CSE470: Software Engineering

BRAC UNIVERSITY

Inspiring Excellence

Project Report

uSavior

(An Academic Aid Website)

Submission Date: 28 August 2023

Group Information

Table of Contents

Introduction

Education has gone beyond conventional limits in our quickly changing society, offering students and teachers with a world of limitless opportunities. Introducing uSavior, is a personalized gateway to a universe of knowledge and growth. uSavior seeks to make education accessible to everyone, particularly those studying Computer Science and Engineering (CSE).

uSavior is an educational environment that has been specifically designed to meet the changing demands of both students and instructors. For students, Our platform provides an incredible range of CSE-related courses covering a broad range of topics.From the realms of coding and algorithms to software engineering and artificial intelligence, uSavior becomes your virtual arena for exploration, learning, and excelling. Our platform enables you to choose courses that align with your interests and goals, ensuring that your education is interesting, relevant, and deeply rewarding. Dive into lecture videos, reading materials, assignments, and quizzes that are designed to provide a comprehensive learning experience.

Instructors also play a significant role in shaping the learning experience. uSavior invites instructors to join our vibrant community and create their own accounts. If you're an expert in your field with a passion for teaching, uSavior provides you with a platform to connect with eager learners. Design your courses, share your insights, and inspire the next generation of learners. Our robust tools and resources make it effortless to create engaging and effective learning materials.

Our dedicated admin team ensures the seamless operation of uSavior. Managing students, courses, and instructors, they work tirelessly to create a platform that facilitates efficient communication.

Functional Requirements

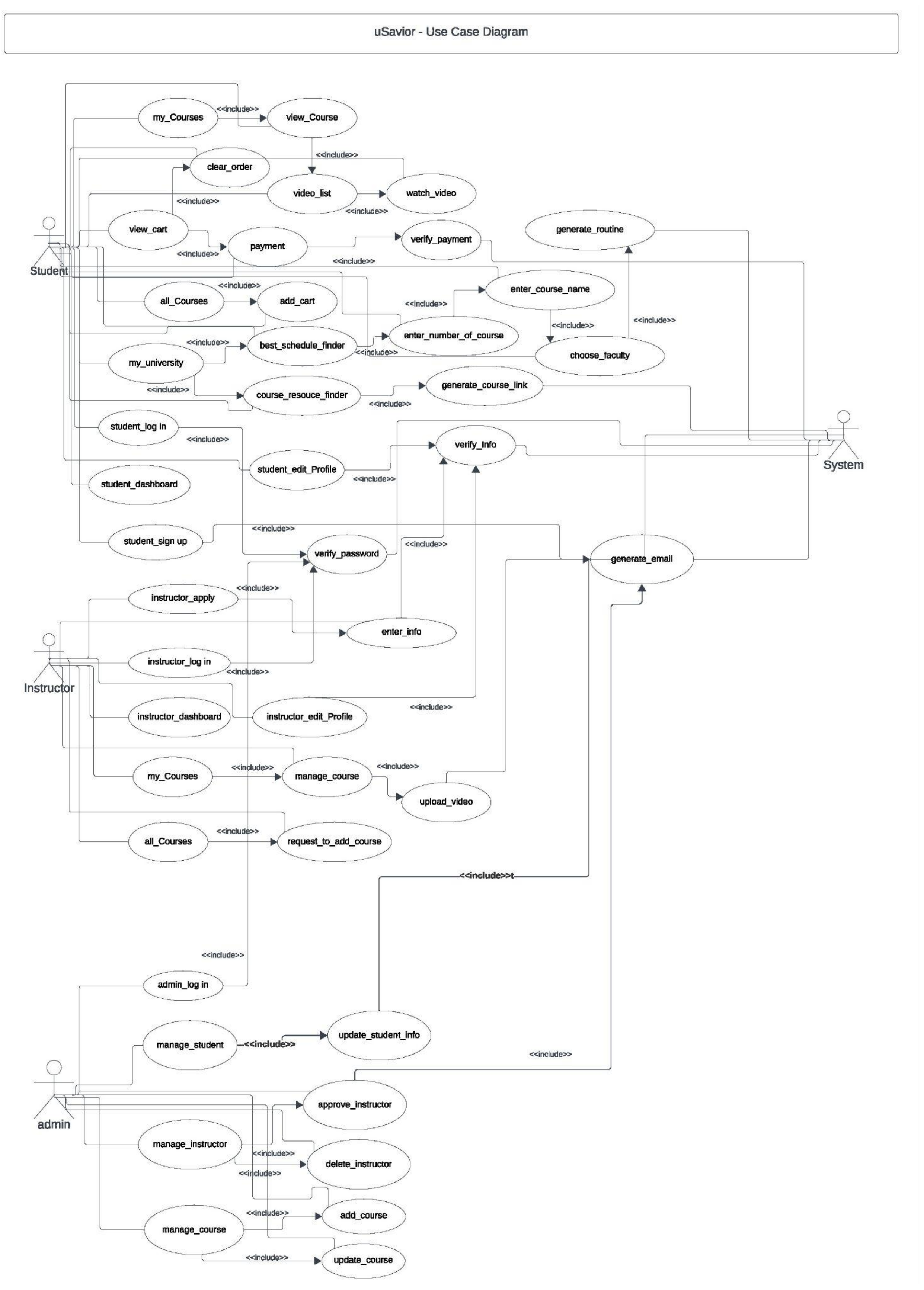
In uSavior, some of the potential functional requirements are:

1. User Accounts:
   - Students and instructors must be able to create their own accounts on the platform.
   - User authentication and secure login mechanisms should be implemented.
2. Course Catalog:
   - The platform should provide plenty of courses covering a wide range of topics.
   - Courses should be categorized and easily searchable for students' convenience.
3. Course Enrollment:
   - Students should be able to browse courses, view detailed descriptions, and enroll in the ones they're interested in.
4. Course Materials:
   - Courses should include lecture videos, reading materials, assignments, quizzes, and other relevant resources.
   - Instructors should be able to upload and manage these course materials.

5. Communication and Notifications:
    - Students should receive notifications about course updates, deadlines, and announcements.
6. Admin Panel:
    - An admin panel is necessary to manage user accounts, courses, and instructor profiles.
    - Admins should have the ability to monitor platform activity, resolve issues, and ensure smooth operations.
7. Payments and Purchases:
    - The platform should support secure payment gateways for students to purchase courses.

Use Case Diagram

A use case diagram is a visual representation of how users might interact with a system. It falls under the UML (Unified Modeling Language) umbrella and is used predominantly in software engineering and systems design to specify, visualize, and document the behavior of software systems. A use case diagram helps teams to clarify the system's main functionalities and the actors that interact with those functionalities.

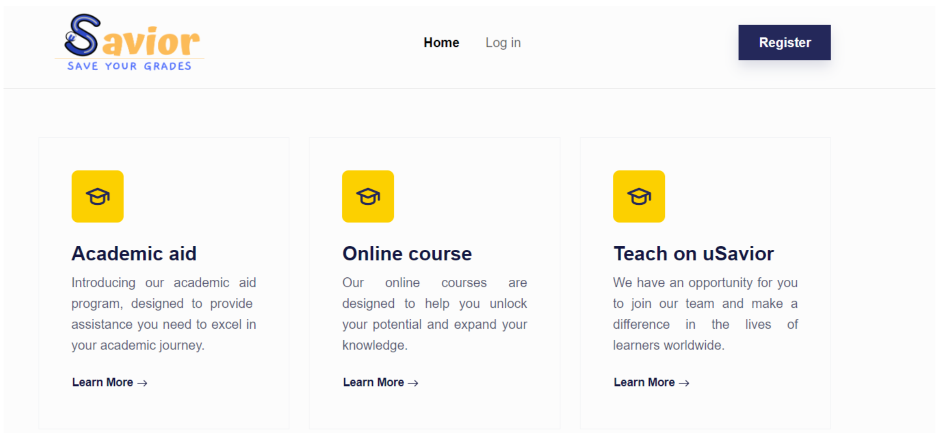

uSavior - Use Case Diagram
Student
Instructor
admin
System
my_Courses
view_Course
clear_order
video_list
watch_video
view_cart
payment
verify_payment
generate_routine
all_Courses
add_cart
enter_course_name
enter_number_of_course
best_schedule_finder
choose_faculty
my_university
course_resouce_finder
generate_course_link
student_log in
verify_Info
student_edit_Profile
student_dashboard
student_sign up
verify_password
generate_email
instructor_apply
enter_info
instructor_log in
instructor_dashboard
instructor_edit_Profile
my_Courses
manage_course
upload_video
all_Courses
request_to_add_course
admin_log in
manage_student
update_student_info
approve_instructor
manage_instructor
delete_instructor
add_course
manage_course
update_course
<<include>>

User Manual

Our web application consists of two main parts. There is the admin interface as well as the user interface. As a result, there are two sections in the user manual.

Homepage

When a user initially accesses this website, both members and non-members receive this homepage.



Academic aid
Introducing our academic aid program, designed to provide assistance you need to excel in your academic journey.

Online course
Our online courses are designed to help you unlock your potential and expand your knowledge.

Teach on uSavior
We have an opportunity for you to join our team and make a difference in the lives of learners worldwide.

Figure: Homepage

This page has 3 buttons:

- Home: These buttons take users to the website's home page or main page.
- Register: Users need to register in order to use uSavior.

