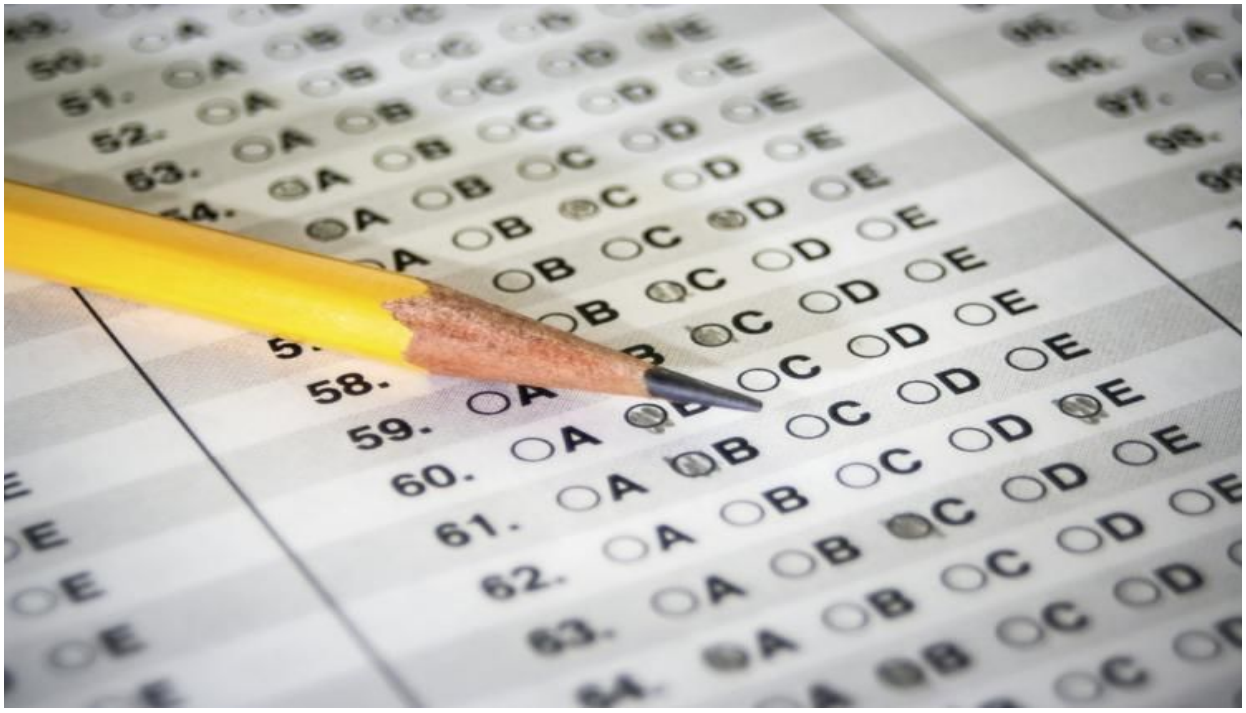


Automatic Question Paper Generator

A Java WebApp to generate Question Paper on a click !

ARTIFICIAL INTELLIGENCE PROJECT REPORT 2017

Submitted *to :*

Prof Poonam Bedi

By:

Deepti Sharma(09)

Sandhya(32)

Shivani Tiwary(36)

Surbhi Chaurasia(42)

M.SC COMPUTER SCIENCE

DUCS, DELHI UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

DELHI UNIVERSITY

NEW DELHI-110007

NOVEMBER 2017

DECLARATION

This is to certify that the Project Report entitled

"AUTOMATIC QUESTION PAPER GENERATOR"

is being submitted to the Department of Computer Science, Delhi University, New Delhi for the course "Artificial Intelligence" in **Master of Science** is a record of bonafide work carried out by us under the supervision of **Prof. Poonam Bedi.** The matter embodied in the declaration has not been submitted in part or full to any university.

DEEPTI SHARMA(09)

SANDHYA(32)

SHIVANI TIWARY(36)

SURBHI CHAURASIA(42)

DEPARTMENT OF COMPUTER SCIENCE

DELHI UNIVERSITY

NEW DELHI-110007

CERTIFICATE

This is to certify that this Project Report entitled "Automatic Question Paper Generator" submitted by **Deepti Sharma , Sandhya , Shivani Tiwary and Surbhi Chaurasia** to the **Department of Computer Science, Delhi University, New Delhi** in partial fulfillment of the requirement for the course "Artificial Intelligence" in **Master of Science** is a project report carried out by them under the supervision of

Prof. Poonam Bedi

To the best of my knowledge this work has not been submitted in part or full to any other University or Institution for the award of any degree or diploma.

Delhi University
New Delhi, India

Dr Poonam Bedi
Professor

ACKNOWLEDGMENT

It is our great pleasure to express our gratitude to all the people who have supported us greatly during this project.

First of all, We would like to give special and sincere thanks to our teacher , **Prof. Poonam Bedi ,** who has provided us with the best of her knowledge and experience and has encouraged us to delve into such a project and learning activity. Her untiring and diligent effort, methodical approach made it possible for us to complete this work on time.

We would like to extend our appreciation to Department of Computer Sciences, Delhi University, New Delhi for motivating us towards project development and providing us with all necessary resources to accomplish the project on time.

As a team , we all are thankful to each other , and our Seniors for guiding us and understanding our needs.

This acknowledgement is incomplete without mentioning our parents and god who have enabled us to reach this place, we are deeply obliged to all the people who helped us and took the patience and time to encouragement to face challenges.

Deepti Sharma(09)
Sandhya(32)
Shivani Tiwary(36)
Surbhi Chaurasia(42)

TABLE OF CONTENT

ABSTRACT

Education being the most growing sector and the most versatile one as well; it most requires a technological boom. Everyday new concepts and ideas are released in the market; the learners not only need to learn them but practice them as well.

A system like AQPG allows the learner to apply the knowledge (s)he has gained so far.

Our system divides users in three categories-

1) Admin
2) Instructor
3) Learner

The admin has been provided with the functionality of adding context to database; and approving the changes required by instructor or learner in a question. (S)he has the authority to create user and dismiss upon request.

Our system generates random questions from the database of the particular subjects; opted by the learner; and judging on the capability of learner; the level is increased. The system creates performance charts to allow the learner to feel competitive and be able to judge which topic he/she needs to practice more.

Realizing the fact that question paper creation is a tedious and time consuming task; our system adds the functionality for the instructor to generate and save the question paper automatically on a click with required preferences. There are other added features for correction and addition of question to database on granted permission by admin.

PROCESS MODEL

A (software/system) process model is a description of the sequence of activities carried out in a project, and the relative order of these activities. There are hundreds of different process models to choose from, e.g.:

- ***Waterfall***
- ***Spiral***
- ***Rapid Prototyping***
- ***Agile methods,***

But most are minor variations on a small number of basic models.

By changing the process model, we can improve and/or tradeoff:

1. **Development speed (time to market)**
2. **Product quality**
3. **Project visibility**
4. **Administrative overhead**
5. **Risk exposure**
6. **Customer relations etc.**

The Three Main Phases:

• design,

• build,

• maintain. (50% of IT activity goes here!)

