1. INTRODUCTION

1.1 ABOUT THE PROJECT

“Scholarship Portal for College” project is developed for scholarship division of colleges. Though media systems are very popular, majority of students are not aware of various schemes of scholarships applicable to them. To have a general awareness on scholarship, this project will be more convenient for each and every educational institution to take active participation in the implementation of various scholarships Schemes under their supervision .The project “Online Scholarship Management System” is a Software package developed for managing student’s scholarship details, and branch details. The software is very helpful to find the eligible candidates from colleges.

The project finds the eligible candidates from students list on the basis of marks, caste, sports and income. The “Online Scholarship Management System” has three modules. They are Admin and Student. The student can register to apply scholarship, after registration the student can login the system using his/her user name and password. The admin add the scholarship details. After login process the admin can view the scholarship application request by the students. Admin can send the approved / reject notification to the student after viewing the request. Once the admin granted the permission for the scholarship, the amount is sanctioned to the student through the online. Even the student can check their status. Scholarship as defined by is a grant or payment made to support a student's education, and it is awarded on the basis of academic or other achievement. If there is any problem with their applications while they are processing, it will also take an extra time for both the reviewing committee as well as the applicant to communicate and correct the errors. Therefore, additional paperwork for the review may cause a delay in the entire procedure. The processes of screening the applicant’s credentials, evaluation of applicant’s form, conducting aptitude test and oral interview are also tedious.

2. SYSTEM STUDY

2.1 EXISTING SYSTEM

At present the college maintains manual records, in the form of registers, files, etc. to store the scholarship details of the students. New scholarships to be issued, involves heavy paper work. It also leads to data redundancy. The analysis part includes a detailed study of the existing system. Difficulty to the students of remote area and outside state residing students to apply and to know the status of sanction.

2.2 DISADVANTAGES

- Notification through news papers
- Application on printed forms
- Manual verification of records
- Manual distribution of the scholarships by the colleges

2.3 PROPOSED SYSTEM

The proposed system has two major benefits. Once the existing system is automated all the drawbacks of the present system can be overcome. Many modifications can be made to make the atmosphere and extractions of data for generating a variety of reports are the factors, which form the basis of development of the proposed system. By using this website students can easily know about the information of scholarship details.

2.4 ADVANTAGES

- Besides newspapers now notification through online which can be viewed any time.
- Now Information available online.
- Sanctions are given online.
- Automation is done through software hence chances of subjectivity, and errors eliminated.

2.5 PROBLEM DEFINITION AND DESCRIPTION

The main goal or aim of our project is to automate the scholarship departments by assigning scholarship officer to every college. Students can apply various scholarship related forms. Admin monitor group of colleges by assigning a scholarship officer to every college. Scholarship Officer maintains the each and every student details. Scholarship Officer sends mail to eligible students about the scholarship.

3. SYSTEM ANALYSIS

3.1 PACKAGES SELECTED

Front End	: PYTHON
Back End	: MySQL Server

3.2 RESOURCES REQUIRED

HARDWARE SPECIFICATION

• Processor	: Dual core processor 2.6.0 GHZ
• RAM	: 4GB
• Hard disk	: 320 GB
• Compact Disk	: 650 Mb
• Keyboard	: Standard keyboard

SOFTWARE SPECIFICATION

• Operating system	: Windows OS
• Front End	: Python
• Back End	: MySQL SERVER
• IDLE	: Python 2.7 IDLE

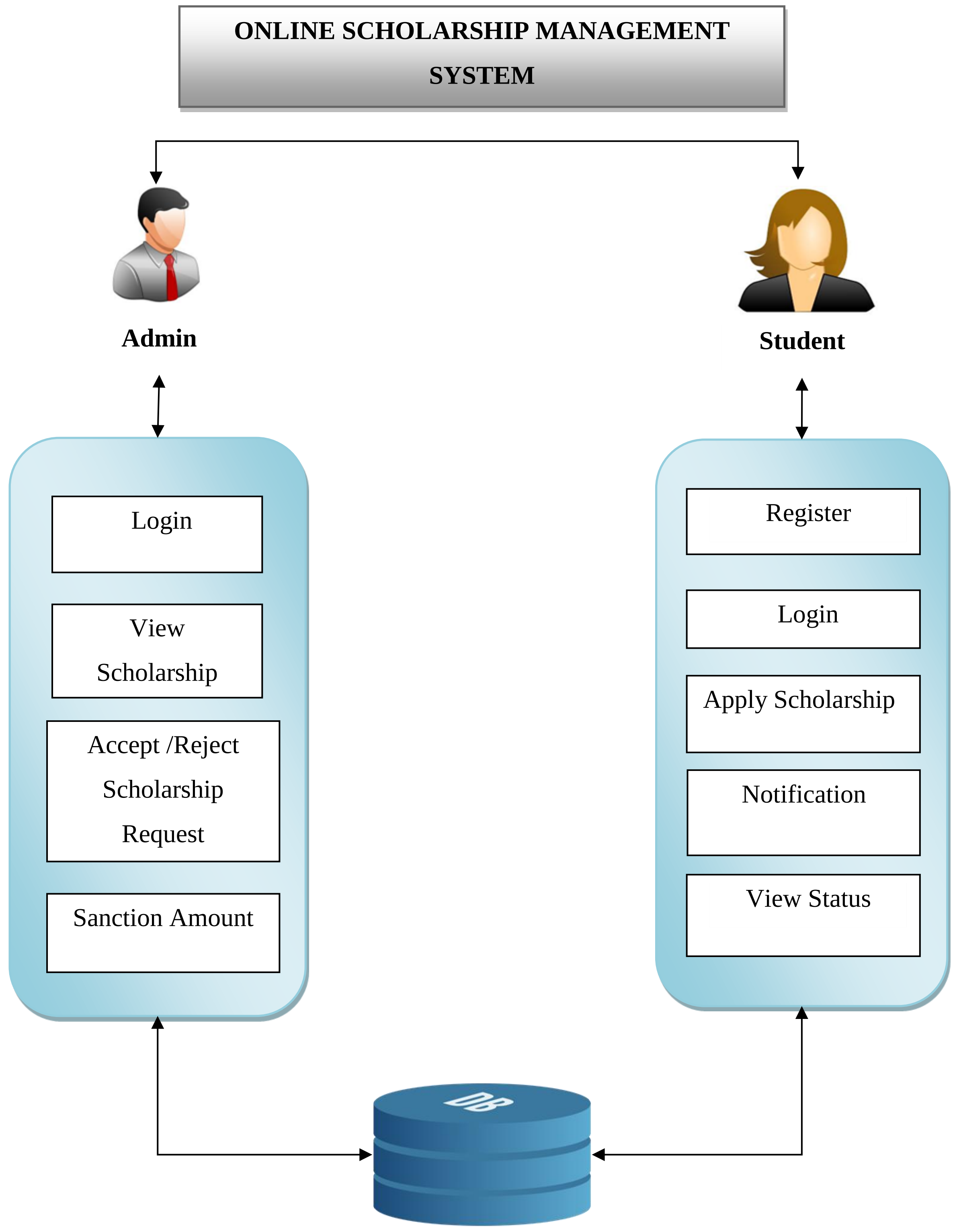
4. SYSTEM DESIGN

4.1 ARCHITECTURAL DESIGN

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs).

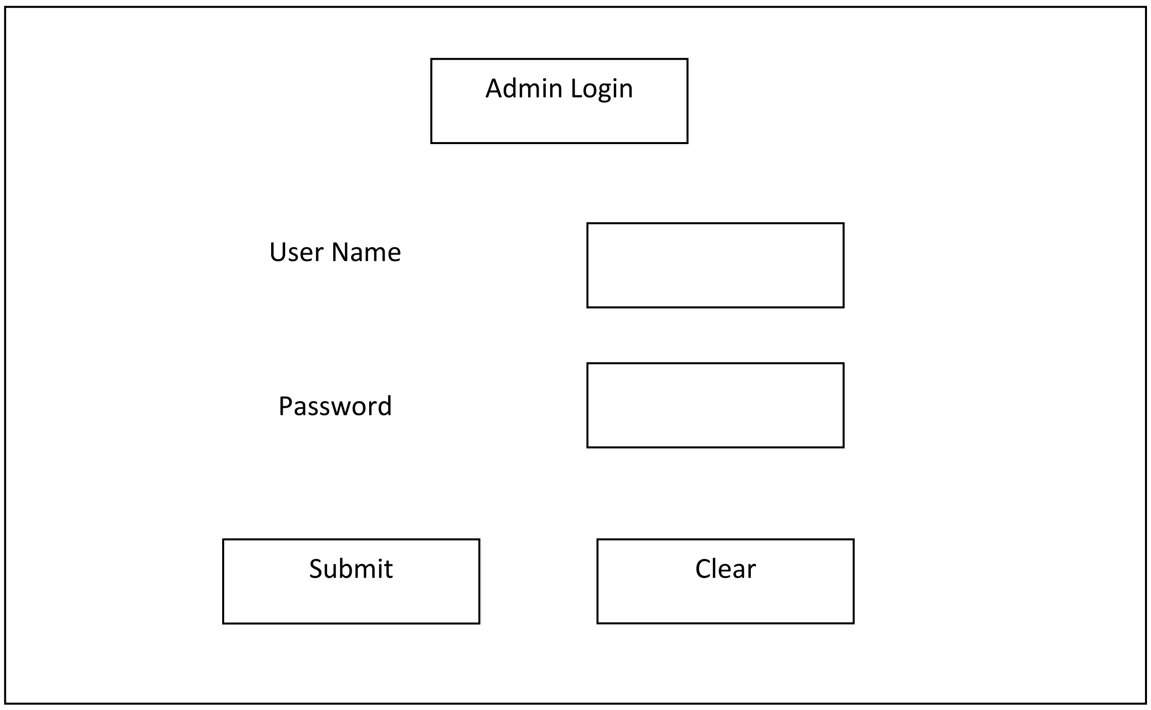
Various organizations define systems architecture in different ways, including:

- An allocated arrangement of physical elements which provides the design solution for a consumer product or life-cycle process intended to satisfy the requirements of the functional architecture and the requirements baseline.
- Architecture comprises the most important, pervasive, top-level, strategic inventions, decisions, and their associated rationales about the overall structure (i.e., essential elements and their relationships) and associated characteristics and behavior.
- If documented, it may include information such as a detailed inventory of current hardware, software and networking capabilities; a description of long-range plans and priorities for future purchases, and a plan for upgrading and/or replacing dated equipment and software
- The composite of the design architectures for products and their life-cycle processes.

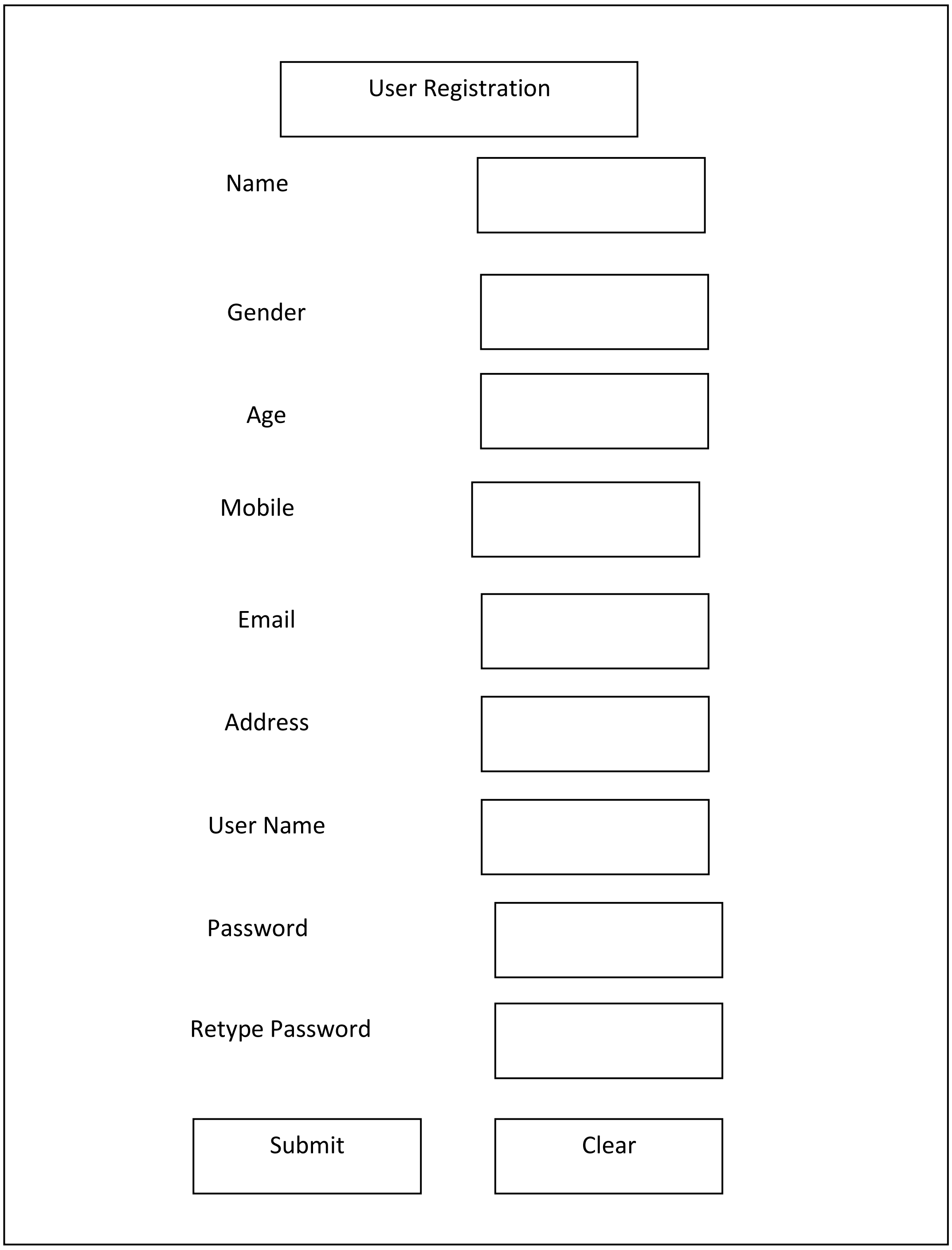

ONLINE SCHOLARSHIP MANAGEMENT SYSTEM
Admin
Student
Login
View Scholarship
Accept /Reject Scholarship Request
Sanction Amount
Register
Login
Apply Scholarship
Notification
View Status
DB