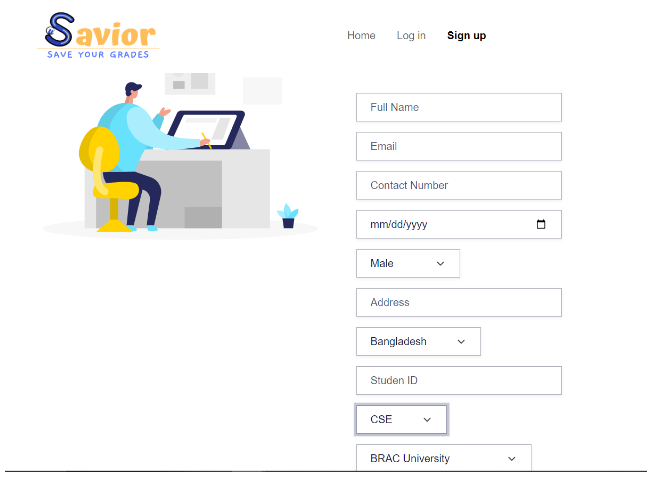
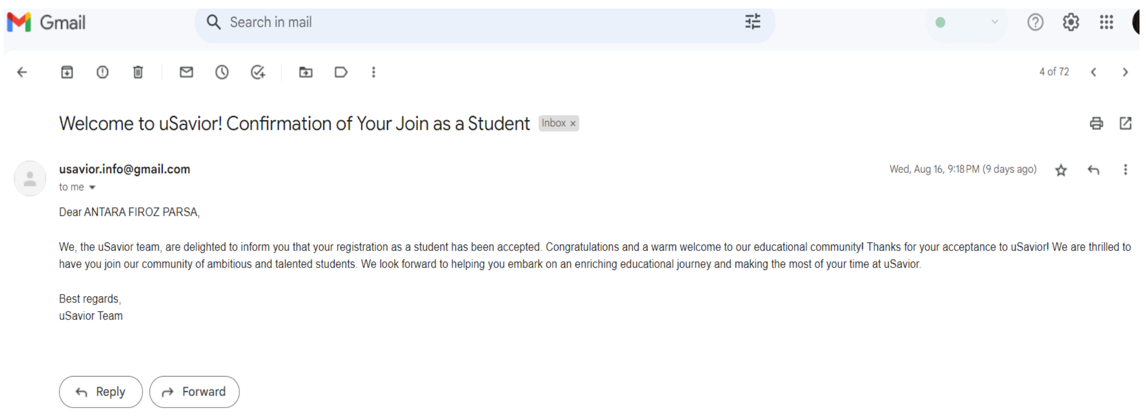


Figure: Register

In the figure, the registration module is depicted. Name,address, date of birth, studentId, department, University name, password etc are required.

To validate the email address, it sends a confirmation message, as seen in figure.



Welcome to uSavior! Confirmation of Your Join as a Student

usavior.info@gmail.com
to me
Wed, Aug 16, 9:18 PM (9 days ago)

Dear ANTARA FIROZ PARSA,

We, the uSavior team, are delighted to inform you that your registration as a student has been accepted. Congratulations and a warm welcome to our educational community! Thanks for your acceptance to uSavior! We are thrilled to have you join our community of ambitious and talented students. We look forward to helping you embark on an enriching educational journey and making the most of your time at uSavior.

Best regards,
uSavior Team

Figure: Confirmation message

- Login: After registration, to enter a personal account, users need to login. The member's password and email address are required to confirm this. Members can login as a student or an instructor. The authentication module is shown in figure.

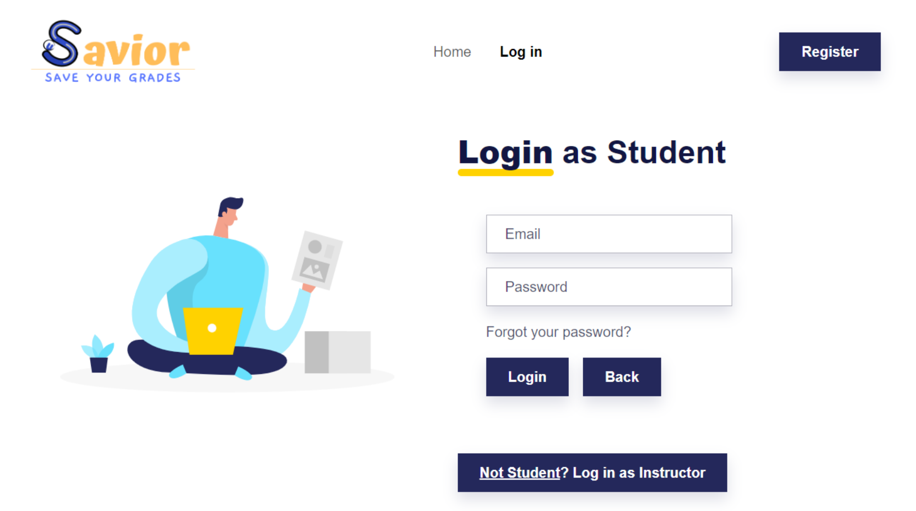
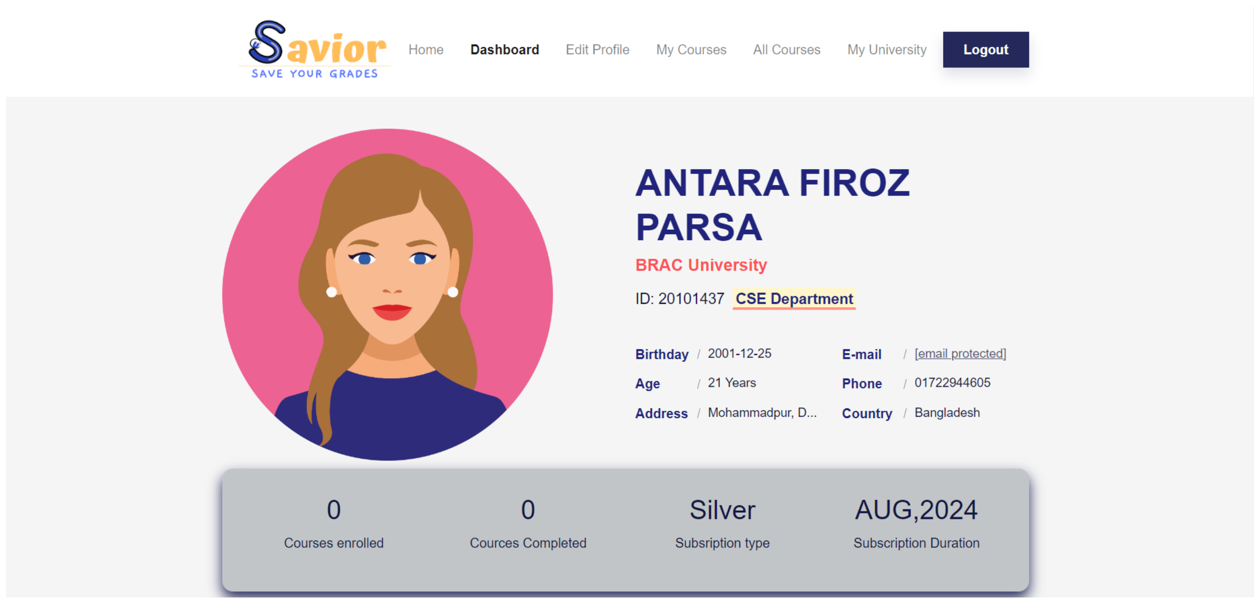


Figure: Log in

Dashboard

After login as a student or an instructor, one can see his/her own profile.



ANTARA FIROZ PARSA

BRAC University

ID: 20101437 CSE Department

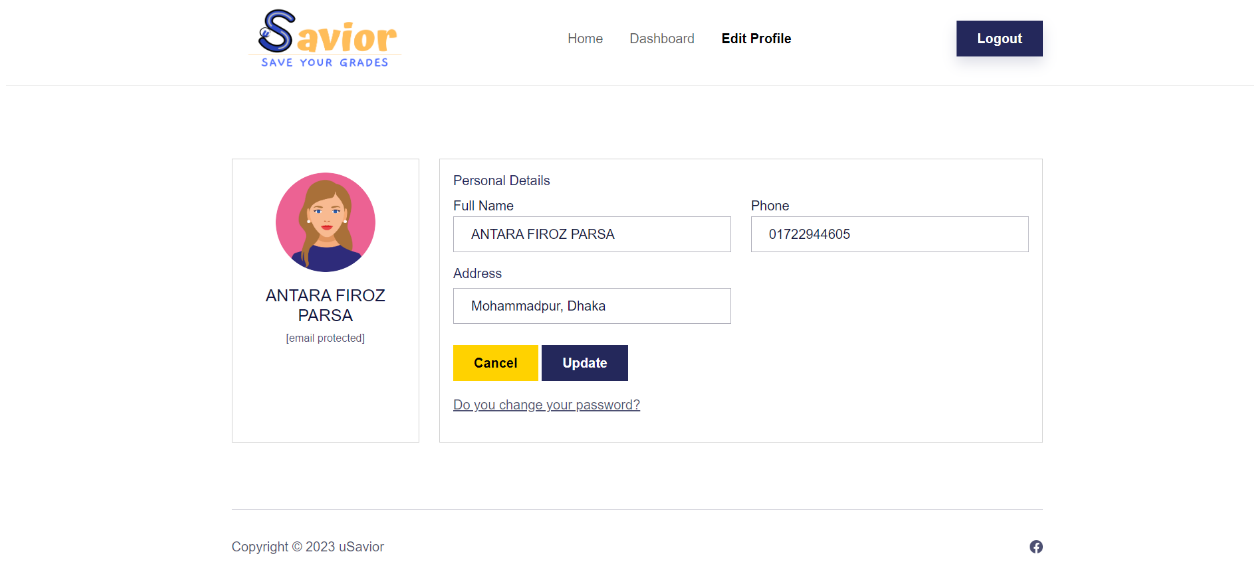
Birthday	2001-12-25	E-mail	[email protected]
Age	21 Years	Phone	01722944605
Address	Mohammadpur, D...	Country	Bangladesh

0	0	Silver	AUG,2024
Courses enrolled	Cources Completed	Subsription type	Subscription Duration

Figure: Dashboard

This page has some buttons:

- Edit profile: If a user wants to update any information about his/her. After editing he/she can again go back to the dashboard or homepage.



