**A Mini Project Report**

**The Question Papers Management**

Submitted in partial fulfilment for the requirements of

**B.E. (CSE) VI Semester Mini Project**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

**by**

**Gandhari Manideep Kumar (160117733158)**

**Mamidala Kruthika (160117733009)**

**Sai Chandra Pothuri (160118733167)**

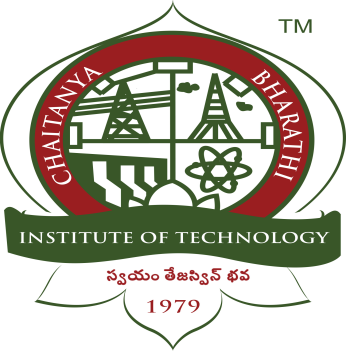
**Harish Akula (160118733096)**

**Aishwarya Jaini (160118733121)**

**V. Rahul (160119737165)**

**Juwairiyyah (160119733129)**

**Akkamgari Shiva Teja Goud (160119737052)**



**Department of Computer Science and Engineering,**

**Chaitanya Bharathi Institute of Technology (Autonomous),**

**(Affiliated to Osmania University, Hyderabad)**

**Hyderabad, TELANGANA (INDIA) – 500 075**

**June-2020**

**CERTIFICATE**

This is to certify that the project titled “**Question papers management”** is the bonafide work carried out by **Harish Akula (160118733096)** a student of B.E.(CSE) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2019-2020, submitted in partial fulfillment of the requirements for the **COSC Internship** Mini Projects and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

|  |  |
| --- | --- |
| **Mentor(s)**  **Smt I. Srujana  (Assistant Professor)** | **Head, CSE Dept.**  **Dr.Y. Ramadevi**  **(Professor and Head)** |
| **Place:**  **Date:** | |

|  |
| --- |
|  |

**CERTIFICATE**

This is to certify that the project titled “**Question papers management”** is the bonafide work carried out by **Aishwarya Jaini(160118733121)**, a student of B.E.(CSE) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2019-2020, submitted in partial fulfillment of the requirements for the **COSC Internship** Mini Projects and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

|  |  |
| --- | --- |
| **Mentor(s)**  **Dr. Sagar**  **(Professor)** | **Head, CSE Dept.**  **Dr.Y. Ramadevi**  **(Professor and Head** |
| **Place:**  **Date:** | |

**CERTIFICATE**

This is to certify that the project titled “**Question papers management”** is the bonafide work carried out by **V. Rahul (160119737165)**, a student of B.E.(I.T) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2019-2020, submitted in partial fulfillment of the requirements for the **COSC Internship** Mini Projects and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

|  |  |
| --- | --- |
| **Mentor(s)**  **Swathi somya mohan**  **(Professor)** | **Head, CSE Dept.**  **<Name & Designation>** |
| **Place:**  **Date:** | |

**CERTIFICATE**

This is to certify that the project titled “**Question papers management”** is the bonafide work carried out by **Gandhari Manideep Kumar(160117733158)**, a student of B.E.(CSE) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2019-2020, submitted in partial fulfillment of the requirements for the **COSC Internship** Mini Projects and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

|  |  |
| --- | --- |
| **Mentor(s)**  **Smt.G.Vanitha**  **- Assistant Professor,**  **Dept of CSE.** | **Head, CSE Dept.**  **Dr.Y. Ramadevi**  **(Professor and Head)** |
| **Place:**  **Date:** | |
|  | |

**CERTIFICATE**

This is to certify that the project titled “**Question papers management”** is the bonafide work carried out by **Sai Chandra Pothuri(160118733167)**, a student of B.E.(CSE) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2019-2020, submitted in partial fulfillment of the requirements for the **COSC Internship** Mini Projects and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

|  |  |
| --- | --- |
| **Mentor(s)**  **<Name and Designation>** | **Head, CSE Dept.**  **<Name & Designation>** |
| **Place:**  **Date:** | |
|  | |

**CERTIFICATE**

This is to certify that the project titled “**Question papers management”** is the bonafide work carried out by **Mamidala Kruthika(160117733009)**, a student of B.E.(CSE) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2019-2020, submitted in partial fulfillment of the requirements for the **COSC Internship** Mini Projects and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

|  |  |
| --- | --- |
| **Mentor(s)**  **Smt.Kavita Agarwal**  **(Assistant Professor)** | **Head, CSE Dept.**  **Dr.Y. Ramadevi**  **(Professor and Head)** |
| **Place:**  **Date:** | |

|  |
| --- |
|  |

**CERTIFICATE**

This is to certify that the project titled “**Question papers management”** is the bonafide work carried out by **Juwairiyyah (160119733129)**, a student of B.E.(CSE) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2019-2020, submitted in partial fulfillment of the requirements for the **COSC Internship** Mini Projects and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

|  |  |
| --- | --- |
| **Mentor(s)**  **Sri.A.Mohan (Assistant Professor)** | **Head, CSE Dept.**  **Dr.Y. Ramadevi**  **(Professor and Head)** |

|  |
| --- |
| **Place:**  **Date:** |
|  |

**CERTIFICATE**

This is to certify that the project titled “**Question papers management”** is the bonafide work carried out by **<name and roll number in bold face>** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, a student of B.E.(CSE) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2019-2020, submitted in partial fulfillment of the requirements for the **COSC Internship** Mini Projects and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

|  |  |
| --- | --- |
| **Mentor(s)**  **<Name and Designation>** | **Head, CSE Dept.**  **<Name & Designation>** |
| **Place:**  **Date:** | |
|  | |

**DECLARATION**

We hereby declare that the project entitled “Question Papers Management” is submitted for the ‘COSC Internship drive’ Mini Projects is our original work and the project has not formed the basis for the award of any other degree, diploma, fellowship or any other similar titles.

**Name(s) and Signature(s) of the Student**

Place:

Date:

**ABSTRACT**

There has always been a problem of having to obtain papers of the past right before exams. Sometimes we don’t get the right paper and sometimes we don’t get papers at all and its time consuming. The ‘Question Papers Management’ utility is a one stop solution for this problem. This utility provides with an application for the students to upload question papers once the exam is done and the Admin has to approve the desired question paper or the Admin can upload question paper himself. The Admin feature is available in a web application and student facilities are provided by an Android mobile application. This utility helps the students to improve their learning, saves their time. This utility also helps faculty to go through previous set of papers for understanding how papers are set over the time.

**ACKNOWLEDGEMENT**

Firstly, we are grateful to COSC for giving us this opportunity to do an internship amidst this period of lockdown. We thank Chaitanya Bharathi Institute of Technology for providing us the scope and supporting our internship. We are thankful to our faculty mentor Sri K. Rajesh Kannan (Assistant Professor, Dept. of IT - CBIT) for providing us his valuable assistance for doing this project and recommending us some features. We are grateful from the bottom of our hearts to our mentor Prithvi M (B.E – ¾ CSE CBIT) for being one of us while doing the project and for his valuable guidance and support all the times to complete the project. Finally, we thank each of our team mates for working hard for all the days and completing this project successfully.





**Table of Contents**

[**The Question Papers Management** (i)](#_Toc44691024)

[**CERTIFICATE(s)** (ii)](#_Toc44691025)

[**DECLARATION** (iii)](#_Toc44691082)

[**ABSTRACT** (iv)](#_Toc44691086)

[**ACKNOWLEDGEMENT** (v)](#_Toc44691087)

[Table of Figures (vi)](#_Toc44691088)

[**Chapter 1. Problem definition** 1](#_Toc44691091)

[Section 1.1. Problem statement 1](#_Toc44691092)

[Section 1.2. Significance and objective 1](#_Toc44691093)

[Section 1.3. Organization 2](#_Toc44691094)

[Chapter 2. **Literature Survey** 3](#_Toc44691095)

[Section 2.1. Existing solutions 3](#_Toc44691096)

[Section 2.2. Related works 4](#_Toc44691097)

[Section 2.3. Tools and technologies used 5](#_Toc44691098)

[**Chapter 3. Design of the proposed system** 7](#_Toc44691099)

[Section 3.1. Block diagram 7](#_Toc44691100)

[Section 3.2. Module description 8](#_Toc44691103)

[**Chapter 4. Implementation of the proposed system** 10](#_Toc44691104)

[Section 4.1. ER diagram of the database 10](#_Toc44691105)

[Section 4.2. Database schema explanation with examples. 11](#_Toc44691107)

[Section 4.3. The utility and how to use it. (Web app) 13](#_Toc44691117)

[Section 4.4. Mobile application walkthrough 19](#_Toc44691131)

[Section 4.5. API and how to use it 26](#_Toc44691140)

[Section 4.6. How the notification system works. 35](#_Toc44691141)

[**Chapter 5. Conclusions** 36](#_Toc44691144)

[Section 5.1. Conclusion 36](#_Toc44691145)

[**Chapter 6. Appendices** 37](#_Toc44691146)

[Section 6.1. Resources 37](#_Toc44691147)

1. **Problem definition**
   1. ****Problem statement****

The ‘Question papers management’ utility:

This utility is for the students who wants to view previous question papers for revision purpose.

The Administrator can upload examination time tables. edit - According to time tables, after the exam ends, users get notified to upload Question paper (automatic) The Admin views all requests for particular exam, branch, year verifies and clicks upload. When uploaded, that exam slot should disable upload option and display paper(automatic). The Admin can change the uploaded picture. There should be a feature so that the admin can login and perform all these activities.

