	Course Name: Advanced Web Technology	**EXPERIMENT NO. 10**	
	Course Code: 20CP314P **Faculty: Komal Singh**	**Branch: CSE**	**Semester: VI**
Submitted by: Shivanjali Srivastav **Roll no: 22BCP110**			

Objective: Implement a web-app using Django framework

Experiment 10:

1) Design a 'Hello World' Web-application using Django framework.

2) Perform CRUD commands in a designed model using the default SQLite database extension in Django.

3) Connect your Django web-application to PostgreSQL database and perform CRUD commands.

Note: Please include snapshots of all commands, terminal sessions, localhost outputs, and log files in your documentation with necessary steps.

TASK 1

1. Create Virtual Environment

In PowerShell:

```
python -m venv env
```

2. Activate Virtual Environment

```
.\env\Scripts\activate
```

It’s activated when your prompt changes to something like:

```
(env) PS D:\sem6\awt\lab\exp10>
```

3. Install Django

```
pip install django
```

```
PS D:\sem6\awt\lab\exp10> python --version
Python 3.12.4
PS D:\sem6\awt\lab\exp10> python -m venv env
PS D:\sem6\awt\lab\exp10> .\env\Scripts\activate
(env) PS D:\sem6\awt\lab\exp10> pip install django
Collecting django
  Downloading Django-5.2-py3-none-any.whl.metadata (4.1 kB)
Collecting asgiref>=3.8.1 (from django)
  Downloading asgiref-3.8.1-py3-none-any.whl.metadata (9.3 kB)
Collecting sqlparse>=0.3.1 (from django)
  Downloading sqlparse-0.5.3-py3-none-any.whl.metadata (3.9 kB)
Collecting tzdata (from django)
  Using cached tzdata-2025.2-py2.py3-none-any.whl.metadata (1.4 kB)
Downloading Django-5.2-py3-none-any.whl (8.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.3/8.3 MB 27.3 kB/s eta 0:00:00
Downloading asgiref-3.8.1-py3-none-any.whl (23 kB)
Downloading sqlparse-0.5.3-py3-none-any.whl (44 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.4/44.4 kB 28.8 kB/s eta 0:00:00
Using cached tzdata-2025.2-py2.py3-none-any.whl (347 kB)
Installing collected packages: tzdata, sqlparse, asgiref, django
Successfully installed asgiref-3.8.1 django-5.2 sqlparse-0.5.3 tzdata-2025.2

[notice] A new release of pip is available: 24.0 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(env) PS D:\sem6\awt\lab\exp10>
```

Figure 1: Successful installation of Django

Run:

```
django-admin startproject helloworld_project
```

```
cd helloworld_project
```

Start a Django App

```
python manage.py startapp hello
```

Add App to Project Settings

Open `helloworld_project/settings.py` and **add `'hello',` to the `INSTALLED_APPS` list**:

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'hello',  # Add this line
]
```

Create a “Hello World” View

In `hello/views.py`, Replace that with the "Hello, World!" view:

```
from django.http import HttpResponse

def hello_world(request):
    return HttpResponse("Hello, World!")
```

Create `hello/urls.py`

Create a new file named `urls.py` inside the `hello` folder and paste this:

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.hello_world),
]
```

Link the app's URL to the main project

In `helloworld_project/urls.py` (located next to `settings.py`), modify like this:

```
from django.contrib import admin
from django.urls import path, include  # include is required

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('hello.urls')),  # this links to your hello app
]
```

Run the server

In terminal:

```
python manage.py runserver
```

```
(env) PS D:\sem6\awt\lab\exp10\helloworld_project> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
April 08, 2025 - 11:52:34
Django version 5.2, using settings 'helloworld_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

WARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server instead.
For more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/
[08/Apr/2025 11:53:24] "GET / HTTP/1.1" 200 13
Not Found: /favicon.ico
[08/Apr/2025 11:53:24] "GET /favicon.ico HTTP/1.1" 404 2453
```

Figure 2: After running the command to start the server

Open browser and go to:

http://127.0.0.1:8000/

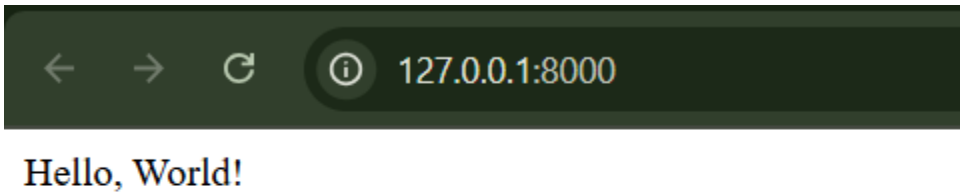


Figure 3: Browser page of http://127.0.0.1:8000/

TASK 2

1. Create a Model

In `hello/models.py`, define a simple model. For example, a `Student`:

```
from django.db import models

class Student(models.Model):
    name = models.CharField(max_length=100)
    roll_no = models.IntegerField()
    branch = models.CharField(max_length=100)

    def __str__(self):
        return self.name
```

