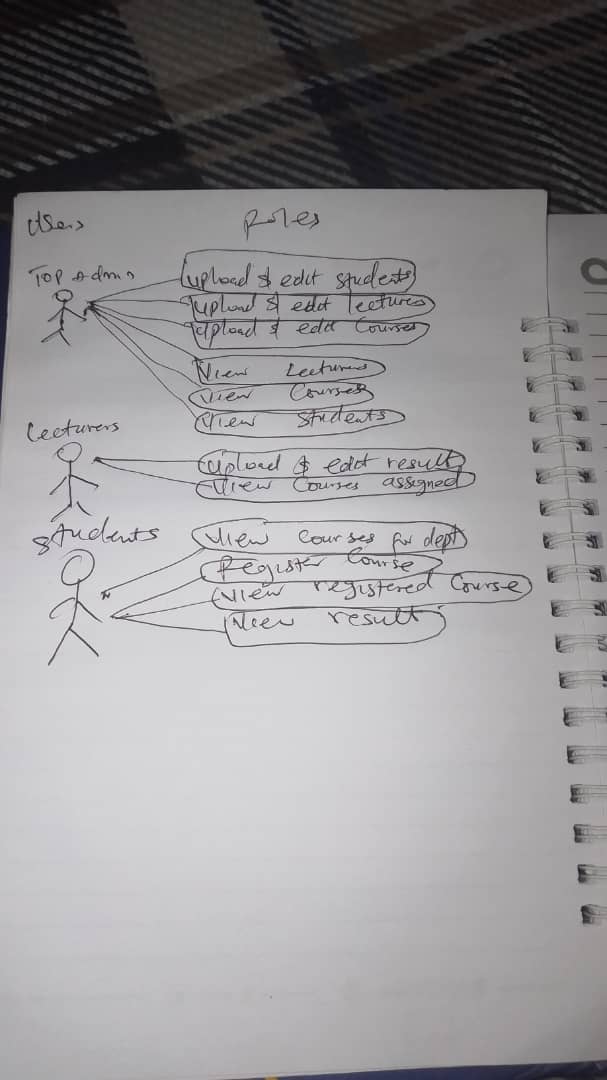
4.1 **SYSTEM SPECIFICATION AND DESIGN**

In the course registration and result processing software, I identified three level users and are modeled as follows:



Above is a use case diagram that represents a summary of activities of the course registration and result processing system. The three level users of the system are:

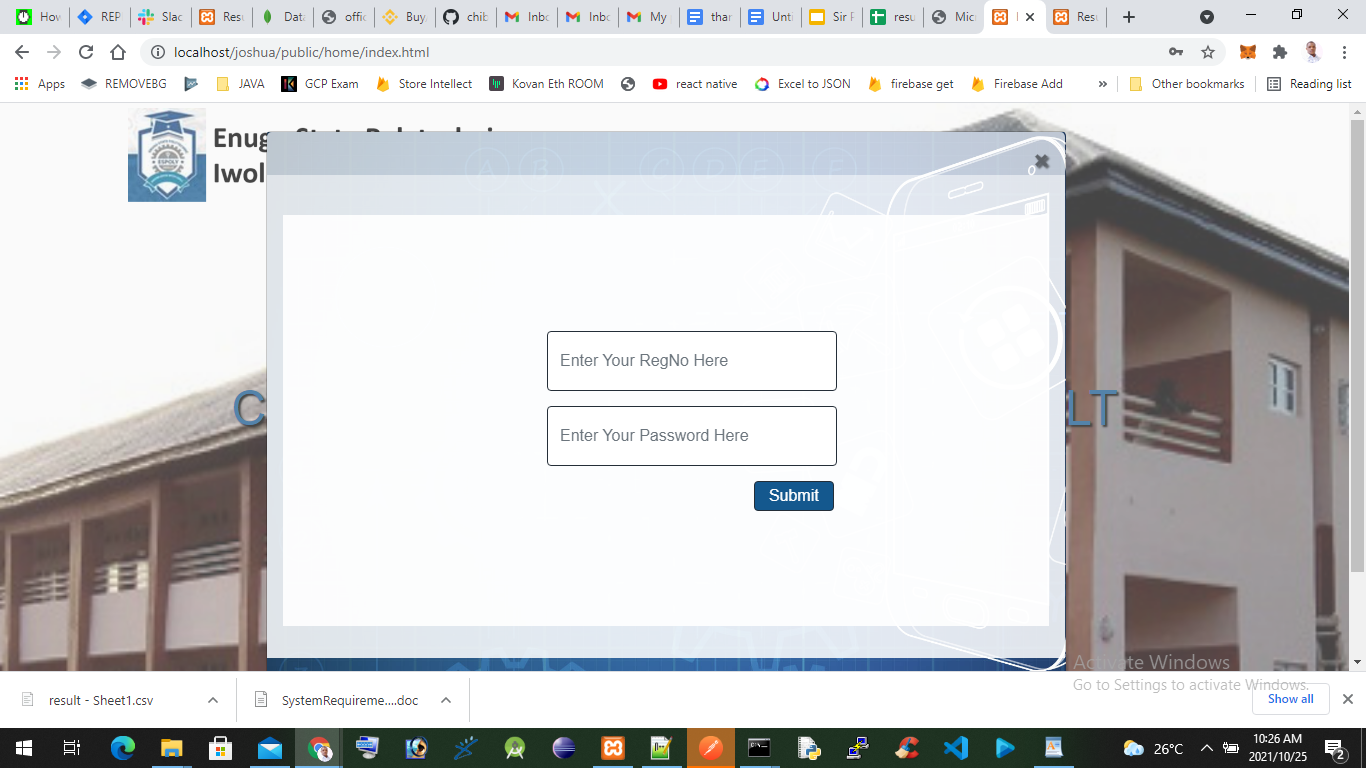
1. Top Admin: This is a staff authorized to have access to the software, possibly the school ICT unit. At the beginning of the session, he first of all updates the session and uploads various files in an excel format. He uploads courses that will be taken that session, students who have registered and lecturers for different courses.