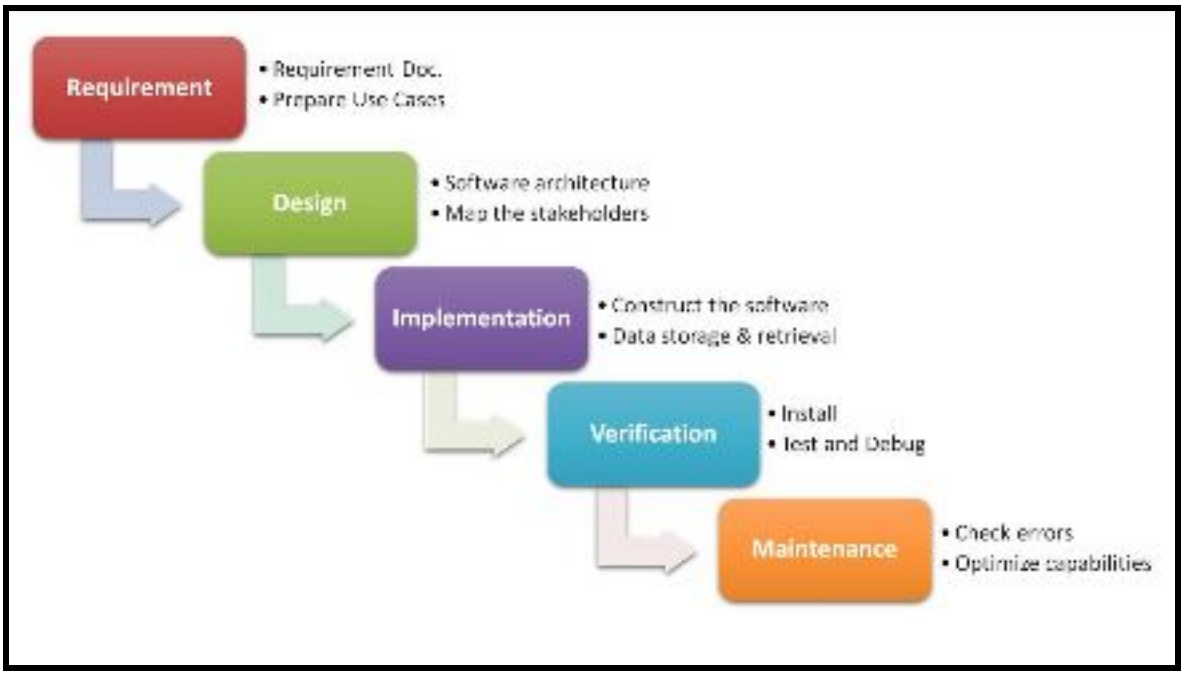
We have opted for the **Waterfall Model.**

The Waterfall Model

- The waterfall model is the classic process model – it is widely known, understood and used.
- In some respect, waterfall is the ”common sense” approach.
- Since the product we are working on is a simple web application which needn’t be delivered incrementally. We opted out for the most easy process model for developing the product. We can simply go from one stage to another after a document was signed by the client.

Advantages:

1. Easy to understand and implement.
2. Identifies deliverables and milestones
3. Document driven: People leave, documents don’t
4. Works well on large/mature products and weak teams



SOFTWARE REQUIREMENT SPECIFICATION

OVERALL DESCRIPTION:

1. Product Functions:

- Adding and Deleting Users, Courses, Questions and related information into database.
- Modifying the same three entities by the Admin.
- Generating Question Paper on a Click By Instructor.
- Practice Sessions for Learner.
- Viewing the profile and editing content by the three type of Users.
- Other expected functions like verifying the user upon login password entry, etc. are there.

2. User Characteristics

- There are three types of users :
  - Instructor
  - Admin
  - Learner

- INSTRUCTOR :
  - should have the knowledge of the course for which he/she exclaims.

- Should have simple knowledge of computer software like clicking buttons and form entry to utilize the software.

- ADMIN:
  - Admin has the privilege to ADD, DELETE and to MODIFY the questions.
  - It is expected that admin would have little more knowledge than the other users about the software functionality to utilize his/her power of manipulating database.
  - He/she should have minimal/ vague data design idea.

- LEARNER :
  - A learner should have basic knowledge of computers of how things work like keyboard entry.

3. General Constraints

- Unauthorised access should be completely denied.
- Only validated users should be allowed to access the system.
- System would use only English Language.
- The database of questions is constrained by the number of questions , which can only increase if the admin increase the size of the database.

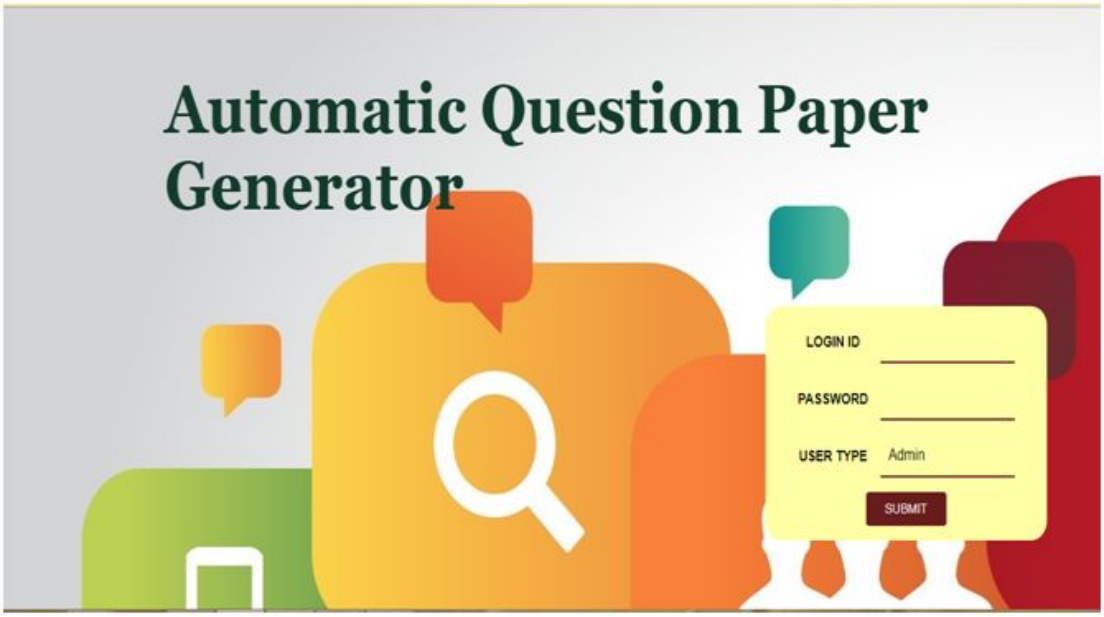
4. Assumptions and Dependencies

- The data base can be populated by Admin only.
- No signup functionality is there , so we assume that each user who wants to use the system knows some admin , whom he/ she might contact for getting the password.
- The software depends the Database File : QUESPPR.

EXTERNAL INTERFACE REQUIREMENTS:

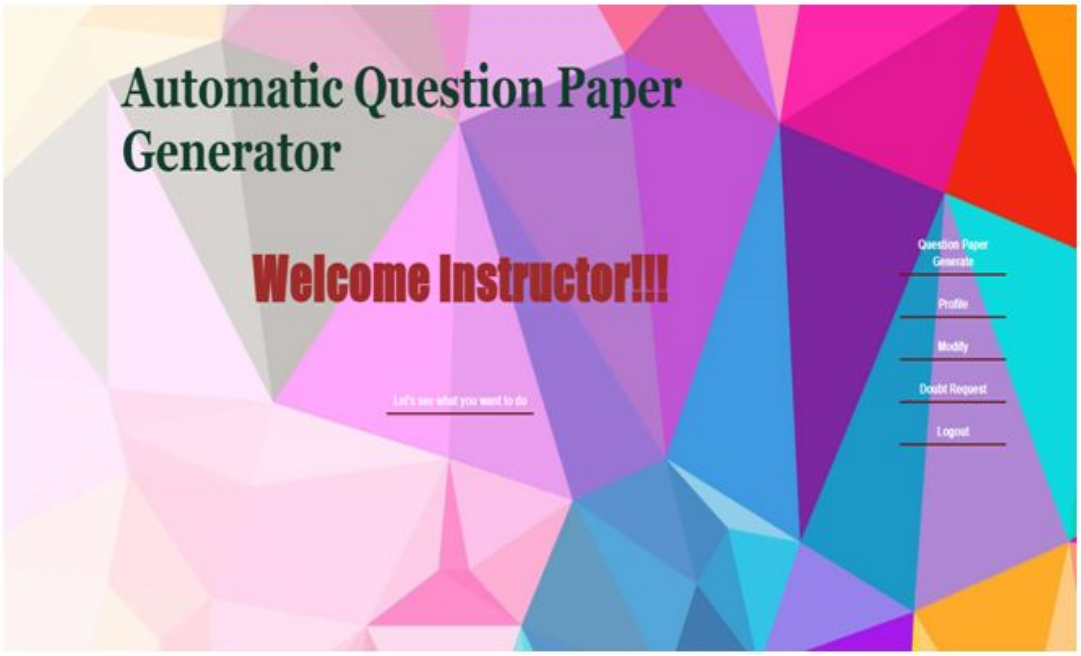
1. User Interface

Screen 1: Login Page

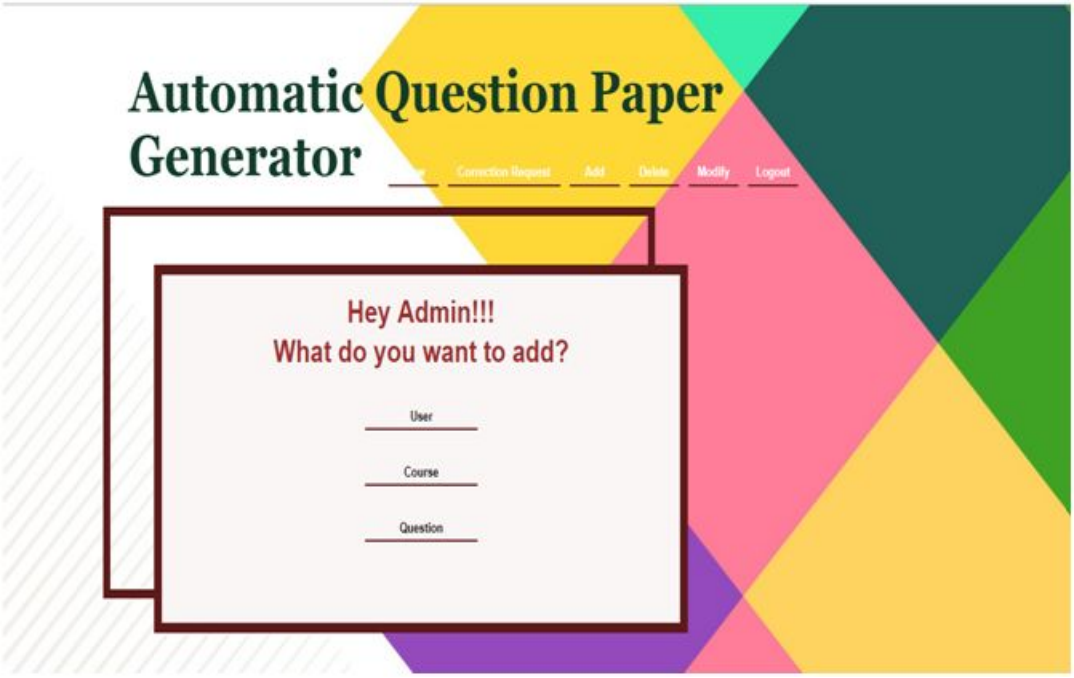


Screen 2: Admin Login Page

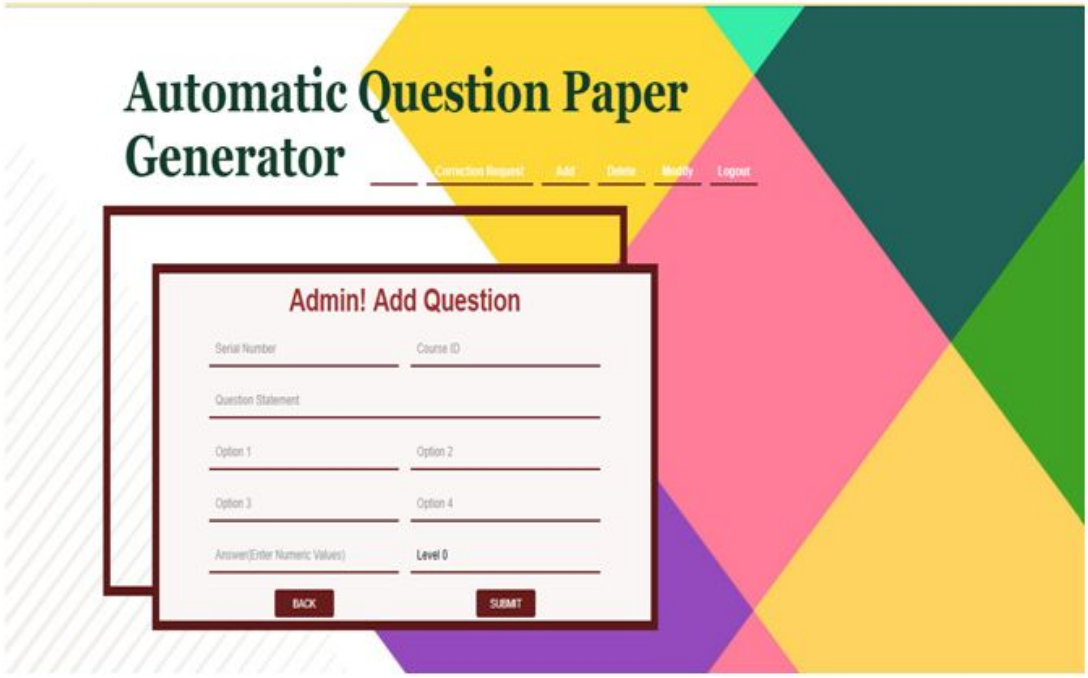
Screen 3: Instructor Login Page



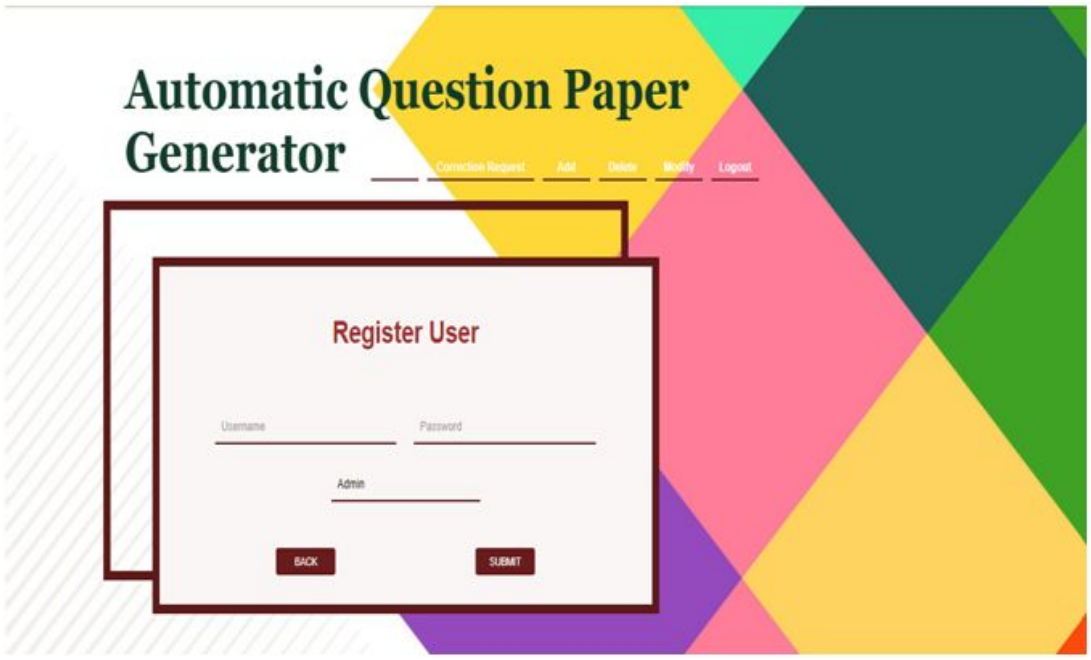
Screen 4: Admin Add Functionality Page