The end users can view previous question papers sorted semester wise, year wise branch wise etc. The user gets upload notification (so that he can upload a picture). There should be a way for the users to register and login.

* 1. ****Significance and objective****

**The objective of this utility is to change and improvise the student’s learning. It helps the students to quickly go through all the previous papers of the subjects before the exam and prepare for the exam accordingly, without having to specifically retrieve the papers through some person digitally or physically, removing the inevitable time consumption for waiting to get a paper.**

**With this utility, all that changes. This tool helps the student get papers by a touch, from a simple click on his/her phone. It helps the student to prepare for exams with less worry, obtaining papers instantly with a click. This improves students’ confidence and thereby the performance in the exams. The faculty can save time on giving previous question papers details and focus more on teaching.**

* 1. ****Organization****

Chaitanya Bharathi Institute of Technology is the organization for this project. Chaitanya Bharathi Institute of Technology is a private engineering college located in Gandipet, near Financial District, Hyderabad, Telangana, India. The college is affiliated to Osmania University and is accredited by the National Board of Accreditation. The institute received an autonomous status in 2013.

CBIT was established in 1979.[4] The institute is affiliated to Osmania University, originally with three undergraduate programs and an intake of 200 students. By 2014 this had grown to nine undergraduate and seven postgraduate courses with a total intake of 9000 students. It was inaugurated by late Dr. Marri Chenna Reddy.

CBIT is the first private engineering college in Telangana.

1. **Literature Survey**
   1. ****Existing solutions****

**The existing ways to obtain question papers for revision are somewhat difficult and time consuming. One has to go to a place where all the physical copies of the papers are stored – i.e. a repository of some sort.**

**In colleges like CBIT, the question papers are stored in the library. One has to physically be present at places like libraries where papers are stored, and wait for the others who are already reading, taking pictures of the papers or who have taken the papers with them. This waiting proves fruitless once the individual realizes that the paper he gets after a long wait, is not the paper they have been looking for.**

**Sometimes a question paper could be the only paper in the storage of papers, and once its lost, there’s almost no way to get it back. Sometimes the papers could be physically damaged, with scratches, or erasure of the paper ink with time, or the papers could be torn, some could be un-readable. Sometimes the paper would not be with the repository. Sometimes one might not be at a place to be able to be physically present at the repository to obtain the paper.**

**The other way could be when the students obtain the paper from other students who had previously given the exams. This method is somewhat unreliable. The student may wait for long, all to know that there is no paper, or that the paper he wanted is not the one the other student provided. But this method helps those who cannot get the paper from the place where papers are stored. This is currently the most popular way among the students to obtain question papers of the past.**

**Some other solutions could be that some people make model papers and previous questions compilation, make books of them and sell in the markets. This too, is not that reliable considering that officials don’t make the compilation of the papers and the compilation could be inconsistent sometimes. Sometimes the institute papers couldn’t be available in the market. And not all students would want to buy a set question papers when they could believe they could get a question paper for free from their friends or the institute. Sometimes the compiled set could be lost.**

**The other way is that faculty commonly provide previous questions as a part of their teaching curriculum hoping the students to note down the questions. This again poses the problem of losing the questions. Another main problem could be that some faculty couldn’t give the questions from the previous question papers. There are some websites which upload the question papers for students. But not all institutes would be available.**

**The solution presented in this project aims to get over the problem. This solution is a way to digitalize question papers and store in a remote database in a cloud-based server. The students can get papers whenever they want and wherever they want.**

* 1. Related works

Existing solutions for equivalent problems are done by government officials for providing the previous question papers for nation-wide recruitment and entrance exams, and board exams.

Previous Question papers for UPSC conducted exams

<https://www.upsc.gov.in/examinations/previous-question-papers?field_exam_name_value=&page=4>

Previous Question papers for CBSE conducted exams

<http://cbse.nic.in/newsite/examination.html>

Previous Question papers for JEE Advanced

<http://www.jeeadv.ac.in/pastqp.php>

Previous Question papers for JEE Main - NTA

<https://jeemain.nta.nic.in/WebInfo/Public/View.aspx?page=4>

Our solution objective is to make a similar utility for the institute (CBIT).

* 1. Tools and technologies used

Android Studio - Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development.

Java - Java is a general-purpose programming language that is class-based and object-oriented, and designed to have as few implementation dependencies as possible. Android application in this project is programmed using Java in Android Studio.

Python - Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.

Django - Django is a Python-based free and open-source web framework that follows the model-template-view architectural pattern. It is used in this project to make the web application for the admin.

Flask and Flask RESTful (for RESTful API)- Flask is a micro web framework written in Python. Flask-RESTful is an extension for Flask that adds support for quickly building REST APIs. An application programming interface (API) is a computing interface which defines interactions between multiple software intermediaries. It defines the kinds of calls or requests that can be made, how to make them, the data formats that should be used, the conventions to follow, etc. Representational state transfer (REST) is a software architectural style that defines a set of constraints to be used for creating Web services. Web service APIs that adhere to the REST architectural constraints are called RESTful APIs.

Postman - Postman is a collaboration platform for API development. Postman's features simplify each step of building an API and streamline collaboration so you can create better APIs—faster.

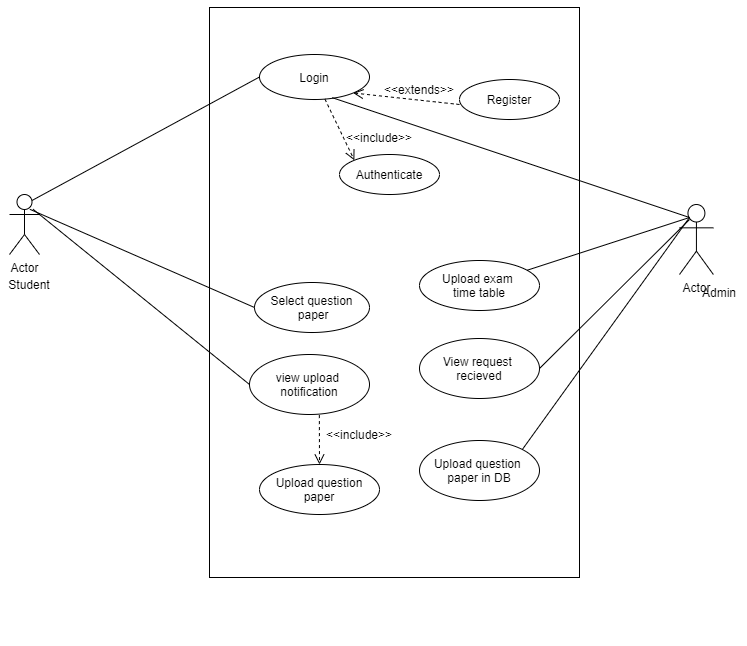
HTML, CSS and JavaScript - Hypertext Markup Language is the standard mark-up language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript

Bootstrap - Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development.

JSON - JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language Standard. It is used in APIs to send and receive the data.

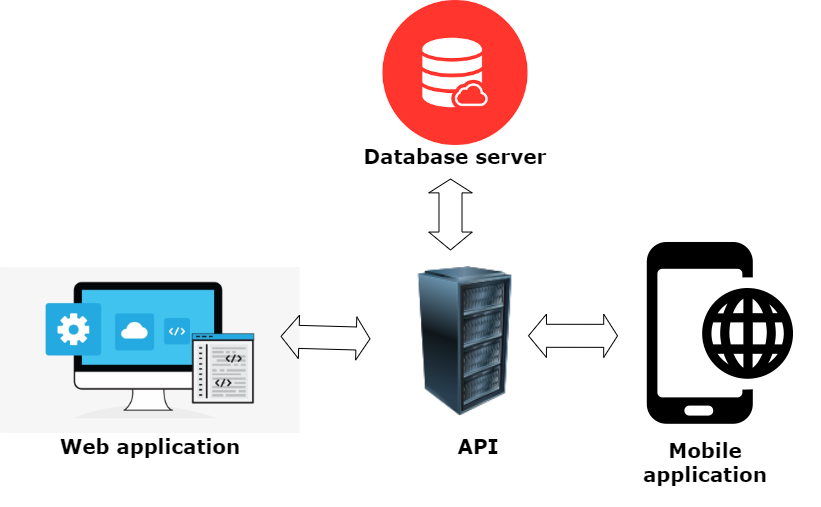
MySQL - MySQL is an open-source relational database management system (RDBMS)."SQL" is the abbreviation for Structured Query Language. A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data.

1. **Design of the proposed system**
   1. Block diagram



* + 1. UML diagram of the proposed system depicting the Admin and user facilities and activities. (The Unified Modeling Language is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.)

The proposed system is represented in the UML diagram depicted above.