2. Register Model in Admin Panel

In `hello/admin.py`:

```
from django.contrib import admin
from .models import Student

admin.site.register(Student)
```

3. Add App to Installed Apps (already done)

Just confirming: `hello` is added to `INSTALLED_APPS` in `settings.py`:

```
'hello',
```

4. Make Migrations

In terminal:

```
python manage.py makemigrations
```

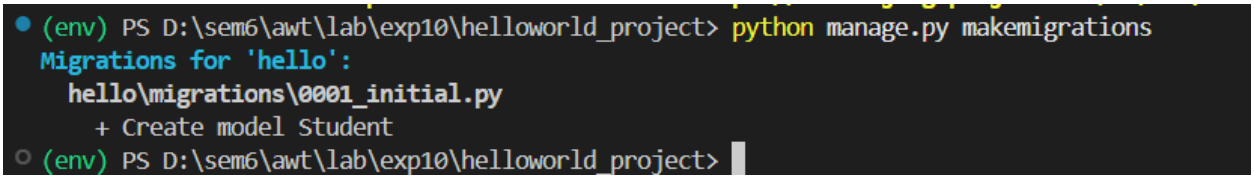


Figure 4

Then apply it:

```
python manage.py migrate
```

```
(env) PS D:\sem6\awt\lab\exp10\helloworld_project> python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, hello, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying hello.0001_initial... OK
  Applying sessions.0001_initial... OK
(env) PS D:\sem6\awt\lab\exp10\helloworld_project>
```

Figure 5

5. Create Superuser (To access Django Admin)

```
python manage.py createsuperuser
```

It will ask:

- Username
- Email
- Password (type twice)

```
Username (leave blank to use 'shivanjalis'):
Email address: shivanjalisriv@gmail.com
Password:
Password (again):
Superuser created successfully.
```

Figure 6

6. Run Server and Open Admin Panel

```
python manage.py runserver
```

Then go to:

http://127.0.0.1:8000/admin/

- Log in with your superuser credentials
- You'll see **Students** section under "HELLO"
- Add some sample student entries there