4.2 I/O FROM DESIGN

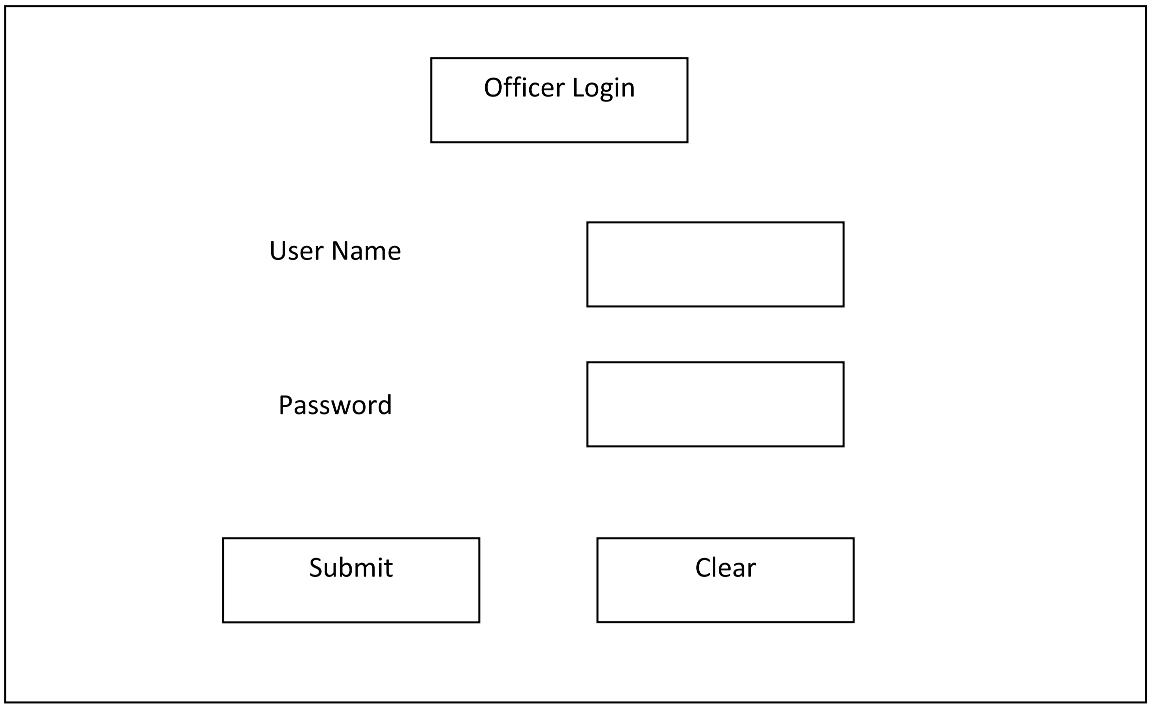
Admin Login



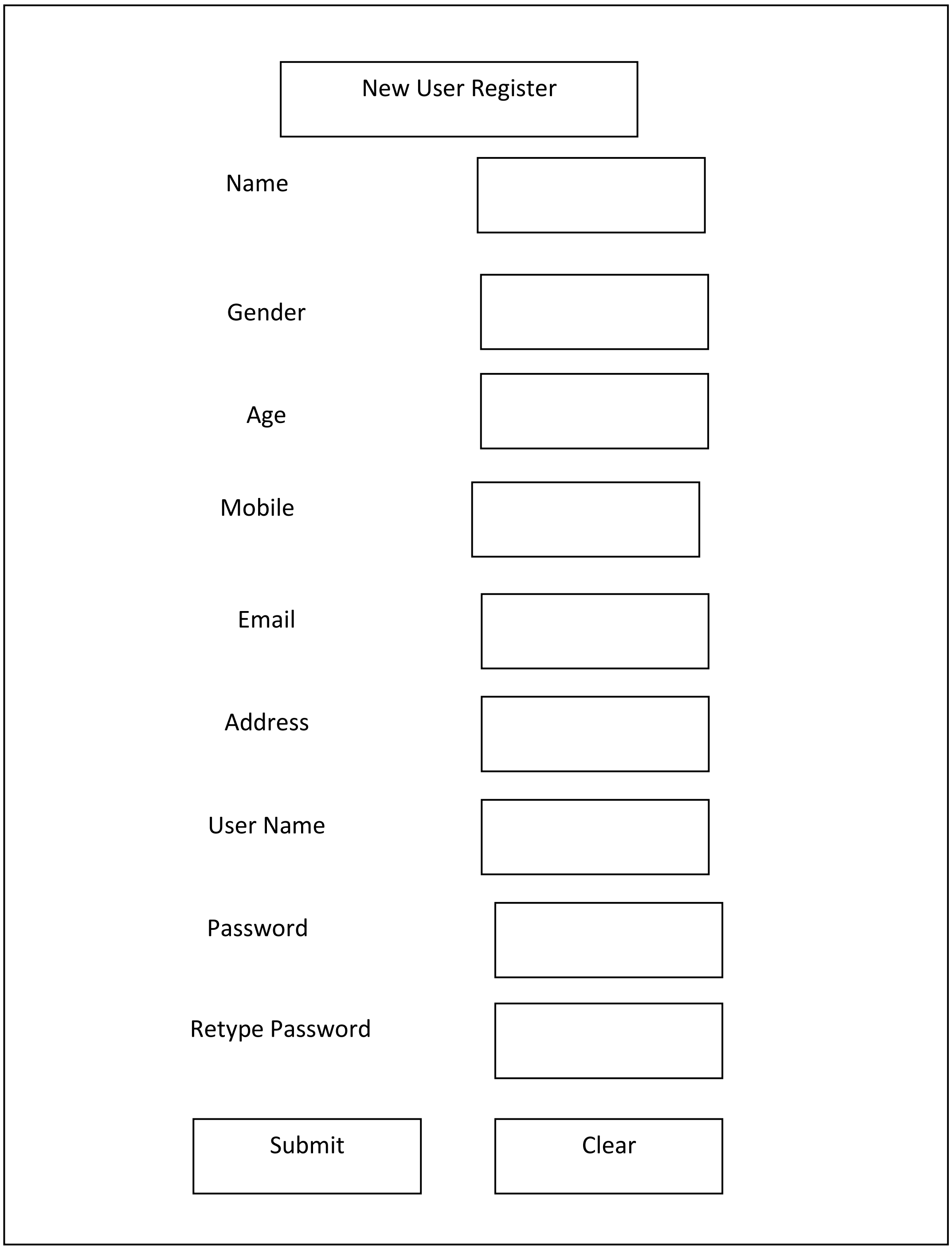
User Registration



Officer Login



New User Registration



4.3 TABLES

A table is a data structure that organizes information into rows and columns. It can be used to both store and display data in a structured format. For example, databases store data in tables so that information can be quickly accessed from specific rows. Websites often use tables to display multiple rows of data on page. Spreadsheets combine both purposes of a table by storing and displaying data in a structured format.

Databases often contain multiple tables, with each one designed for a specific purpose. For example, a company database may contain separate tables for employees, clients, and suppliers. Each table may include its own set of fields, based on what data the table needs to store. In database tables, each field is considered a column, while each entry (or record), is considered a row. A specific value can be accessed from the table by requesting data from an individual column and row.

Table Name: applytb

Field	**Type**	**Null**	**Default**
id	bigint(20)	Yes	NULL
UserName	varchar(250)	Yes	NULL
Mobile	varchar(250)	Yes	NULL
schloarship	varchar(250)	Yes	NULL
Certificate1	varchar(500)	Yes	NULL
Certificate2	varchar(500)	Yes	NULL
SAmount	varchar(20)	Yes	NULL
Percentage	varchar(250)	Yes	NULL
SanctionAmount	varchar(250)	Yes	NULL
Status	varchar(250)	Yes	NULL

Table Name: schloartb

Field	**Type**	**Null**	**Default**
id	bigint(10)	Yes	NULL
ScholarshipName	varchar(250)	Yes	NULL
ScholarshipType	varchar(250)	Yes	NULL

Announced	varchar(250)	Yes	NULL
Year	varchar(250)	Yes	NULL
Amount	varchar(250)	Yes	NULL
Payment	varchar(250)	Yes	NULL
Certificate1	varchar(250)	Yes	NULL
Certificate2	varchar(250)	Yes	NULL
OtherInfo	varchar(500)	Yes	NULL
Image	varchar(500)	Yes	NULL

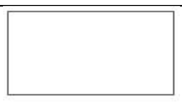
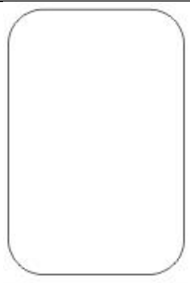
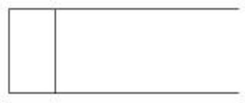
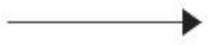
Table Name: studenttb

Field	**Type**	**Null**	**Default**
id	bigint(20)	Yes	NULL
RegisterNo	varchar(250)	Yes	NULL
Name	varchar(250)	Yes	NULL
Gender	varchar(250)	Yes	NULL
Mobile	varchar(250)	Yes	NULL
Email	varchar(250)	Yes	NULL
Address	varchar(500)	Yes	NULL
Department	varchar(250)	Yes	NULL
Batch	varchar(250)	Yes	NULL
Year	varchar(250)	Yes	NULL
Shift	varchar(250)	Yes	NULL

4.4 DATA FLOW DIAGRAM

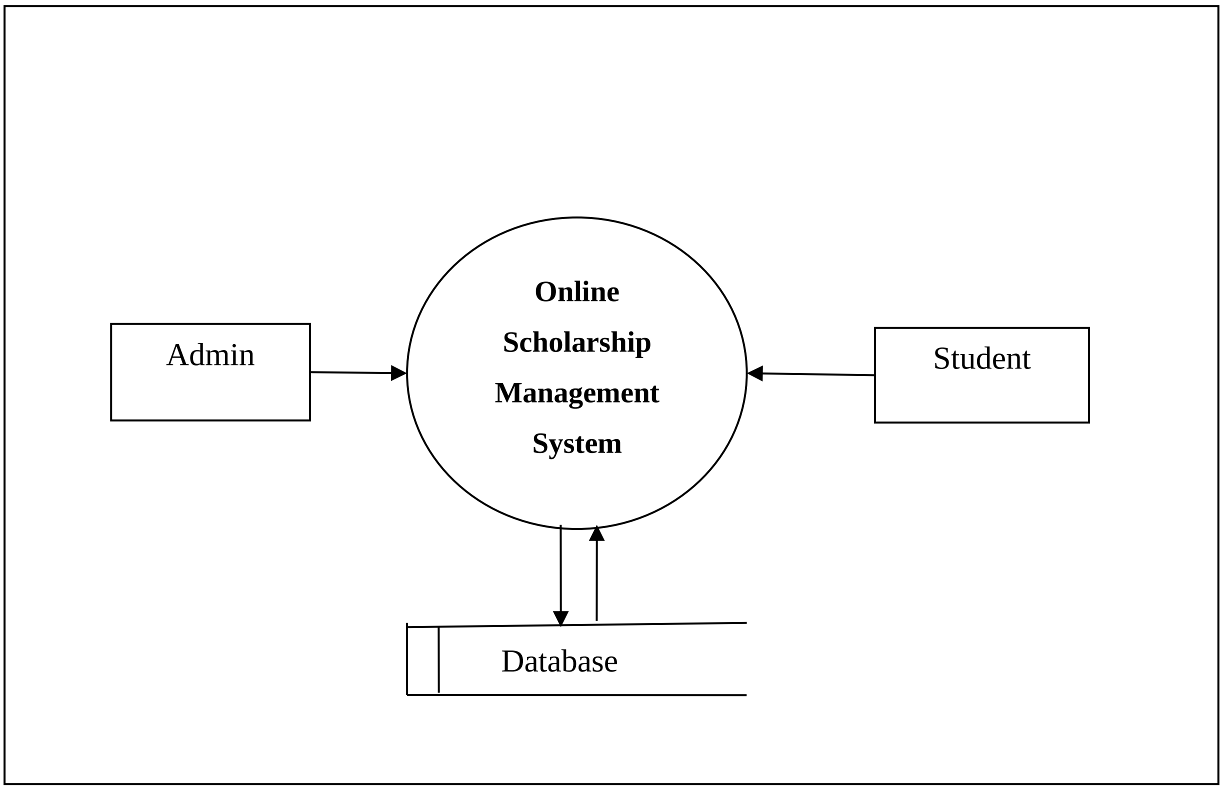
A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

Data flow Symbols:

Symbol	Description
	An **entity**. A source of data or a destination for data.
	A **process** or task that is performed by the system.
	A **data store**, a place where data is held between processes.
	A **data flow**.

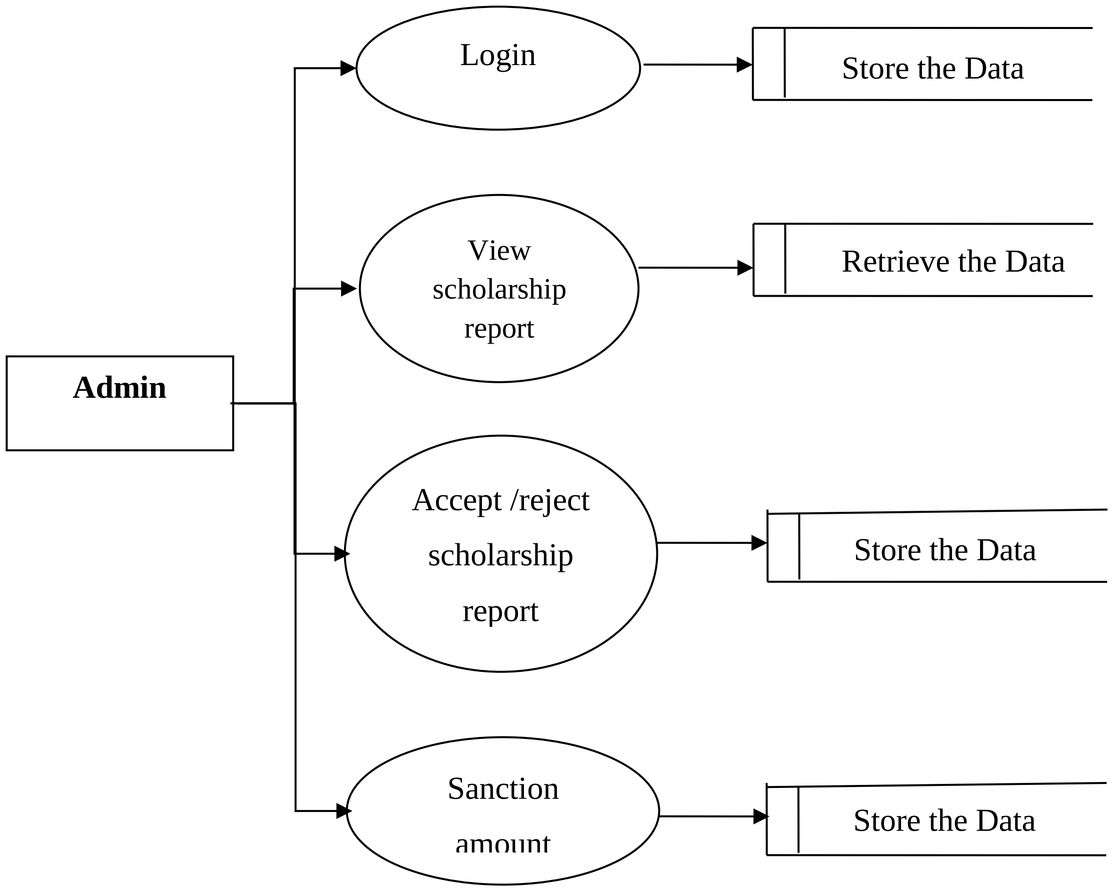
LEVEL 0

The Level 0 DFD shows how the system is divided into 'sub-systems' (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.