* + 1. A figure depicting the architecture of the ‘Question Papers Management’ utility. The Web application for the Administrators, the mobile application for the end users interact with the API and API interacts with the database server.
  1. Module description

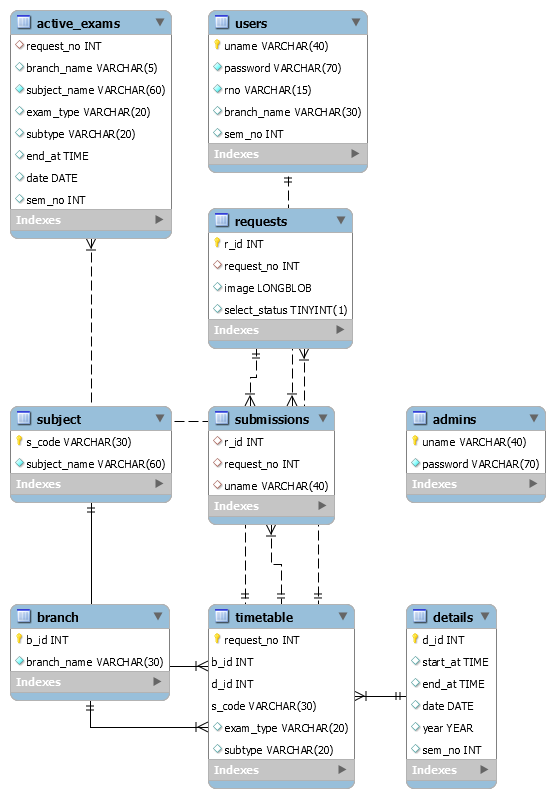
The student can login, register, upload the question paper, update or remove it. The student will be authenticated when he logs in, by the API. Student can search the question paper, get notified to upload a paper and he can upload the question paper. The Admin can login, Admin will be authenticated, admin can upload timetable, view the request received, and upload the question paper in the database.

For this project, the Admin utilities are developed in a web application with Django. Web app facilitates actions that can be done by the admin in the diagram and the description above. The user utility is developed as an android mobile application. The user can utilise the application for all the facilities shown in the diagram and the description. Data is stored in a database hosted in a cloud-based server. The intermediary connecting agent is the API.

All the necessary data is to be stored in the database for the admin and for the end user purpose. For that, the admin and the user applications interact with the middle agent API to store or retrieve data in the database.

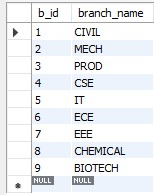
All the images, timetable data, subject information, branch information, data information, exam information, timings, user info, admin info image data, all are stored in the database. The Web application and Mobile application have to make HTTP requests and the API interacts with the database and sends back the respective application a suitable response. And with that response the respective applications function accordingly. For both the web and mobile application to function properly, the API must be configured properly adhering to the requirements of the either tool.

1. **Implementation of the proposed system**
   1. ER diagram of the database



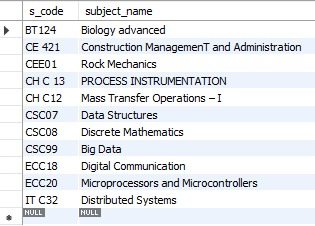
* + 1. ER diagram of the Database schema used in this project.
  1. Database schema explanation with examples.

The branch table stores information about the branches. An id is given to each branch.



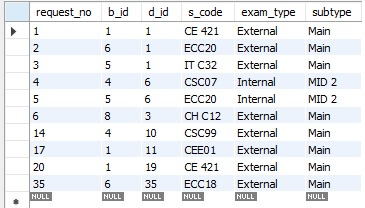
* + 1. Table branch

The subject table gives the information of the subjects present and the subject code



* + 1. Table subject

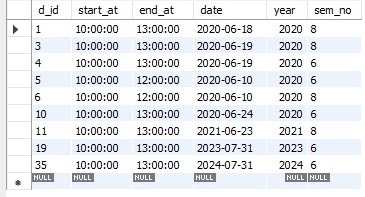
The timetable table contains the information about the branch, timings (can be obtained from d\_id using details table), subject information, exam type and sub type of the exam (Main, MID 1, MID 2 etc)



* + 1. Table timetable

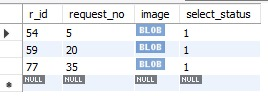
The request\_no in timetable table gives all the information about a particular exam.

The details table stores information about the timings, date, year of the exam and the semester number of the exam. A d\_id can cater to multiple exams in the timetable table.



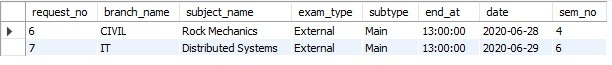
* + 1. table details

The requests table stores the images uploaded by the user or admin with the select status(indicating the status of whether the image has been selected or not), request\_no which gives the exam information from the timetable table , image and r\_id which is the ID. The image is in LONGBLOB format.



* + 1. Table requests

The active\_exams table stores all the information about the exams which are active and the android app can use the information in this table to notify users.



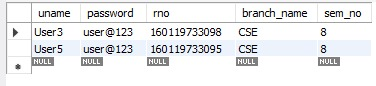
* + 1. Table active\_exams

The admin table stores the information about the admin username and password.



* + 1. Table admin

The users table stores the information about the user’s username, password, roll number, semester and branch



* + 1. Table users

The submissions table stores the information about the submissions made by a user. The information pertaining to a particular request\_no(for a particular exam) in the submissions table is deleted once the admin approves a question paper or once the admin himself uploads a paper for a particular exam.



* + 1. Table submissions
  1. The utility and how to use it. (Web app)

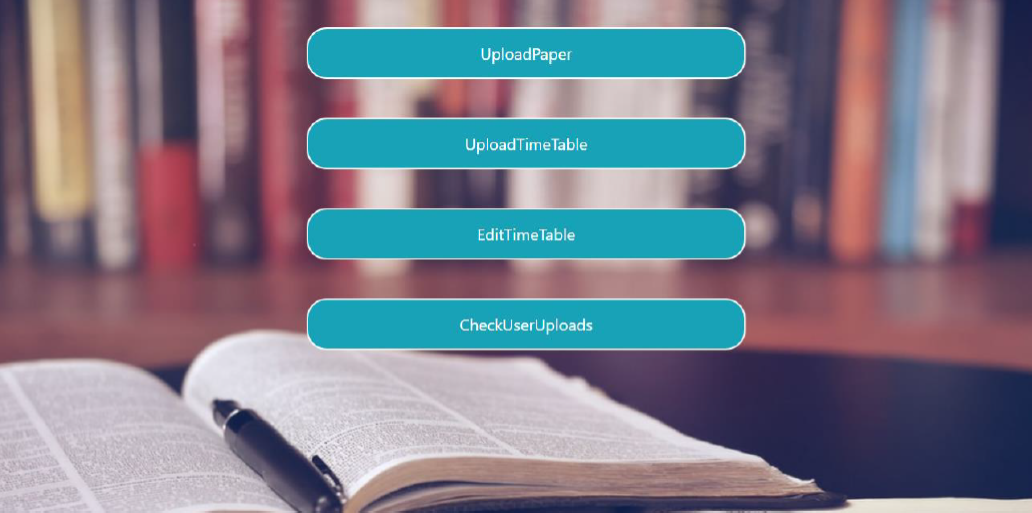
The Django web app is designed for the Administrator utilization purpose. To start with, go to the /paper URL.

It directs to the login page. The Admin has to enter the correct username and password credentials to log in to the utility. After that, the admin is redirected to the home page if correct credentials are entered.



* + 1. The web app admin login page

This is the home page that should be seen if correctly logged in. From here, the admin can click on the corresponding button to either upload paper, upload timetable, edit (update or delete the timetable) and check user uploads.



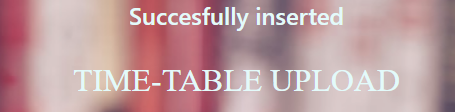
* + 1. The admin homepage

The upload timetable helps admin to upload a timetable. Clicking the button should redirect you to a page. The page should look like the one in the picture that follows. The picture looks like this.



* + 1. Admin upload timetable page

A message of success is displayed if the timetable insertion is done correctly.

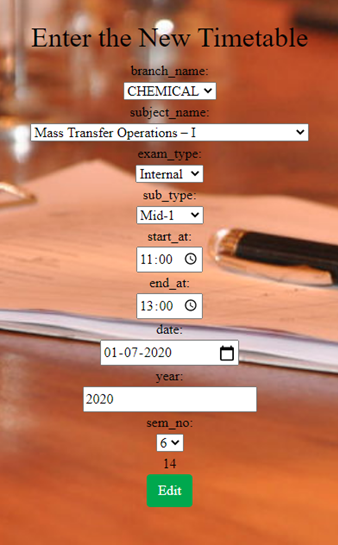
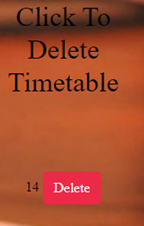


* + 1. Admin timetable upload success.

The edit timetable redirects to the page below. To proceed, enter correct details and submit.  


* + 1. Admin timetable editing page where admin has to open a timetable

You will be redirected to the page that has options to edit timetable or delete as shown in images that follow. Giving new data and pressing submit, will change the timetable.

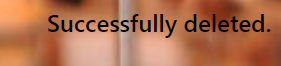
* + 1. Admin timetable update and delete features

A message of success is displayed on successful update.



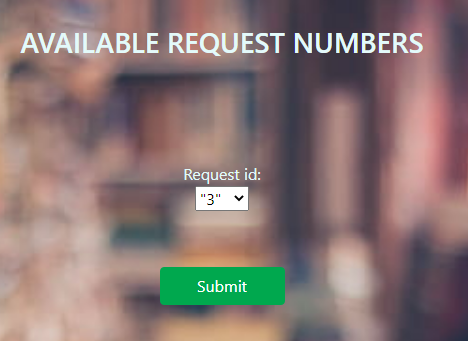
* + 1. Successful update of timetable

To delete the timetable, press the delete button. A message of deletion is displayed if deleted.