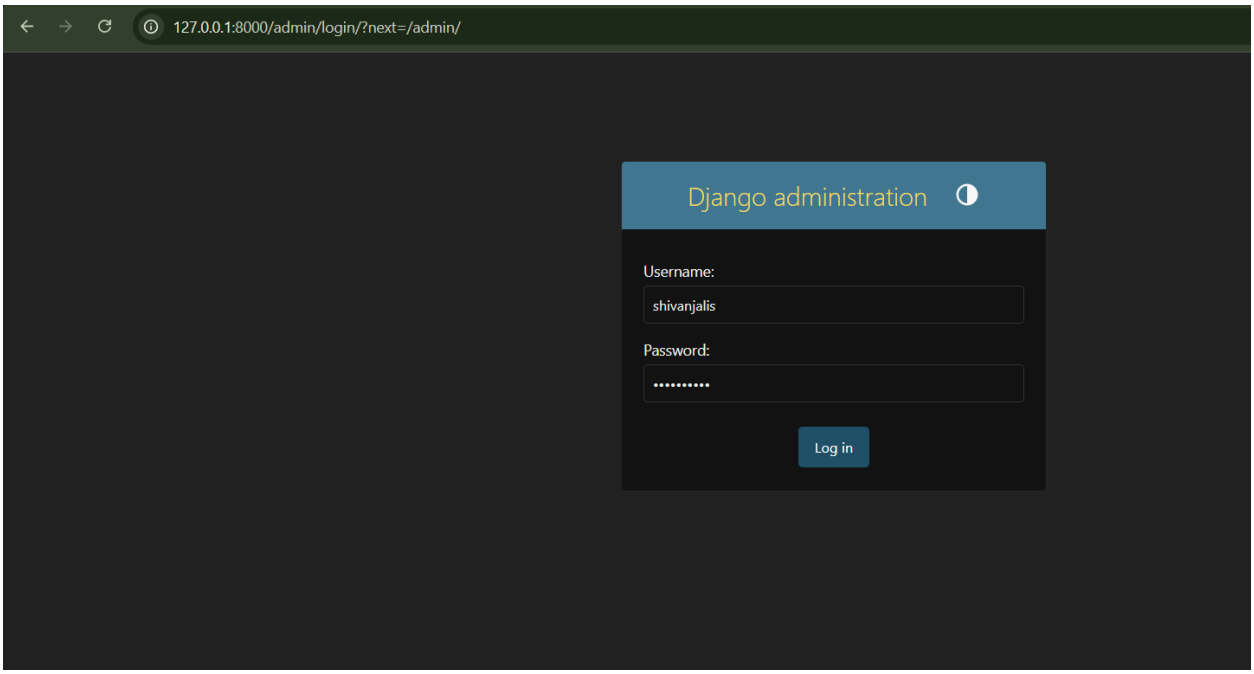


Figure 7: Admin Login Page

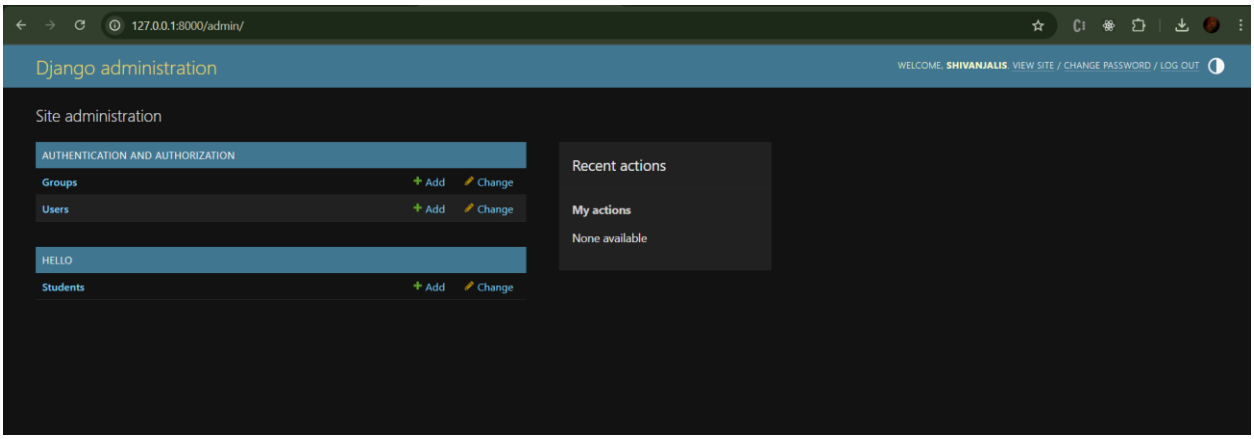