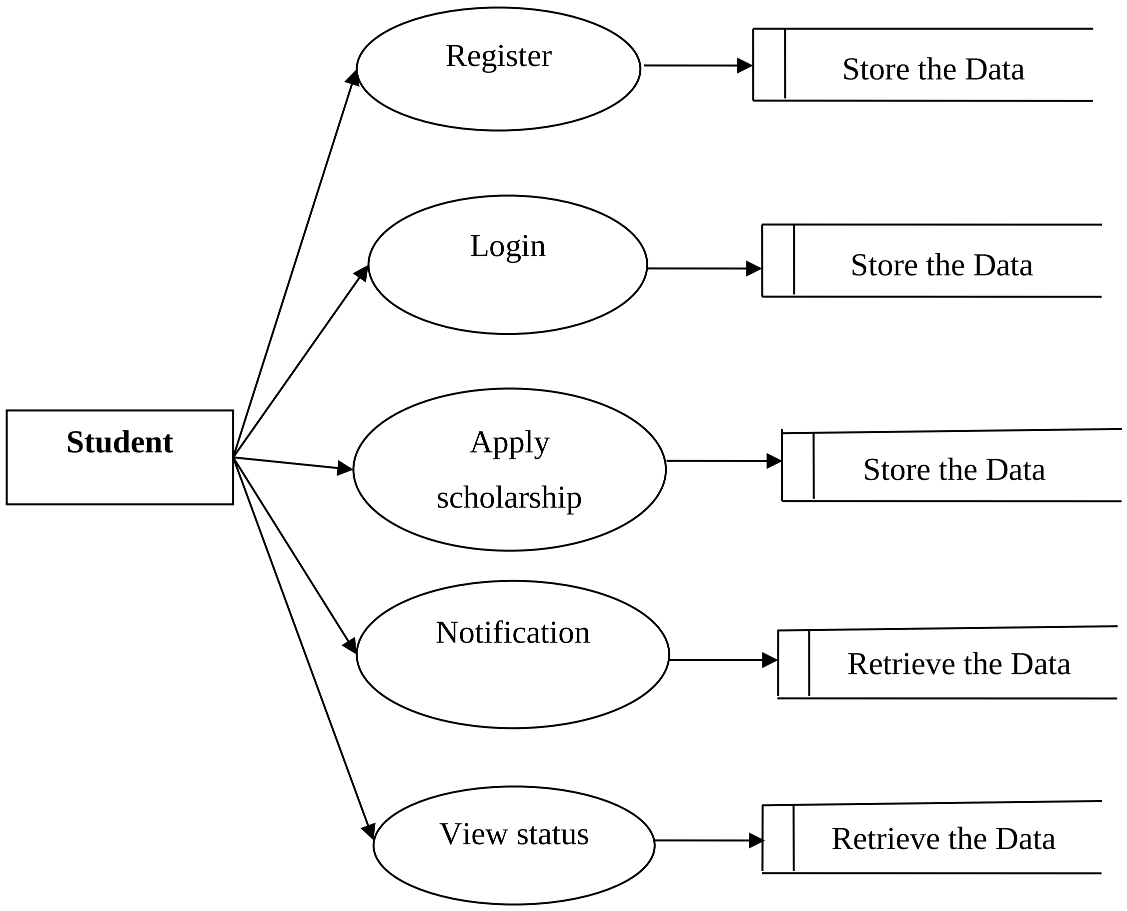
LEVEL 1

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.



LEVEL 2

A Data Flow Diagram (DFD) tracks processes and their data paths within the business or system boundary under investigation. A DFD defines each domain boundary and illustrates the logical movement and transformation of data within the defined boundary. The diagram shows 'what' input data enters the domain, 'what' logical processes the domain applies to that data, and 'what' output data leaves the domain. Essentially, a DFD is a tool for process modeling and one of the oldest.



4.5 NORMALIZATION

Database Normalization is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anamolies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables.

Normalization is used for mainly two purposes,

- Eliminating redundant (useless) data.
- Ensuring data dependencies make sense i.e data is logically stored.

Normalization Rule

Normalization rules are divided into the following normal forms:

1. First Normal Form
2. Second Normal Form
3. Third Normal Form

First Normal Form (1NF)

For a table to be in the First Normal Form, it should follow the following 4 rules:

1. It should only have single (atomic) valued attributes/columns.
2. Values stored in a column should be of the same domain
3. All the columns in a table should have unique names.
4. And the order in which data is stored, does not matter.

Id	Name	gende r	location	Address	phone	Age	Email
2374	Ram	Male	Kk nagar	Trichy	9876540388	25	ram@gmail.com
2375	Arul	Male	Anna nagar	Trichy	9787890878	30	raul@gmail.com

Second Normal Form (2NF)

For a table to be in the Second Normal Form,

1. It should be in the First Normal form.
2. And, it should not have Partial Dependency.

Id	Phone
2374	9876543210
2375	9876543211

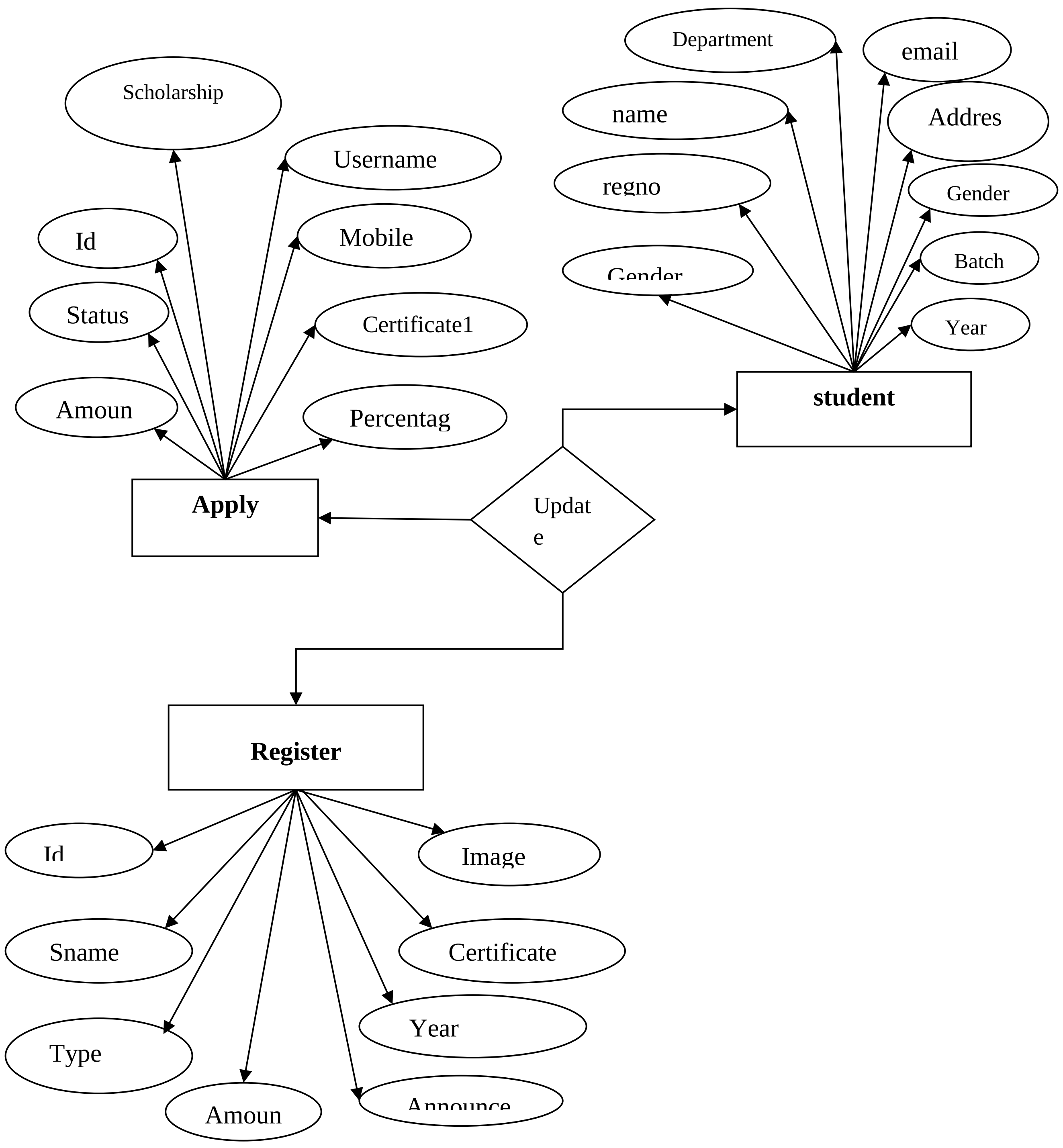
Third Normal Form (3NF)

A table is said to be in the Third Normal Form when,

1. It is in the Second Normal form.
2. And, it doesn't have Transitive Dependency.

Name	gender	location	Address	Age	Email
Ram	Male	Kk nagar	Trichy	25	ram@gmail.com
Arul	Male	Anna nagar	Trichy	30	raul@gmail.com

4.6 E-R DIAGRAM



4.7 DATA DICTIONARY

FIELD NAME	**TYPE**	**DESCRIPTION**	**SAMPLE VALUES**
id	bigint(20)	Specify the id	125454
UserName	varchar(250)	Specify the user name	Naveen
Mobile	varchar(250)	Specify the mobile	9632105487
schloarship	varchar(250)	Specify the scholarship	Ng
Certificate1	varchar(500)	Specify the certificate 1	File 1.png
Certificate2	varchar(500)	Specify the certificate 2	File 2.png
SAmount	varchar(20)	Specify the sanction amount	25000
Percentage	varchar(250)	Specify the percentage	2%
Status	varchar(250)	Specify the status	Successfully
ScholarshipName	varchar(250)	Specify the scholarship name	pg
ScholarshipType	varchar(250)	Specify the scholarship type	Education
Year	varchar(250)	Specify the year	2025
Amount	varchar(250)	Specify the amount	25000
Payment	varchar(250)	Specify the payment	25000
Image	varchar(500)	Specify the image	Image.jpg
RegisterNo	varchar(250)	Specify the register number	GTTY2562
Name	varchar(250)	Specify the name	Naveen
Gender	varchar(250)	Specify the gender	Male
Email	varchar(250)	Specify the email id	navee@gmail.com
Address	varchar(500)	Specify the address	Trichy
Department	varchar(250)	Specify the department	Bsc cs
Batch	varchar(250)	Specify the batch	3rd batch
Year	varchar(250)	Specify the year	2nd year

5. SYSTEM DEVELOPMENT

5.1 FUNCTIONAL DOCUMENTATION

MODULES

Admin

- Login
- View Scholarship Request
- Accept /Reject Scholarship Request
- Sanction Amount

Student

- Register
- Login
- Apply Scholarship
- Notification
- View Status

MODUEL DESCRIPTION

Admin

- Login
  In this module, the admin can login in the system using his/her username and password.
- View Scholarship Request
  In this module, the admin can view the student scholarship request.
- Accept /Reject Scholarship Request
  In this module, the admin can accept/ reject the student scholarship request.
- Sanction Amount
  In this module used to make a amount sanction. This module contains user's card details like name, card no, amount etc.

Student

- Register

There is registration form available where new student can create their account by providing required information to the system. The registration form details are like name, email, gender, mobile number, address, and etc. These details are stored in the database. And then can getting to the username and password in the system.

- Login

  In this module, the student can login in the system using username and password.
- Apply Scholarship

  In this module, the student can apply the scholarship.
- Notification

  If the admin can update the status of accept/ reject request and amount sanction, the student can get the automatic notification.
- View Status

  After received notification the student can check the status of accept/ reject request and amount sanction in this module.

5.2 SPECIAL FEATURES OF LANGUAGE/UTILITY

Frond End: Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. In July 2018, Van Rossum stepped down as the leader in the language community. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation. Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach. While offering choice in coding methodology, the Python philosophy rejects exuberant syntax (such as that of Perl) in favor of a simpler, less-cluttered grammar. As Alex Martelli put it: "To describe something as 'clever' is not considered a compliment in the Python culture."Python's philosophy rejects the Perl "there is more than one way to do it" approach to language design in favour of "there should be one—and preferably only one—obvious way to do it".

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of CPython that would offer marginal increases in speed at the cost of clarity.[When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. CPython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter. An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name a tribute to the British comedy group Monty Python and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard for and bar.

A common neologism in the Python community is pythonic, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called un pythonic. Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as Pythonists, Pythonistas, and Pythoneers. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Python’s initial development was spearheaded by Guido van Rossum in the late 1980s. Today, it is developed by the Python Software Foundation. Because Python is a multiparadigm language, Python programmers can accomplish their tasks using different styles of programming: object oriented, imperative, functional or reflective. Python can be used in Web development, numeric programming, game development, serial port access and more.

There are two attributes that make development time in Python faster than in other programming languages:

1. Python is an interpreted language, which precludes the need to compile code before executing a program because Python does the compilation in the background. Because Python is a high-level programming language, it abstracts many sophisticated details from the programming code. Python focuses so much on this abstraction that its code can be understood by most novice programmers.
2. Python code tends to be shorter than comparable codes. Although Python offers fast development times, it lags slightly in terms of execution time. Compared to fully compiling languages like C and C++, Python programs execute slower. Of course, with the processing speeds of computers these days, the speed differences are usually only observed in benchmarking tests, not in real-world operations. In most cases, Python is already included in Linux distributions and Mac OS X machines.