2. The student: The student can register for a course, he can revisit the site, view registered courses for a session and semester, and he can also view his result.

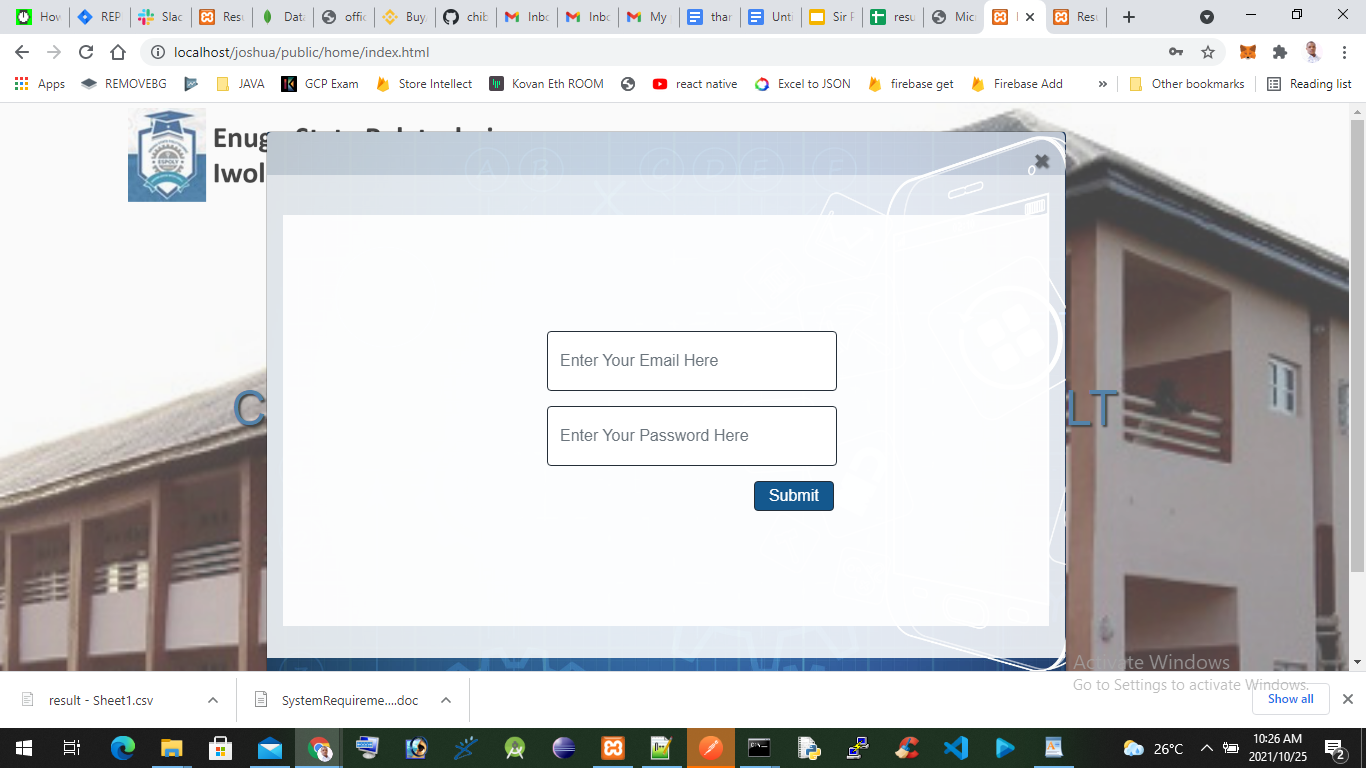
3. The Lecturer: This is a staff authorized to have access to the software, and also incharge of a particular course, he first of all gets a list of the students who registered his course from the Top Admin and the result of the student for the semester and session via an excel file.