Figure 8: Admin dashboard

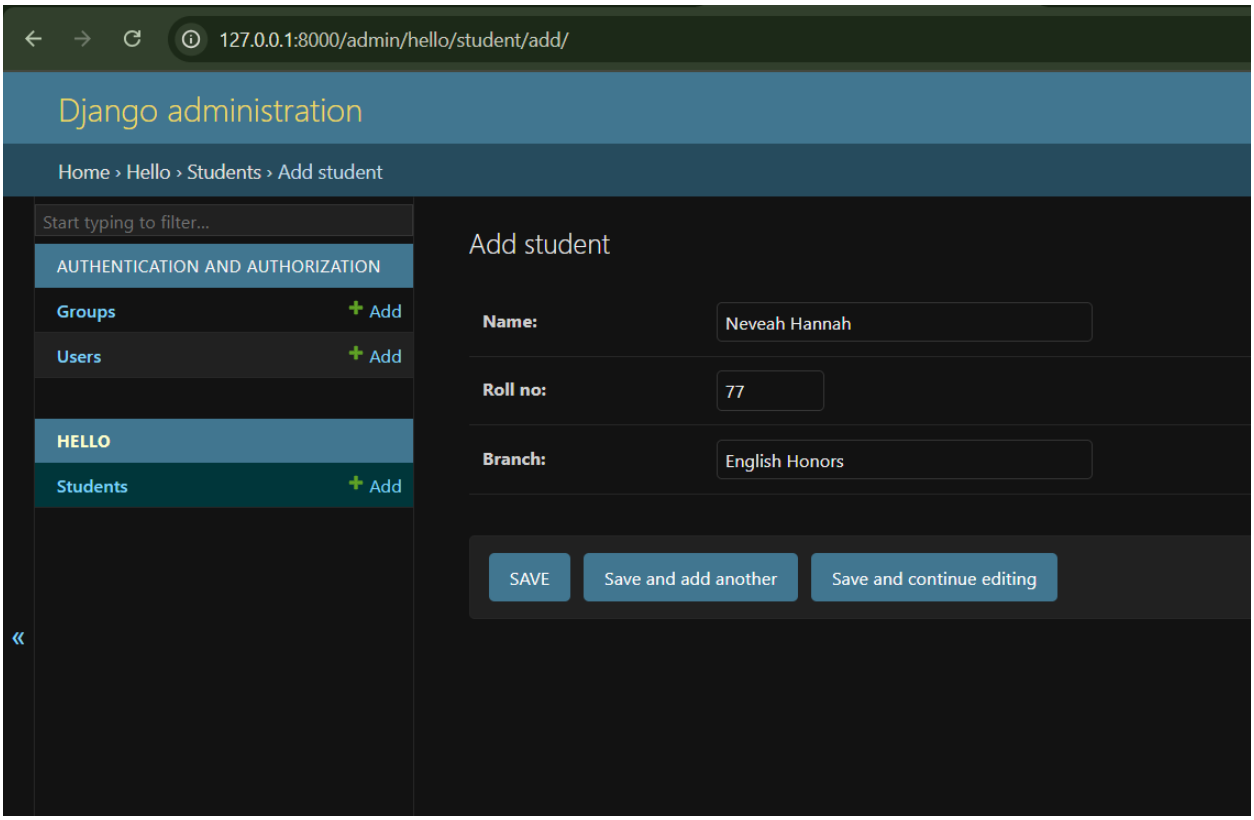


Figure 9: C (Create)

