School of Computer Science and Electrical Engineering
University of Ottawa

GNG - 5300 | Fall 2024
Full Stack Software Development

Assignment 2
Student Management System

Name	Student Number
Ahmad El Mokahal	300059298

Instructor: Masoud Dorrikhteh
Date: 2024/09/30

Setup Process

The project was set up according to the instructions on the Django tutorial page. A virtual environment was created using the `venv` Python module, and the latest version of Django was installed. No other packages were installed for the creation of this project. An initial commit was made to the project GitHub repository under “Project Setup”.

Challenges Encountered

No significant challenges were encountered in the implementation of this project, the Django documentation page was used extensively to implement all of the main features. All of the Student Management System features were implemented as described in the project requirements.

Implementation

Student Models

To create a Django model for a Student. A “students” application was created, and the model was added to the students/models.py directory. The model contains the following attributes: first name, last name, email, date of birth, enrollment date, and grade. The following is a code snippet for that class.

```
class Student(models.Model):
    first_name = models.CharField(max_length=50)
    last_name = models.CharField(max_length=50)
    email = models.EmailField(unique=True)
    date_of_birth = models.DateField()
    enrollment_date = models.DateField()
    grade = models.IntegerField()
```

Admin Interface

The students/admin.py file was modified to register the Student model in the admin panel and to display all student attributes in the panel with the following code:

```
@admin.register(Student)
class StudentAdmin(admin.ModelAdmin):
    list_display = ('first_name', 'last_name', 'enrollment_date')
```

An admin account was created using the following credentials:

Username: admin
Password: admin123

Views and Templates

Four views were created in the students/views.py file: student_list, student_details, student_create, and student_edit. To represent all of the operations that can be performed on a Student record. Each view processes the request and renders the appropriate template for that view. The following templates were created for this project in the students/templates directory:

- base.html: To create a common page that can be used to apply Bootstrap styling to all of the other templates.
- student_details.html: To display all of the details for a single student record.
- student_form.html: To display a common form that can be used for both creating a new student record and editing an existing student record.
- student_list.html: To display a list of all the existing student records in a table, and the buttons to View/Edit each record as well as a button to create new records.

The following url routes were created for to represent each new page:

`/student_list` the main route for this application which shows the list of student records.
`/<int>/` the route used to display a specific record depending on its ID integer value in the database.
`/<int>/edit` the route used to edit a student record.
`/add/` the route used to display a form to add a new record.

Forms

A custom form called StudentForm was created in the students/forms.py directory for adding and updating student records. Validation functions were created for the email and grade input fields. The `clean_email` function ensures that emails must contain an ‘@’ and throwing a ValidationError if it violates the format. The `clean_grade` function ensures that the input grade is non-empty and between 1 and 12.

Authentication

User authentication was added to the system by creating a user login page, where only one admin account can be used to log in and add, edit, or delete student records from the list. A user can view the list of records without being logged in. Using the Django auth decorators, the `@login_required` decorator was used on each protected function in the students/views.py file. These functions include student_create, student_edit, and student_delete. The template for student_list.html was modified to ensure that create, edit, and delete options are only visible to authenticated users using the CSRF token issued when a user is authenticated.

Search Functionality

To implement the ability to search for student records by name, the students_list template was modified to include a search input. The student_list view function was modified to include a query that is used to filter the student objects and return the matching records to be displayed in the list.

```
def student_list(request):
    query = request.GET.get('q', '')
    if query:
        students = Student.objects.filter(
            first_name__icontains=query
        ) | Student.objects.filter(
            last_name__icontains=query
        )
    else:
        students = Student.objects.all()
    return render(request, 'students/student_list.html', {'students':
students, 'query': query})
```