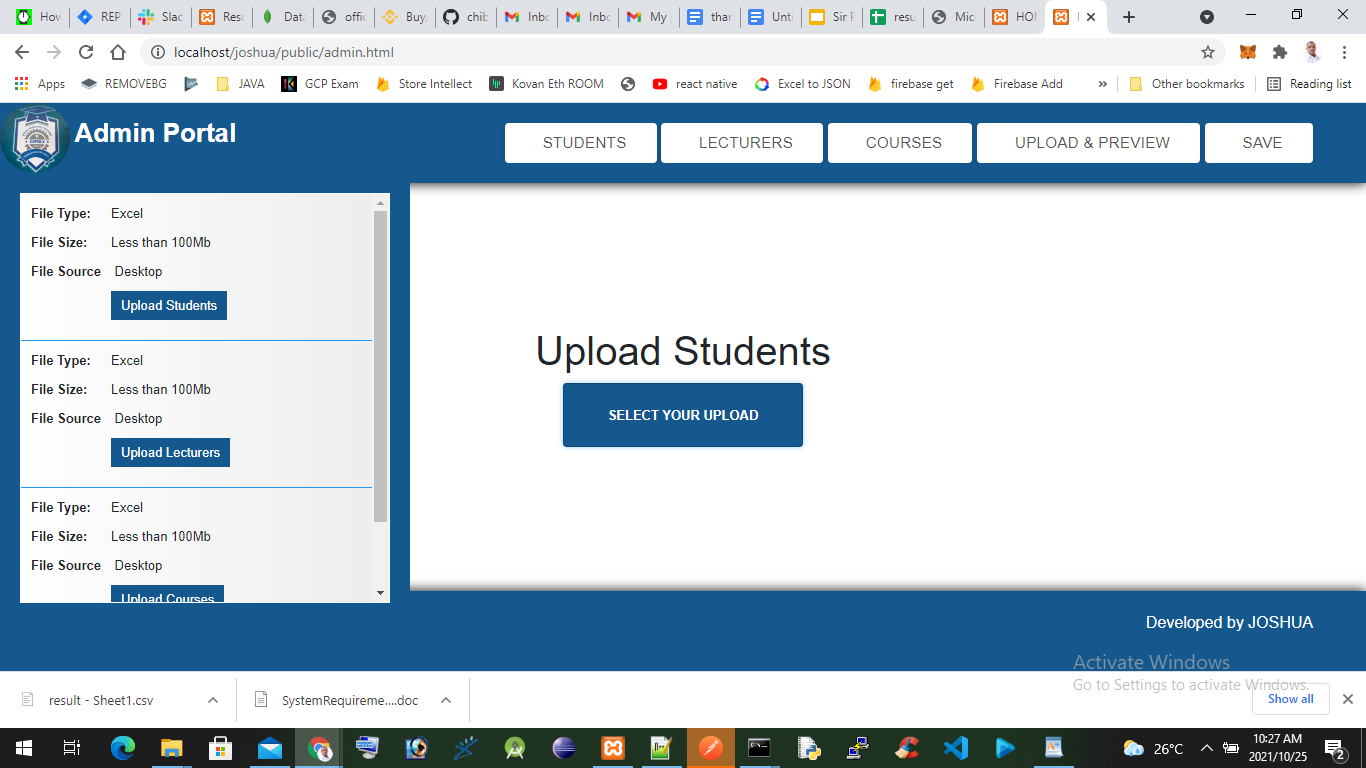
4.1.1 **The Input Design**

The input design shows the template for the user’s input. Data is entered into the system through the input form. The students enter data through the course registration forms while the admin officer and lecturers have separate forms for uploading excel files.