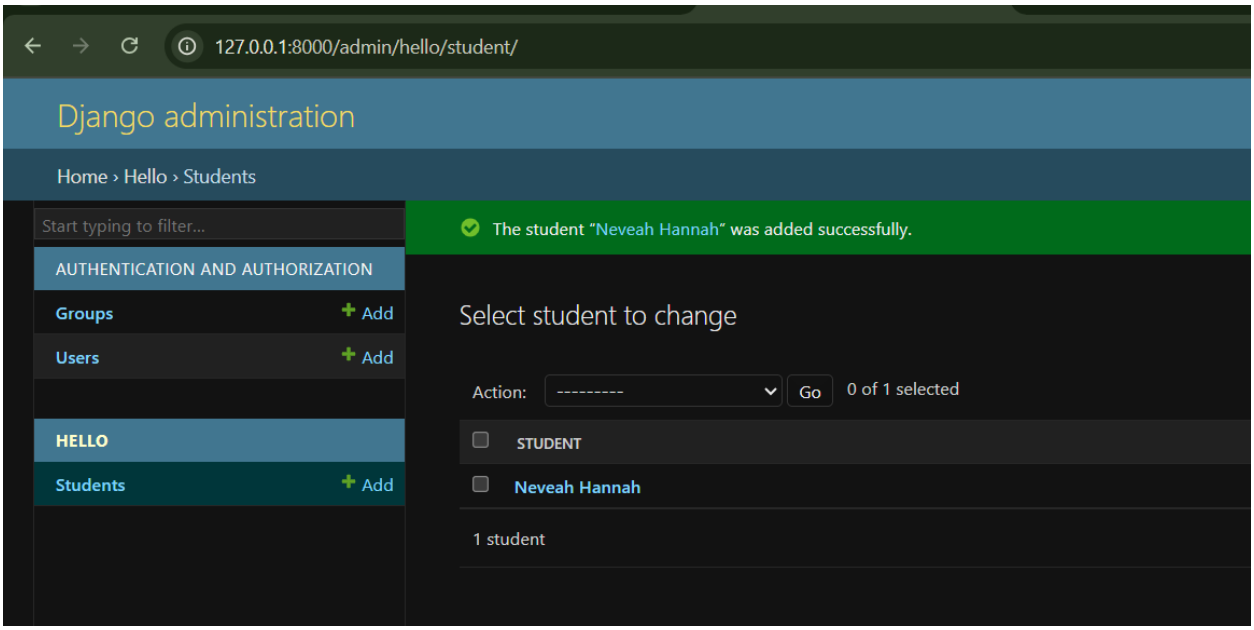


Figure 10: Successful C(create) operation

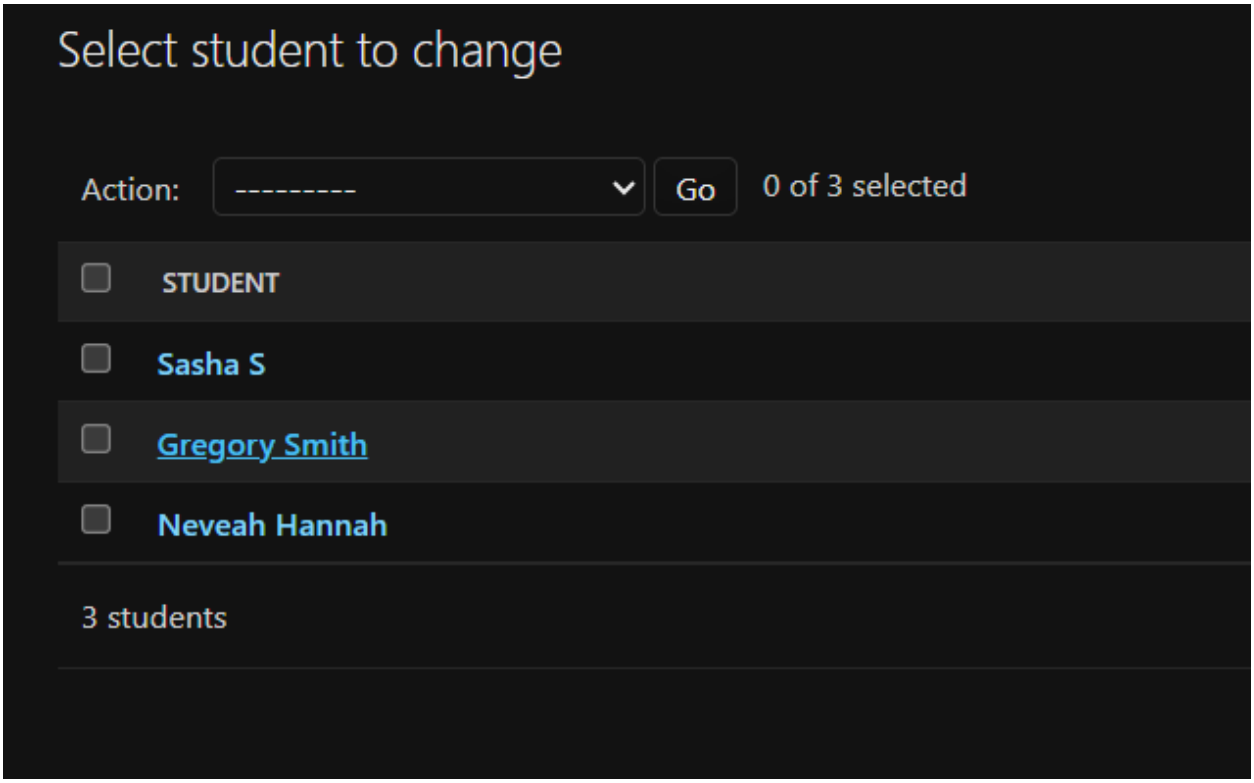


Figure 11: R(read)