* + 1. Successful delete of timetable

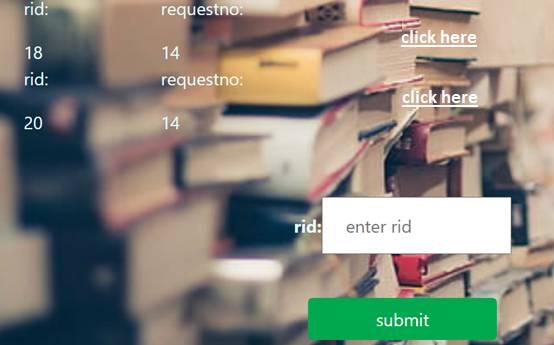
The admin can approve images once the user uploads them. The check user uploads button takes the admin to this page



* + 1. Admin exam search page

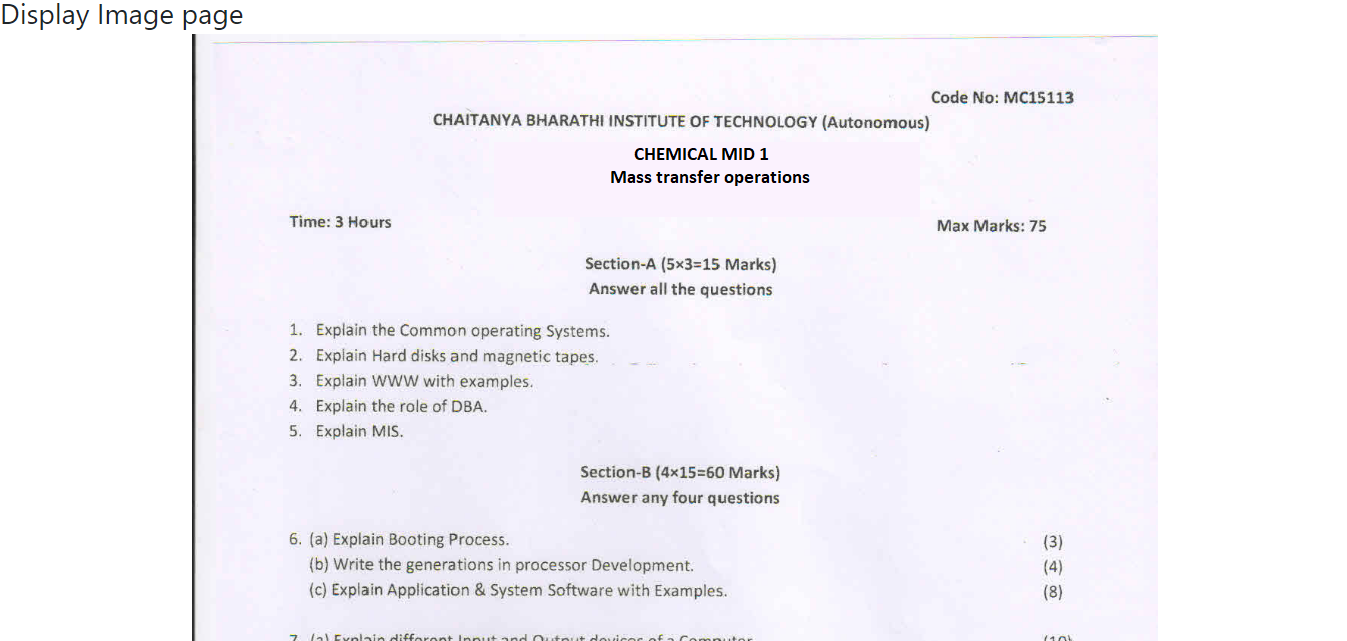
Admin can then select the request id to check a particular subject.

Once selected, the images for the exams are displayed below.



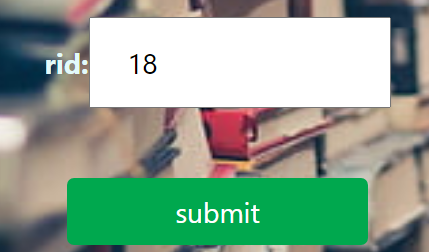
* + 1. Admin paper checking page

To check an image, the click button corresponding to the image is used. The Admin can check each image corresponding and each image has a unique rid. The image appears as follows.



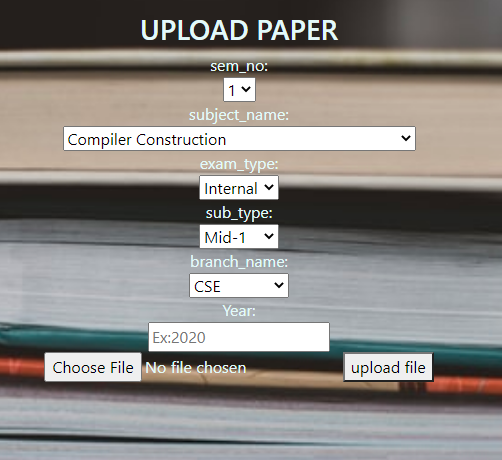
* + 1. Admin paper display page

To approve a paper, admin has to enter the rid corresponding to the paper and press the submit button. Once the paper is approved, the remaining papers get deleted for a request no (i.e. for an exam of a particular subject on particular time for a sem.)



* + 1. Admin accepting paper by entering an rid(r\_id)

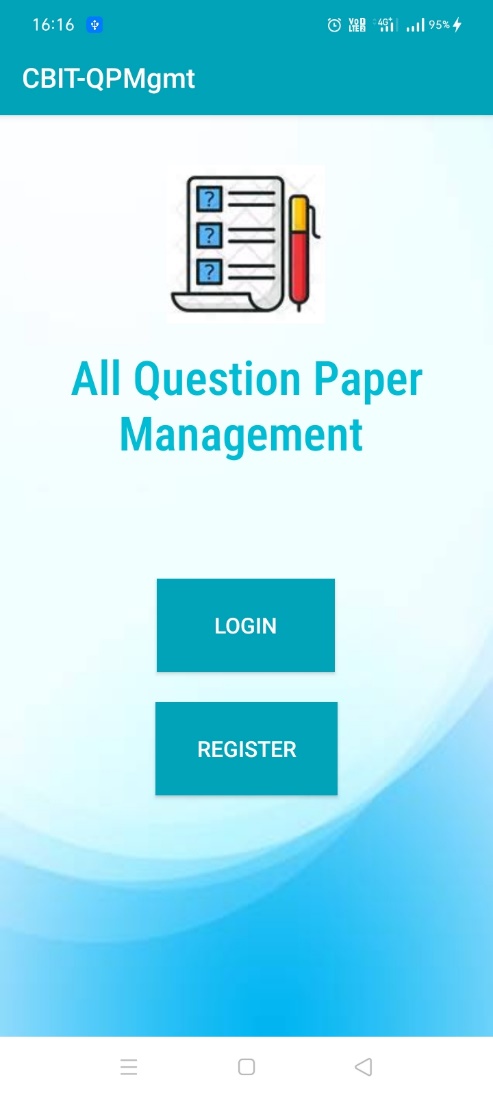
The admin can upload paper himself, instead of accepting from students. The Admin has to first set the exam for which they want to upload the paper for, then upload the paper using choose file button and confirming by pressing upload file button.



