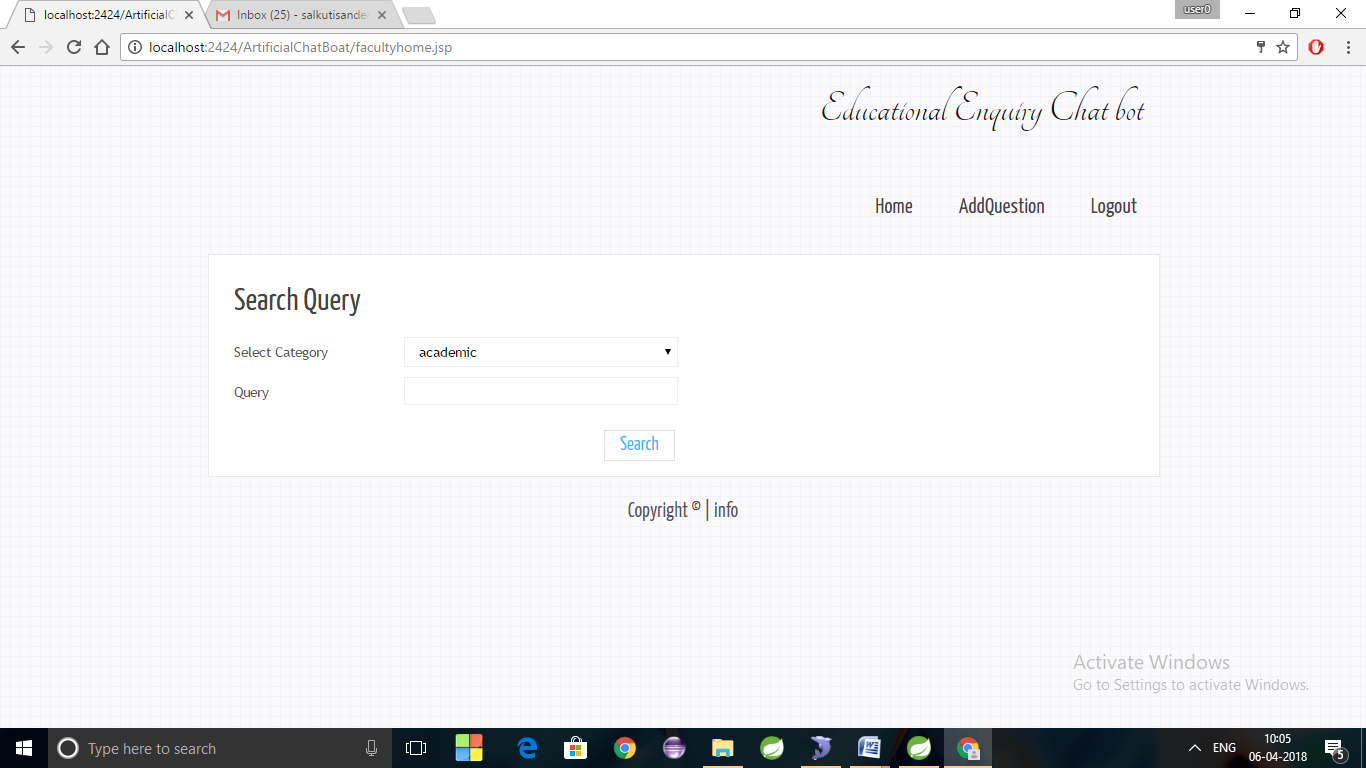
**Fig 7.3.2 View updated answers page**

This is section is for viewing updated questions he can see updated answer and can download I there is any file.