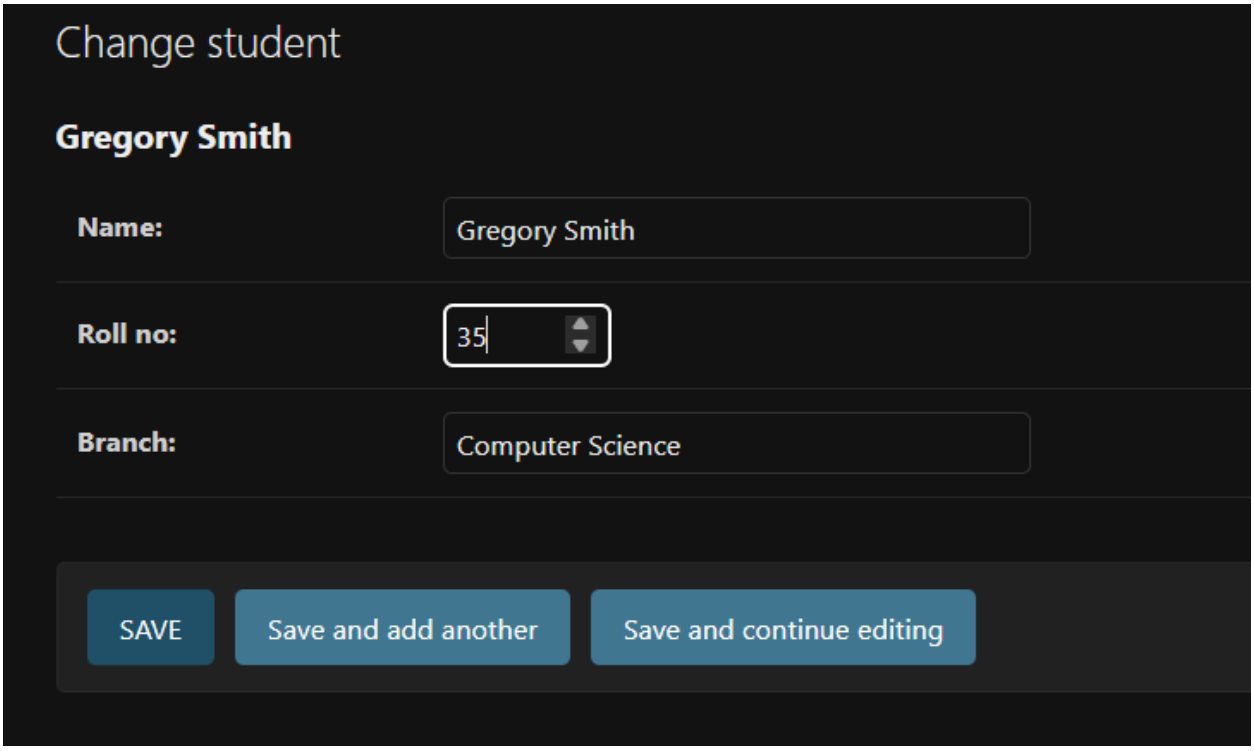


Figure 12: U(Update)

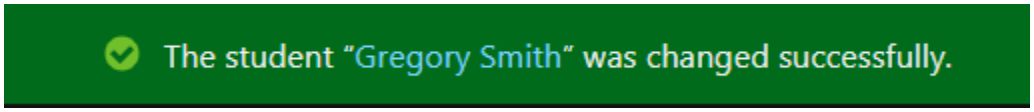


Figure 13: Successful U(update) operation

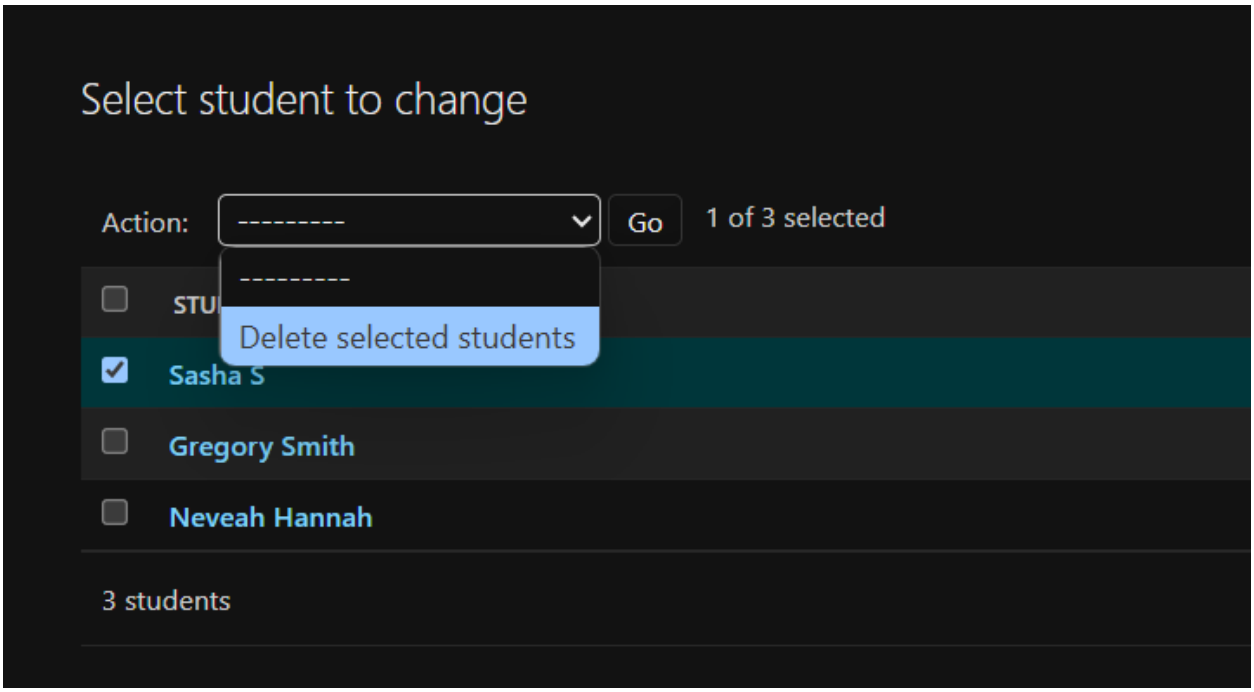


Figure 14: D(delete)