* + 1. Admin – upload paper page

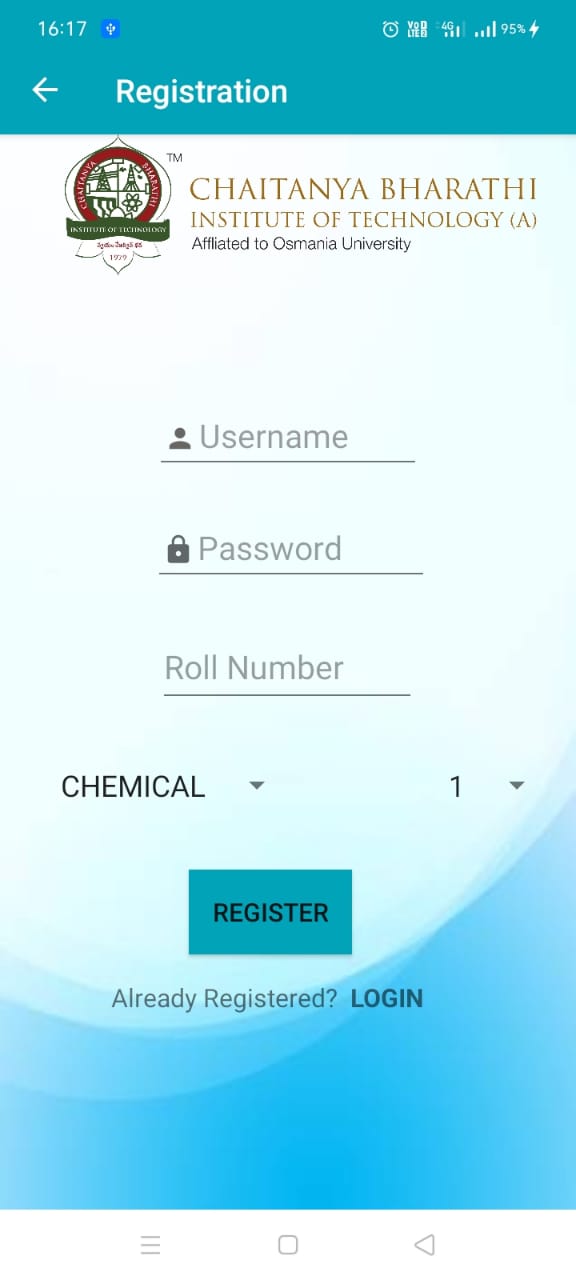
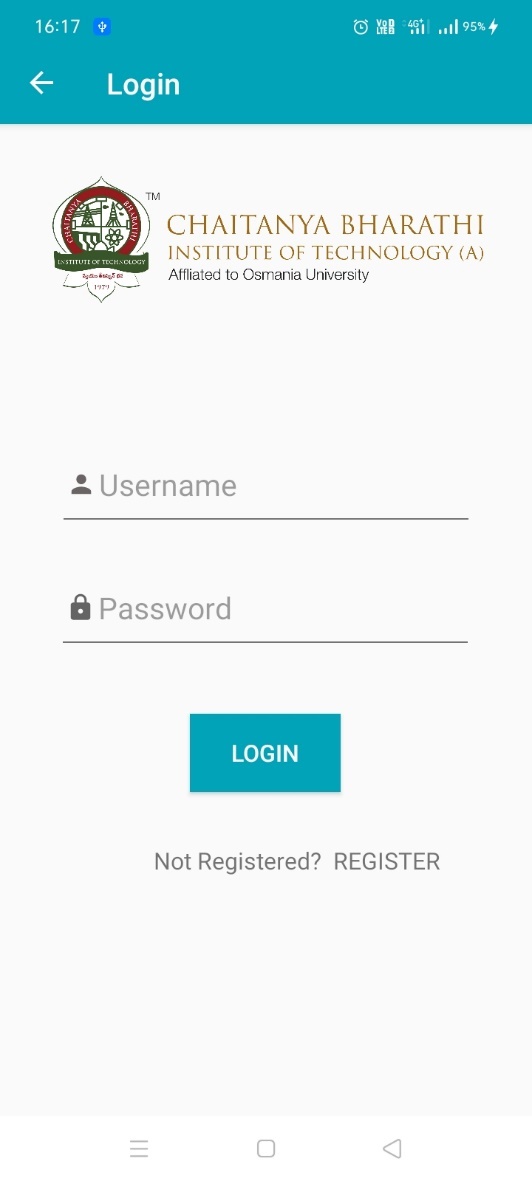
Once the admin uploads the paper or approves a request from the user to accept the question paper, all the other requests for the exam get deleted.

* 1. Mobile application walkthrough



* + 1. The page that appears to the user once they open the app after installing. The user can log in if they have an account already with the login button, or create one account with the register option.

The mobile application, opening, takes user to a home page where user can choose to login or register. While registering, The user has to enter the username, password, roll number, branch and sem number to register for an account.

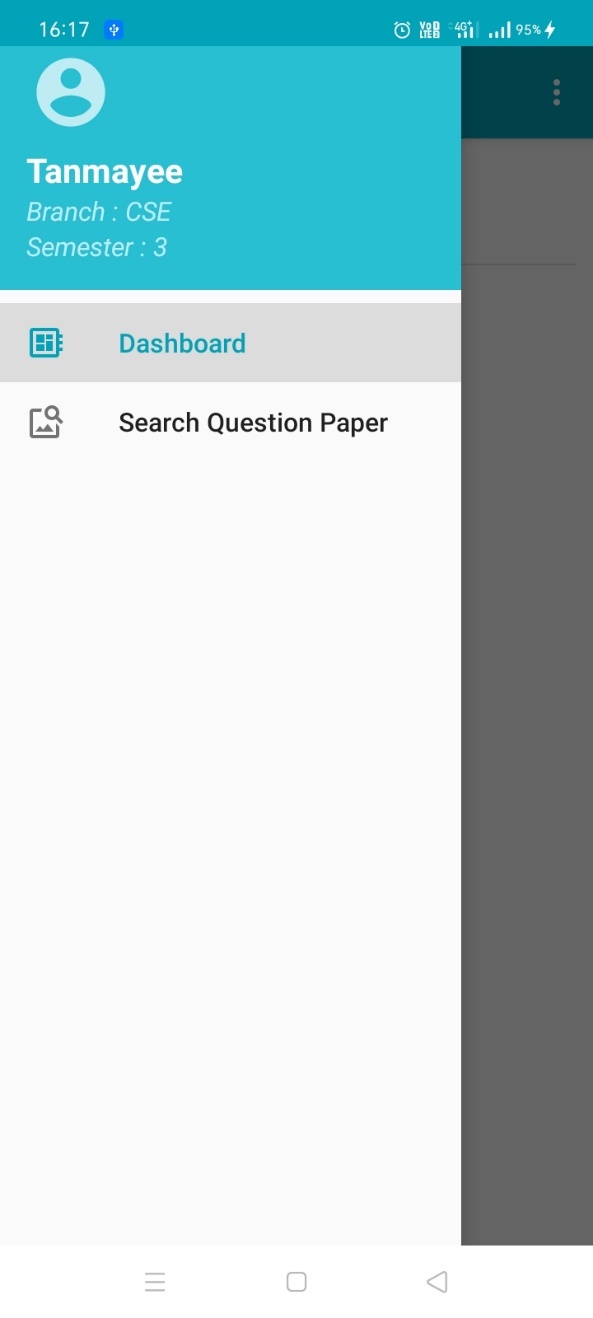
 

* + 1. User registration (on the left) and User login (On the right) pages of the android mobile application.

While registering, the user cannot provide a username that has already been registered.

The user cannot provide another roll number that has already been used by a user having another account. That way, only one user of a particular roll number can register. After registering, the user can log in using the login facility. User has to provide the correct credentials (The correct username and password) that they have provided while registering for an account.

Once the user has successfully logged in to the application, User will be directed to the dashboard page where the active exams are displayed once they are over and paper can be submitted. The user can navigate to the other page using the sidebar.

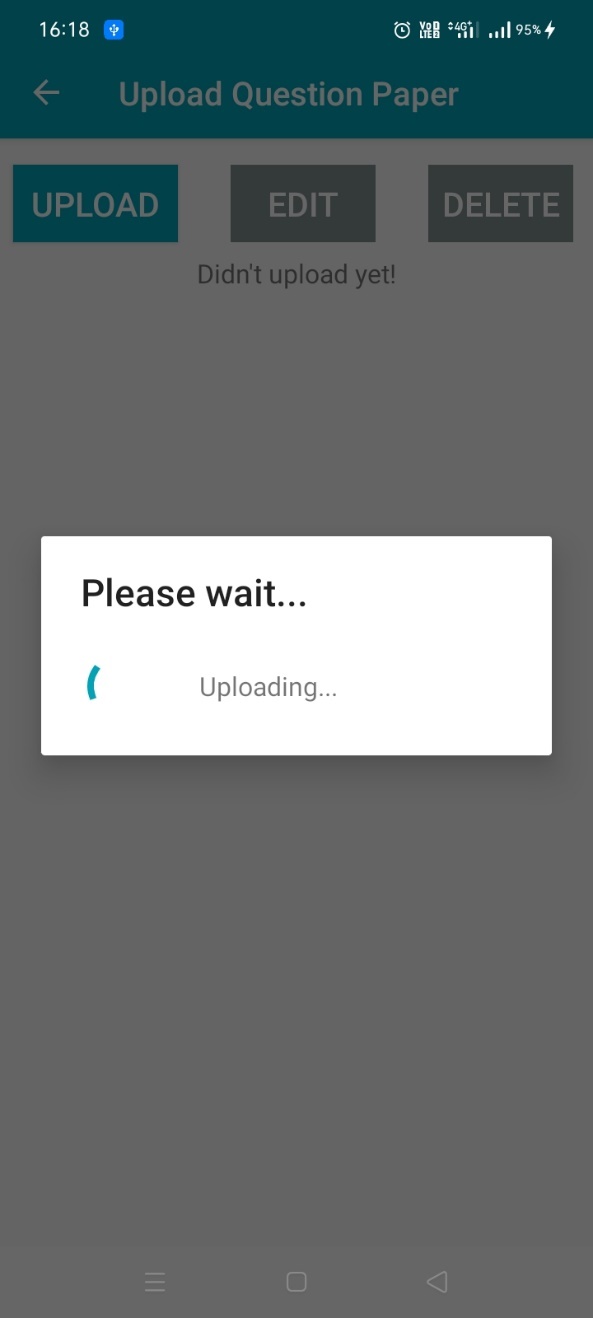
 

* + 1. Dashboard page (On the left), Navigation sidebar feature on the right showing the dashboard and search question paper utilities.

In the dashboard, after selecting a subject, user can upload a question paper corresponding to the exam the user selects.