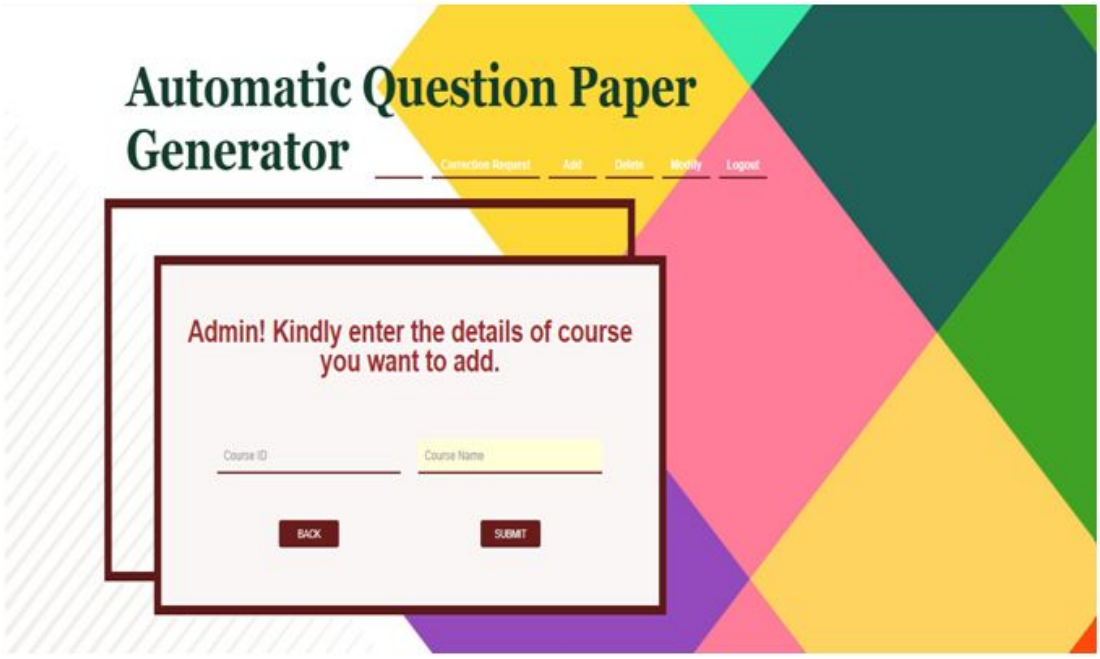
Screen 5: Admin Add Question Page



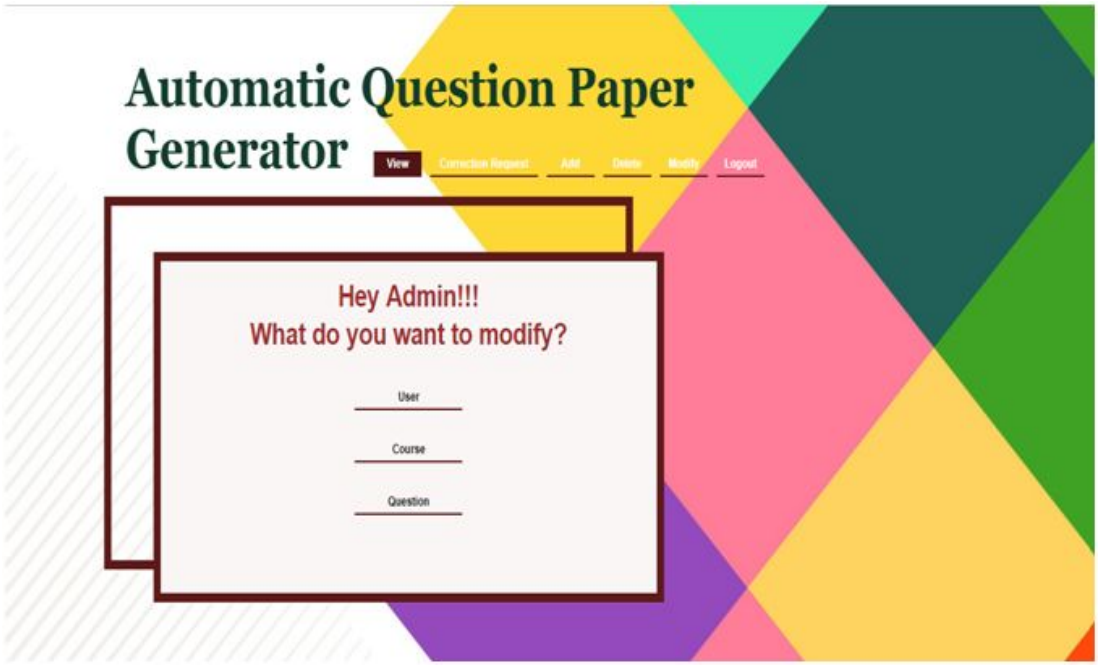
Screen 6: Admin Add User Page



Screen 7: Admin Add Courses Page



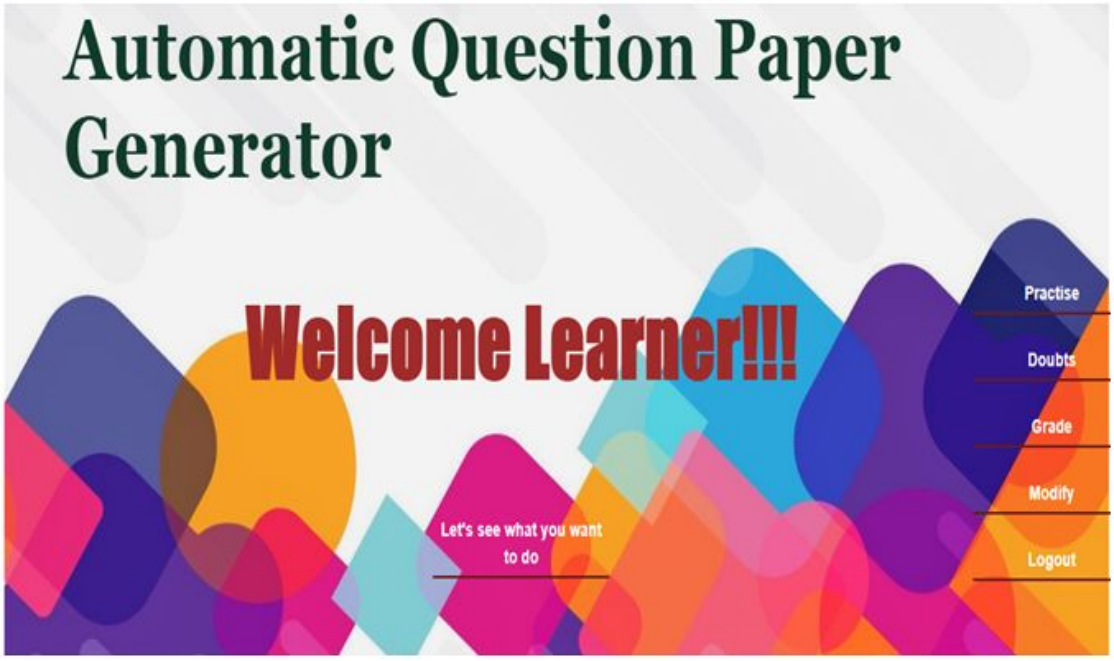
Screen 8: Admin Modify Functionality Page