Back End: My SQL

MySQL is the world's most used open source relational database management system (RDBMS) as of 2008 that run as a server providing multi-user access to a number of databases. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack—LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python." Free-software-open source projects that require a full-featured database management system often use MySQL.For commercial use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: TYPO3, Joomla, Word Press, phpBB, MyBB, Drupal and other software built on the LAMP software stack. MySQL is also used in many high-profile, large-scale World Wide Web products, including Wikipedia, Google(though not for searches), ImagebookTwitter, Flickr, Nokia.com, and YouTube.

Inter images

MySQL is primarily an RDBMS and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, MySQL Workbench is actively developed by Oracle, and is freely available for use.

Graphical

The official MySQL Workbench is a free integrated environment developed by MySQL AB, that enables users to graphically administer MySQL databases and visually design database structures. MySQL Workbench replaces the previous package of software, MySQL GUI Tools. Similar to other third-party packages, but still considered the authoritative MySQL frontend, MySQL Workbench lets users manage database design & modeling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).MySQL Workbench is available in two editions, the regular free and open source Community Edition which may be downloaded from the MySQL website, and the proprietary Standard Edition which extends and improves the feature set of the Community Edition.

6. TESTING

6.1 TYPES OF TESTING DONE

Testing is a set activity that can be planned and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it is vital success of the system.

Testing Objectives:

There are several rules that can serve as testing objectives, they are

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

If testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrates that software functions appear to the working according to the specification, that performance requirements appear to have been met.

There are three ways to test a program

1. For Correctness
2. For Implementation efficiency
3. For Computational Complexity.

Tests used for implementation efficiency attempt to find ways to make a correct program faster or use less storage. It is a code-refining process, which reexamines the implementation phase of algorithm development. Tests for computational complexity amount to an experimental analysis of the complexity of an algorithm or an experimental comparison of two or more algorithms, which solve the same problem.

The data is entered in all forms separately and whenever an error occurred, it is corrected immediately. A quality team deputed by the management verified all the necessary documents and tested the Software while entering the data at all levels.

TYPES OF TESTING

The development process involves various types of testing. Each test type addresses a specific testing requirement. The most common types of testing involved in the development process are:

- Unit Test
- Integration Test
- System Test
- Validation Test

Unit Testing:

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. These units have specific behavior. The test done on these units of code is called unit test. Unit test depends upon the language on which the project is developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing:

In integration testing modules are combined and tested as a group. Modules are typically code modules, individual applications, source and destination applications on a network, etc. Integration Testing follows unit testing and precedes system testing. Testing after the product is code complete. Betas are often widely distributed or even distributed to the public at large in hopes that they will buy the final product when it is released.

System Testing

System testing is defined as testing of a complete and fully integrated software product. This testing falls in black-box testing wherein knowledge of the inner design of the code is not a pre-requisite and is done by the testing team. It is the final test to verify that the product to be delivered meets the specifications mentioned in the requirement document. It should investigate both functional and non-functional requirements.

Validation Testing

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. Validation Testing ensures that the product actually meets the client's needs.

7. USER MANUAL

7.1 INSTALLATION PROCEDURE

Step 1: Download the Full Installer

- Open a browser window and navigate to the Python.org Downloads page for Windows.
- Under the “Python Releases for Windows” heading, click the link for the Latest Python 3 Release - Python 3.x.x. As of this writing, the latest version was Python 3.8.4.
- Scroll to the bottom and select either Windows x86-64 executable installer for 64-bit or Windows x86 executable installer for 32-bit.
- If you aren’t sure whether to select the 32-bit or the 64-bit installer, then you can expand the box below to help you decide.
- When the installer is finished downloading, move on to the next step.

Step 2: Run the Installer

- Once you’ve chosen and downloaded an installer, run it by double-clicking on the downloaded file. A dialog box like the one below will appear:

There are four things to notice about this dialog box:

1. The default install path is in the `AppData/` directory of the current Windows user.
2. The Customize installation button can be used to customize the installation location and which additional features get installed, including `pip` and IDLE.
3. The Install launcher for all users (recommended) checkbox is checked default. This means every user on the machine will have access to the `py.exe` launcher. You can uncheck this box to restrict Python to the current Windows user.
4. The Add Python 3.8 to `PATH` checkbox is unchecked by default. There are several reasons that you might not want Python on `PATH`, so make sure you understand the implications before you check this box.

- The full installer gives you total control over the installation process.
- Customize the installation to meet your needs using the options available on the dialog box. Then click Install Now. That’s all there is to it!

8. CONCLUSION

8.1 SUMMARY OF THE PROJECT

In this work, an online mobile-based system e-scholarship was implemented that replaces the paper-based method. The online mobile -based application allowed students to apply for scholarship online irrespective of their geographical locations and also provide them with feedback services. The software was develop using object oriented analysis and design. It is hoped that effective implementation of this software product would eliminate many problems discovered during system's investigation. We therefore make the following recommendations. Online should endeavor to adopt the e-Scholarship System software develop in this research work in order to automate their various scholarship application processes. So that the workload associated with the manual system can be reduced considerably.

8.2 FUTURE POSSIBILITIES

The scope of the project is the system on which the software is installed, i.e. the project is developed as a Web based application, and it will work for a particular institute. But later on the project can be modified to operate it E-Scholarship. In future that can create a Mobile application for this project.

BIBLIOGRAPHY

BOOK REFERENCES

1. Heinold, Brian. "A practical introduction to Python programming." (2021).
2. Kneusel, Ronald T. Practical deep learning: A Python-based introduction. No Starch Press, 2021.
3. Dhruv, Akshit J., Reema Patel, and Nishant Doshi. "Python: the most advanced programming language for computer science applications." Science and Technology Publications, Lda (2021): 292-299.
4. Sundnes, Joakim. Introduction to scientific programming with Python. Springer Nature, 2020.
5. Hill, Christian. Learning scientific programming with Python. Cambridge University Press, 2020.

WEBSITE REFERENCES

1. https://medium.com/javarevisited/10-free-python-tutorials-and-courses-from-google-microsoft-and-coursera-for-beginners-96b9ad20b4e6
2. https://www.bestcolleges.com/bootcamps/guides/learn-python-free/
3. https://www.programiz.com/python-programming
4. https://realpython.com/
5. https://www.codecademy.com/learn/learn-python

APPENDIX

SAMPLE CODE

```
flask import Flask, render_template, request, session, flash,send_file

import mysql.connector


app = Flask(__name__)

app.config['SECRET_KEY'] = 'aaa'


@app.route('/')

def home():

return render_template('index.html')


@app.route('/AdminLogin')

def AdminLogin():

return render_template('AdminLogin.html')


@app.route('/NewStudent')

def NewStudent():

return render_template('NewStudent.html')


@app.route('/StudentLogin')

def StudentLogin():
```

```
return render_template('StudentLogin.html')


@app.route('/NewScholarship')

def NewScholarship():

return render_template('NewScholarship.html')


@app.route("/adminlogin", methods=['GET', 'POST'])

def adminlogin():

error = None

if request.method == 'POST':

if request.form['uname'] == 'admin' and request.form['password'] == 'admin':

conn = mysql.connector.connect(user='root', password='', host='localhost',

database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM studenttb")

data = cur.fetchall()

return render_template('AdminHome.html', data=data)


else:

return render_template('index.html', error=error)

@app.route("/AdminHome")

def AdminHome():
```

```
conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM studenttb")

data = cur.fetchall()

return render_template('AdminHome.html', data=data)


@app.route("/newsch", methods=['GET', 'POST'])

def newsch():

if request.method == 'POST':

regno = request.form['sname']

uname = request.form['Scholarship']

gender = request.form['Announced']

mobile = request.form['year']

email = request.form['amt']

address = request.form['payment']

depart = request.form['cer1']

Batch = request.form['cer2']

year = request.form['info']

import random

file = request.files['file']

fnew = random.randint(1111, 9999)

savename = str(fnew) + ".png"
```

```
file.save("static/upload/" + savename)


conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cursor = conn.cursor()

cursor.execute(

"insert into schloartb values('','" + regno + "','" + uname + "','" + gender + "','" + mobile + "','"
+ email + "','" + address + "' ,'" + depart + "','" + Batch + "','" + year + "','" + savename + "')")

conn.commit()

conn.close()


flash("Record Saved!")

return render_template('NewScholarship.html')


@app.route("/Remove")

def Remove():

id = request.args.get('id')

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cursor = conn.cursor()

cursor.execute(

"delete from schloartb where id='" + id + "'")

conn.commit()

conn.close()
```

```
flash('scholarship  info Remove Successfully!')

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM schloartb  ")

data = cur.fetchall()

return render_template('ScholarshipInfo.html', data=data)


@app.route("/ScholarshipInfo")

def ScholarshipInfo():

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM schloartb  ")

data = cur.fetchall()

return render_template('ScholarshipInfo.html', data=data)


@app.route("/newstudent", methods=['GET', 'POST'])

def newstudent():

if request.method == 'POST':

regno = request.form['regno']

uname = request.form['uname']

gender = request.form['gender']
```

```
mobile = request.form['mobile']

email = request.form['email']

address = request.form['Address']

depart = request.form['depart']

Batch = request.form['Batch']

year = request.form['year']

Shift = request.form['Shift']


conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cursor = conn.cursor()

cursor.execute(

"insert into studenttb values('','" + regno + "','" + uname + "','" + gender + "','" + mobile + "','"
+ email + "','" + address + "' ,'" + depart + "','" + Batch + "','" + year + "','" + Shift + "')")

conn.commit()

conn.close()


conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM studenttb  ")

data = cur.fetchall()


flash("Record Saved!")
```

```
return render_template('NewStudent.html', data=data)


@app.route("/studentlogin", methods=['GET', 'POST'])
def studentlogin():
if request.method == 'POST':
username = request.form['uname']
password = request.form['password']
session['uname'] = request.form['uname']


conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')
cursor = conn.cursor()
cursor.execute("SELECT * from studenttb where RegisterNo='" + username + "' and name='"
+ password + "'")
data = cursor.fetchone()
if data is None:
return render_template('index.html')
return 'Username or Password is wrong'
else:
session['mob'] = data[4]
conn = mysql.connector.connect(user='root', password='', host='localhost',

database='1collegesscholarshipdb')
cur = conn.cursor()
```

```
cur.execute("SELECT * FROM studenttb where RegisterNo='" + username + "' and Name='"
+ password + "'")

data = cur.fetchall()


flash("you are successfully logged in")

return render_template('StudentHome.html', data=data)


@app.route('/StudentHome')

def StudentHome():


regno = session['uname']

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM studenttb where RegisterNo='" + regno + "' ")

data = cur.fetchall()

return render_template('StudentHome.html', data=data)


@app.route("/Search")

def Search():

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM schloartb  ")
```

```
data = cur.fetchall()

return render_template('Search.html', data=data)



@app.route("/search", methods=['GET', 'POST'])

def search():

if request.method == 'POST':

Scholarship = request.form['Scholarship']



conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM schloartb  where ScholarshipType='" + Scholarship + "'")

data = cur.fetchall()

return render_template('Search.html', data=data)



@app.route("/Apply")

def Apply():

id = request.args.get('id')

session['id'] = id



conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cursor = conn.cursor()
```

```
cursor.execute("SELECT  *  FROM schloartb  where  id='" + id + "'")

data = cursor.fetchone()


if data:

sname = data[1]

pay = data[6]


else:

return 'No Record Found!'


return render_template('Apply.html', sname=sname, pay=pay)


@app.route("/apply", methods=['GET', 'POST'])

def apply():

if request.method == 'POST':

sname = request.form['sname']

payment = request.form['payment']

mark = request.form['Mark']

pmark = (float(mark) / 100) * float(payment)

file1 = request.files['file1']

file1.save("static/upload/" + file1.filename)

file2 = request.files['file2']

file2.save("static/upload/" + file2.filename)
```

```
conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cursor = conn.cursor()

cursor.execute(

"insert into applytb values('','" + session['uname'] + "','" + session['mob'] + "','" + sname + "','"
+ file1.filename + "','" + file2.filename + "','"+ payment +"','"+ mark +"','"+ str(pmark)
+"','waiting')")

conn.commit()

conn.close()


flash("Record Saved!")

return render_template('Apply.html')


@app.route("/Status")

def Status():

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM applytb where UserName='" + session['uname'] + "'   ")

data = cur.fetchall()

return render_template('Status.html', data=data)


@app.route("/ApplyInfo")
```

```
def ApplyInfo():

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM applytb where status='waiting'   ")

data = cur.fetchall()

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM applytb")

data1 = cur.fetchall()

return render_template('ApplyInfo.html', data=data,data1=data1)


@app.route("/Accept")

def Accept():

id = request.args.get('id')

mob = request.args.get('mob')

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cursor = conn.cursor()

cursor.execute(

"update  applytb set status='Approved' where id='" + id + "'")

conn.commit()

conn.close()
```

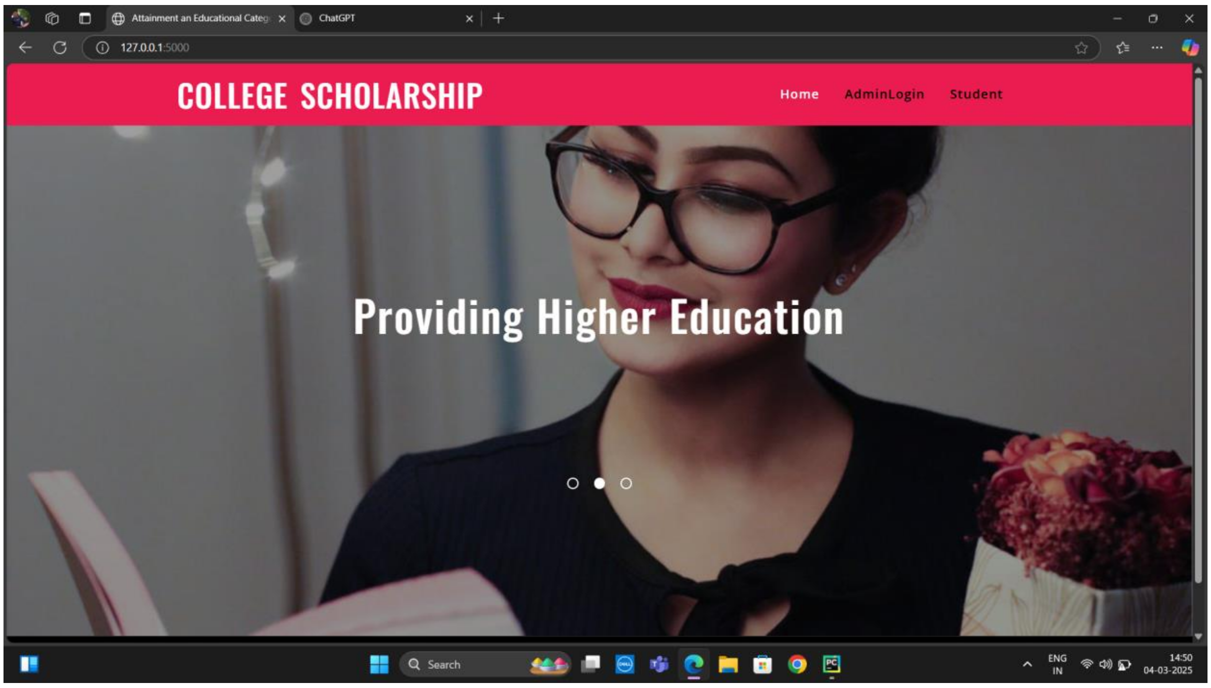
```
flash('scholarship  info Approved Successfully!')

sendmsg(mob, 'scholarship  info Approved Successfully!')

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM applytb where status='waiting' ")

data = cur.fetchall()


conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM applytb where status !='waiting' ")

data1 = cur.fetchall()

return render_template('ScholarshipInfo.html', data=data,data1=data1)


@app.route("/Reject")

def Reject():

id = request.args.get('id')

mob = request.args.get('mob')

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cursor = conn.cursor()

cursor.execute(

"update  applytb set status='Reject' where id='" + id + "'")
```