ANTARA FIROZ PARSA
[email protected]

Personal Details

Full Name: ANTARA FIROZ PARSA

Phone: 01722944605

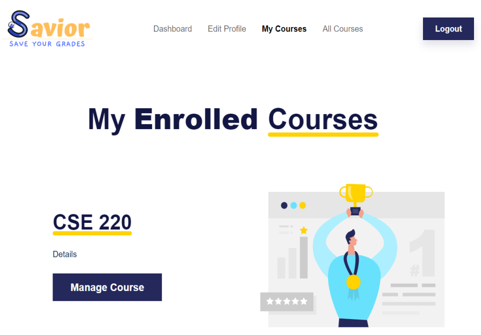
Address: Mohammadpur, Dhaka

Copyright © 2023 uSavior

Figure: Edit profile

My Courses

As Instructor: Here Instructors can see all the courses that they permit to teach. Instructor can upload lecture videos and course materials into their courses

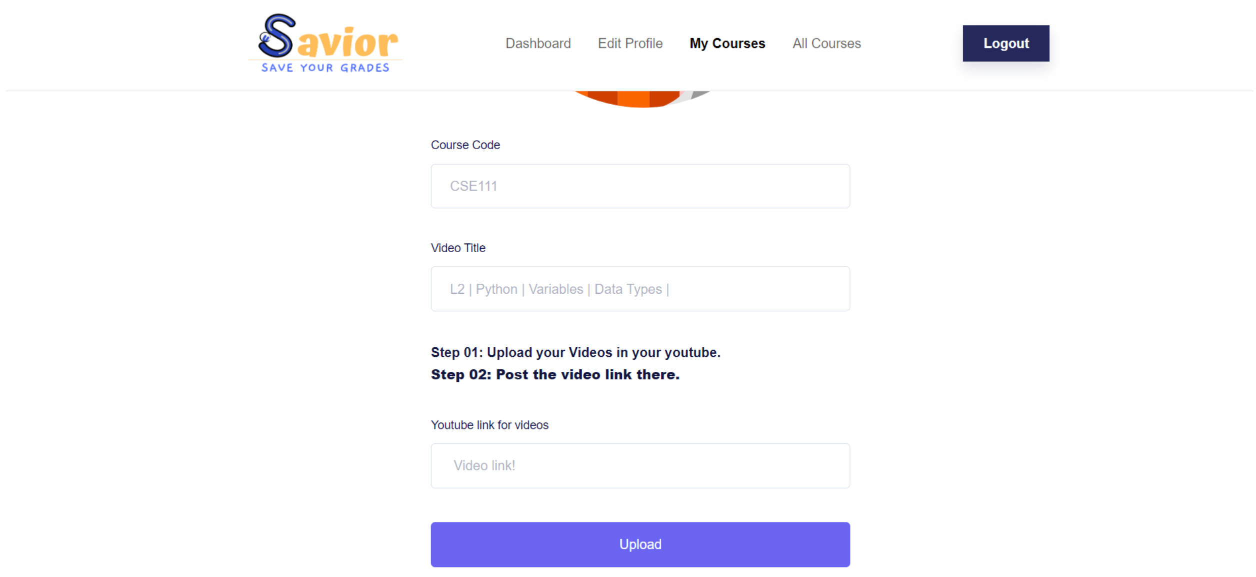


My Enrolled Courses

CSE 220

Figure: My Course

- Manage Course: Here, instructors can upload the lecture videos and course materials.



Course Code

CSE111

Video Title

L2 | Python | Variables | Data Types |

Step 01: Upload your Videos in your youtube.
Step 02: Post the video link there.

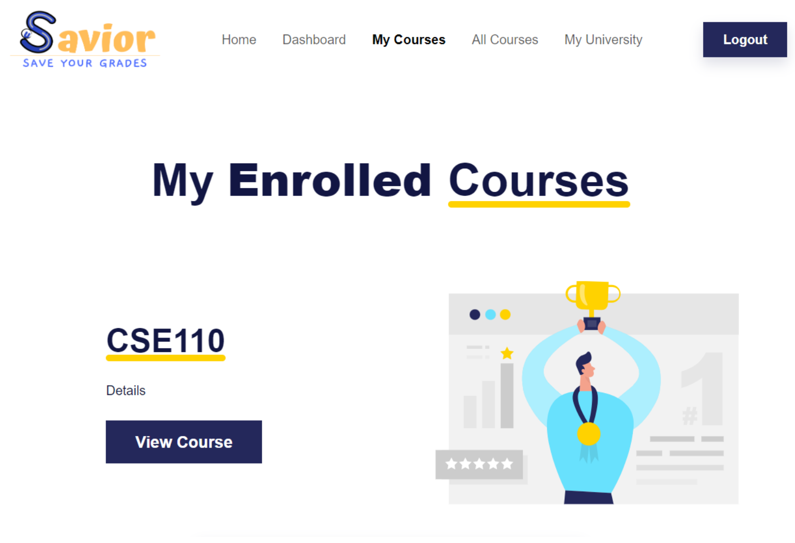
Youtube link for videos

Video link!

Upload

Figure: Manage Course

As Student: After the completion of payment, In my courses students can see all the courses that they purchased.



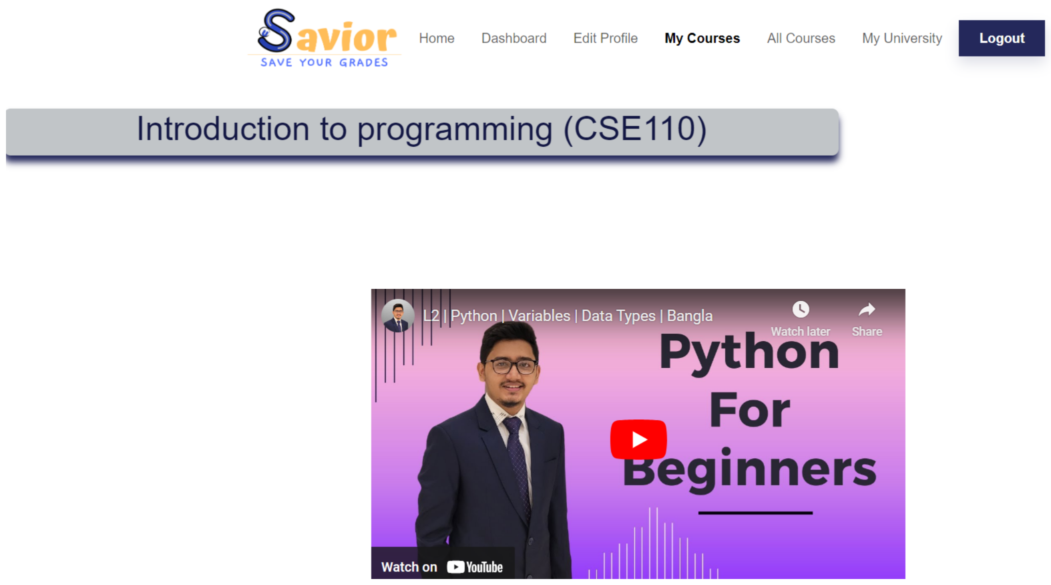
My Enrolled Courses

CSE110

Details

Figure: My Course

View Course: Here, students can see the lecture videos and course materials.

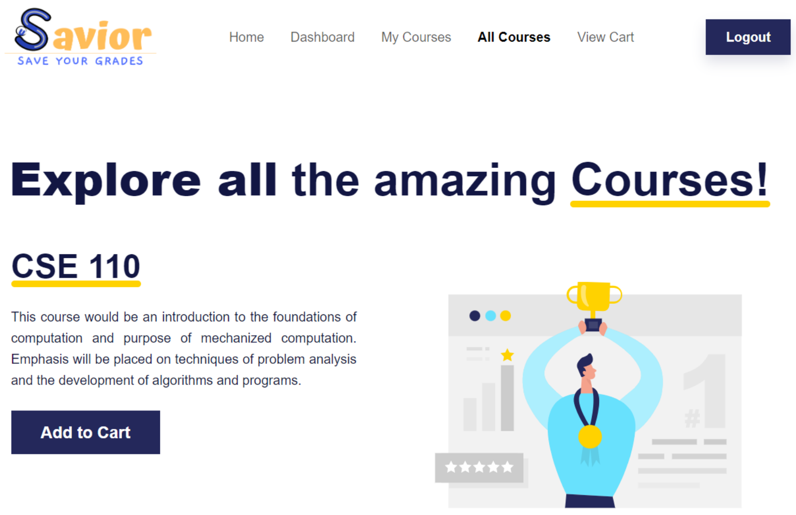


Introduction to programming (CSE110)

Figure: Play Video

All Courses

As Student: In all courses, students can explore all the courses that uSavior provides and Add to cart the courses.



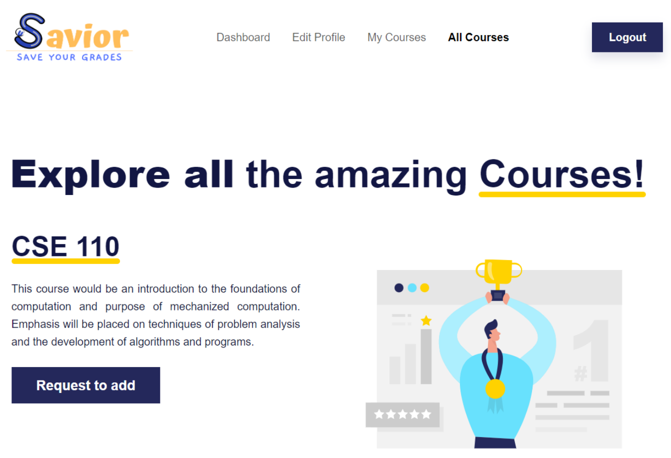
Explore all the amazing Courses!

CSE 110

This course would be an introduction to the foundations of computation and purpose of mechanized computation. Emphasis will be placed on techniques of problem analysis and the development of algorithms and programs.

Figure: All Course

As Instructor: Similarly, Instructors can also see all courses from their account and request the courses to admin that they prefer to teach.



Explore all the amazing Courses!

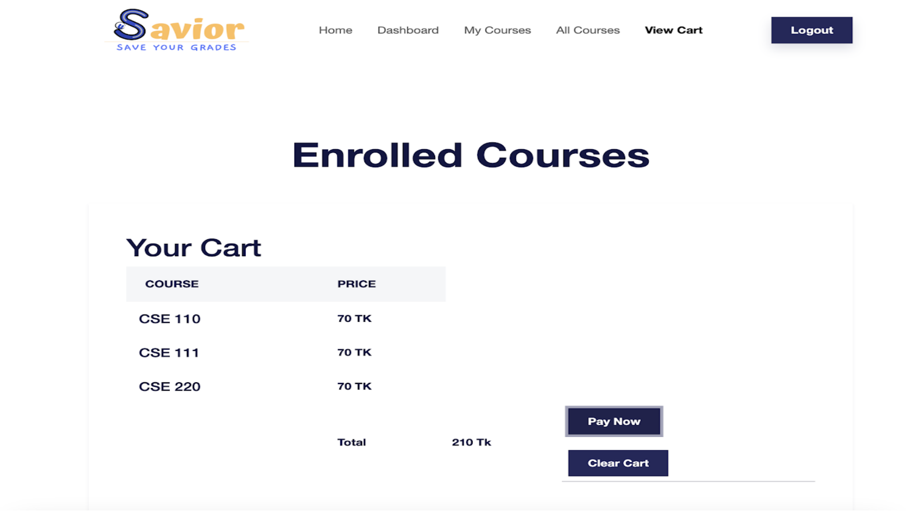
CSE 110

This course would be an introduction to the foundations of computation and purpose of mechanized computation. Emphasis will be placed on techniques of problem analysis and the development of algorithms and programs.