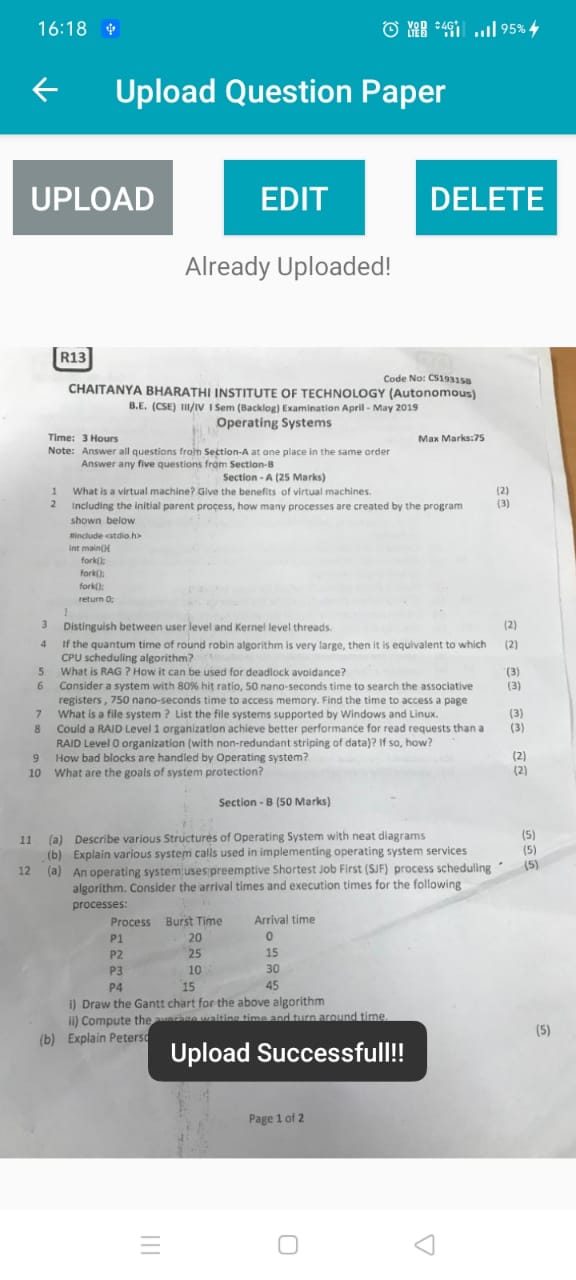
The user can select and upload an image of the question paper.

Once the user uploaded the question paper, user can either change the image uploaded by pressing the edit button or delete the upload by pressing the delete button.

* + 1. The upload question paper page (Left) showing the options to upload, edit and delete the uploaded question paper. The paper being uploaded (Right)

The user can either change his upload or delete the upload before the admin accepts submissions for an exam.

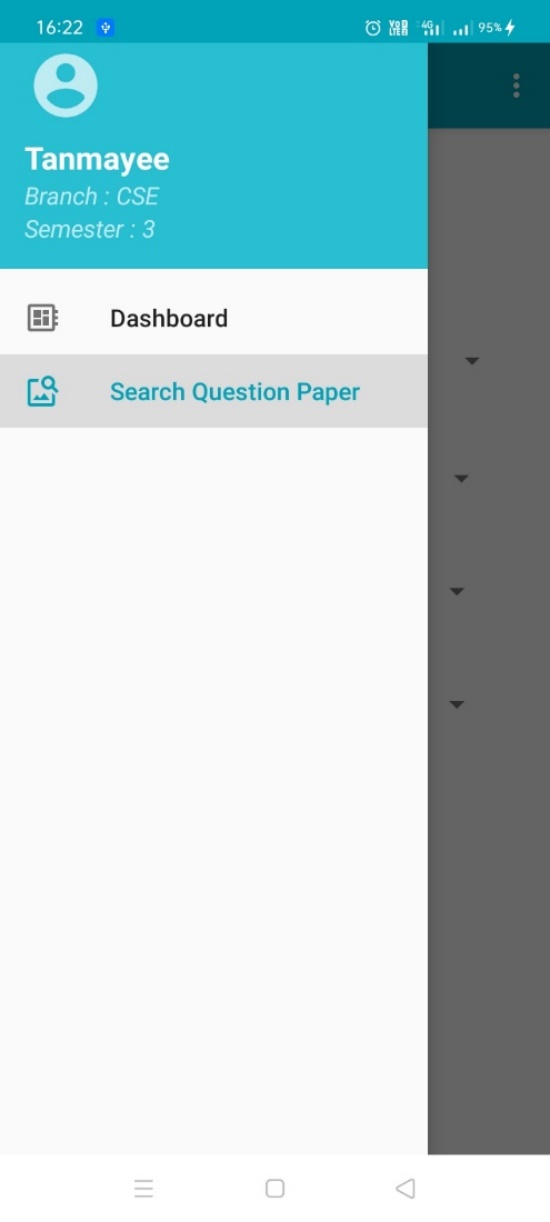
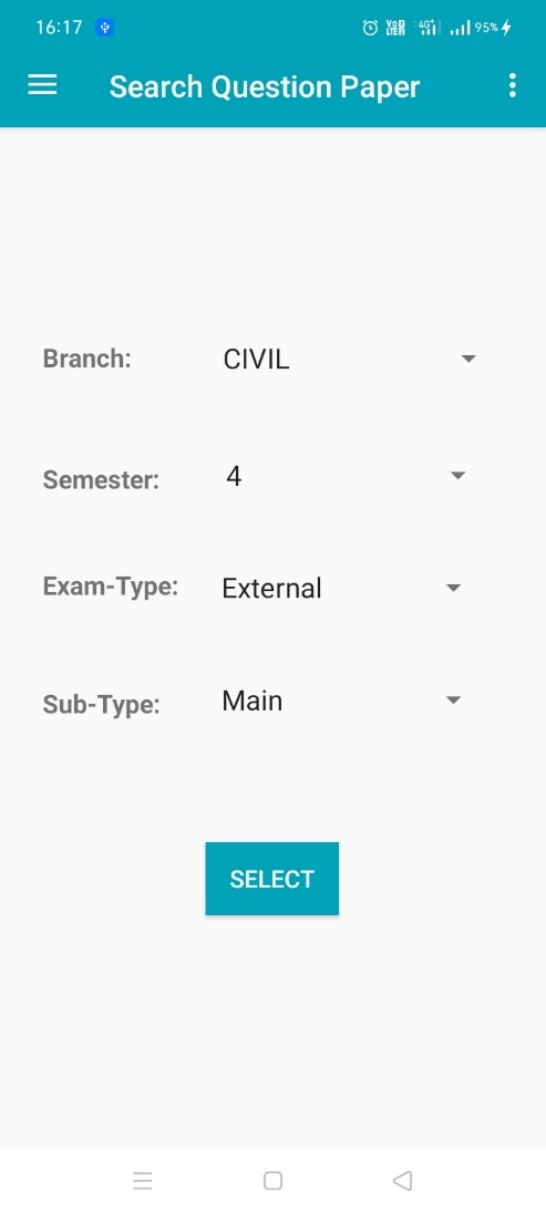
 

* + 1. The upload question paper after a successful upload(Left) and The Question paper after a successful delete.

After that, the user image would either get deleted from the database or get approved accordingly and user cannot change the uploaded image or delete the uploaded image him/herself.

The default page where the user is in, is the dashboard page. The user can navigate to ‘Search Question Paper’ to search question papers by branch, semester, exam type (Internal, External) and subtype (MID 1, MID 2, Main, Backlog).

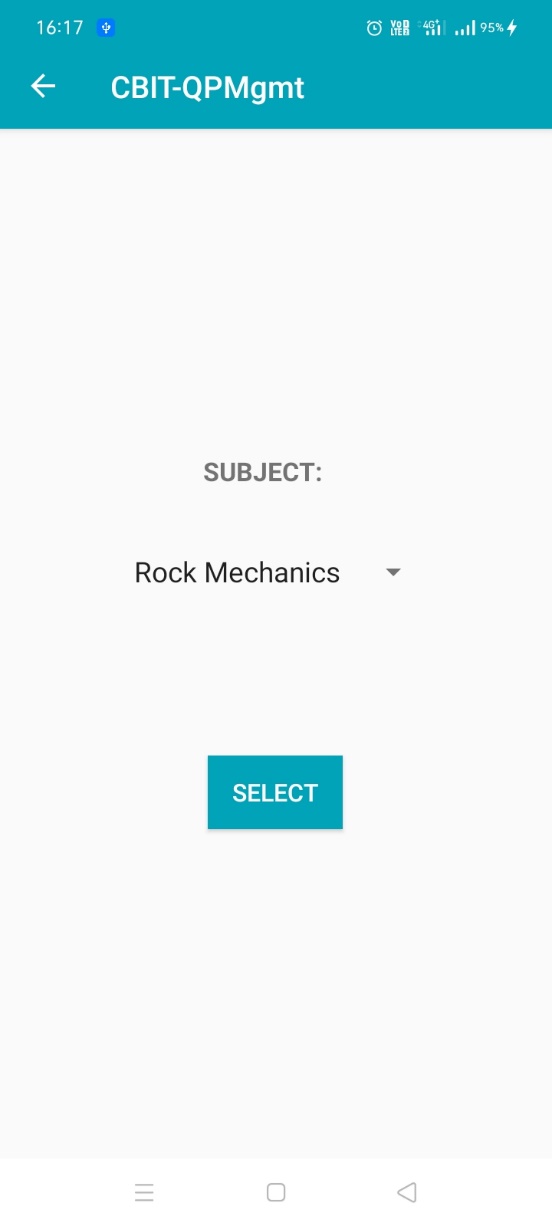
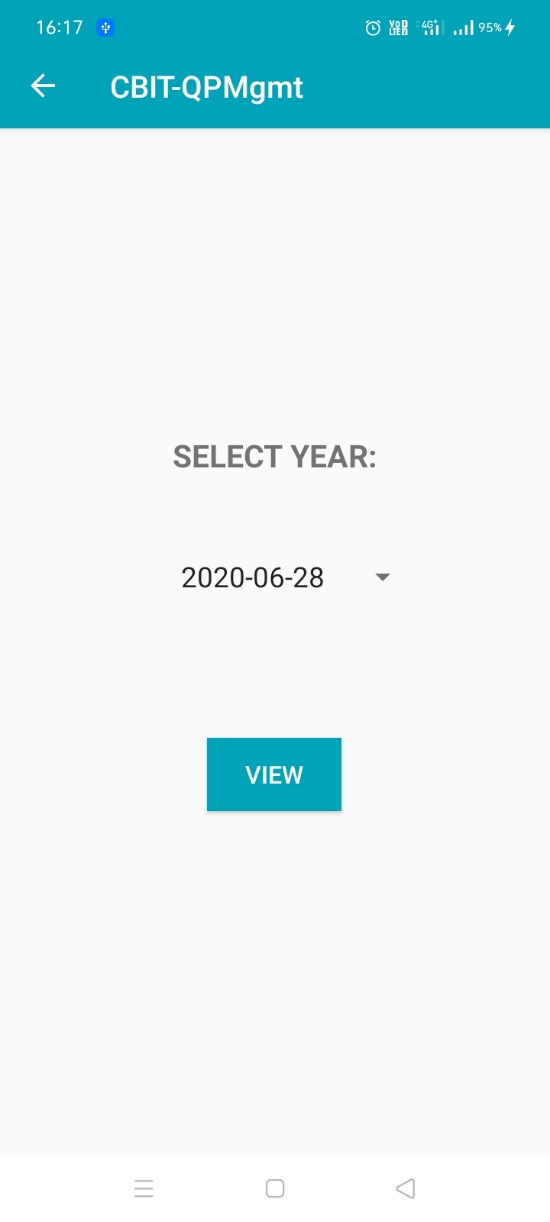
Once the user clicks the Search Question Paper, he will be shown fields to be set, as show in in the images that follow.