Pagination

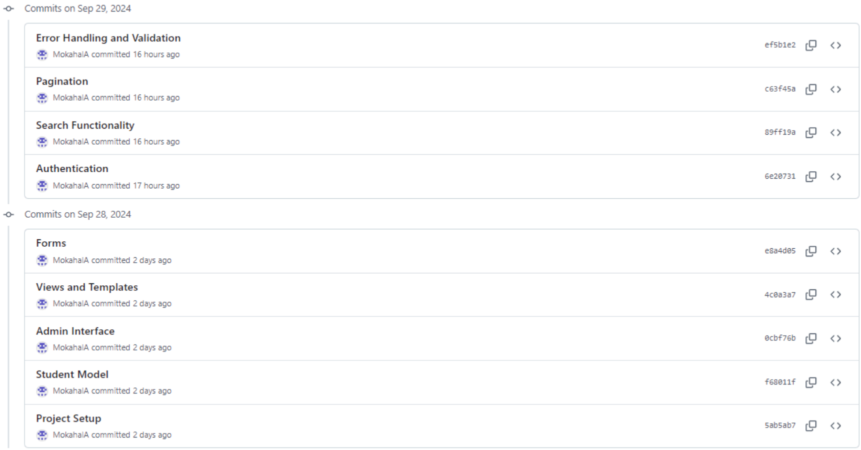
The pagination feature was implemented to restrict the number of displayed records on a page to 10 records. The user can see the next page of records by navigating using a “Next” button. The student_list.html template was modified to add pagination controls and text to display which page the user is viewing. The student_list view function was modified to include the Paginator from the Django.core package.

```
def student_list(request):
    query = request.GET.get('q', '')
    students = Student.objects.all()

    # Filter students based on the search query
    if query:
        students = students.filter(
            first_name__icontains=query
        ) | students.filter(
            last_name__icontains=query
        )

    # Implement pagination
    paginator = Paginator(students, 10)
    page_number = request.GET.get('page')
    page_obj = paginator.get_page(page_number)

    context = {
        'students': page_obj,
        'query': query,
    }
    return render(request, 'students/student_list.html', context)
```

Error Handling & Validation

All of the input fields that exist in the application were coded to include validators to require the user to enter an input & to prompt the user whenever the input is invalid. Throwing a ValidationError whenever the input is unexpected. The search results were ordered by first name to ensure consistent results each time the user views the list of records and to avoid the UnorderedObjectListWarning.

The Github commits for each implemented feature are shown below:



Commits on Sep 29, 2024

Error Handling and Validation
MokahalA committed 16 hours ago
ef5b1e2

Pagination
MokahalA committed 16 hours ago
c63f45a

Search Functionality
MokahalA committed 16 hours ago
89ff19a

Authentication
MokahalA committed 17 hours ago
6e20731

Commits on Sep 28, 2024

Forms
MokahalA committed 2 days ago
e8a4d05

Views and Templates
MokahalA committed 2 days ago
4c0a3a7

Admin Interface
MokahalA committed 2 days ago
0cbf76b

Student Model
MokahalA committed 2 days ago
f68011f

Project Setup
MokahalA committed 2 days ago
5ab5ab7

Application Demo

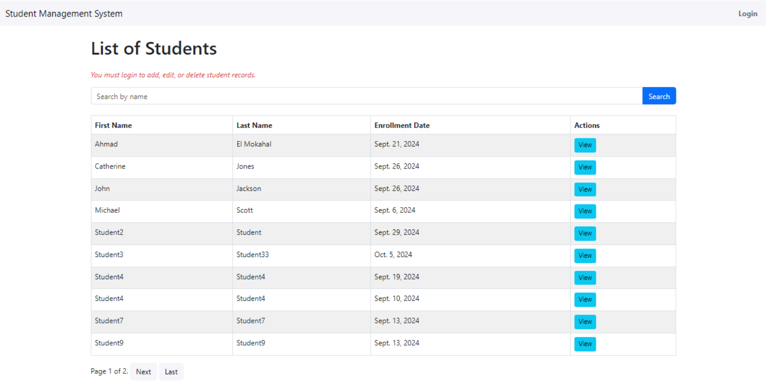
The Django application can be found in the following repository, before you run the application for the first time, you must make the necessary migrations first by executing the following command:

```
python manage.py migrate
```

To run the application from the root directory of the project, execute the following command:

```
python manage.py runserver
```

Navigating to the http://127.0.0.1:8000/students/ route shows the main page which displays the list of student records. While the user is not logged in as an admin, they can view each record and search from the list.



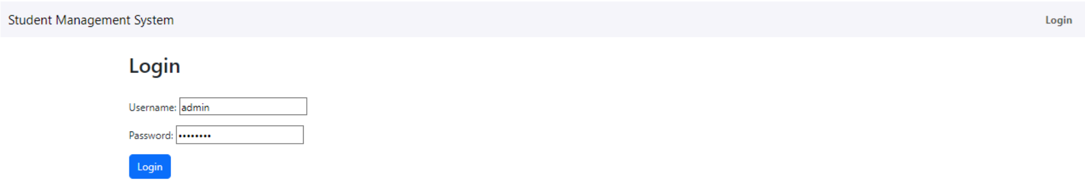
List of Students

You must login to add, edit, or delete student records.