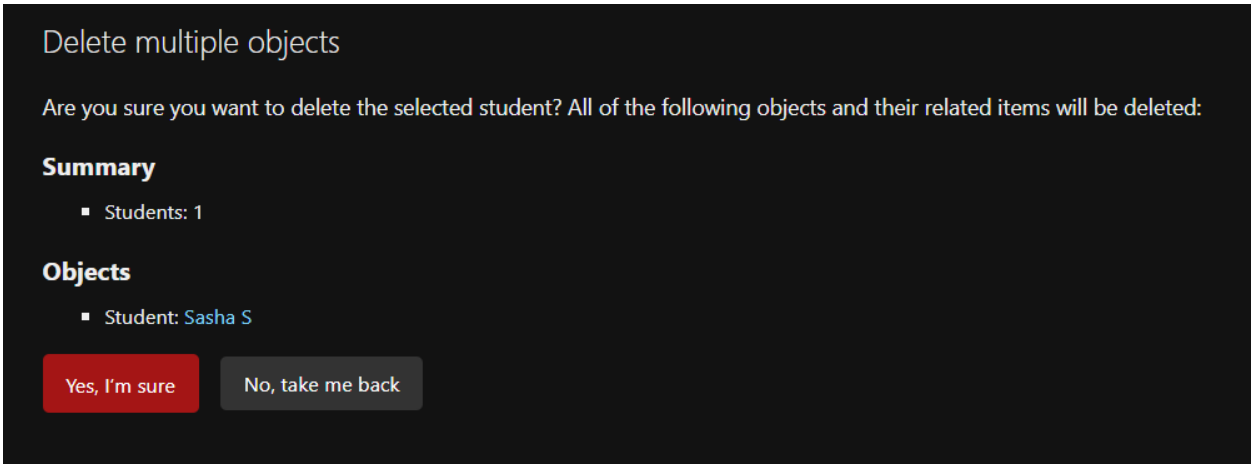


Figure 15: D(delete)