* + 1. The page where the user can set the branch semester, exam type and sub type of the exam and view the subjects corresponding to those requirements.

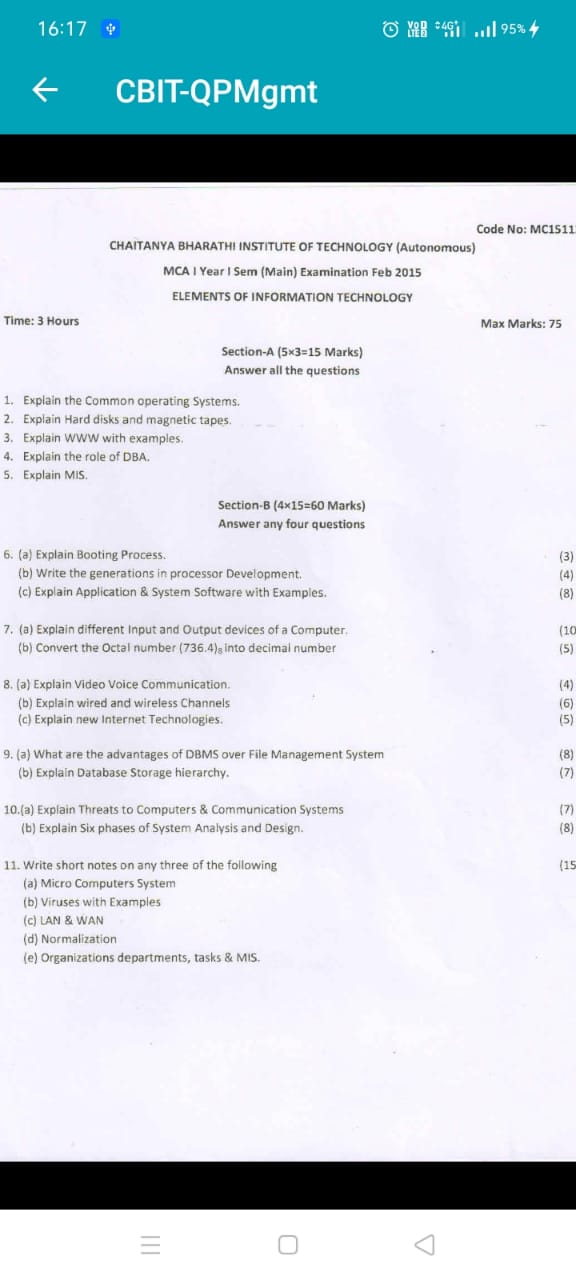
The user can search for a question paper by entering the branch, semester, exam-type and subtype. The user has to enter a valid set of inputs for the app to display the results. The results will be subjects corresponding to those inputs. After user enters valid set of inputs in the search question paper page, the user will be displayed a list of subjects corresponding to those fields (Branch, Semester, Exam-type, subtype).

The user has to then select a subject he wants to get papers for. The images for those operations in the application are shown as follows.

* + 1. On the left is the subject selection utility. On selecting a subject one can set date for the question paper the user wants to see, in the app which looks as in the picture on the right.

After pressing view, the user will be displayed subject list year wise. Then the user can select an item shown to display an image corresponding to the year, subject, branch, semester, exam type and exam’s subtype.



* + 1. The question paper that is displayed to the user for the paper the user chose to see.
  1. API and how to use it

The API is designed so that it can accept requests from web app and mobile app to send or receive data. The data sent can either be used to insert into the database, delete, update or retrieve information appropriately.

The API takes in some data sent by the API caller and API returns a message accordingly, whether the action to be performed by the API is successful or not – the API returns an error if the data is not found or when there is an internal server error or connection error when connecting to the database. API is useful as it maintains the common database for both admin and end users. It’s a flexible way to do operations with the data to get a desired response or change in the database. This API is stored in a server or a similar location and API callers have to make calls to specific endpoints of the API. The following table is a set of endpoints, the data to send, the response you might receive and what happens in back-end. The following is table of the details of the API endpoints, data to be sent, response that will be received and how the API works in the back end.

**Admin endpoints:**

|  |  |  |
| --- | --- | --- |
| Endpoint | Particular | Information |
| /admin-create-timetable (POST) | Data to be sent | b\_id, s\_code, exam\_type, subtype, start\_at, end\_at, date, year, sem\_no |
| Response received | A message of success or error (internal server error or database related error.) or an error that required data not found is returned. |
| Back-end working detail | Timing and sem info. is added to details table if not already present. Exam info is added to timetable table if not already present. This information is added to active\_exams table if not already present. |
| /admin-timetable-delete (POST) | Data to be sent | request\_no (can be obtained by another endpoint by providing necessary details) |
| Response received | A message of successful deletion or error (internal server error or database related error.) or an error that required data not found is returned. |
| Back-end working detail | Obtains d\_id (which gives timing and sem info). Delete all entries in requests and timetable table for the request\_no. Delete entries corresponding to d\_id in details table if no other entry (than the deleted row) in timetable table has d\_id |
| /admin-timetable-update (POST) | Data to be sent | request\_no, b\_id, s\_code, exam\_type, subtype, start\_at, end\_at, date, year, sem\_no |
| Response received | A message of a successful update or error (internal server error or database related error.) or an error that required data not found is returned. |
| Back-end working detail | The current d\_id is obtained (old d\_id). The timings and sem info are added to details table if not already present (for given fields). The d\_id is obtained for the timings and sem provided (d\_id changes if the timings or sem is changed, let’s call it new d\_id). The timetable is updated with the d\_id obtained, and the other fields sent. If the old d\_id is not the same as the new d\_id, then its confirmed that timings are updated, and the details entry corresponding to old d\_id is deleted if no other row in the timetable table depends on the old d\_id. The corresponding details in the active\_exams table are updated. |
| /admin-reqno-details  (GET) | Data to be sent | request\_no |
| Response received | All the r\_id, request\_no, image data is sent back for a given request\_no, to the caller. An error message is displayed for any internal server error. |
| Back-end working detail | The API retrieves the r\_id, request\_no, image (BLOB data encoded in base64 string by the API) for a particular request\_no, and sends the JSON list back to the API caller. |
| /admin-qpreq (POST) | Data to be sent | b\_id, sem\_no, exam\_type, subtype, s\_code, year, image |
| Response received | A message of successful insertion or error (internal server error or database related error.) or an error that required data not found is returned. |
| Back-end working detail | request\_no is selected for the given details, and image is converted to BLOB from base64 string, and is inserted with a select status of 1 into the requests table. All other entries with the request\_no in requests table are deleted (with select\_status of 0 or 1). The entries for the request\_no in submissions table and active\_exams table are deleted |
| /admin-qpreq (GET) | Data to be sent | r\_id |
| Response received | r\_id, request\_no and base 64 encoded image is returned or an error is returned (internal server error.) |
| Back-end working detail | All the r\_id, request\_no and image details for an r\_id are retrieved from the requests table and are sent to the caller by converting the BLOB to base64 encoded string. |
| /admin-false-select  (GET) | Data to be sent | - |
| Response received | All the distinct request\_no’s which have a select\_status = 0 or an error message is returned if server error occurs. |
| Back-end working detail | All the distinct request\_no’s are queried for and returned, which have a select\_status = 0 |
| /admin-delete-req  (POST) | Data to be sent | request\_no, r\_id |
| Response received | A message of success is returned if worked or a message of internal server error or database error is returned if not successful |
| Back-end working detail | All the entries in requests table with the request\_no sent, except for the entry with an r\_id are deleted. The undeleted entry with the r\_id is set a select\_status of 1. The entries in active exams and submissions corresponding to a request\_no are deleted. |
| /admin-get-subjects  (GET) | Data to be sent | - |
| Response received | All the subject\_name and s\_code details are returned if successful. Or else an error message is returned. |
| Back-end working detail | All the subject\_name and s\_code are retrieved from the subject table. |
| /admin-get-timetable  (GET) | Data to be sent | b\_id, sem\_no, exam\_type, subtype, s\_code, year |
| Response received | request\_no, b\_id, s\_code, exam\_type, subtype, start\_at, end\_at, date, year, sem\_no are returned if successful or a message of data not present is returned or an error message is returned. |
| Back-end working detail | request\_no is obtained from the timetable joined with details table from the given fields and the [request\_no, b\_id, s\_code, exam\_type, subtype, start\_at, end\_at, date, year, sem\_no] are retrieved for a request\_no from timetable joined with details table. |
| /admin-login (POST) | Data to be sent | uname, password |
| Response received | Access token is returned if login is successful or a message of invalid credentials is returned if false credentials are provided. |
| Back-end working detail | The username and password are searched in the database. If any row has the data matching, then an access token (JWT – JSON Web Token) is generated and returned. |