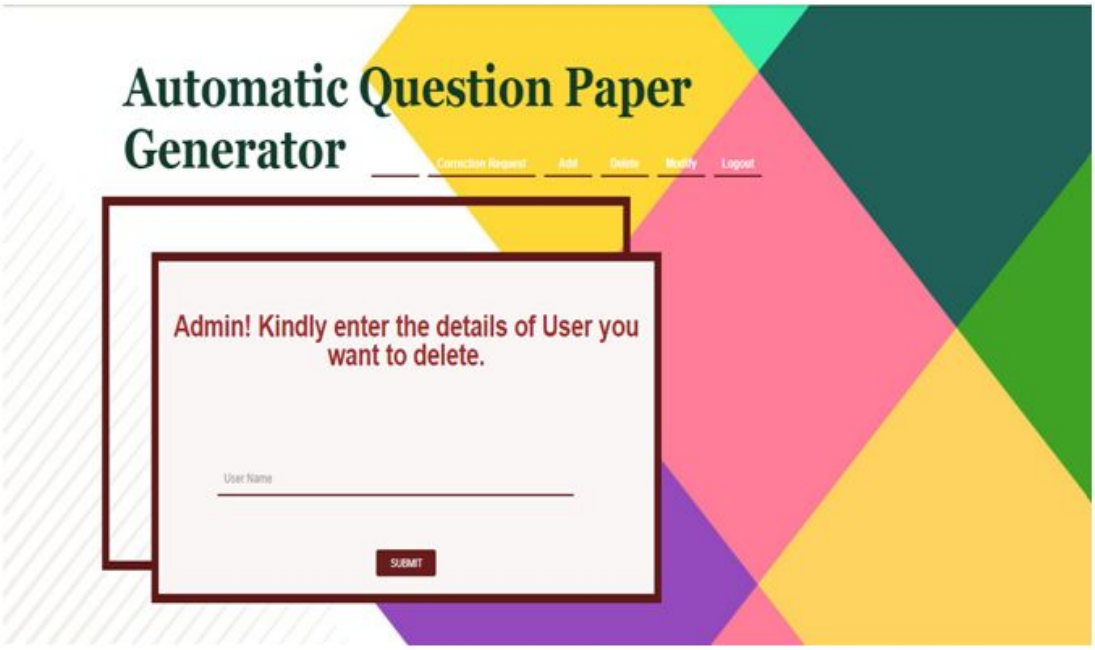
Screen 9: Admin Delete Functionality Page

Screen 10: Admin Delete Question Page

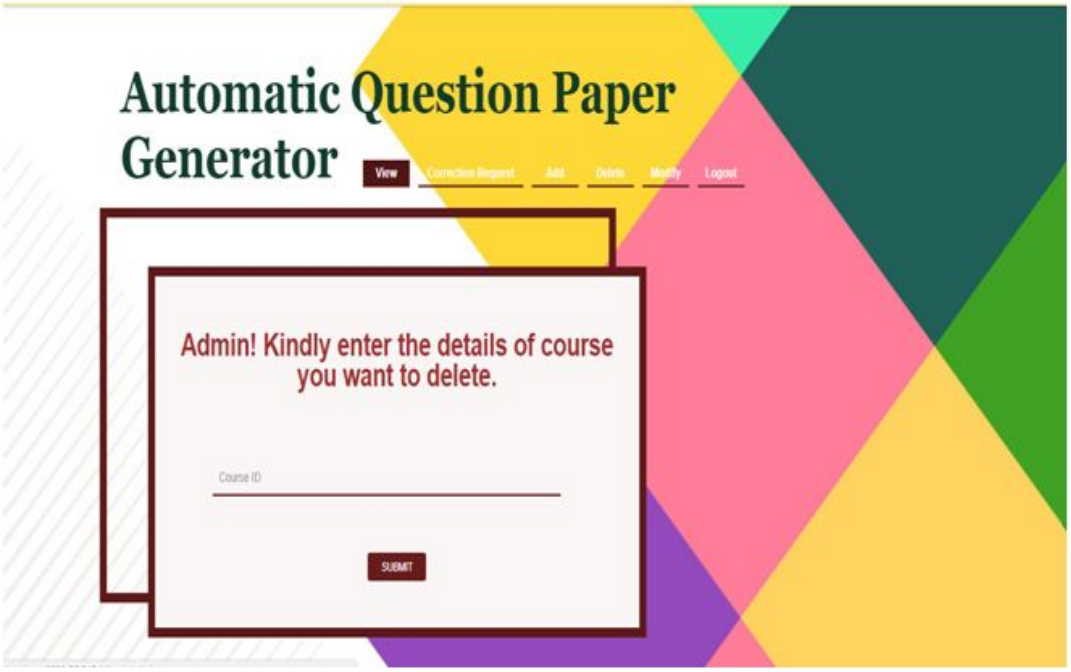
Screen 11: Learner's Login Page



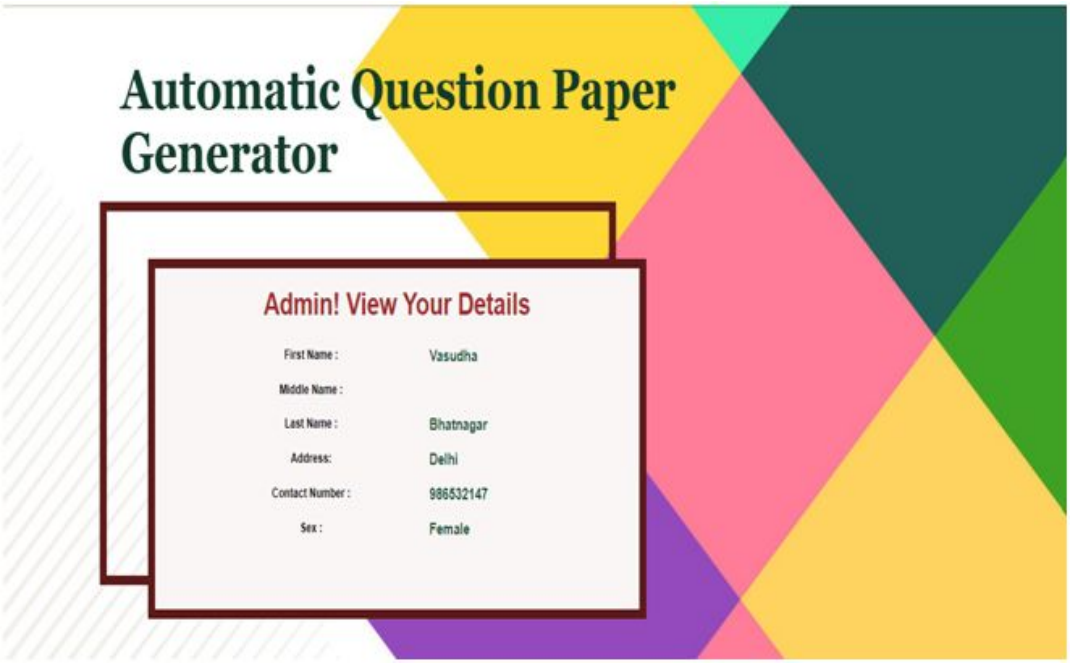
Screen 10: Admin Delete User Page



Screen 11: Admin Delete Course Page



Screen 12: Admin View Profile Page



2. Hardware Interface

- We require a nominal display; high deficiency screen is not necessary.
- Also internet connection is required to allow the software to use the database located at the server.

3. Software Interface

- We require a special database known as “QUESTION PAPER INFO” which is available to the application for allowing the various to carry out their functions smoothly.
- The database is the crux of this software , larger the database , larger is the variation of question that strikes a user at time , hence giving excellent practice sessions.