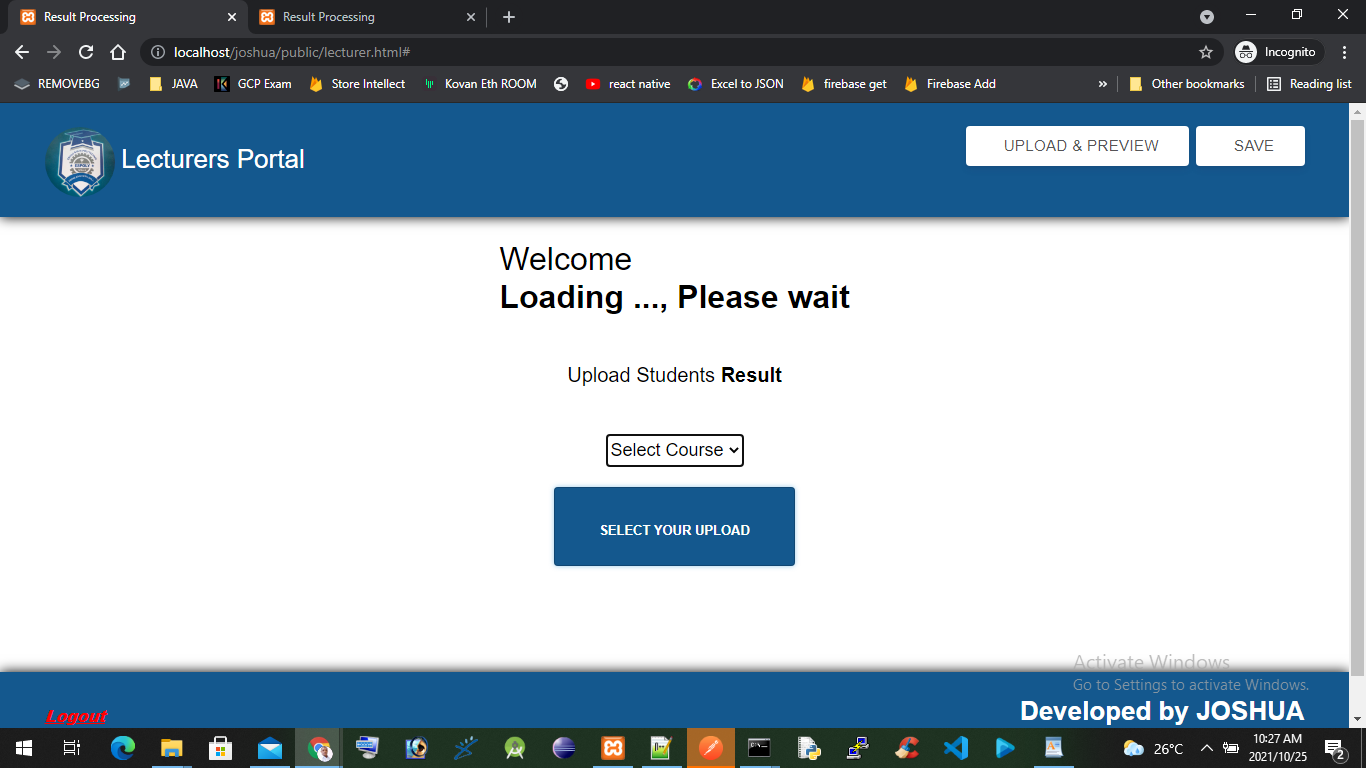
Student Login Form

Lecturers Login Form



Admin File Upload Dashboard

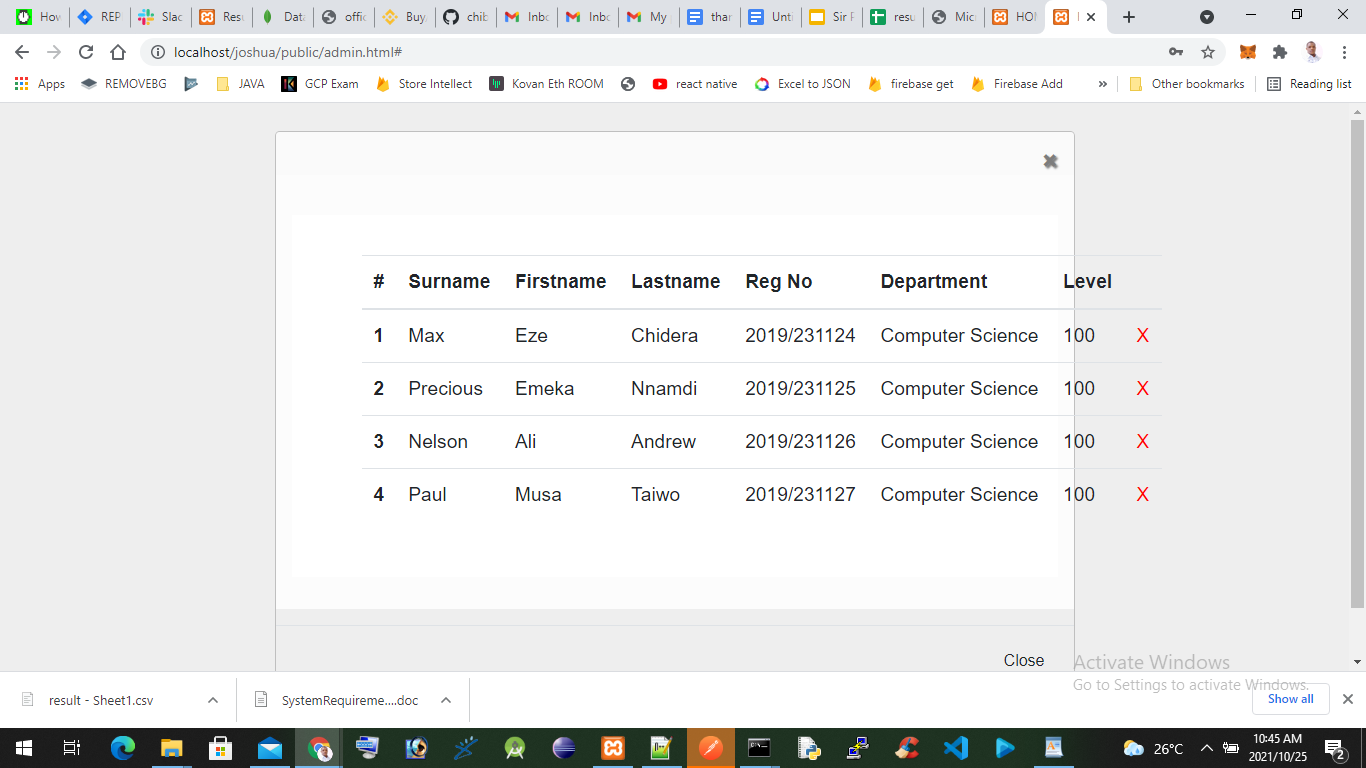
Lecturers Result Upload Dashboard

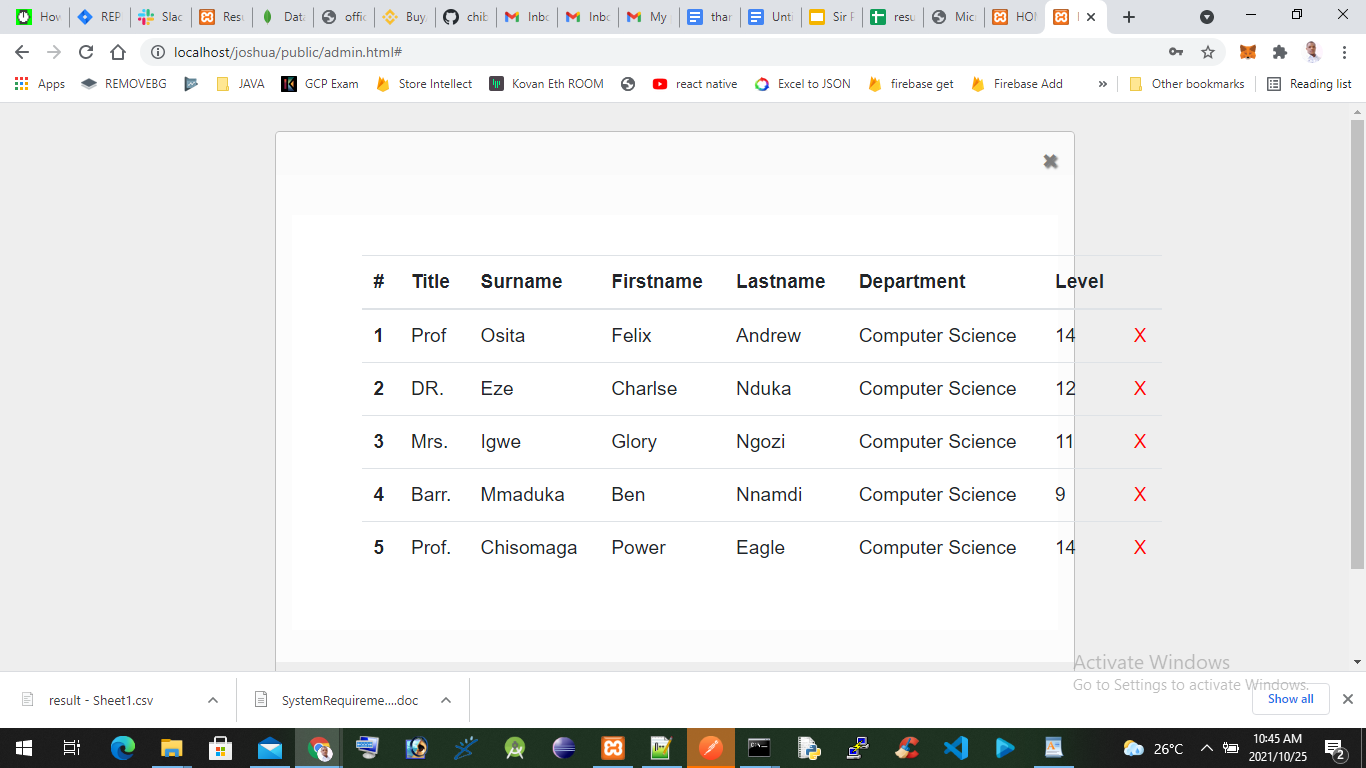