Figure: All Course

View Cart

In the view cart, students can see all the courses that they added to the cart.



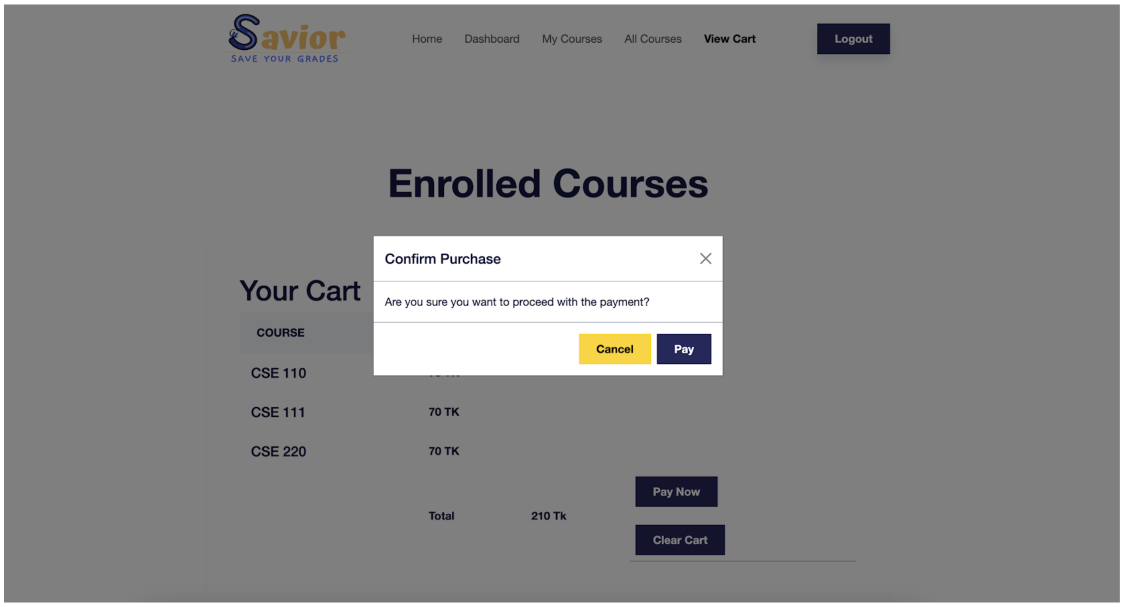
Enrolled Courses

Your Cart

COURSE	PRICE
CSE 110	70 TK
CSE 111	70 TK
CSE 220	70 TK

Total 210 Tk

Figure: View Cart



Enrolled Courses

Confirm Purchase

Are you sure you want to proceed with the payment?

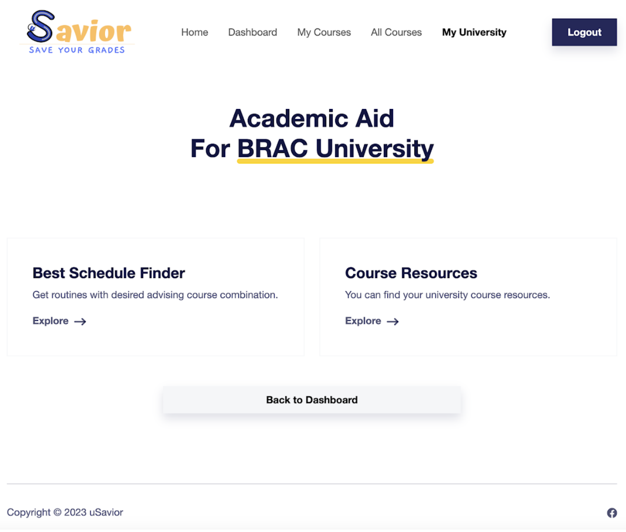
Your Cart

COURSE	
CSE 110	
CSE 111	70 TK
CSE 220	70 TK
Total	210 Tk

Figure: View Cart Confirmation

My University

There are 2 features here. Best schedule finder and Courses. Best schedule finder is to check all possible timing of students' courses. Courses will provide a resource link to students for any courses.



Academic Aid
For BRAC University

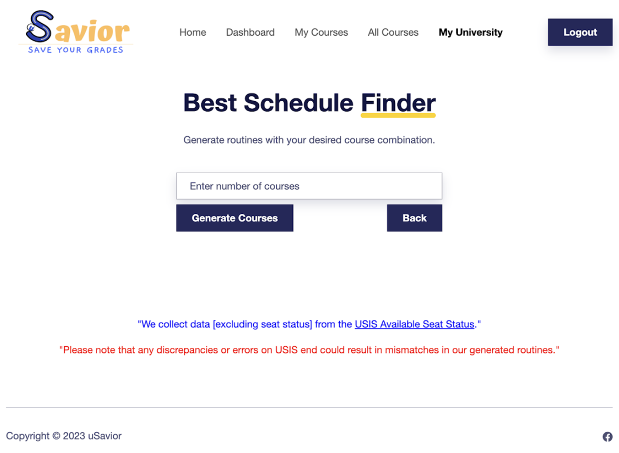
Best Schedule Finder
Get routines with desired advising course combination.

Course Resources
You can find your university course resources.

Copyright © 2023 uSavior

Figure: My University

1. **Best Schedule Finder [Student]:** Here, students can select courses and faculties according to their preferences and it will generate all possible schedules for those courses.



Best Schedule Finder

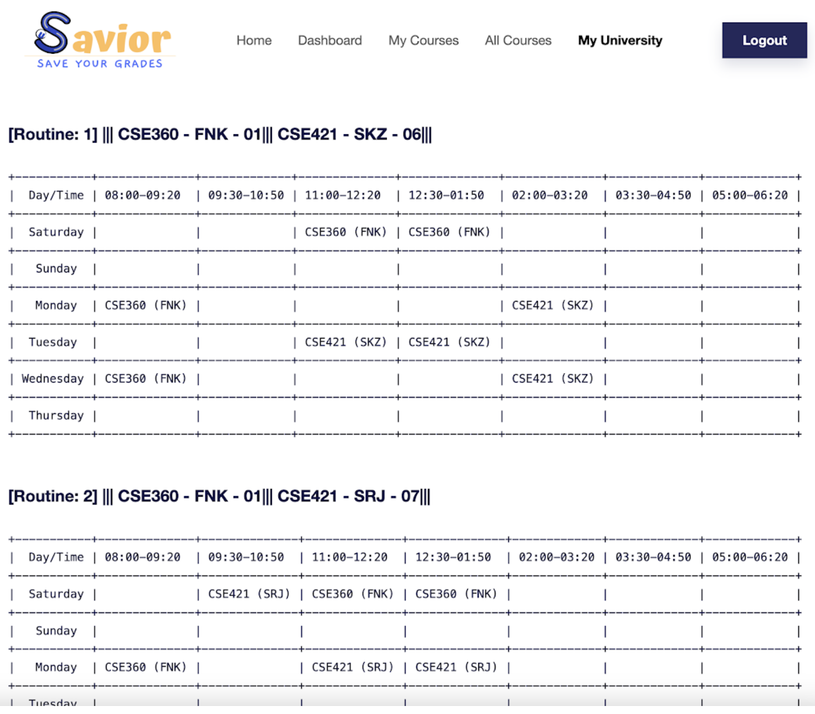
Generate routines with your desired course combination.

"We collect data [excluding seat status] from the USIS Available Seat Status."

"Please note that any discrepancies or errors on USIS end could result in mismatches in our generated routines."

Copyright © 2023 uSavior

Figure: My University 1.1



[Routine: 1] ||| CSE360 - FNK - 01||| CSE421 - SKZ - 06|||

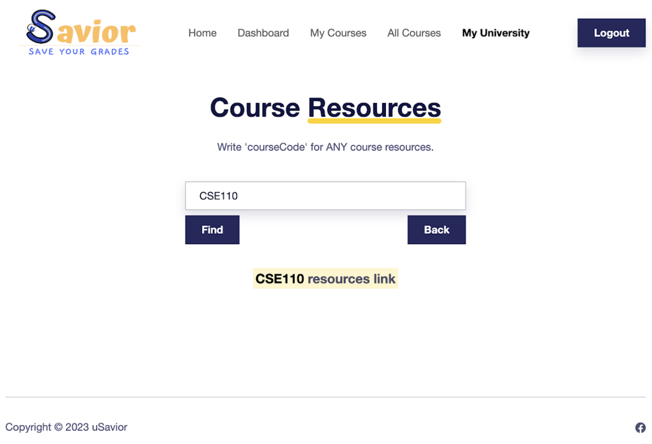
```
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|  Day/Time | 08:00-09:20 | 09:30-10:50 | 11:00-12:20 | 12:30-01:50 | 02:00-03:20 | 03:30-04:50 | 05:00-06:20 |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|  Saturday |             |             | CSE360 (FNK)| CSE360 (FNK)|             |             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|   Sunday  |             |             |             |             |             |             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|   Monday  | CSE360 (FNK)|             |             |             | CSE421 (SKZ)|             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|  Tuesday  |             |             | CSE421 (SKZ)| CSE421 (SKZ)|             |             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
| Wednesday | CSE360 (FNK)|             |             |             | CSE421 (SKZ)|             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|  Thursday |             |             |             |             |             |             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
```

[Routine: 2] ||| CSE360 - FNK - 01||| CSE421 - SRJ - 07|||

```
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|  Day/Time | 08:00-09:20 | 09:30-10:50 | 11:00-12:20 | 12:30-01:50 | 02:00-03:20 | 03:30-04:50 | 05:00-06:20 |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|  Saturday |             | CSE421 (SRJ)| CSE360 (FNK)| CSE360 (FNK)|             |             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|   Sunday  |             |             |             |             |             |             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|   Monday  | CSE360 (FNK)|             | CSE421 (SRJ)| CSE421 (SRJ)|             |             |             |
+-----------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|  Tuesday  |             |             |             |             |             |             |             |
```

Figure: My University 1.2

2. **Course Resources [Student]:** Here, Students can select any courses and it will provide the resource link of that course.