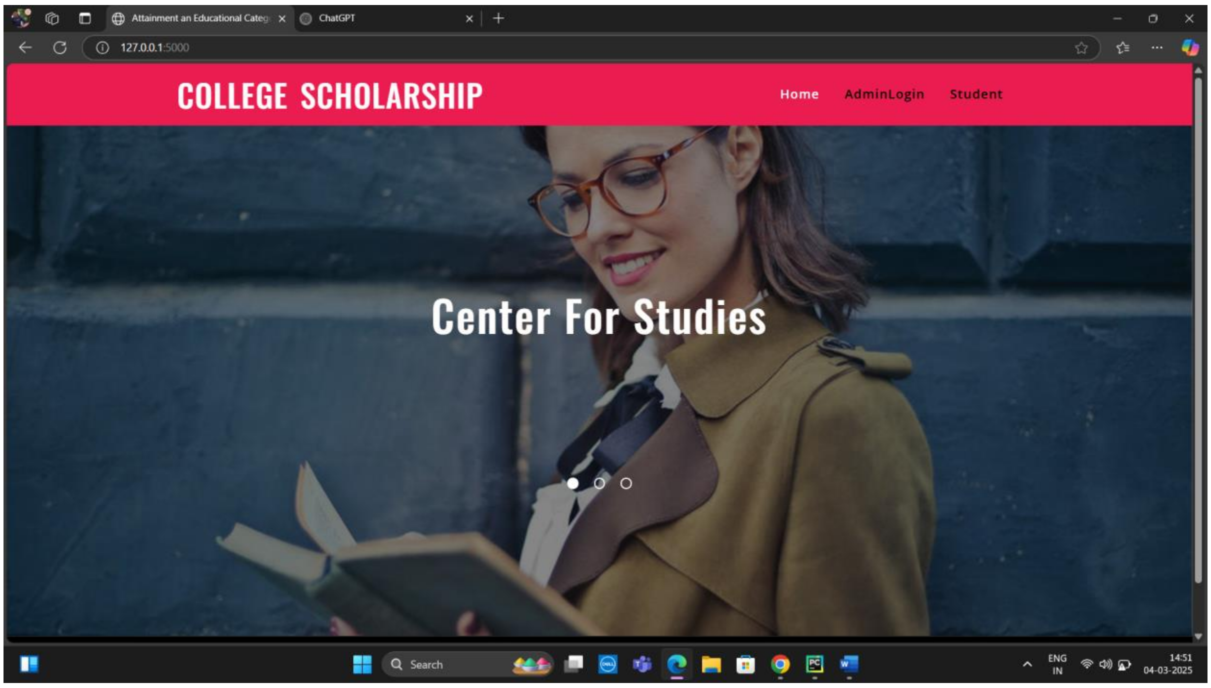
```
conn.commit()

conn.close()

flash('scholarship  info Reject !')

sendmsg(mob,'cholarship  info Reject')

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM applytb where status='waiting' ")

data = cur.fetchall()


conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cur = conn.cursor()

cur.execute("SELECT * FROM applytb where status !='waiting' ")

data1 = cur.fetchall()


return render_template('ScholarshipInfo.html', data=data,data1=data1)


@app.route("/down1")

def down1():

id = request.args.get('id')

print(id)

conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')
```

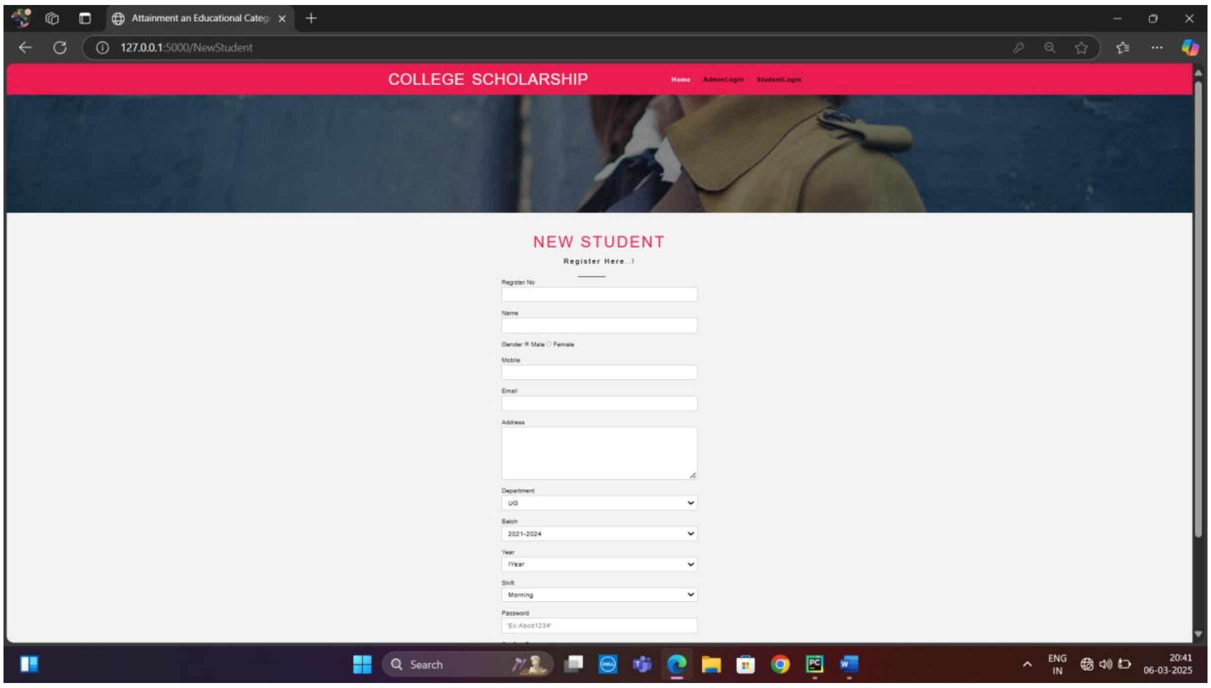
```
cursor = conn.cursor()

cursor.execute("SELECT * from applytb where id ='" + id + "'   ")

data = cursor.fetchone()

if data is None:



return 'Assingment Not Upload'

else:

print(data[4])

filename = data[4]



return send_file('static/upload/' + filename, as_attachment=True)

@app.route("/down2")

def down2():

id = request.args.get('id')

print(id)



conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1collegesscholarshipdb')

cursor = conn.cursor()

cursor.execute("SELECT * from applytb where id ='" + id + "'   ")

data = cursor.fetchone()

if data is None:
```

```
return 'Assingment Not Upload'

else:

print(data[5])

filename = data[5]


return send_file('static/upload/' + filename, as_attachment=True)


def sendmsg(targetno,message):

import requests

requests.post("http://smsserver9.creativepoint.in/api.php?username=fantasy&password=5966
92&to=" + targetno + "&from=FSSMSS&message=Dear user  your msg is " + message + "
Sent By FSMSG
FSSMSS&PEID=1501563800000030506&templateid=1507162882948811640")


if __name__ == '__main__':

app.run(debug=True, use_reloader=True)
```

SAMPLE OUTPUTS







NEW STUDENT

Register Here..!

Register No

Name

Gender [✓ Male] Female

Mobile

Email

Address

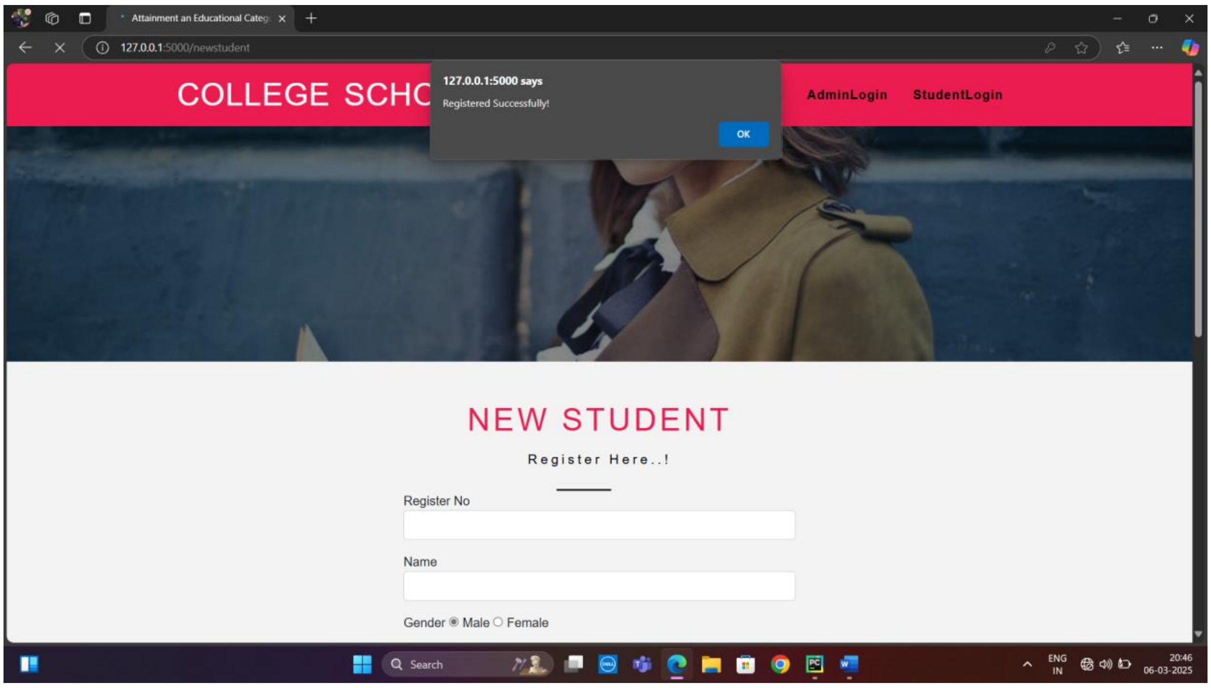
Department
UG

Batch
2021-2024

Year
IYear

Shift
Morning

Password
'Ex:Abcd1234'



127.0.0.1:5000 says

Registered Successfully!

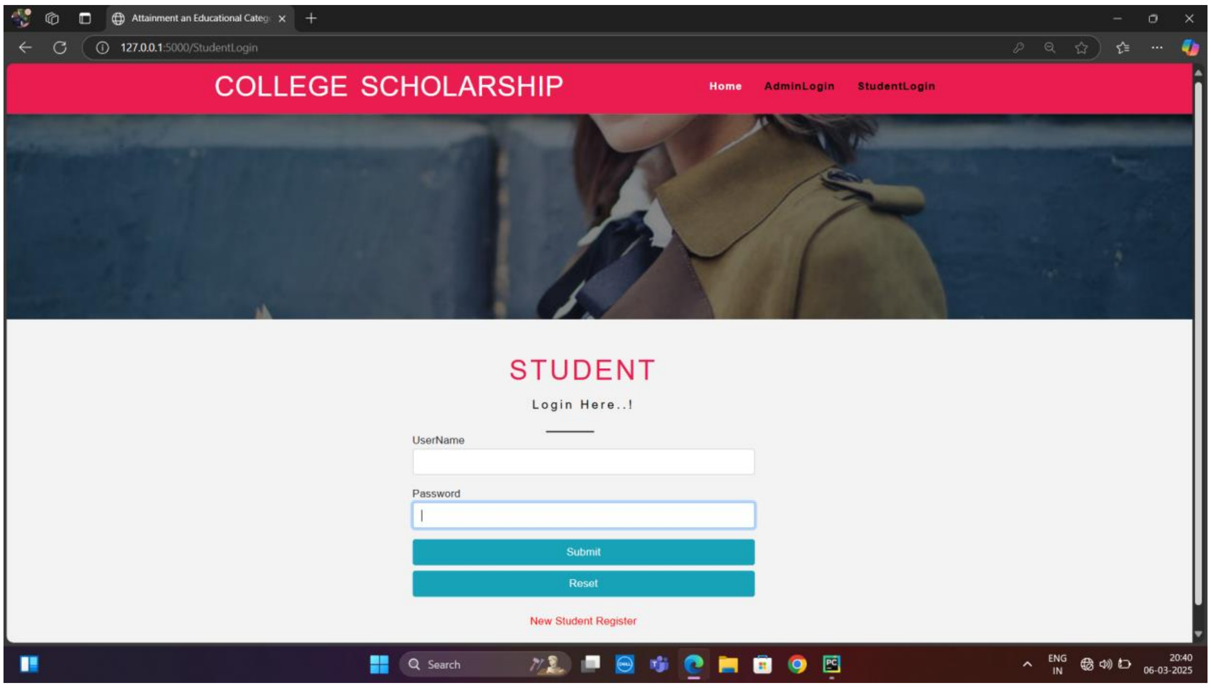
NEW STUDENT

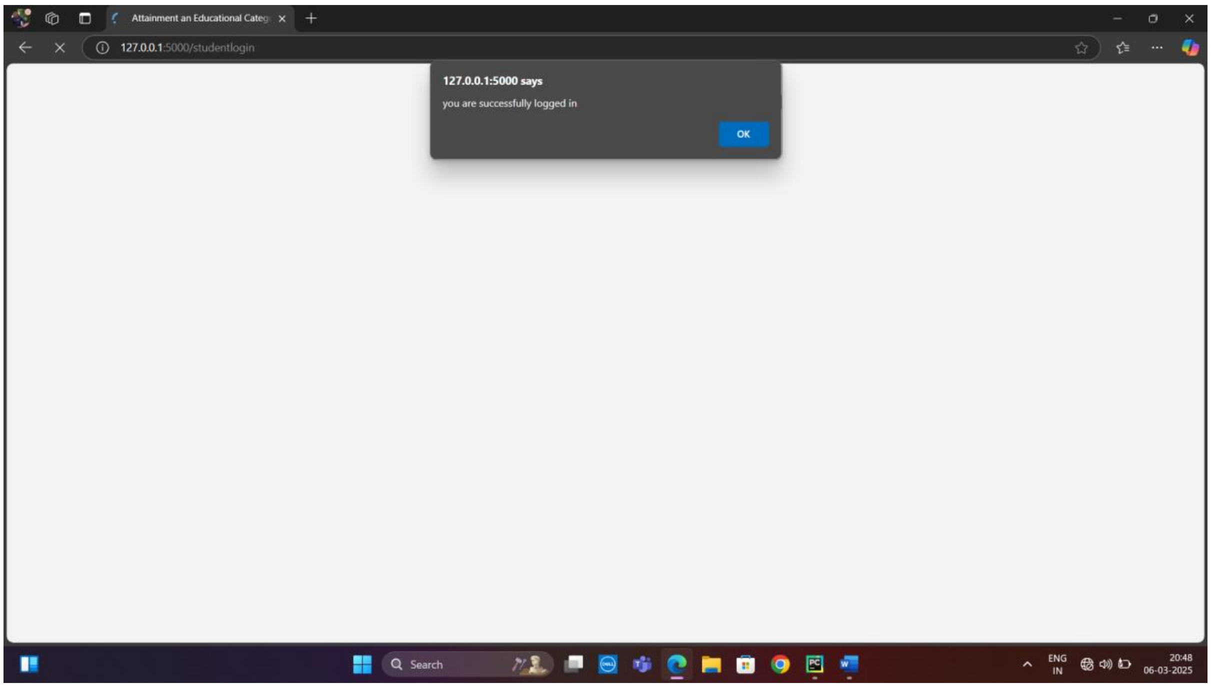
Register Here..!