****

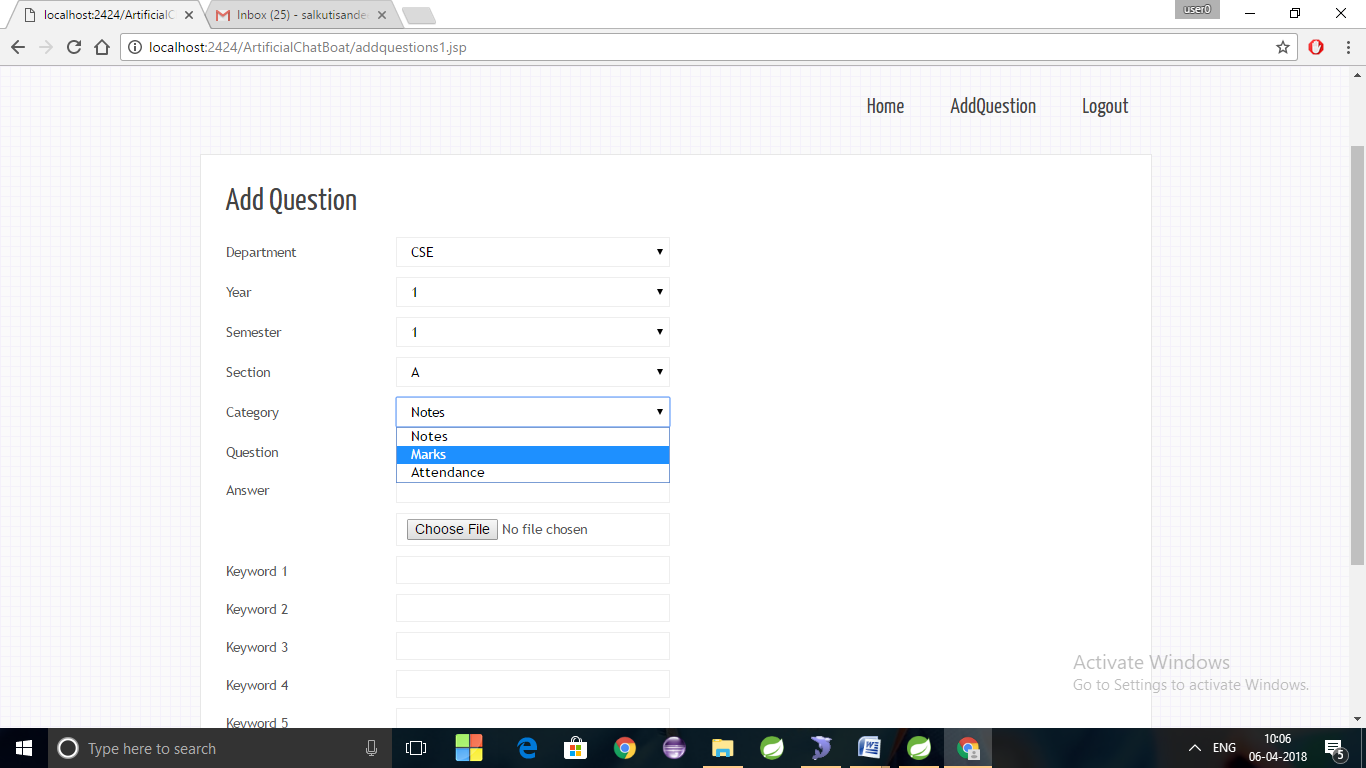
**Fig 7.3.3 Add feedback page**

This section is for giving feedbacks.

****

**Fig 7.4 Faculty page**

This is faculty home page where they can ask queries, Add questions and logout their account.

****

**Fig 7.4.1 Add question page**

This is add question page of faculty where they can add only questions related to notes, marks, attendance.

**8. SUMMARY AND CONCLUSIOON**

The conclusion of the application is, we are able be hosted on a cloud platform. The users can access this system from any place and at any time. The response time to the queries of the user will depend upon the internet speed of the user. If user has a decent internet connection, he/she will get the answers to his/her queries in the usual time. The usual reply time will be around 3-5 seconds as the process involves fetching the keywords from the users query, searching it in the knowledge base and then showing the output. This process will take some time, which is estimated to be 4 seconds approximately. If the user has a bad internet connection, it will take some more time for him to get the output. But even in the worst case, the response time will not exceed 15 seconds

**9. REFERENCES**

1. S.R. Bharamagoudar, Geeta R.B. and S.G Totad,- Web Based Student Information Management System, Andhra Pradesh, vol. 2, June 2013
2. IEEE - Android Based Smart Learning and Attendance Management System, June 2015, 4655
3. http://students.iitk.ac.in/eclub/assets/documentations/summer12/cha t-bot.pdf
4. Vishal Bhalla, Tapodhan Singla, AnkitGahlot and Vijay Gupta, “Bluetooth Based Attendence Management System”, International Journal of Innovations in Engineering and Technology (IJIET) Vol.3 Issue 1 October 2013, ISSN:2319-1058.
5. Saurabh Walia and Satinderjit kaur Gill, -A Framework for Web Based Student Record Management System using PHP, Himachal Pradesh, vol. 3, August 2014.
6. C. I. Nass, and S. Brave, Wired for speech: How voice activates and advances the human-computer relationship: MIT Press Cambridge, 2005.
7. Y.-P. Yang, “An Innovative Distributed Speech Recognition Platform for Portable, Personalized and Humanized Wireless Devices,” Computational Linguistics and Chinese Language Processing, vol. 9, no. 2, pp. 77-94, 2004.
8. J. P. Campbell Jr, “Speaker recognition: a tutorial,” Proceedings of the IEEE, vol. 85, no. 9, pp. 1437-1462, 1997.
9. C.-H. Lee, "From knowledge-ignorant to knowledge-rich modeling: a new speech research paradigm for next generation automatic speech recognition", 2004.
10. V. Bhargava, and N. Maheshwari, “An Intelligent Speech Recognition System for Education System,” 2009.
11. E. Loper, and S. Bird, "NLTK: The natural language toolkit." pp. 63-70, 2002.
12. S. Bird, "NLTK: the natural language toolkit." pp. 69-72, 2006.