Course Resources

Write 'courseCode' for ANY course resources.

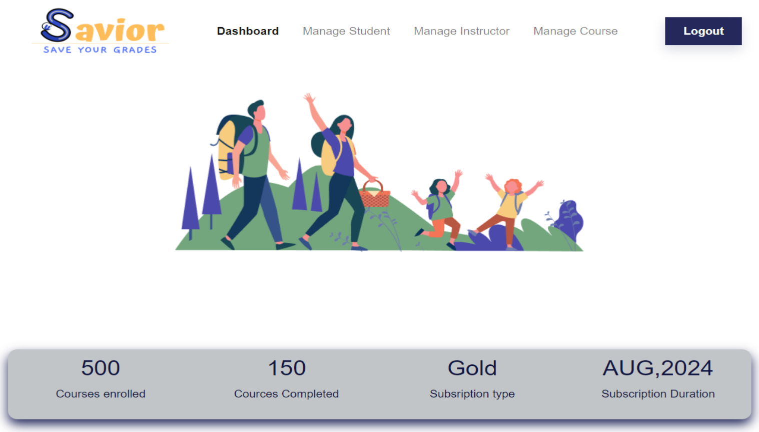
CSE110

Copyright © 2023 uSavior

Figure: My University 1.3

Admin Dashboard

After logging in as administrator, a dashboard will appear.

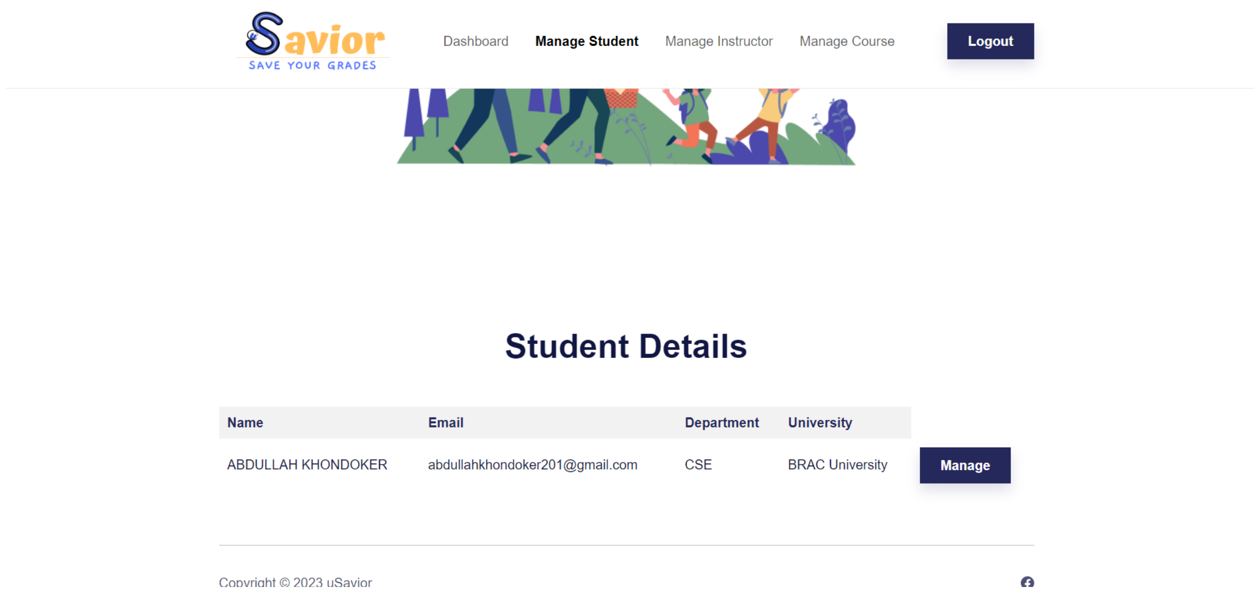


500	150	Gold	AUG,2024
Courses enrolled	Cources Completed	Subsription type	Subscription Duration

Figure: Admin Dashboard

This page has some buttons:

- Manage Student: Admin can manage the students who purchased courses.



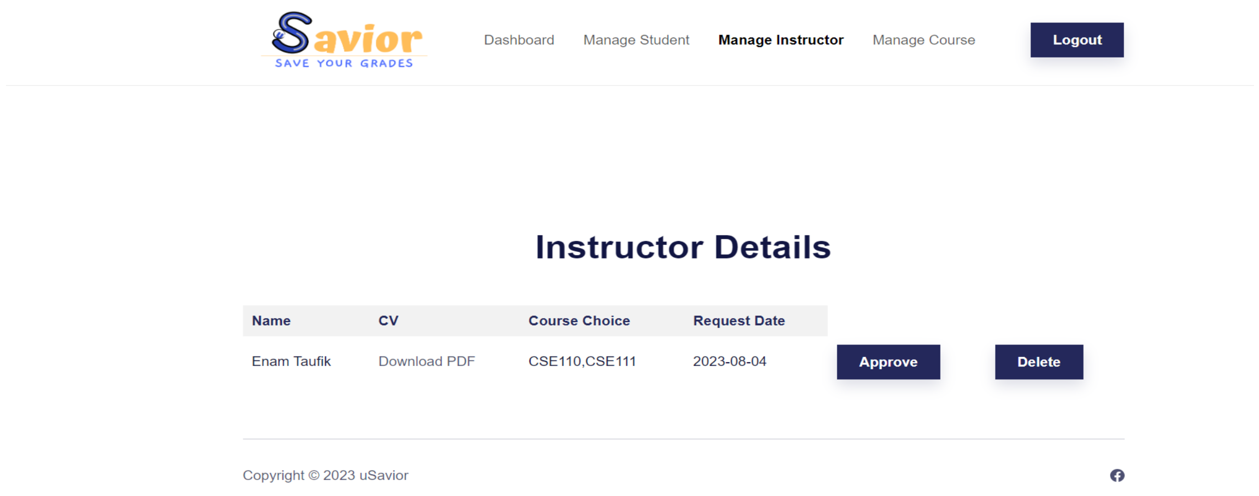
Student Details

Name	Email	Department	University	
ABDULLAH KHONDOKER	abdullahkhondoker201@gmail.com	CSE	BRAC University	Manage

Copyright © 2023 uSavior

Figure: Manage Student

- Manage Instructor: Admin can see the instructors who request for courses and can approve or delete them.



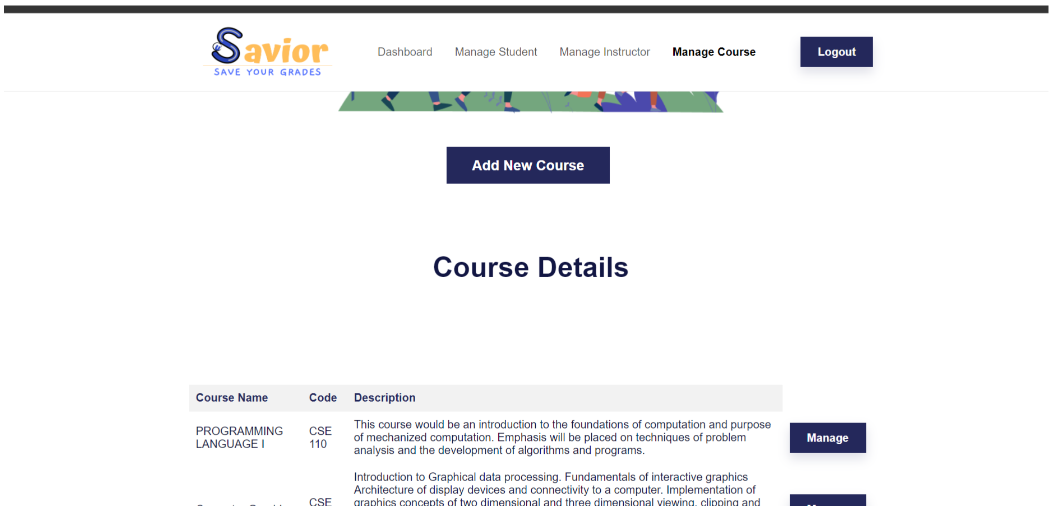
Instructor Details

Name	CV	Course Choice	Request Date		
Enam Taufik	Download PDF	CSE110,CSE111	2023-08-04	Approve	Delete

Copyright © 2023 uSavior

Figure: Manage Instructor

- Manage Courses: Here, admin can manage or edit the courses and according to students preferences add new courses.



Course Details

Course Name	Code	Description
PROGRAMMING LANGUAGE I	CSE 110	This course would be an introduction to the foundations of computation and purpose of mechanized computation. Emphasis will be placed on techniques of problem analysis and the development of algorithms and programs.
	CSE	Introduction to Graphical data processing. Fundamentals of interactive graphics Architecture of display devices and connectivity to a computer. Implementation of graphics concepts of two dimensional and three dimensional viewing, clipping and