FUNCTIONAL REQUIREMENTS:

1. *Fr 1: LoginDao.Validate()*

Input : Login Bean Object

Login Bean stores the

1) Login Name
2) User Type
3) Password

Output : Integer Status (0/1)

Processing :

The Validate() function creates a connection to the database and retrieves the data record where the Login name and password matches the ones that has been provided by the user , if they are valid then obviously 1 record will be fetched from the database and hence the status will be 1 , else the status would be 0.

2. *Fr 2: AddCourseDao.Insertion()*

Input : Course Bean Object

Course Bean stores the

1) Course Id
2) Course Name

Output : Integer Status(0, 1, -1)

Processing :

The Insertion() function takes the course bean and queries the database for inserting the given values in the Course bean using the prepared statement , if everything goes right , then the status is returned 1 , else a failed database insertion is reported via 0 status and exceptions are thrown .

3. *Fr 3: AddQuestionDao.Insertion()*

Input : Question Bean Object

Question bean stores the

1) Serial Number
2) Course Id
3) Question Statement
4) Option 1
5) Option 2
6) Option 3
7) Option 4
8) Answer
9) Level

Output : Integer Status(0, 1, -1)

Processing :

The Insertion() function takes the question bean and queries the database for inserting the given values in the Question bean using the prepared statement , if everything goes right , then the status is returned 1 , else a failed database insertion is reported via 0 status and exceptions are thrown .

4. *Fr 4: RegisterUserDao.Insertion()*

Input : Login Bean Object

Login Bean stores the

1) Login Name
2) User Type
3) Password

Output : Integer Status (0, 1, -1)

Processing :

The Insertion() function takes the Login bean and queries the database for inserting the given values in then Login bean using the prepared statement , if everything goes right , then the status is returned 1 , else a failed database insertion is reported via 0 status and exceptions are thrown .

5. *Fr 5: DeleteCourseDao.Deletion()*

Input : Course Bean Object

Course Bean stores the

1) Course Id
2) Course Name

Output : Integer Status(0, 1, -1)

Processing :

The Deletion() function takes the course bean and queries the database for deleting the given value of Course id in the Course bean using the prepared statement , if everything goes right , then the status is returned 1 , else a failed database insertion is reported via 0 status and exceptions are thrown .

6. *Fr 6: DeleteQuestionDao.Deletion()*

Input : Question Bean Object

Question bean stores the

1) Serial Number
2) Course Id
3) Question Statement
4) Option 1
5) Option 2
6) Option 3
7) Option 4
8) Answer
9) Level

Output : Integer Status(0, 1, -1)

Processing :

The Deletion() function takes the question bean and queries the database for deleting the given value of Serial Number in the Question bean using the prepared statement , if everything goes right , then the status is returned 1 , else a failed database insertion is reported via 0 status and exceptions are thrown .

7. *Fr 7: DeleteUserDao.Deletion()*

Input : Login Bean Object

Login Bean stores the

1) Login Name
2) User Type
3) Password

Output : Integer Status (0, 1, -1)

Processing :

The Deletion() function takes the Login bean and queries the database for deleting the given value of Login Name in the Login bean using the prepared statement , if everything goes right , then the status is returned 1 , else a failed database insertion is reported via 0 status and exceptions are thrown .

8. *Fr 8: QuestionDao.RetrieveQuestion()*

Input : ArrayList<String>

An ArrayList of Strings containing the Course Id of all the Courses selected by the Instructor for generating the Paper.

Output : ArrayList<Question Bean>

An array list of the questions retrieved along with the Attributes.

Processing :

The retrieveQuestion() function retrieves all the questions of all the levels from the database and create a list of them for generating question paper in our ProcessChoice.jsp . Only questions of those subjects appear which were selected by the Instructor.

9. Fr 9: SubjectDao.displaySubject()

Input : None
Output : ArrayList<Subject Bean>
An array list of the Subjects available in the system.

Processing :
The displaySubject() function retrieves all the records of the Course Table in database and form an array list by reading record by record all the subjects available in the system so that when the learner wants to choose the subjects for the practice session , we can show him /her the available choices and also for the instructor to generate the paper the choices can be displayed via the database.

10. Fr 10:ViewDao.Retrieve()

Input : View Bean Object
View Bean stores the
1) Login Name
2) Type
3) First Name
4) Second Name
5) Last Name
6) Address
7) Contact Number
8) Birthdate
9) Sex

Output : Integer Status

Processing :
The retrieve() function is used to retrieve the information of the user from the database to display on the profile page using the login name which is already present in the view bean inputted to the function from the session attribute and all the other attributes are set using the login name. Success is indicated by 1.

PERFORMANCE REQUIREMENTS:

Static:

- The application has a limit on the current users.
- There is only one major database file “QUESPPR” which contains all the information about all the users and courses and questions.

Dynamic:

- Typically the application can respond within 1-5s after processing the requested query of the user.
- In case the user request the download of the file , it take more time when more number of questions are requested. Maximum 10 s.
- Throughput is dependent on the terminal and the database also, expecting a nominal system, we can state the throughput after multiple test only.
- The database usually reponses back within 0.8 ms at max. For smaller queries and not more than 1 second for larger accesses.

DESIGN CONSTRAINTS:

Standard Compliance:

The system on which the application is installed requires no standards to be followed.

Hardware limitations:

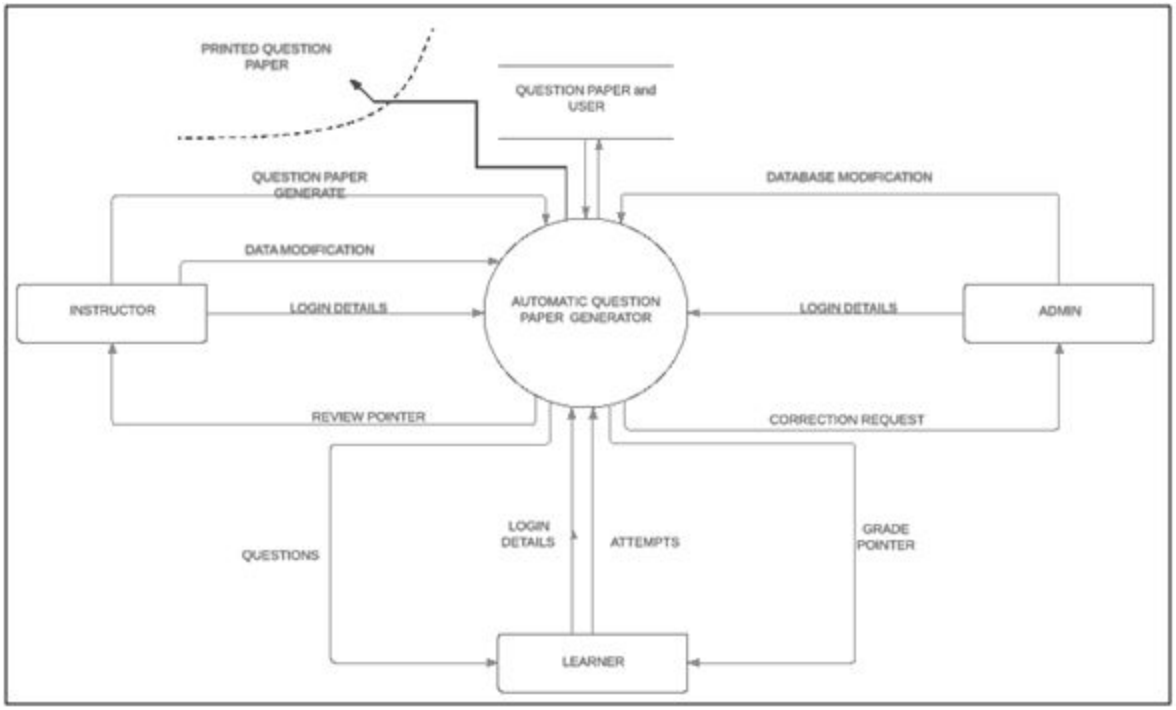
The software alone is efficient no special hardware support required.