Register No

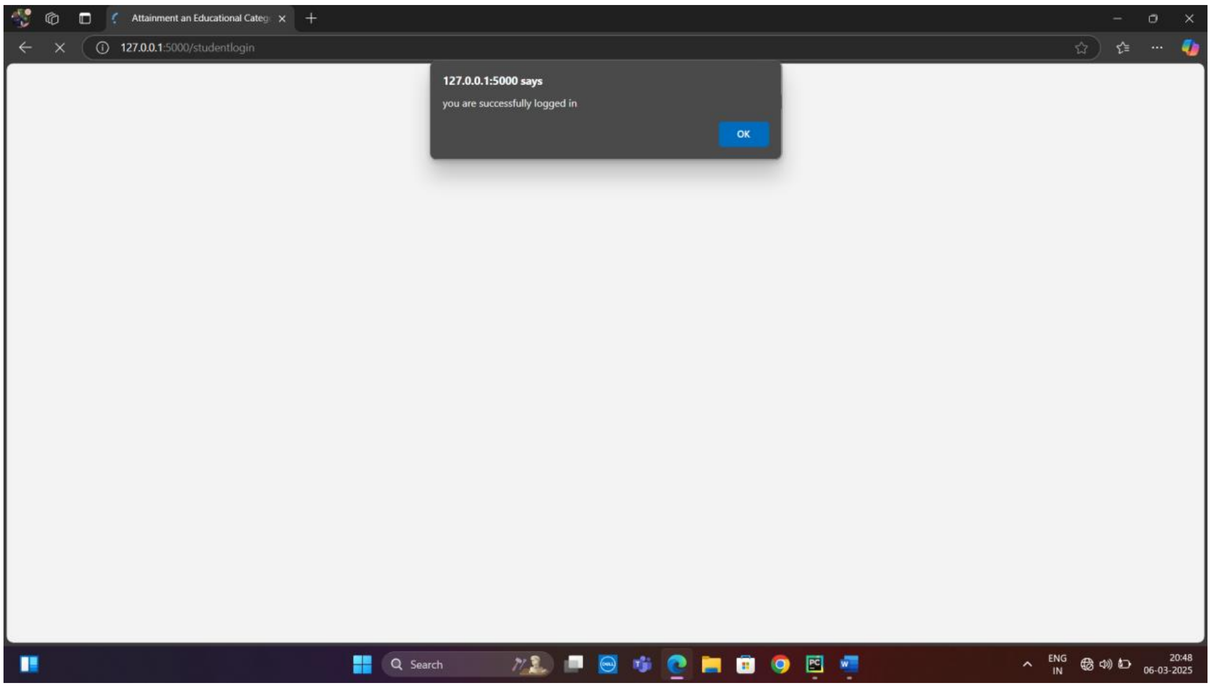
Name

Gender [✓ Male] Female



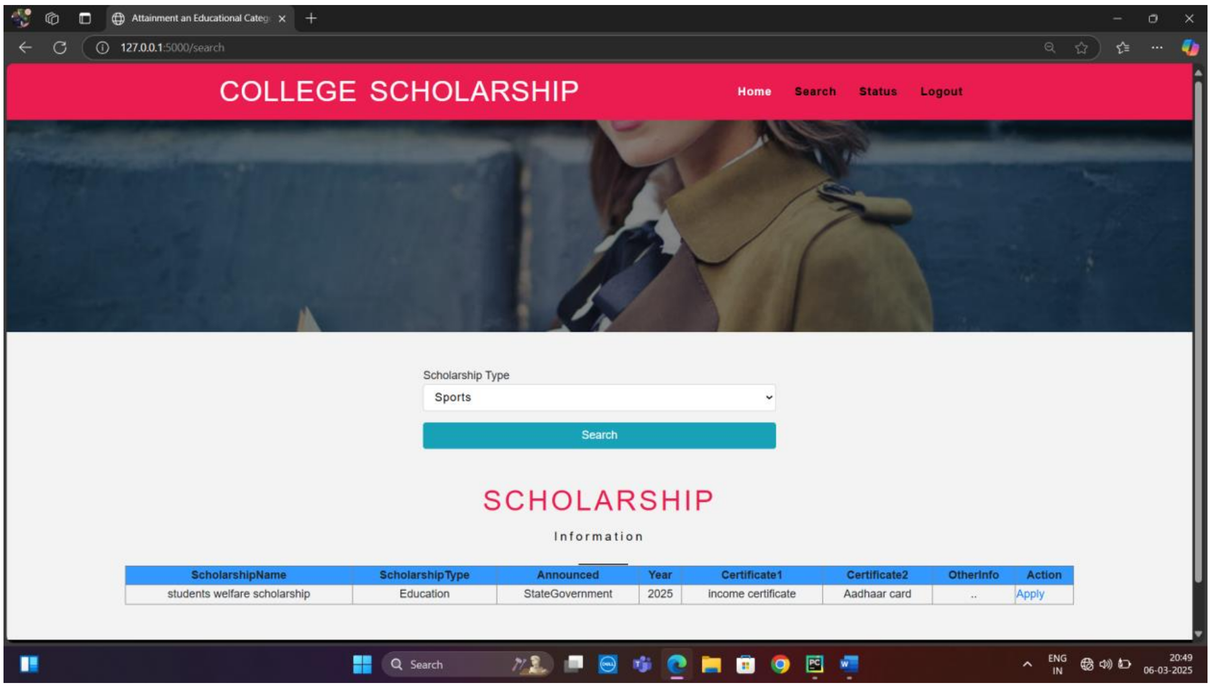


you are successfully logged in


127.0.0.1:5000 says

you are successfully logged in

OK



Scholarship Type

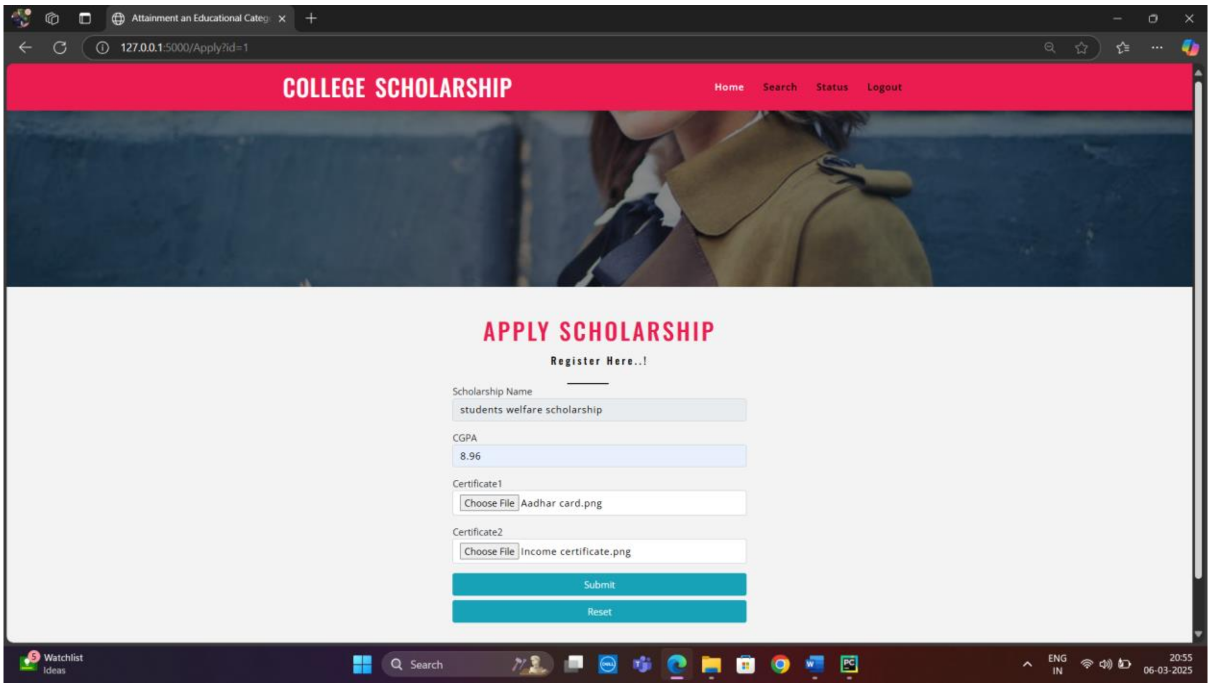
Sports

Search

SCHOLARSHIP

Information

ScholarshipName	ScholarshipType	Announced	Year	Certificate1	Certificate2	OtherInfo	Action
students welfare scholarship	Education	StateGovernment	2025	income certificate	Aadhaar card	..	Apply



APPLY SCHOLARSHIP

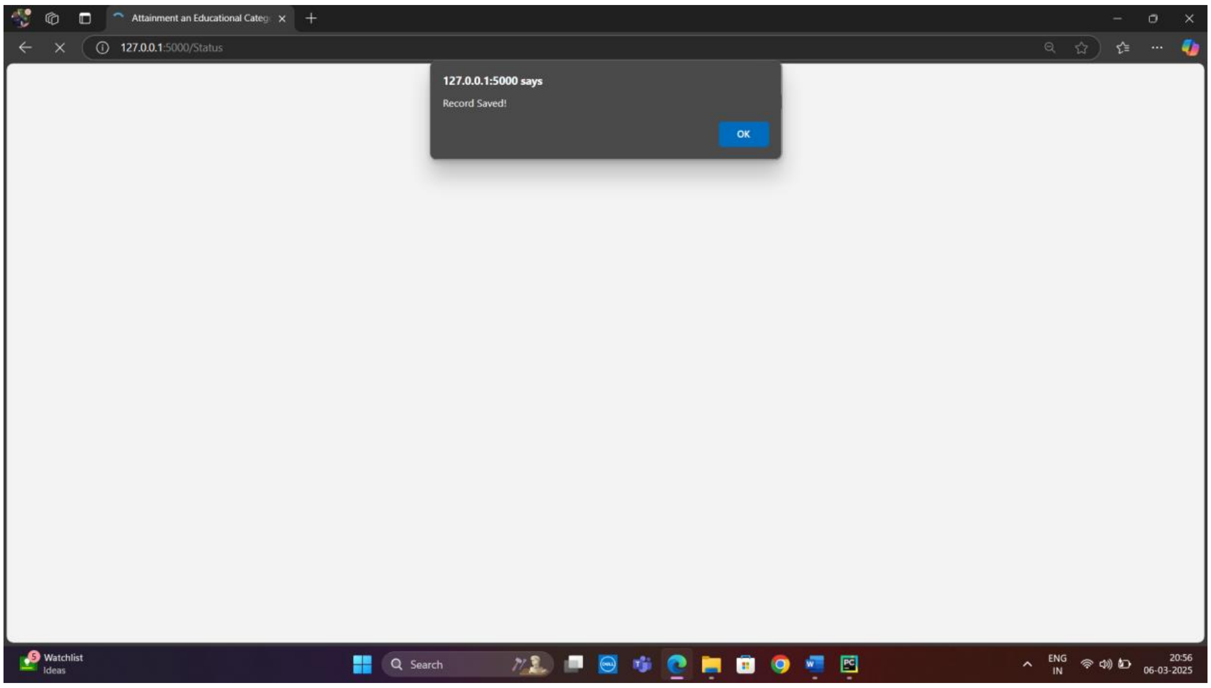
Register Here..!