**User endpoints:**

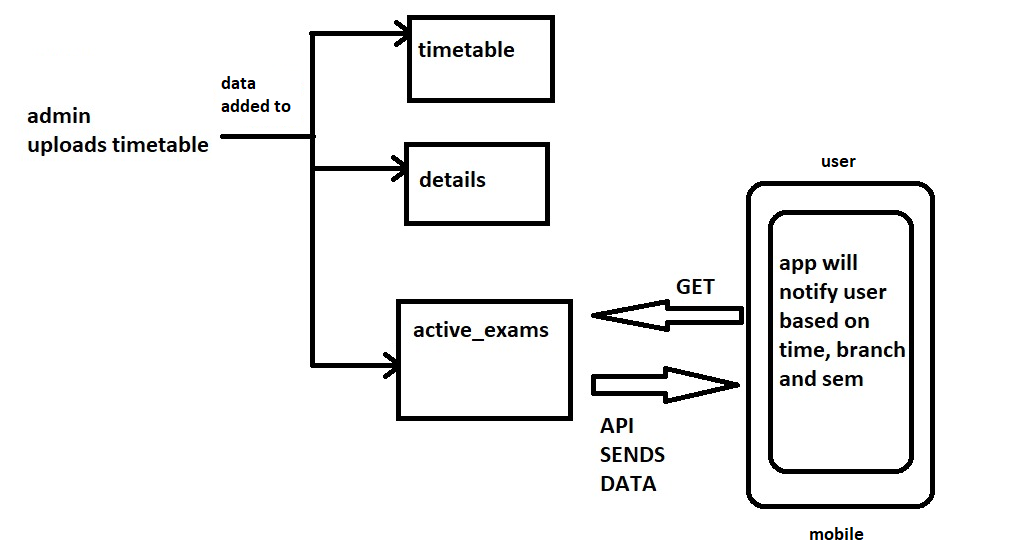
|  |  |  |
| --- | --- | --- |
| /qpreq  (POST) | Data to be sent | request\_no, image (base64 encoded string), uname |
| Response received | A message of successful update is returned if the operation is successful. Or else a message of internal server error or database error is returned or an error message stating action not allowed to perform is returned (this is when admin approves/uploads a paper already) or an error that required data not found is returned. |
| Back-end working detail | The requests table is checked if there is an image having select\_status = 1 for the request\_no, if such an entry is there, an error is returned that action can’t be performed. Otherwise, the image is converted to blob data and inserted to the requests table. The r\_id is obtained using the last insert id for the connection made and inserted into the submissions table along with uname, if not already present. |
| /qpreq  (GET) | Data to be sent | request\_no |
| Response received | r\_id, request\_no, image are returned for the given request\_no if successful or a message of error is returned. |
| Back-end working detail | The details for a request\_no are returned for the row where the select\_status = 1 |
| /qp-update (POST) | Data to be sent | request\_no, image, uname |
| Response received | A message of successful update is returned if successful or else a message of error is returned. (database or internal server error) or an error message stating action not allowed to perform is returned (this is when admin approves/uploads a paper already) or an error that required data not found is returned. |
| Back-end working detail | The requests table is checked if there is an image having select\_status = 1 for the request\_no, if such an entry is there, an error is returned that action can’t be performed. Otherwise, the request\_no and uname are searched in the submissions table and r\_id is obtained. The row corresponding to r\_id in requests table is updated with the new image. |
| /qp-delete (POST) | Data to be sent | request\_no, uname |
| Response received | A message of successful deletion is returned or else a message of error is returned. (database or internal server error) or an error message stating action not allowed to perform is returned (this is when admin approves/uploads a paper already) or an error that required data not found is returned. |
| Back-end working detail | The requests table is checked if there is an image having select\_status = 1 for the request\_no, if such an entry is there, an error is returned that action can’t be performed. Otherwise, the entry in the submissions having the uname and r\_id are deleted. The entry in the requests table with the r\_id and request\_no are deleted. |
| /get-active-exams  (GET) | Data to be sent | branch\_name, sem\_no |
| Response received | A list of JSONs of request\_no, branch\_name, subject\_name, exam\_type, subtype, end\_at, date, sem\_no are returned if successful or else an error message is returned. |
| Back-end working detail | All the row data containing all columns of the active\_exams data are returned for a given branch\_name or sem\_no |
| /get-subjects (GET) | Data to be sent | branch\_name, sem\_no, exam\_type, subtype |
| Response received | A list of JSONs of distinct subject\_name’s are returned if successful or an error message is returned for connection error. |
| Back-end working detail | timetable, details, subject, branch tables are joined and queried for subject\_name for the fields provided. |
| /get-uploads (GET) | Data to be sent | uname, request\_no |
| Response received | r\_id, uname, request\_no if worked successfully or else an error message is returned. |
| Back-end working detail | The row corresponding to the uname, request\_no is searched for and returned if found. |
| /get-yearwise (GET) | Data to be sent | branch\_name, sem\_no, exam\_type, subtype, subject\_name |
| Response received | A list of JSON objects with request\_no , subject\_name , date ordered by descending order of date(most recent of the dates first) if successful or else an error message is returned. |
| Back-end working detail | timetable, subject, details, branch tables are joined and queried for the rows having the data sent, and the rows having the data are ordered in descending order of date. |
| /user-register (POST) | Data to be sent | uname, password, rno, branch\_name, sem\_no |
| Response received | A message of success is returned if successful or else error messages of either existing uname or rno is returned |
| Back-end working detail | An existing uname or rno is checked in users table, if already there, the respective error message is returned, or else registration will be successful. If the operation isn’t successful an error message is returned. |
| /user-login (POST) | Data to be sent | uname, password |
| Response received | Access token is returned along with uname, rno, branch\_name, sem\_no if login is successful or a message of invalid credentials is returned if false credentials are provided. |
| Back-end working detail | The uname and password are searched in the database. If any row has the data matching, then an access token (JWT – JSON Web Token) is generated and returned along with the other information in that row excluding password. |
| /get-uname-info (GET) | Data to be sent | uname |
| Response received | rno, branch\_name, sem\_no are returned if successful or a message is returned. |
| Back-end working detail | The uname is searched in the database. If any row has the data matching, rno, branch\_name, sem\_no are returned. Or else an error is returned. |

Note that all of these endpoints – Admin endpoints and User endpoints require the respective JWT (JSON Web Token) authentication for the corresponding endpoints (excluding the /admin-login, /user-login, /user-register endpoints).

Providing an invalid JSON Web token makes an API issue an invalid token response stating signature verification failed.

Not providing JWT for the JWT restricted endpoints will return an error stating that authorization is required and that the token is missing.

* 1. How the notification system works.



* + 1. The notification system – how it works

When the admin uploads timetable, the data is added to active\_exams. The data stays in the active\_exams until the admin accepts a paper or uploads a paper him/herself or deletes timetable. The application makes GET request to retrieve the data of the examinations having the branch and sem number of the user, and the data is stored in user device. Based on that, the application issues a notification for the user to upload paper for the exam.



* + 1. Notification in the mobile application so that the user can upload a question paper for the exam.

1. **Conclusions**
   1. Conclusion

This solution is a great way for students to revise question papers of the past without any trouble of physically going to the repository where papers are stored or contacting another student to obtain papers. It also helps paper setters and faculty to easily obtain previous papers and get an idea of how papers are being set over the years, the level of difficulty, difficulty in each section, chapter, where more questions are asked etc. This will help students and faculty save their time and possibly increasing individual grades as well.

1. **Appendices**
   1. Resources

Visit the documentation pages of the technologies used from the links provided below.

* Android documentation: <https://developer.android.com/docs>
* Django documentation: <https://docs.djangoproject.com/en/3.0/>
* Flask RESTful documentation: <https://flask-restful.readthedocs.io/en/latest/>
* Flask JWT extended documentation: <https://flask-jwt-extended.readthedocs.io/en/stable/>
* JWT introduction: <https://jwt.io/introduction/>