Reliability and Fault Tolerance:

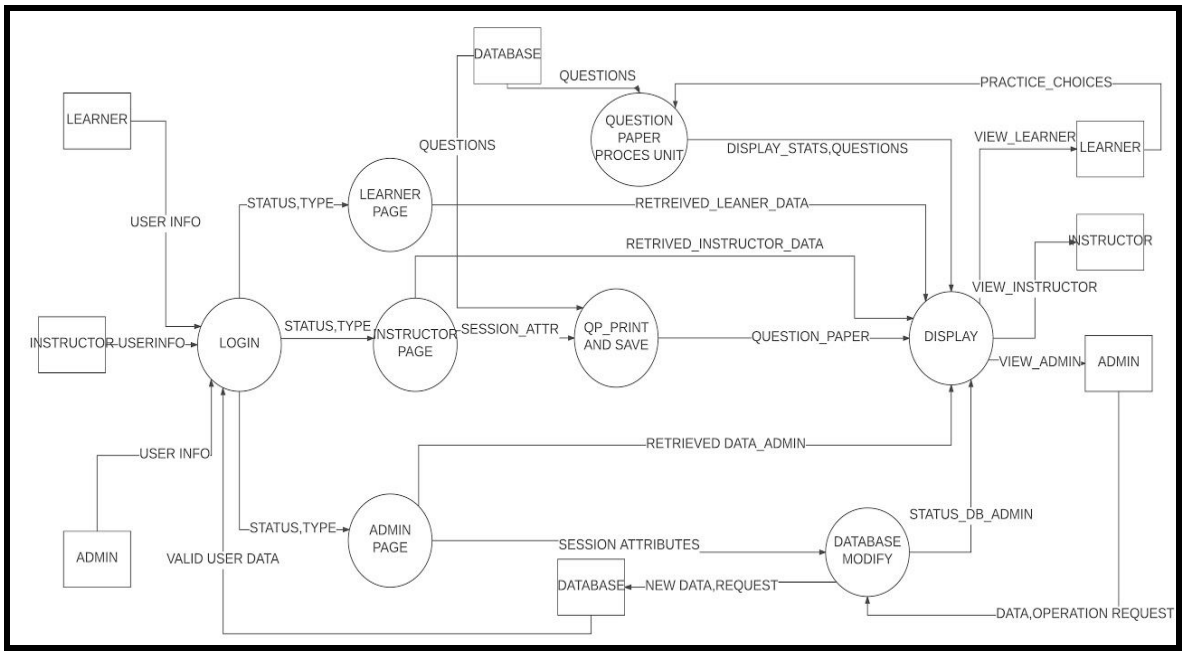
It is a reliable and fault tolerating software.

DATA FLOW DIAGRAM:

1. *Context Level Diagram / Level 0:*


PRINTED QUESTION PAPER
QUESTION PAPER and USER
QUESTION PAPER GENERATE
DATABASE MODIFICATION
DATA MODIFICATION
INSTRUCTOR
LOGIN DETAILS
AUTOMATIC QUESTION PAPER GENERATOR
LOGIN DETAILS
ADMIN
REVIEW POINTER
CORRECTION REQUEST
QUESTIONS
LOGIN DETAILS
ATTEMPTS
GRADE POINTER
LEARNER

2. *Level 1 Dfd :*


DATABASE
QUESTIONS
PRACTICE_CHOICES
LEARNER
QUESTION PAPER PROCES UNIT
QUESTIONS
DISPLAY_STATS,QUESTIONS
VIEW_LEARNER
LEARNER
STATUS,TYPE
LEARNER PAGE
RETREIVED_LEANER_DATA
USER INFO
RETRIVED_INSTRUCTOR_DATA
INSTRUCTOR
VIEW_INSTRUCTOR
INSTRUCTOR-USERINFO
LOGIN
STATUS,TYPE
INSTRUCTOR PAGE
SESSION_ATTR
QP_PRINT AND SAVE
QUESTION_PAPER
DISPLAY
VIEW_ADMIN
ADMIN
USER INFO
RETRIEVED DATA_ADMIN
STATUS_DB_ADMIN
STATUS,TYPE
ADMIN PAGE
SESSION ATTRIBUTES
DATABASE MODIFY
ADMIN
VALID USER DATA
DATABASE
NEW DATA,REQUEST
DATA,OPERATION REQUEST

DATA DICTIONARY :

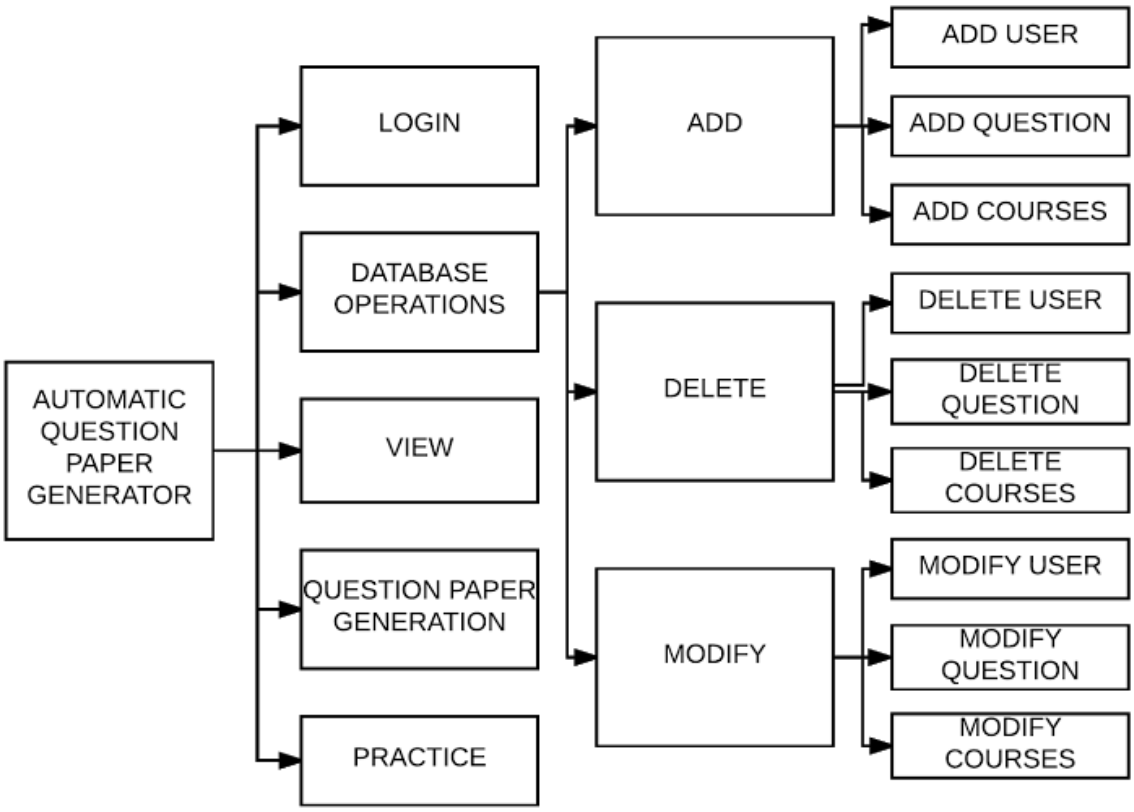
LEVEL 1 DFD:

DATA	DESCRIPTION
USER INFO	**Login Name :** a unique Identification ; *Varchar type* **Password :** secured password key for authentication ; *password type*
STATUS	Indicating a successful operation , *Boolean Value(0/1)*
TYPE	Indicating type of user, *Char type("A","I","L")*
RETRIEVED_LEARNER_DATA	Indicating retrieved data from database, *Array of Learner types (Varchar, Integer)*
PRACTICE_CHOICES	**Subjects :** a list of subjects the student wants to practice; *Varchar Type*
QUESTIONS	*Arraylist* of all the questions.
DISPLAY_STATS	**Count:** number of questions answered correctly *Integer*
VIEW_LEARNER	**Learner Data:** data retrieved from the database, *Multiple Type*
SESSION_ATTR	**Login Name :** since the login name is unique to let us know which user is running the session. *Varchar Type*