Scholarship Name: students welfare scholarship

CGPA: 8.96

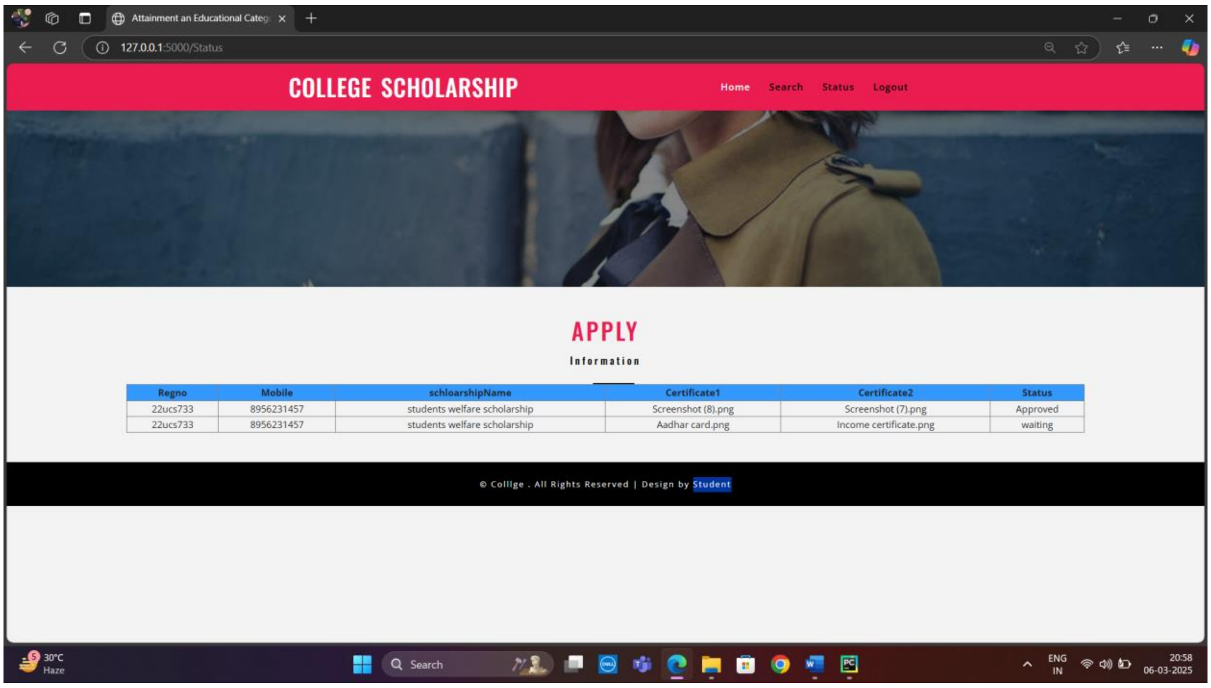
Certificate1: Choose File Aadhar card.png

Certificate2: Choose File Income certificate.png



127.0.0.1:5000 says

Record Saved!

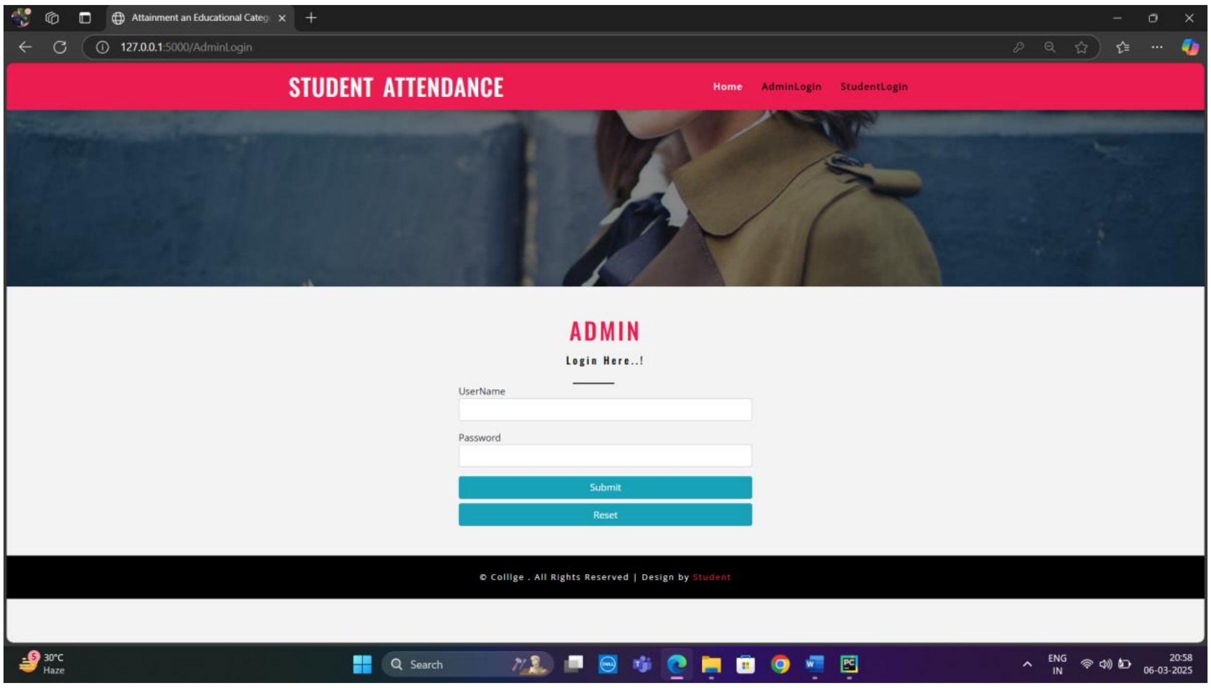


APPLY

Information

Regno	Mobile	schloarshipName	Certificate1	Certificate2	Status
22ucs733	8956231457	students welfare scholarship	Screenshot (8).png	Screenshot (7).png	Approved
22ucs733	8956231457	students welfare scholarship	Aadhar card.png	Income certificate.png	waiting

© Colllge . All Rights Reserved | Design by Student



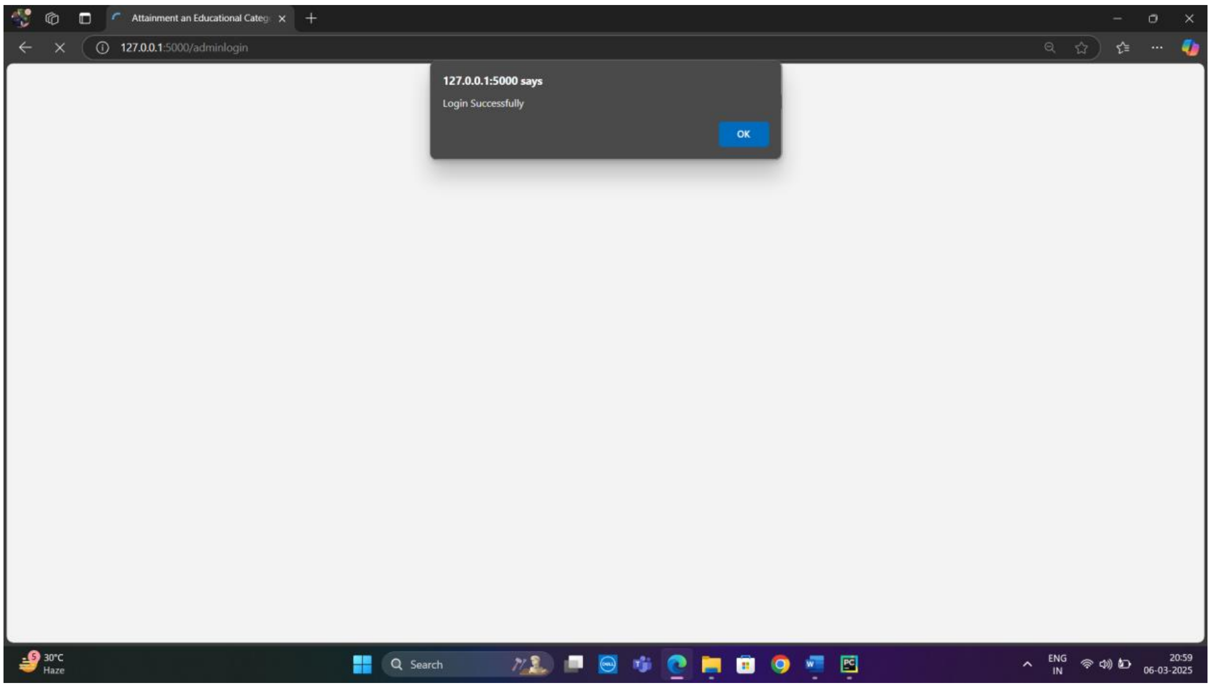
ADMIN

Login Here..!

UserName

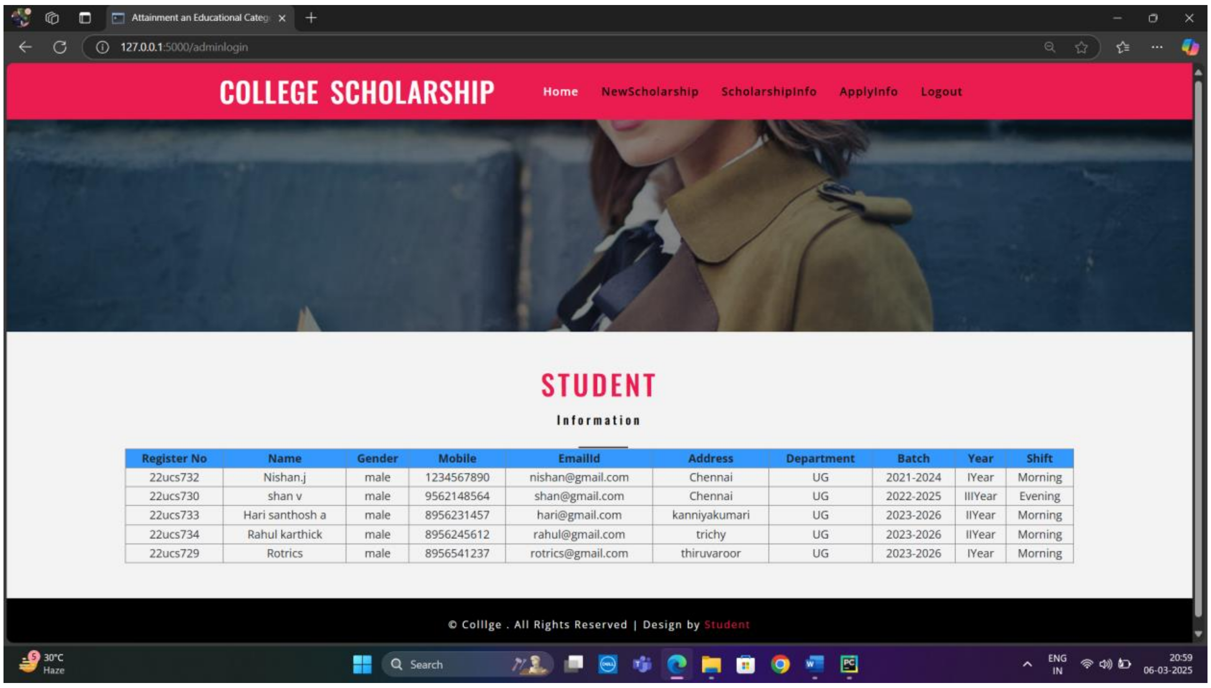
Password

© Colllge . All Rights Reserved | Design by Student



127.0.0.1:5000 says

Login Successfully

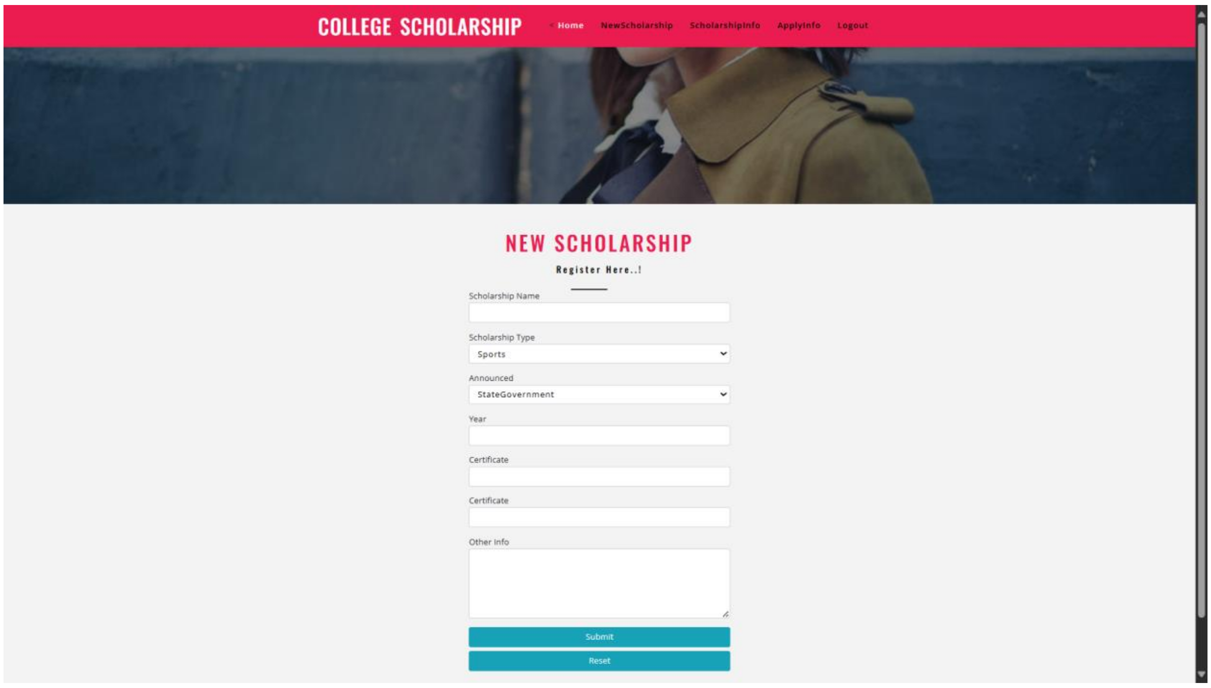


STUDENT

Information

Register No	Name	Gender	Mobile	EmailId	Address	Department	Batch	Year	Shift
22ucs732	Nishan.j	male	1234567890	nishan@gmail.com	Chennai	UG	2021-2024	IYear	Morning
22ucs730	shan v	male	9562148564	shan@gmail.com	Chennai	UG	2022-2025	IIIYear	Evening
22ucs733	Hari santhosh a	male	8956231457	hari@gmail.com	kanniyakumari	UG	2023-2026	IIYear	Morning
22ucs734	Rahul karthick	male	8956245612	rahul@gmail.com	trichy	UG	2023-2026	IIYear	Morning
22ucs729	Rotrics	male	8956541237	rotrics@gmail.com	thiruvaroor	UG	2023-2026	IYear	Morning

© Colllge . All Rights Reserved | Design by Student


COLLEGE SCHOLARSHIP

Home NewScholarship ScholarshipInfo ApplyInfo Logout

NEW SCHOLARSHIP

Register Here..!