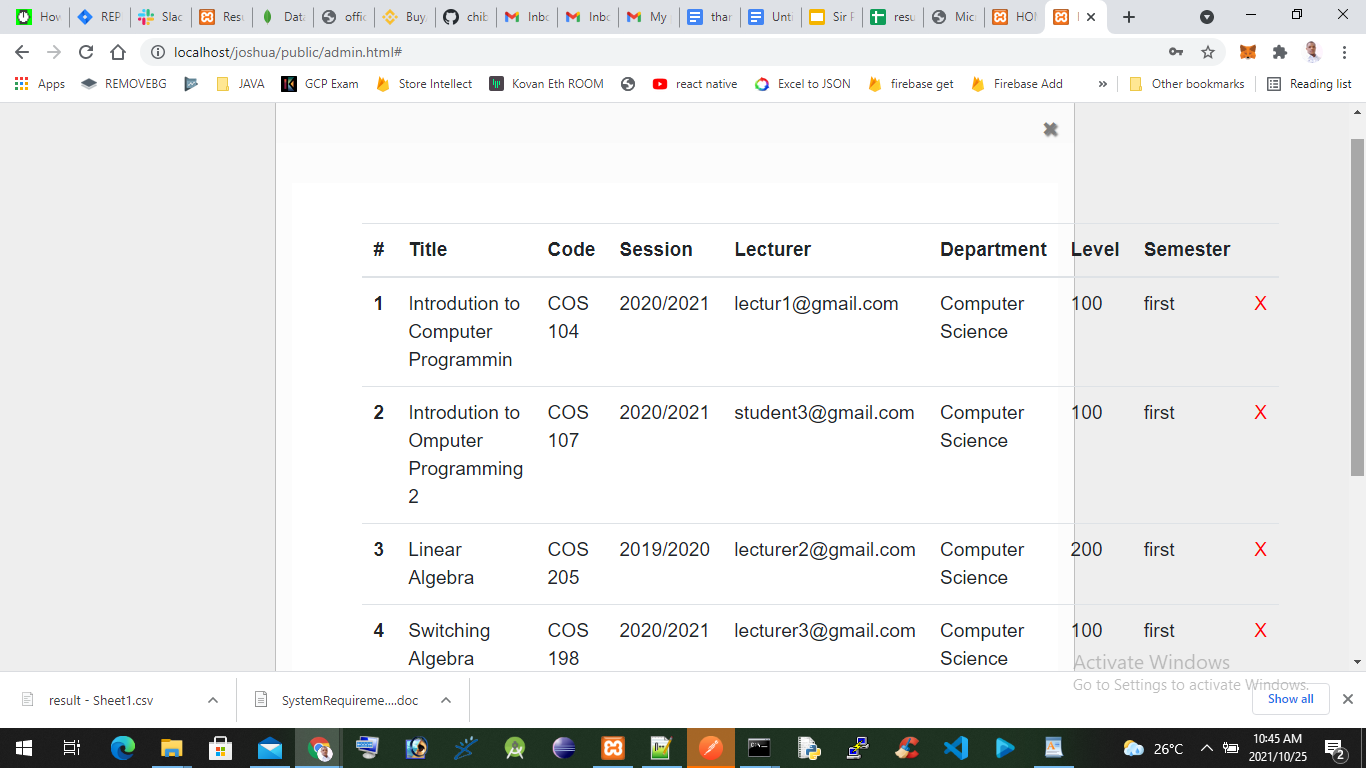
4.1.2 **The Output Design**

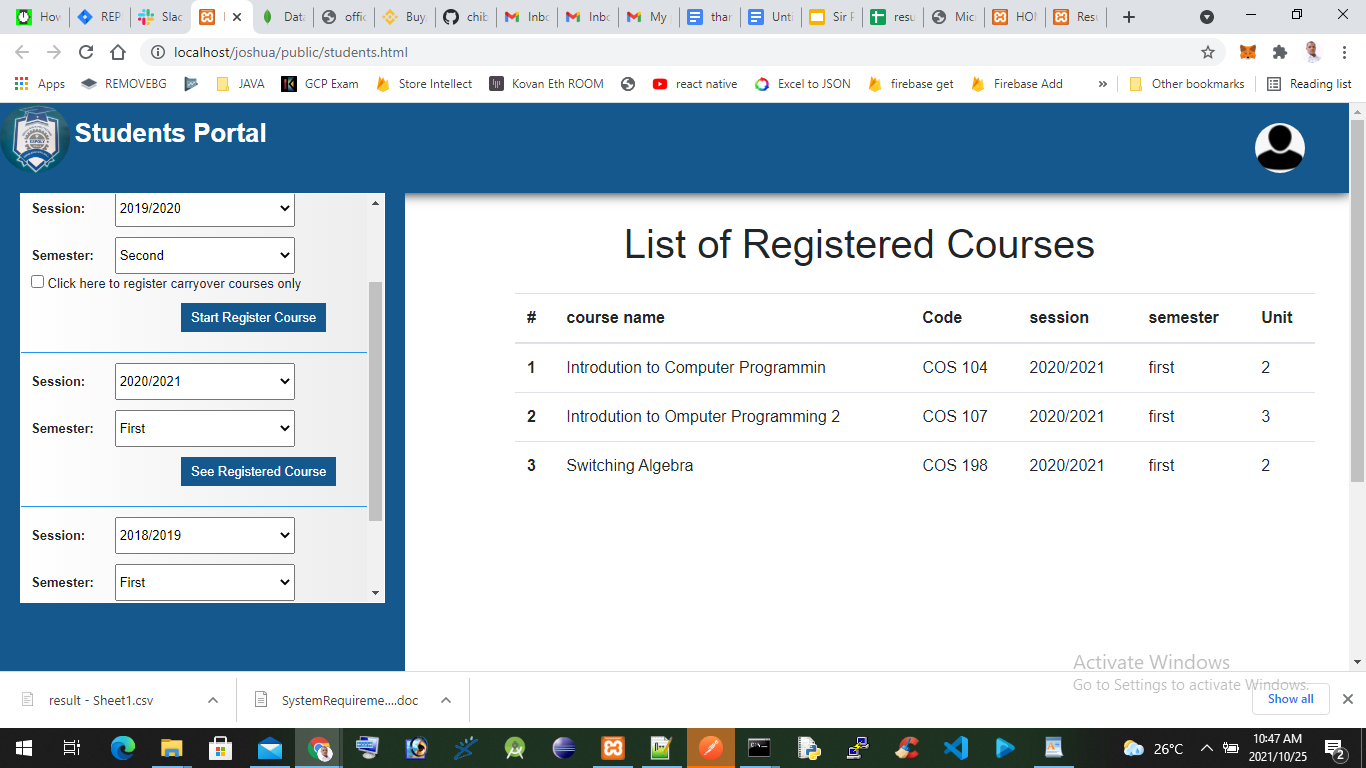
Output Design The output design is the design of the results the user expects to get when he logs in and uses the software. The major thing a student does with the software is to register courses and view results online. After registering for courses online, he/she can view the registered courses, view and print his or her result online. The admin also can view courses, lecturers and registered students.