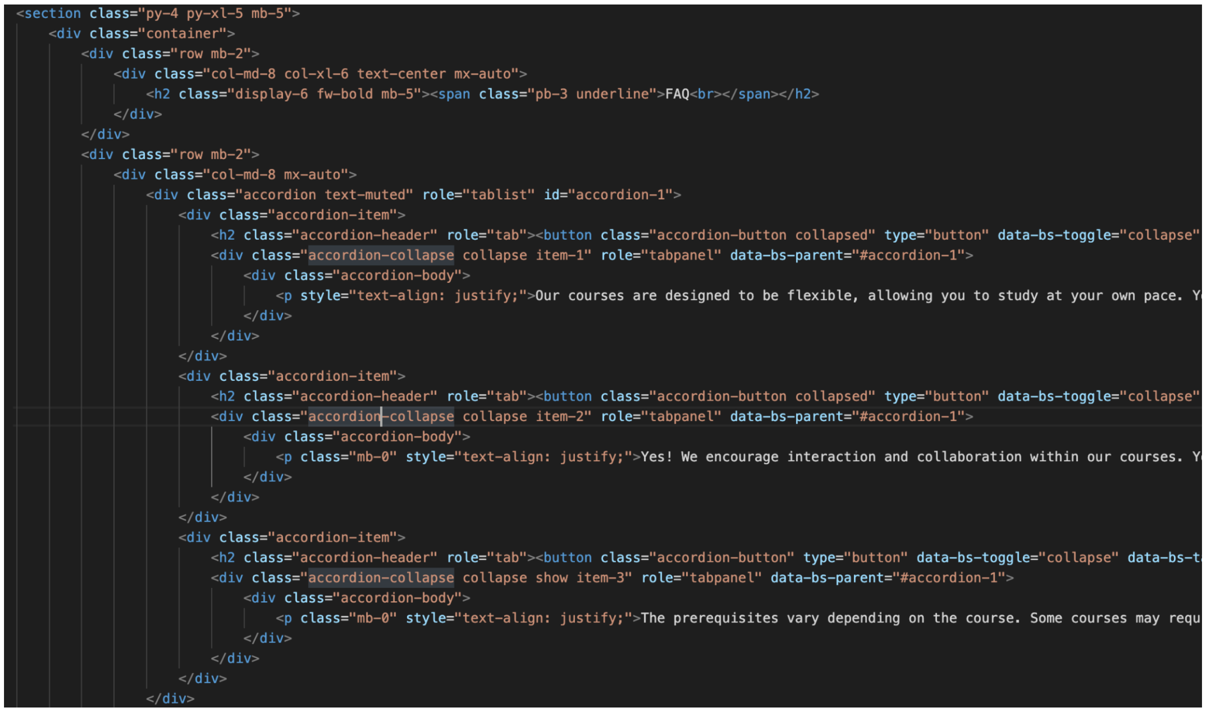
Figure: Course Details

- Logout: With this option, admin or users can sign out from their account.

Frontend Development

The web application utilizes the following front-end programming languages: **HTML, CSS, and JavaScript.**

HTML was used to create the basic structure of this webpage.

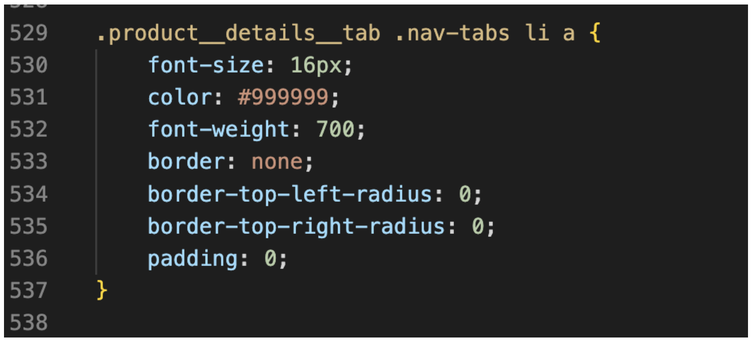


```html
<section class="py-4 py-xl-5 mb-5">
    <div class="container">
        <div class="row mb-2">
            <div class="col-md-8 col-xl-6 text-center mx-auto">
                <h2 class="display-6 fw-bold mb-5"><span class="pb-3 underline">FAQ<br></span></h2>
            </div>
        </div>
        <div class="row mb-2">
            <div class="col-md-8 mx-auto">
                <div class="accordion text-muted" role="tablist" id="accordion-1">
                    <div class="accordion-item">
                        <h2 class="accordion-header" role="tab"><button class="accordion-button collapsed" type="button" data-bs-toggle="collapse"
                        <div class="accordion-collapse collapse item-1" role="tabpanel" data-bs-parent="#accordion-1">
                            <div class="accordion-body">
                                <p style="text-align: justify;">Our courses are designed to be flexible, allowing you to study at your own pace. Y
                            </div>
                        </div>
                    </div>
                    <div class="accordion-item">
                        <h2 class="accordion-header" role="tab"><button class="accordion-button collapsed" type="button" data-bs-toggle="collapse"
                        <div class="accordion-collapse collapse item-2" role="tabpanel" data-bs-parent="#accordion-1">
                            <div class="accordion-body">
                                <p class="mb-0" style="text-align: justify;">Yes! We encourage interaction and collaboration within our courses. Y
                            </div>
                        </div>
                    </div>
                    <div class="accordion-item">
                        <h2 class="accordion-header" role="tab"><button class="accordion-button" type="button" data-bs-toggle="collapse" data-bs-t
                        <div class="accordion-collapse collapse show item-3" role="tabpanel" data-bs-parent="#accordion-1">
                            <div class="accordion-body">
                                <p class="mb-0" style="text-align: justify;">The prerequisites vary depending on the course. Some courses may requ
                            </div>
                        </div>
                    </div>
                </div>
```

Figure: HTML code snippet

CSS was utilized to style and design the appearance of elements on this webpage.

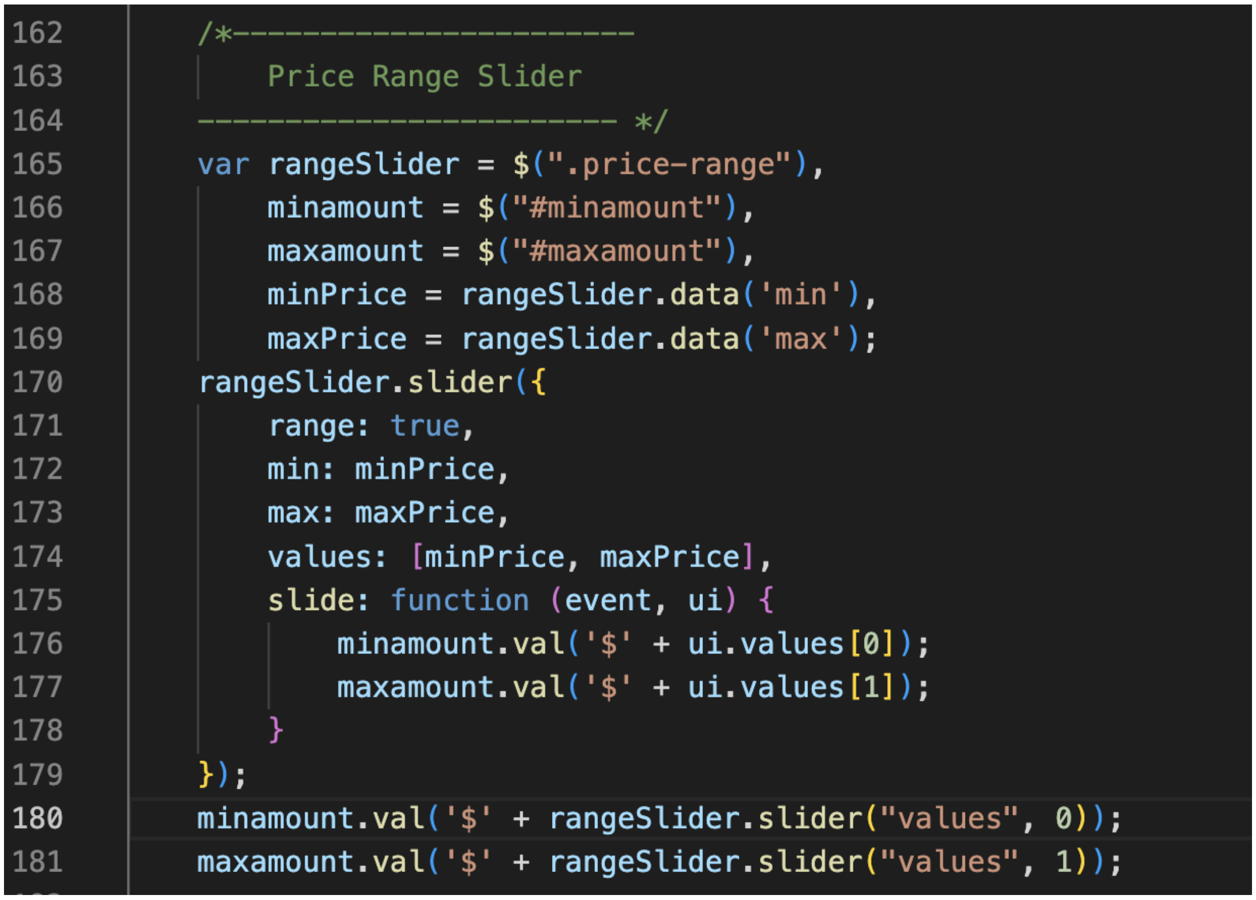
Bootstrap was also used to enhance the webpage's layout and responsiveness



```css
.product__details__tab .nav-tabs li a {
    font-size: 16px;
    color: #999999;
    font-weight: 700;
    border: none;
    border-top-left-radius: 0;
    border-top-right-radius: 0;
    padding: 0;
}
```

Figure: CSS Code Snippet

Javascript was additionally employed to incorporate various dynamic features.



```
/*----------------------
    Price Range Slider
------------------------ */
var rangeSlider = $(".price-range"),
    minamount = $("#minamount"),
    maxamount = $("#maxamount"),
    minPrice = rangeSlider.data('min'),
    maxPrice = rangeSlider.data('max');
rangeSlider.slider({
    range: true,
    min: minPrice,
    max: maxPrice,
    values: [minPrice, maxPrice],
    slide: function (event, ui) {
        minamount.val('$' + ui.values[0]);
        maxamount.val('$' + ui.values[1]);
    }
});
minamount.val('$' + rangeSlider.slider("values", 0));
maxamount.val('$' + rangeSlider.slider("values", 1));
```

Figure: Javascript Code Snippet

For example, the provided JavaScript code snippet initializes a price range slider, allowing users to select a range of prices with interactive input fields.

Backend Development

Our website was developed using the MVC (Model-View-Controller) pattern, ensuring a structured design.
It catered to three primary member categories:

1. Admins
2. Instructors
3. Students

Each had their distinct roles and functionalities. This approach facilitated efficient management and interaction within the website's ecosystem.