Scholarship Name

Scholarship Type: Sports

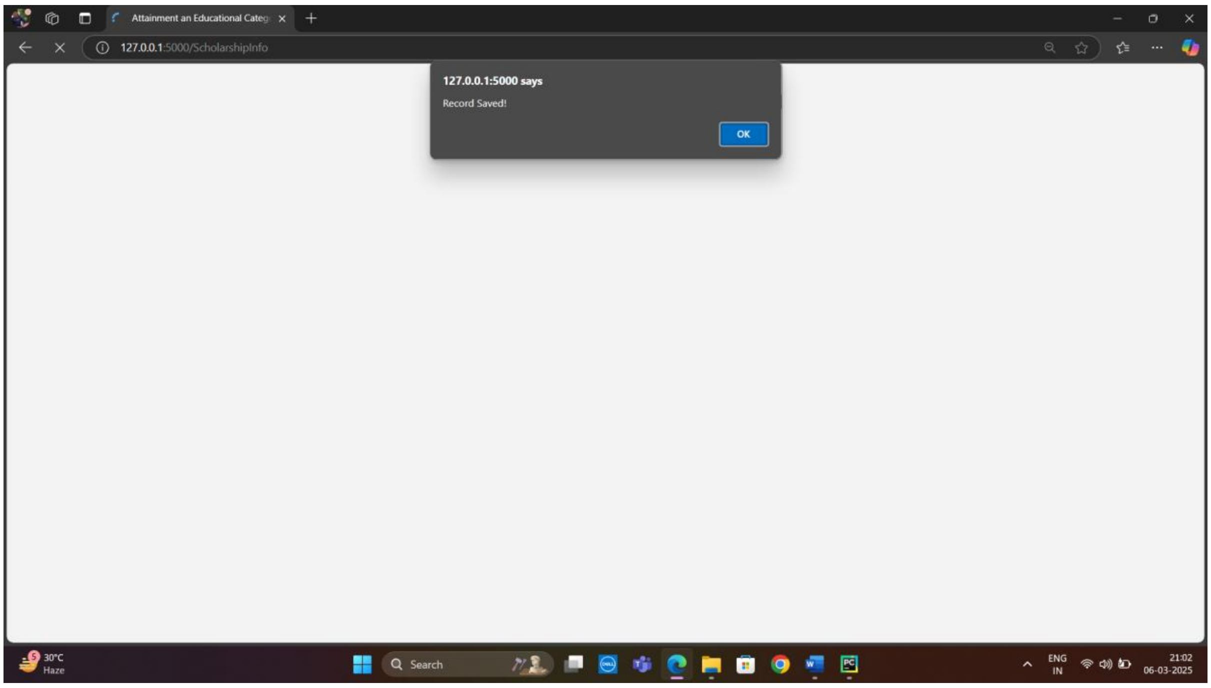
Announced: StateGovernment

Year

Certificate

Certificate

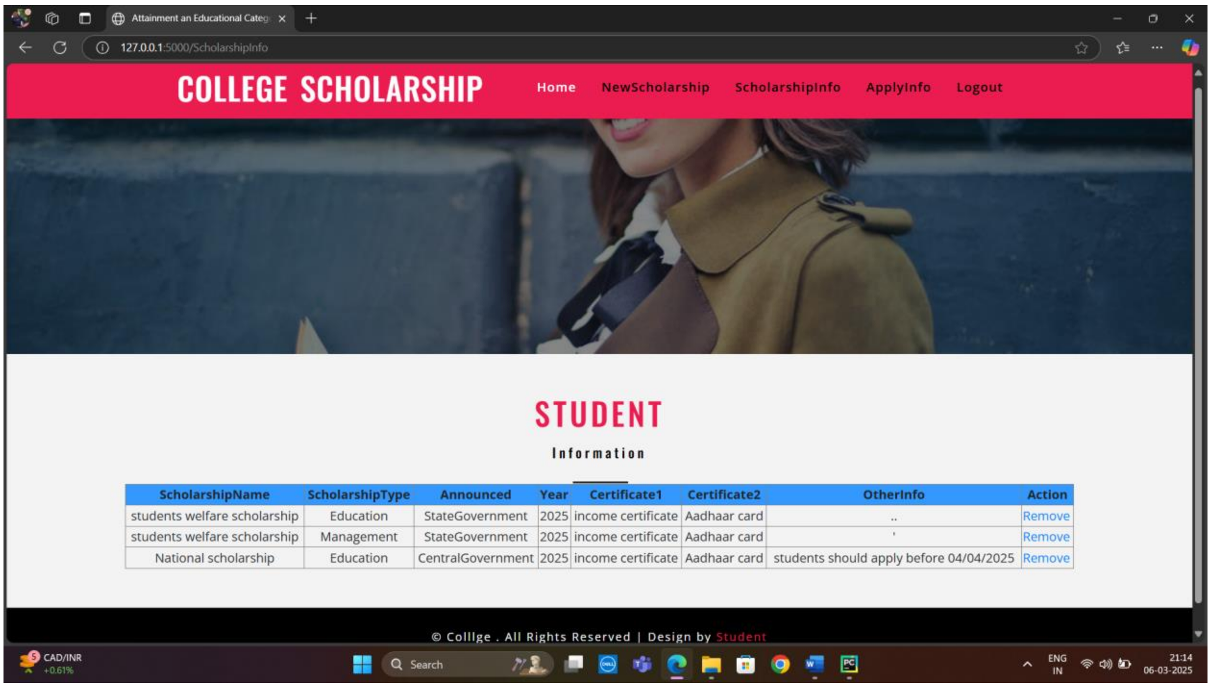
Other Info


Attainment an Educational Categ

127.0.0.1:5000/ScholarshipInfo

127.0.0.1:5000 says

Record Saved!

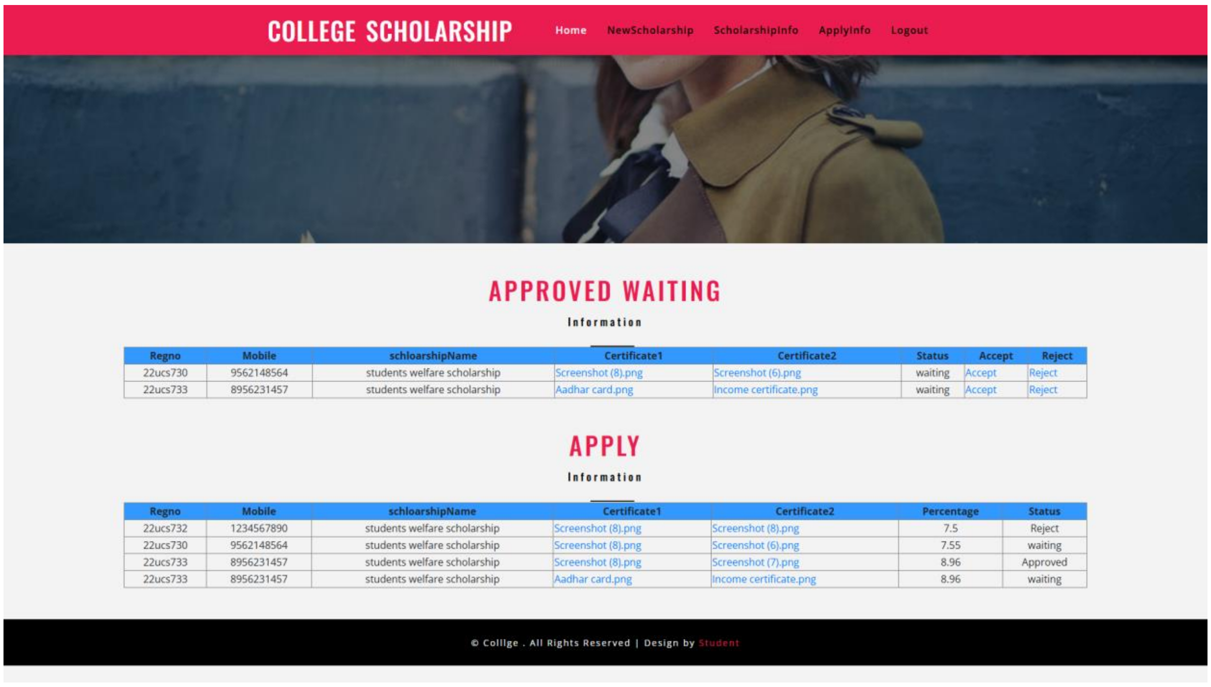


STUDENT

Information

ScholarshipName	ScholarshipType	Announced	Year	Certificate1	Certificate2	OtherInfo	Action
students welfare scholarship	Education	StateGovernment	2025	income certificate	Aadhaar card	..	Remove
students welfare scholarship	Management	StateGovernment	2025	income certificate	Aadhaar card	'	Remove
National scholarship	Education	CentralGovernment	2025	income certificate	Aadhaar card	students should apply before 04/04/2025	Remove

© Colllge . All Rights Reserved | Design by Student



APPROVED WAITING

Information

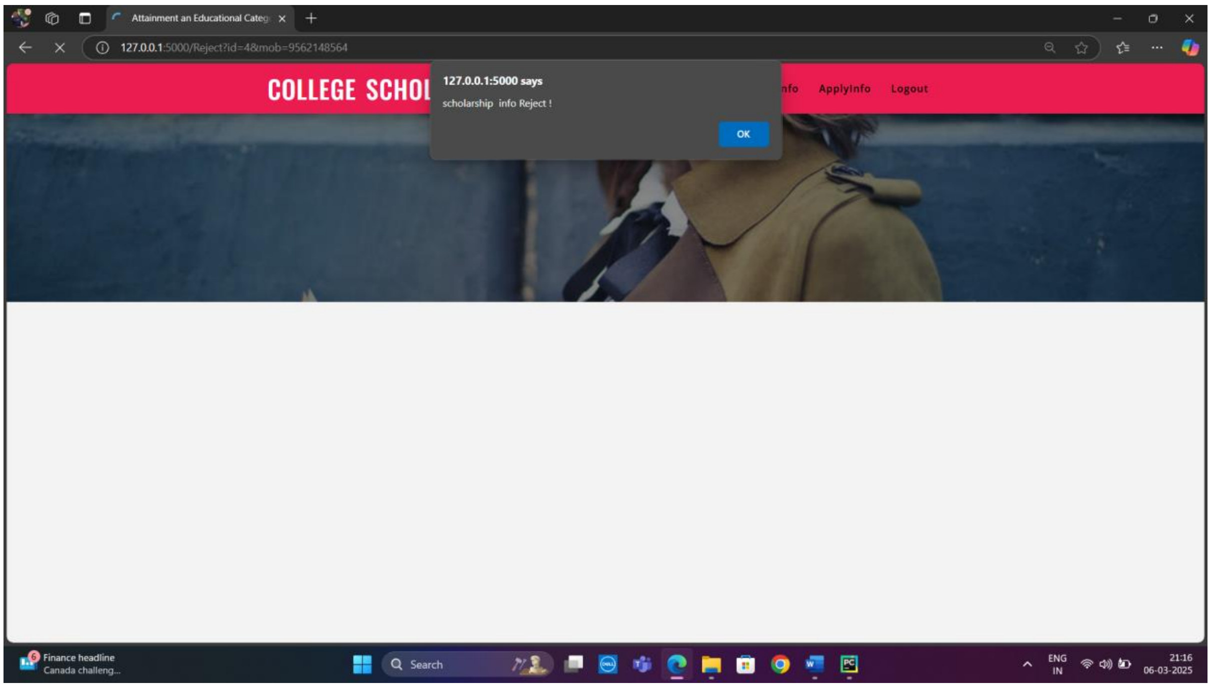
Regno	Mobile	schloarshipName	Certificate1	Certificate2	Status	Accept	Reject
22ucs730	9562148564	students welfare scholarship	Screenshot (8).png	Screenshot (6).png	waiting	Accept	Reject
22ucs733	8956231457	students welfare scholarship	Aadhar card.png	Income certificate.png	waiting	Accept	Reject

APPLY

Information

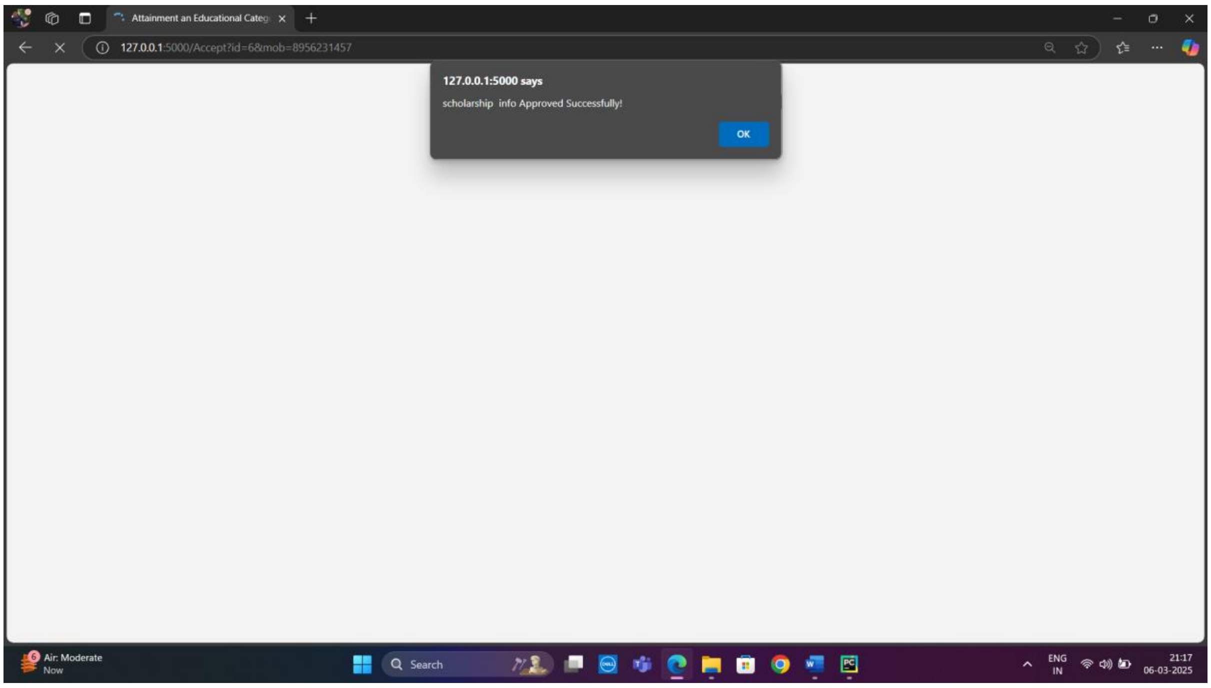
Regno	Mobile	schloarshipName	Certificate1	Certificate2	Percentage	Status
22ucs732	1234567890	students welfare scholarship	Screenshot (8).png	Screenshot (8).png	7.5	Reject
22ucs730	9562148564	students welfare scholarship	Screenshot (8).png	Screenshot (6).png	7.55	waiting
22ucs733	8956231457	students welfare scholarship	Screenshot (8).png	Screenshot (7).png	8.96	Approved
22ucs733	8956231457	students welfare scholarship	Aadhar card.png	Income certificate.png	8.96	waiting

© Colllge . All Rights Reserved | Design by Student



127.0.0.1:5000 says

scholarship info Reject !


Attainment an Educational Categ
127.0.0.1:5000/Accept?id=6&mob=8956231457
OK
Air: Moderate
Now
Search
ENG
IN
21:17
06-03-2025

127.0.0.1:5000 says

scholarship info Approved Successfully!