RETRIEVED_INSTRUCTOR_DATA	Indicating retrieved data from database, *Array of Learner types (Varchar, Integer)*
QUESTION_PAPER	*Arraylist* of Questions.
VIEW_INSTRUCTOR	**Instructor Data:** data retrieved from the database, *Multiple Type*
RETRIEVED_DATA_ADMIN	Indicating retrieved data from database, *Array of Learner types (Varchar, Integer)*
STATUS_DB_ADMIN	Indicating a successful operation , *Boolean Value(0/1)*
NEW_DATA	Data Entered by Admin in form for database updation, addition. *Multiple Type*
DATA	Data Entered by Admin in form for database updation, addition and deletion *Multiple Type*
OPERATION_REQUEST	Deletion Query
VIEW_ADMIN	**Admin Data:** data retrieved from the database, *Multiple Type*

DESIGN :

1. *System Design*



2. *Data Design*

The database stores information about USERS, COURSES and QUESTIONS.

I. Users have a unique ID associated with them. The Database keeps track of their Login Name, First Name, Second Name, Password, Birth date, Sex, Address, and Contact Number.

II. The system has three possible types of user: Instructors, Learners (from any age group) and Admin.

III. Instructor has certain special attributes given as: Qualification, Specialization, Contribution to Question Paper, Review Pointer.

IV. While Learners have special properties like Degree, Grade Pointer, Attempts, Corrections Reported, and Level (defined in the system: beginner, expert).

V. Admin is a special type of privileged who can add questions to the questions table.

VI. Each and Every Instructor teaches many courses/subjects and each course can be taught by many instructors.

VII. Each Course has a special Id and Name. Each Course added in the system's database has at least one question of its own.

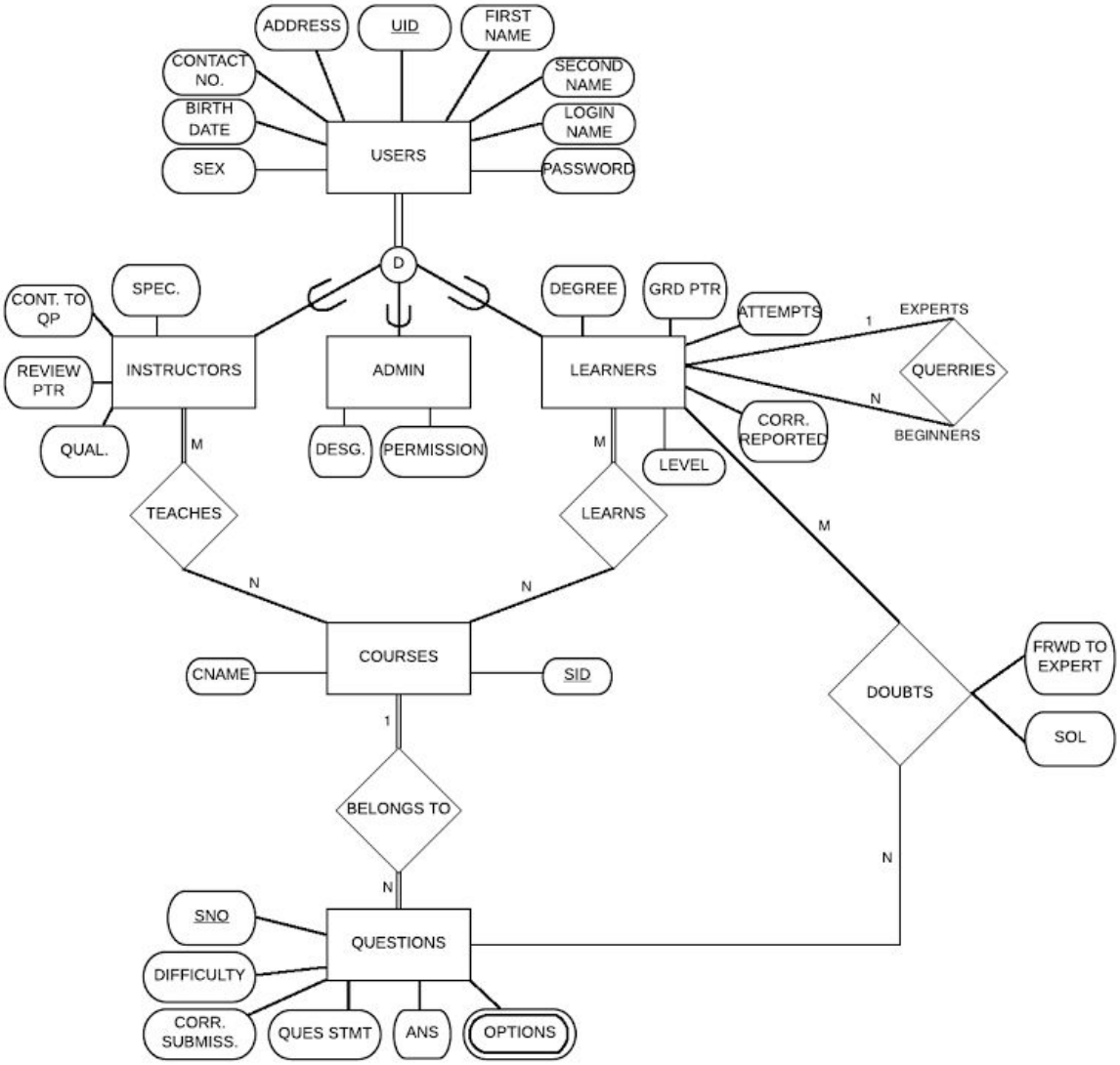
VIII. Questions are stored with the following properties saved: Unique serial number,, Question Statement, Answer, Four Options, Correct Submissions, Difficulty Level(Easy , Difficult, Medium).

IX. Each Question belongs to a particular course only.

X. A learner can learn as many courses as (s) he wants.

XI. Each "Beginner" learner is assigned an "expert" for help.

XII. A learner can mark questions and forward doubts to expert. Learners doubt certain questions. A question can be doubted by many Learners


ADDRESS
UID
FIRST NAME
CONTACT NO.
SECOND NAME
BIRTH DATE
LOGIN NAME
SEX
USERS
PASSWORD
D
SPEC.
CONT. TO QP
REVIEW PTR
INSTRUCTORS
QUAL.
ADMIN
DESG.
PERMISSION
DEGREE
GRD PTR
ATTEMPTS
LEARNERS
LEVEL
CORR. REPORTED
EXPERTS
QUERRIES
BEGINNERS
M
N
1
TEACHES
LEARNS
COURSES
CNAME
SID
DOUBTS
FRWD TO EXPERT
SOL
BELONGS TO
SNO
DIFFICULTY
CORR. SUBMISS.
QUES STMT
ANS
OPTIONS
QUESTIONS

CONCLUSION AND FUTURE DEVELOPMENT

The project we have developed so far can be easily extended to many ideas and can keep on growing. Some changes that we plan to make are :

1. **Increase the levels of question difficulty and allowing the teacher to replace a question from the given one.**
2. **Allowing chat between learner and an expert to sought out doubts.**
3. **Allowing more choices for the learner to practice his own way**
4. **Allowing String search using NLP for deleting a question by Admin.**
5. **Allowing the questions to not get repeated for the user.**
6. **Graphing the daily practice and concentration of the student just like the Github Graph of Contribution.**
7. **Graphing the Instructors capability to contribute corrections.**

And many More....