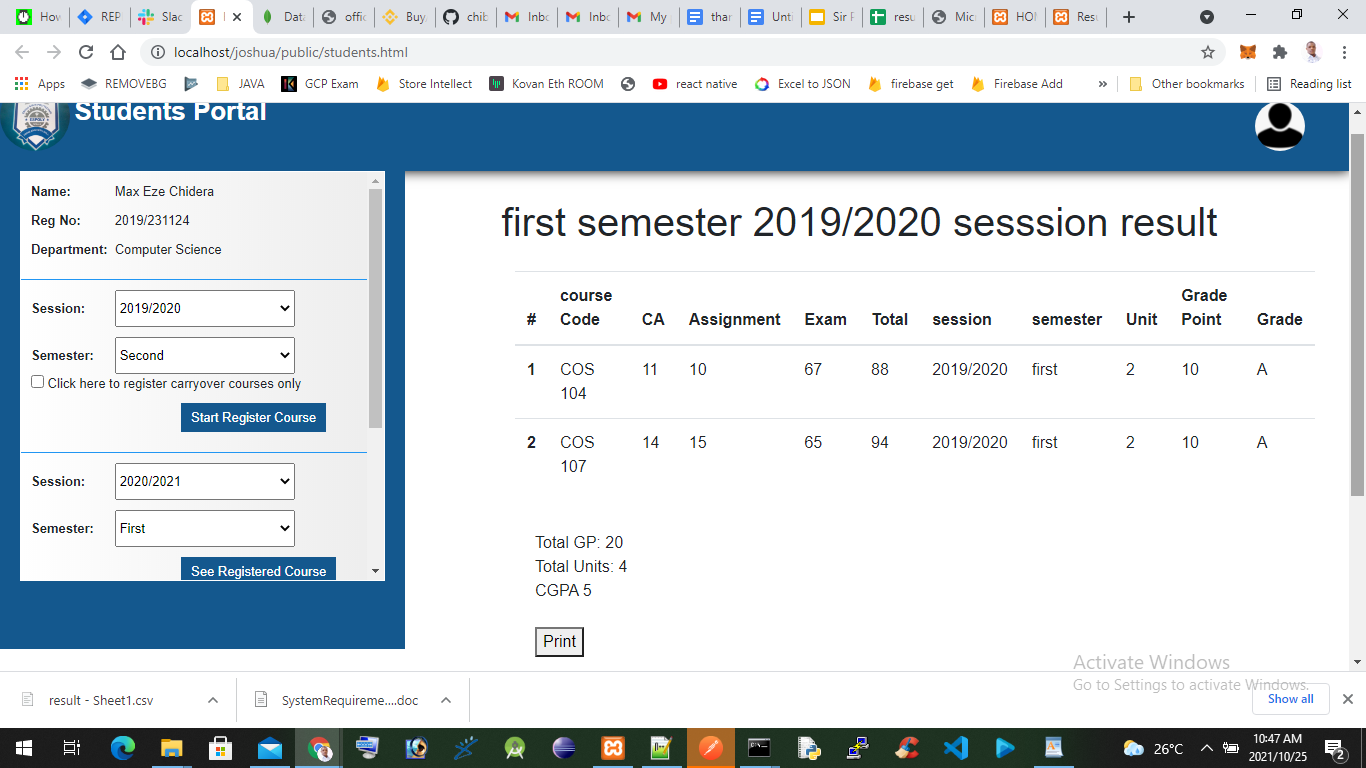
Top Admin Students List



Top Admin Lecturers List

Top Admin Courses List

Students Registered Course Page

Students Print Result Page

4.1.3 Security Specification

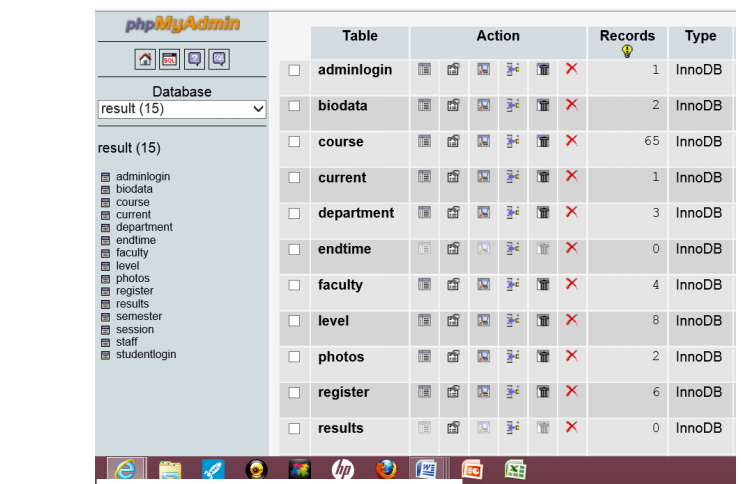
To deploy secure software effectively, you need clear, consistent, testable, and measurable software security requirements.

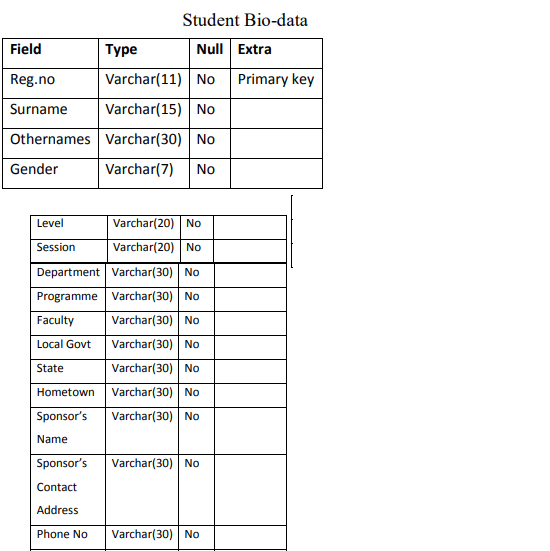
The following requirements were taken into consideration to secure the project from attackers and vulnerabilities.

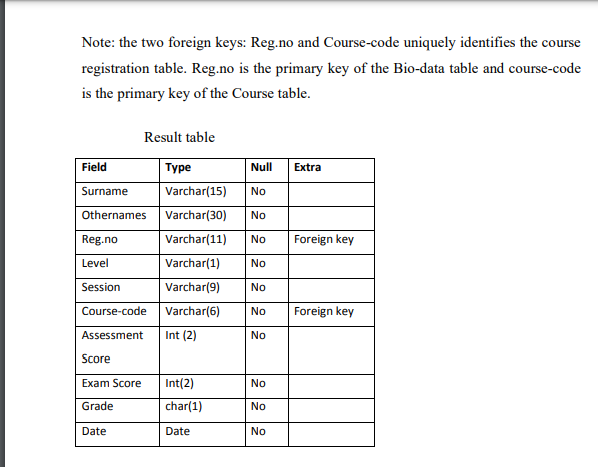
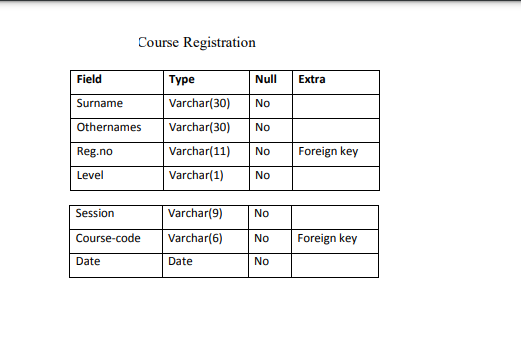
1. Applying Authorization and Authentication checks on every public web route
2. Enforce HTTPS transport security enforcement in production
3. Enforce CSRF attack prevention
4. Enforce Session Fixation protection
5. Enforce Security Header Integration  
   i. HTTP Strict Transport Security for secure requests  
   ii. X-Content-Type-Options integration  
   iii. Cache Control on static resources  
   iv. X-XSS-Protection integration
6. Password hashing enforcement
7. Use of OAuth2/JWT specification for authorization

4.1.4 Database Design Specification

Database is a collection of entities with related information. In the design of the course registration and result process system, the various related entities are: admin login, student login, bio-data, course, level, result, semester and session file. The Student Bio-data table shows the student’s profile, the course table contains all courses that are offered in the department. The registration table pulls data from the Bio-data table and the course table. The result table is a combination of the student’s biodata and the course registered, score, grade and grade-point. The Admin table is the necessary profile information that the Admin would have in order to access the software.







4.2 **Software Development Tools**

The following tools were used to develop the software

1. **Programming Languages** - HTML, CSS, Javascript for frontend and Nodejs

For the backend.

**HTML**

HTML is the standard markup language for creating Web pages.

* HTML stands for Hyper Text Markup Language
* HTML is the standard markup language for creating Web pages
* HTML describes the structure of a Web page
* HTML consists of a series of elements
* HTML elements tell the browser how to display the content
* HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc

**CSS**

CSS stands for cascading stylesheet is the language we use to style an HTML document.

CSS describes how HTML elements should be displayed.

This tutorial will teach you CSS from basic to advanced.

**Javascript**

JavaScript to program the behavior of web pages, As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

**Node.js** is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server-side and client-side scripts.

1. **Database**: A MongoDb database was used to develop the software

MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. Its secured and highly scalable hence the best for the software.