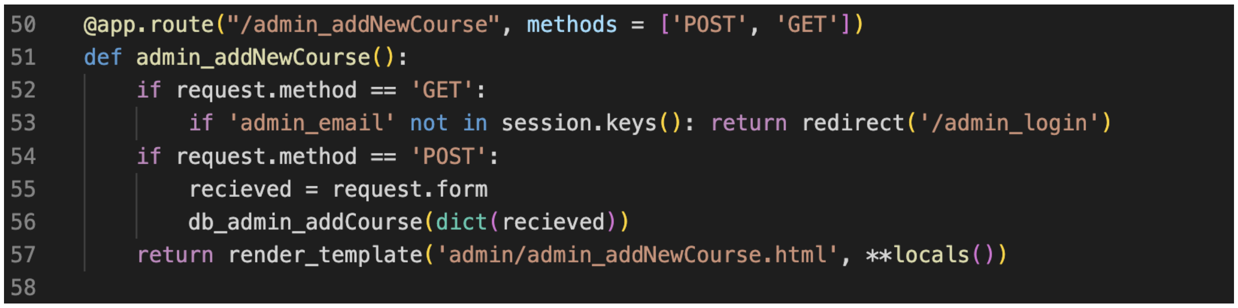
Types of Users and their Functionalities

Admin
The admin controller in the project employs a range of distinct functions to effectively manage various aspects of the admin's responsibilities within the web application. These functions can be broken down into the following key roles:

Function	Description
admin_login()	Handles admin login, validates credentials, and redirects to the admin dashboard upon successful login or to the login page if unsuccessful.
admin_dashboard()	Displays the admin dashboard, requiring authentication and rendering admin-specific data.
admin_StuManagement()	Renders the student management page for admins to view and manage student information, requiring authentication.
admin_InsManagement()	Renders the instructor management page for admins to view and manage instructor information, requiring authentication.
admin_CrsManagement()	Renders the course management page for admins to view and manage course information, requiring authentication.

admin_addNewCourse()	Handles adding a new course by admins, processing form data, and interacting with the database.
download_CV(file_id)	Allows downloading instructor CVs by admins, fetching and sending the CV file to the user.
admin_approve_Ins(file_id)	Approves an instructor application, generating a password, sending an approval email with login credentials, and updating the instructor's status.
admin_delete_Ins(file_id)	Deletes an instructor application, sending a rejection email and removing the instructor's application and data.
admin_logout()	Logs out admins, removing their session, and redirecting to the home page.

For example, The *'admin_addNewCourse'* function manages the addition of new courses through the admin interface. It verifies admin authentication for GET requests and processes form data for POST requests, which is then used to add course details to the database. The function also handles rendering the relevant template.



```
@app.route("/admin_addNewCourse", methods = ['POST', 'GET'])
def admin_addNewCourse():
    if request.method == 'GET':
        if 'admin_email' not in session.keys(): return redirect('/admin_login')
    if request.method == 'POST':
        recieved = request.form
        db_admin_addCourse(dict(recieved))
    return render_template('admin/admin_addNewCourse.html', **locals())

```

Figure: Add New Course by Admin Snippet

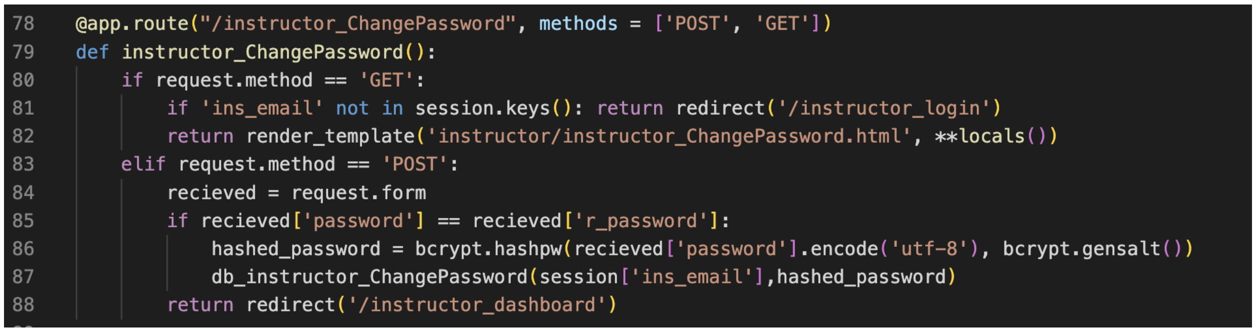
Instructor

The instructor controller is responsible for managing a diverse range of features and functionalities tailored to user instructors within the project. A list of functions dedicated to instructors, accompanied by their respective descriptions is given below:

Function	**Description**
instructor_login():	Handles instructor login, validates credentials, and redirects to the instructor dashboard upon successful login or to the login page if unsuccessful.

instructor_registration():	Manages instructor registration, verifies unique email, processes uploaded PDF files, sends a confirmation email, and registers instructor details.
instructor_dashboard()	Displays the instructor dashboard, requiring authentication and rendering instructor-specific data.
instructor_edit_profile()	Allows instructors to edit their profile information, saving changes to the database and redirecting to the instructor dashboard.
instructor_my_courses():	Renders the instructor's courses page, accessible only after authentication.
instructor_view_AllCourses()	Displays all courses for instructors to view, requiring authentication.
instructor_upload_video()	Presents the interface for instructors to upload videos, accessible post authentication.
instructor_ChangePassword()	Handles password change for instructors, updating hashed passwords and redirecting to the instructor dashboard.
instructor_logout()	Logs out instructors, removing their session and redirecting to the home page.

For example, The '*instructor_ChangePassword*' function manages the process of changing an instructor's password. For GET requests, it verifies instructor authentication and renders the password change template. For POST requests, it compares the entered password with its repetition, then securely hashes the new password and updates it in the database using `db_instructor_ChangePassword`. The instructor is then redirected to the dashboard.



```python
@app.route("/instructor_ChangePassword", methods = ['POST', 'GET'])
def instructor_ChangePassword():
    if request.method == 'GET':
        if 'ins_email' not in session.keys(): return redirect('/instructor_login')
        return render_template('instructor/instructor_ChangePassword.html', **locals())
    elif request.method == 'POST':
        recieved = request.form
        if recieved['password'] == recieved['r_password']:
            hashed_password = bcrypt.hashpw(recieved['password'].encode('utf-8'), bcrypt.gensalt())
            db_instructor_ChangePassword(session['ins_email'],hashed_password)
        return redirect('/instructor_dashboard')
```

Figure: Code Snippet for Instructor Password Change Function

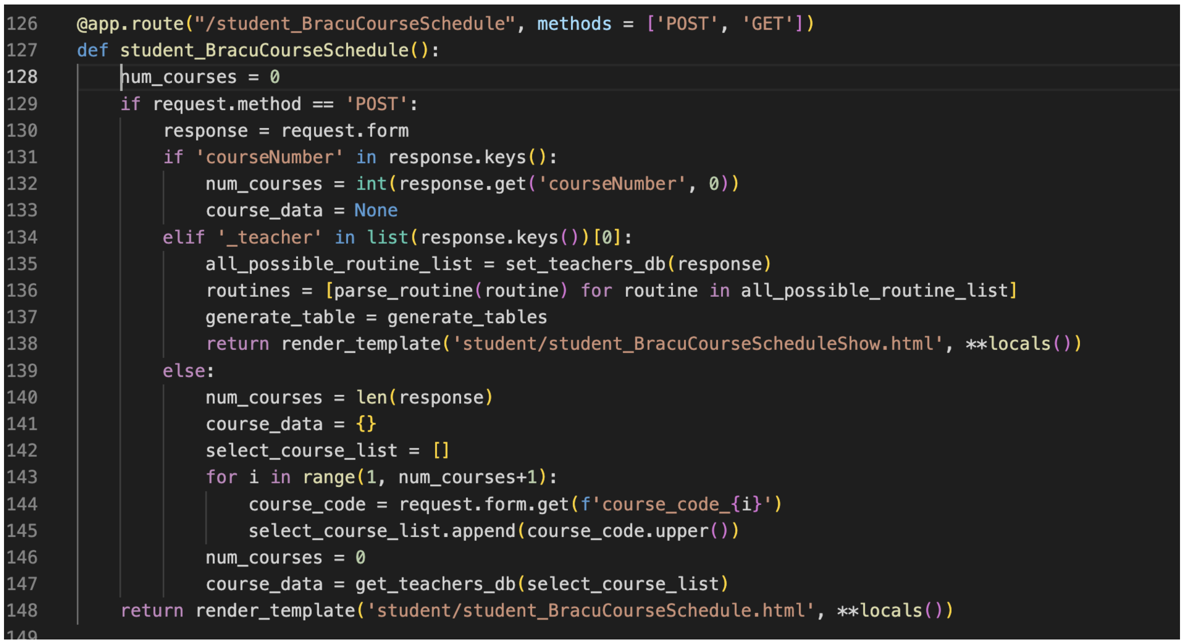
Student

The student controller within the provided code encompasses a diverse set of functions that collectively manage pivotal aspects of students' engagement within the web application. The functions, along with their respective descriptions, are presented below:

Function	Description
student_login()	Handles student login, validates credentials, and redirects to the student dashboard upon successful login or to the login page if unsuccessful.
student_registration()	Manages student registration, verifies unique email, validates password match, hashes the password, sends a confirmation email, and registers student details.
student_dashboard()	Displays the student dashboard, requiring authentication, and calculates the student's age based on their birthdate.
student_edit_profile()	Allows students to edit their profile information, saving changes to the database, and redirecting to the student dashboard.
student_my_courses()	Renders the student's enrolled courses page, accessible only after authentication.
student_my_university()	Renders the student's university information page, accessible only after authentication.
student_View_AllCourses()	Displays all available courses for students to view, requiring authentication.
student_View_Cart()	Renders the student's cart page, likely for adding and purchasing courses, accessible post authentication.
student_play_video()	Presents the interface for students to play videos, accessible after authentication.
student_video_list()	Displays a list of available videos for students to view, requiring authentication.
student_ChangePassword()	Handles password change for students, updating hashed passwords and redirecting to the student dashboard.

student_specific_course_details()	Displays detailed information about a specific course, presumably after the student selects a course.
student_BracuCourseSchedule()	Handles viewing the course schedule for BRAC University, allowing students to search for course routines based on various criteria.
student_BracuCourseResource():	Handles viewing course resources for BRAC University, enabling students to search for resources related to a specific course.
student_logout()	Logs out students, removing their session, and redirecting to the home page.