First Name	Last Name	Enrollment Date	Actions
Ahmad	El Mokahal	Sept. 21, 2024	View
Catherine	Jones	Sept. 26, 2024	View
John	Jackson	Sept. 26, 2024	View
Michael	Scott	Sept. 6, 2024	View
Student2	Student	Sept. 29, 2024	View
Student3	Student33	Oct. 5, 2024	View
Student4	Student4	Sept. 19, 2024	View
Student4	Student4	Sept. 10, 2024	View
Student7	Student7	Sept. 13, 2024	View
Student9	Student9	Sept. 13, 2024	View

Page 1 of 2.

To log in, navigate to the top right corner and click the “Login” button. This will take the user to the login form. The admin credentials used are: username: **admin** password: **admin123** You may navigate back to the main screen by clicking on the header in the top left.

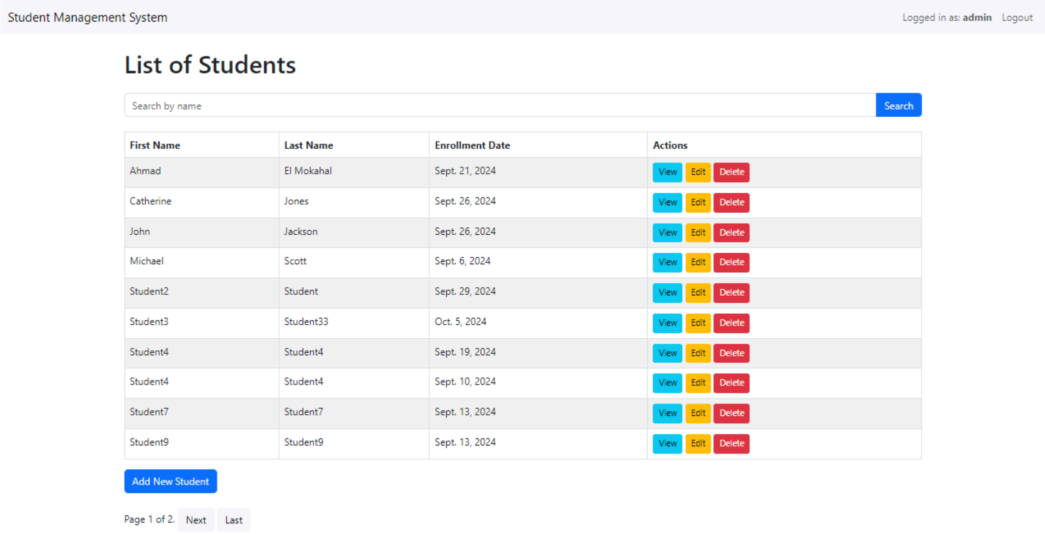


Login

Username: admin

Password: ••••••••

Once logged in, the user is redirected back to the main page where they can now edit, add, and delete records. The user can log out using the button in the top right corner.

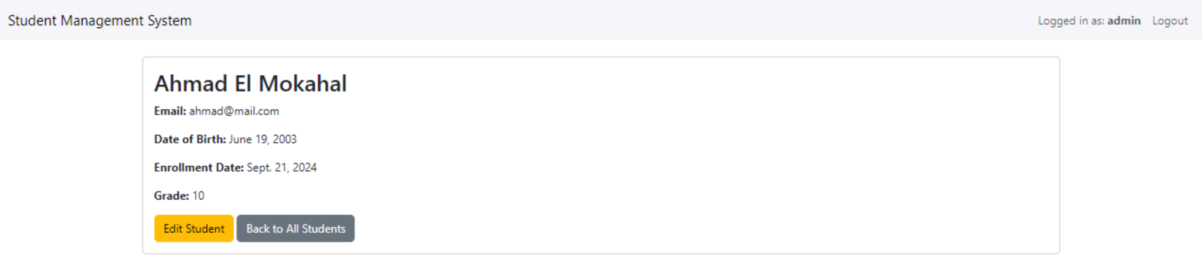


List of Students

First Name	Last Name	Enrollment Date	Actions
Ahmad	El Mokahal	Sept. 21, 2024	View Edit Delete
Catherine	Jones	Sept. 26, 2024	View Edit Delete
John	Jackson	Sept. 26, 2024	View Edit Delete
Michael	Scott	Sept. 6, 2024	View Edit Delete
Student2	Student	Sept. 29, 2024	View Edit Delete
Student3	Student33	Oct. 5, 2024	View Edit Delete
Student4	Student4	Sept. 19, 2024	View Edit Delete
Student4	Student4	Sept. 10, 2024	View Edit Delete
Student7	Student7	Sept. 13, 2024	View Edit Delete
Student9	Student9	Sept. 13, 2024	View Edit Delete

Page 1 of 2.