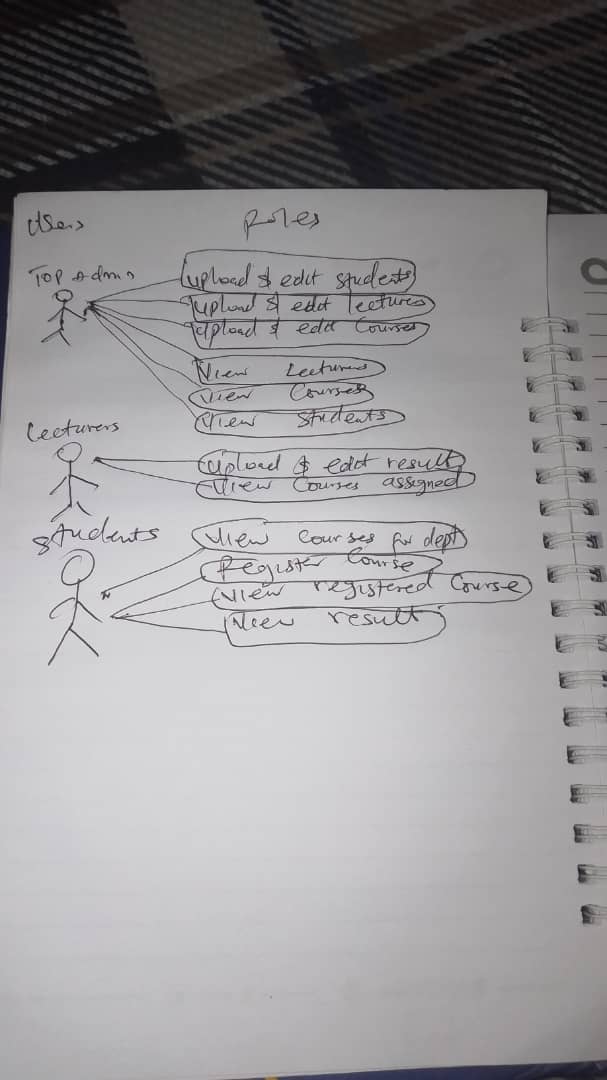
1. Web Server: Nodejs and Apache server was used to server the backend scripts and static html pages respectively.

4.4 **Implementation of the proposed system**

The new result processing system automates all operations and administrative activities related to course registration and examinations result processing in a department of a polytechnic. Thus the basic activities carried out in a manual system are performed by a computer. The system has three levels of users, the students, the lecturers and Top Admin. The primary source of a student's list comes from the admission list. The admission list is typed in an excel spreadsheet and submitted to the Top admin. He uploads the file into the Mongodb database for storage vai the Top Admin dashboard. When a student comes to register for courses, he logs into the portal with a registration number and a password.

This is to enable the student to log on to the portal. After logging in, the portal opens a page with the students' information. This is to show that this student has logged into the software.

When this is done, he will be given access to register courses, edit courses or print registered courses. Student registration is stored into the MongoDb database in a registration table. Lecturers prepare students’ results in an excel spreadsheet with their own systems. The result is saved in an excel format and he or she uploads the softcopy result into the MongoDb database via his or her dashboard. The system provides a component that is capable of tracking each student’s result from the result sheet and automatically calculates the GPA for each semester. With this, the student is able to view all his results together with the GPA for the semester. The software was designed by articulating the manual processes and automating the processes. The automation is carried out by presenting each manual data process as data objects. The data objects represent real-world entities e.g. course registration. It has a set of attributes or properties and operations that can be carried out on it. The object-oriented software captures the various data objects of the system and establishes relationships and interaction among the objects. In the result processing system, data objects are represented as a class object. The classes are: student class; course class; registration class; result class and administration class. Each class is an entity of its own with a known set of characteristics and operations that can be performed on it. The student class contains the student admission list. The administrator can add or delete a student’s record. He can also print the list of registered students. The “course-class” contains all the courses that are offered in the department. The administrator can equally add or delete courses according to the current national universities commission (NUC) and departmental specifications. The “registration-class” contains the name and registration number of students together with the courses registered for a semester. If a student registered for a course that he will not take, he will be required to meet the administrator to drop the course. Only the administrator is authorized to access the registration table. The result class contain the student’s result uploaded from excel file. The administrator’s class contains the administrator’s ID and his profile. He performs the following functions. Add or delete a student’s record, Add or delete courses, Upload result, Calculate GPA and CGPA.



**Functionality of the Use Case diagram.**

The functional use case explains the functionality of each of the activities symbolized in the use case diagram.

1. Upload Biodata:- this is a web-page that inserts the student’s profile data into the database system. The raw data is gotten from the department and transferred into a digital form by typing into an excel template. The excel data is then uploaded into the system.

2. Register/Edit Courses:- this page allows a student to register for his/her courses online.

3. Print Courses:- A view is created where the student can view all registered courses before printing the courses.

4. View Results:- A student is also given the rights to see results of all registered courses.

5. Upload Students:- this is a module that transfers a student's list from excel file to the portal system.

6. Add Courses:- the web-page inserts into the database system all courses and course description that is offered in the department.

7. Add & Edit Staff:- this is a page that adds the names and qualifications of all staff in the department into the database.

8. Print Class List:- A list of class members is generated from database. This list comes from the excel data and uploaded by the administrator in the upload student’s page.

9. Print Course List:- this generates a list of all courses offered in the department. The list is generated according to the specification by the admin officer.

10. Upload Results:- the admin is given the rights to upload student’s results that is given by the examiner.

11. Print Results:- the admin can print the student’s result from the database

12. Update session:- the admin officer has the privileges to allow student’s registration at the beginning of the session and terminates the registration at the end of the session.