For example, the *student_BracuCourseSchedule* function is used to allow students to generate all possible routine combinations for specific courses and sections at BRAC University. In this function, when a POST request is received, the function processes the form data to determine the number of selected courses. It checks if a specific course number was provided, and if so, it prepares routine data by interacting with the database. Alternatively, if a teacher's name is detected, it generates a course schedule table and renders a template to display the schedule.



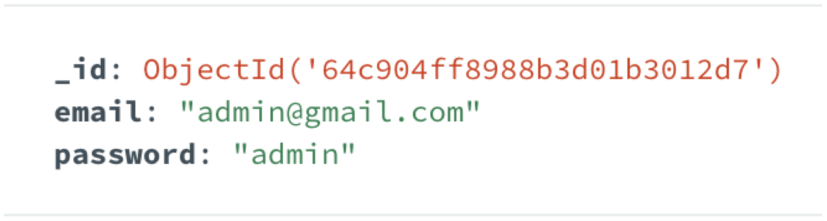
```python
@app.route("/student_BracuCourseSchedule", methods = ['POST', 'GET'])
def student_BracuCourseSchedule():
    num_courses = 0
    if request.method == 'POST':
        response = request.form
        if 'courseNumber' in response.keys():
            num_courses = int(response.get('courseNumber', 0))
            course_data = None
        elif '_teacher' in list(response.keys())[0]:
            all_possible_routine_list = set_teachers_db(response)
            routines = [parse_routine(routine) for routine in all_possible_routine_list]
            generate_table = generate_tables
            return render_template('student/student_BracuCourseScheduleShow.html', **locals())
        else:
            num_courses = len(response)
            course_data = {}
            select_course_list = []
            for i in range(1, num_courses+1):
                course_code = request.form.get(f'course_code_{i}')
                select_course_list.append(course_code.upper())
            num_courses = 0
            course_data = get_teachers_db(select_course_list)
    return render_template('student/student_BracuCourseSchedule.html', **locals())
```

Figure: Code Snippet for Student Course Schedule Generator Function

Database Design

In our project, MongoDB serves as the primary database management system, accommodating three distinct databases: ‘Admin,’ ‘Local,’ and ‘Usavior’ Within the ‘Usavior’ database, we have organized data into four collections:

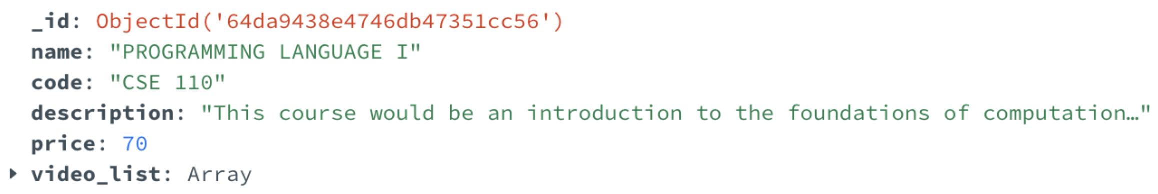
1. **Admin:** The attributes for this collection are _id, email and password.



```
_id: ObjectId('64c904ff8988b3d01b3012d7')
email: "admin@gmail.com"
password: "admin"
```

Figure: Admin database attributes

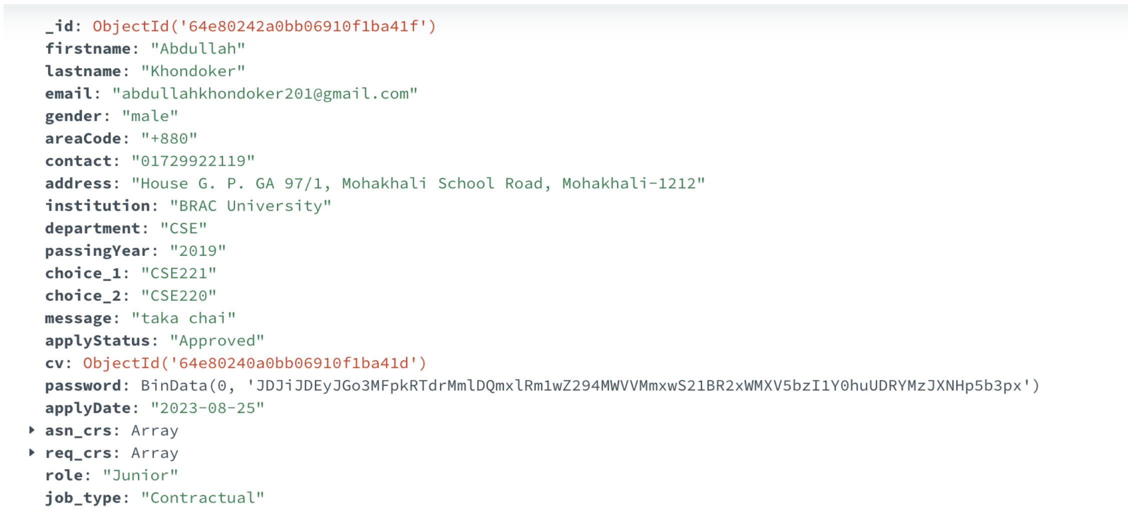
2. **Courses:** The attributes for this collections are_id, name, code, description, price, video_list



```
  _id: ObjectId('64da9438e4746db47351cc56')
  name: "PROGRAMMING LANGUAGE I"
  code: "CSE 110"
  description: "This course would be an introduction to the foundations of computation…"
  price: 70
▸ video_list: Array
```

Figure: Courses database attributes

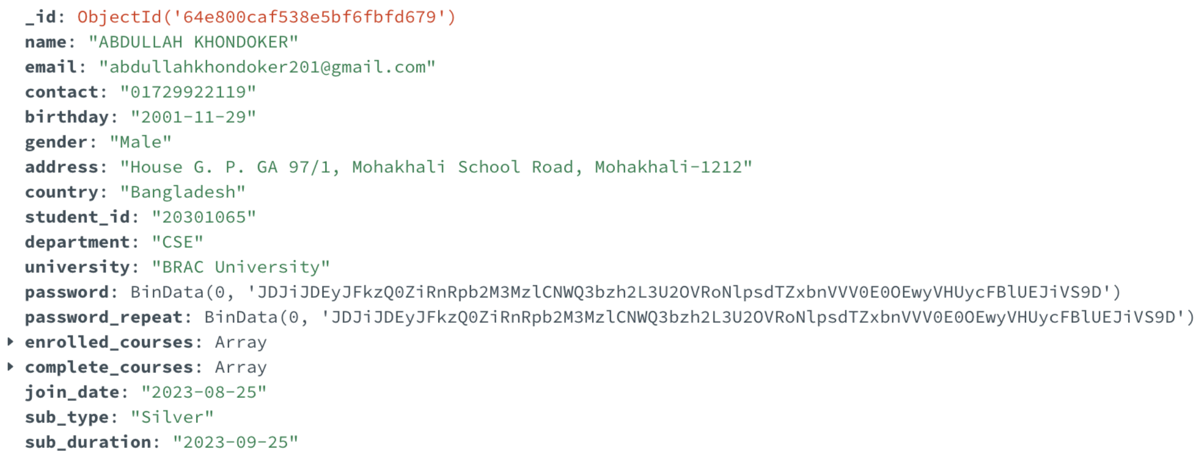
3. **Instructor:** The attributes for this collections are _id, firstname, lastname, email, gender, areaCode, contact, address, institution, department, passingYear, choice_1, choice_2, message, applyStatus, cv, password, applyDate, asn_crs, req_crs, role, job_type



```
  _id: ObjectId('64e80242a0bb06910f1ba41f')
  firstname: "Abdullah"
  lastname: "Khondoker"
  email: "abdullahkhondoker201@gmail.com"
  gender: "male"
  areaCode: "+880"
  contact: "01729922119"
  address: "House G. P. GA 97/1, Mohakhali School Road, Mohakhali-1212"
  institution: "BRAC University"
  department: "CSE"
  passingYear: "2019"
  choice_1: "CSE221"
  choice_2: "CSE220"
  message: "taka chai"
  applyStatus: "Approved"
  cv: ObjectId('64e80240a0bb06910f1ba41d')
  password: BinData(0, 'JDJiJDEyJGo3MFpkRTdrMmlDQmxlRm1wZ294MWVVMmxwS21BR2xWMXV5bzI1Y0huUDRYMzJXNHp5b3px')
  applyDate: "2023-08-25"
▸ asn_crs: Array
▸ req_crs: Array
  role: "Junior"
  job_type: "Contractual"
```

Figure: Instructor database attributes

4. **Student:** The attributes of this collection are id, name, email, contact, birthday, gender, address, country, student_id, department, university, password, password_repeat, enrolled_courses, complete_courses, join_date, sub_type, sub_duration.



```
  _id: ObjectId('64e800caf538e5bf6fbfd679')
  name: "ABDULLAH KHONDOKER"
  email: "abdullahkhondoker201@gmail.com"
  contact: "01729922119"
  birthday: "2001-11-29"
  gender: "Male"
  address: "House G. P. GA 97/1, Mohakhali School Road, Mohakhali-1212"
  country: "Bangladesh"
  student_id: "20301065"
  department: "CSE"
  university: "BRAC University"
  password: BinData(0, 'JDJiJDEyJFkzQ0ZiRnRpb2M3MzlCNWQ3bzh2L3U2OVRoNlpsdTZxbnVVV0E0OEwyVHUycFBlUEJiVS9D')
  password_repeat: BinData(0, 'JDJiJDEyJFkzQ0ZiRnRpb2M3MzlCNWQ3bzh2L3U2OVRoNlpsdTZxbnVVV0E0OEwyVHUycFBlUEJiVS9D')
▸ enrolled_courses: Array
▸ complete_courses: Array
  join_date: "2023-08-25"
  sub_type: "Silver"
  sub_duration: "2023-09-25"
```

Figure: Student database attributes

Technology (Framework, Languages)

Framework: Flask with MVC pattern

Languages: Python, Cython, HTML, CSS

Github Repository

Link: https://github.com/abdullah1065/uSavior

Website Link: https://usavior.vercel.app/home

Individual Contribution

ID	Name	Contribution
20101437	Antara Firoz Parsa	Edit Profile, Add course, My course.
20301065	Abdullah Khondoker	API System, Web Scraping, Authentication, Encryption
20301078	Md. Iftekhar Islam Tashik	User Registration, Admin Management
20301398	Enam Ahmed Taufik	Dashboard, View Cart, Student Course Panel.