To demonstrate viewing a specific record, click on the “View” button corresponding to a record:



Ahmad El Mokahal

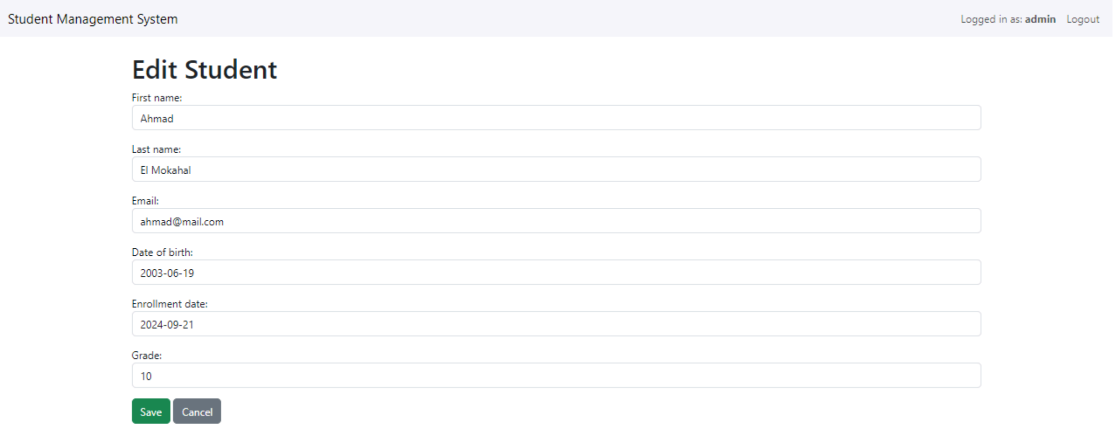
Email: ahmad@mail.com

Date of Birth: June 19, 2003

Enrollment Date: Sept. 21, 2024

Grade: 10

To demonstrate updating a student record:



Student Management System

Logged in as: **admin**

Edit Student

First name: Ahmad

Last name: El Mokahal

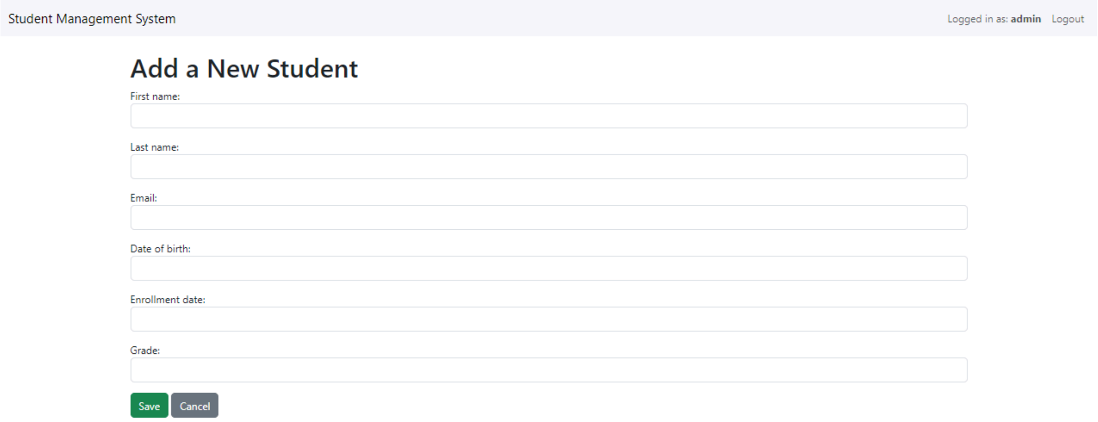
Email: ahmad@mail.com

Date of birth: 2003-06-19

Enrollment date: 2024-09-21

Grade: 10

To demonstrate adding a student record:



Student Management System

Logged in as: **admin**

Add a New Student

First name:

Last name:

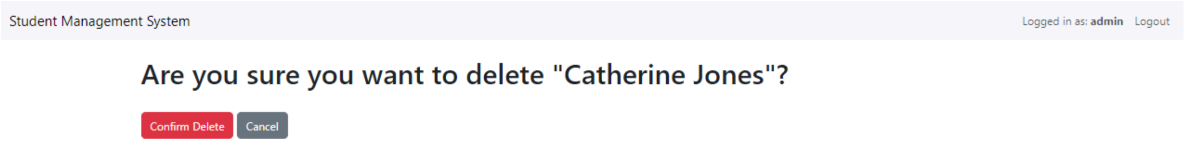
Email:

Date of birth:

Enrollment date:

Grade:

To demonstrate attempting to delete a new record for “Catherine Jones”:



Student Management System

Logged in as: **admin**

Are you sure you want to delete "Catherine Jones"?