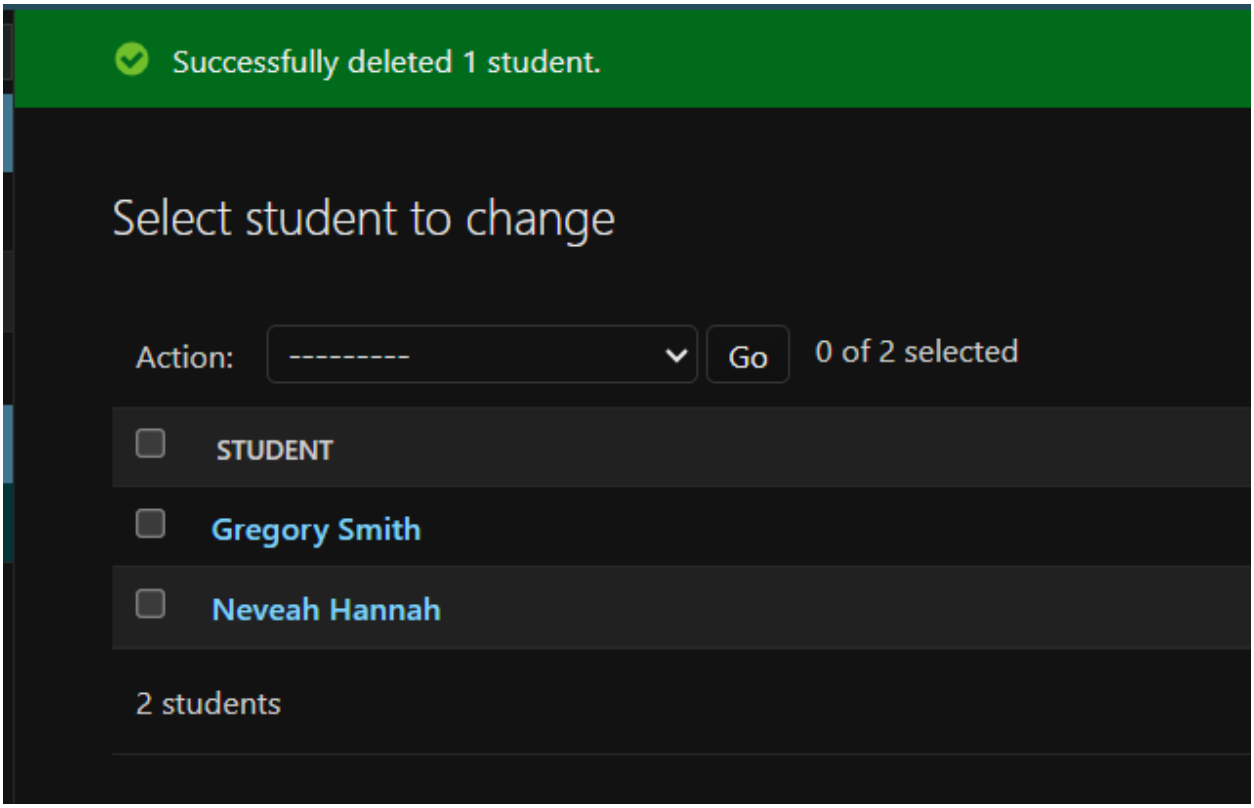


Figure 16: Successful D(delete) operation

TASK 3

STEP 1: Install PostgreSQL Driver

In your VS Code terminal (make sure (env) is active):

```
pip install psycopg2-binary
```

```
(env) PS D:\sem6\awt\lab\exp10\helloworld_project> pip install psycopg2-binary
Collecting psycopg2-binary
  Downloading psycopg2_binary-2.9.10-cp312-cp312-win_amd64.whl.metadata (5.0 kB)
Downloading psycopg2_binary-2.9.10-cp312-cp312-win_amd64.whl (1.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.2/1.2 MB 185.1 kB/s eta 0:00:00
Installing collected packages: psycopg2-binary
Successfully installed psycopg2-binary-2.9.10

[notice] A new release of pip is available: 24.0 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Figure 17: Successful installation of PostgreSQL database adapter for Python

STEP 2: Create PostgreSQL Database

Run this in PowerShell:

```
psql -U postgres
```

Enter your PostgreSQL password.

```
(env) PS D:\sem6\awt\lab\exp10\helloworld_project> psql -U postgres
Password for user postgres:

psql (17.2)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

postgres=#
```

Figure 18: Postgres Terminal

Then:

```
CREATE DATABASE hellodb;
\q
```

Now you have a DB named `hellodb`.

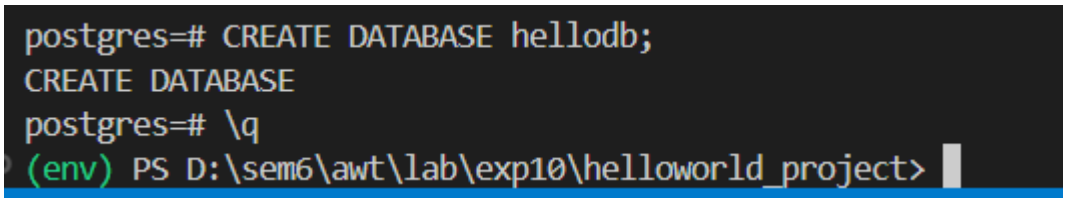


Figure 19: Quitting from Postgres Terminal

STEP 3: Update settings.py

In VS Code, open:

```
helloworld_project/settings.py
```

Scroll down to the `DATABASES` section and replace it like this:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'hellodb',           # DB name
        'USER': 'postgres',          # PostgreSQL username
        'PASSWORD': 'your_password', # replace with your actual password
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

Replace 'your_password' with the real password you use for PostgreSQL.

STEP 4: Apply Migrations to PostgreSQL

Still in terminal:

```
python manage.py migrate
```

This will create all the necessary tables in the new PostgreSQL DB (`hellodb`).

```
(env) PS D:\sem6\awt\lab\exp10\helloworld_project> python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, hello, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying hello.0001_initial... OK
  Applying sessions.0001_initial... OK
(env) PS D:\sem6\awt\lab\exp10\helloworld_project>
```

Figure 20

STEP 5: Test with CRUD

The app hello already has a model (`Student`), and I did CRUD via the Django admin in Task 2.

Note: I had to create a superuser again.

Let's reuse that:

1. Make sure `hello` is in INSTALLED_APPS
2. Run:

   ```
   python manage.py runserver
   ```

3. Go to: `http://127.0.0.1:8000/admin`
4. Login as superuser
5. Add, update, and delete Student records as usual

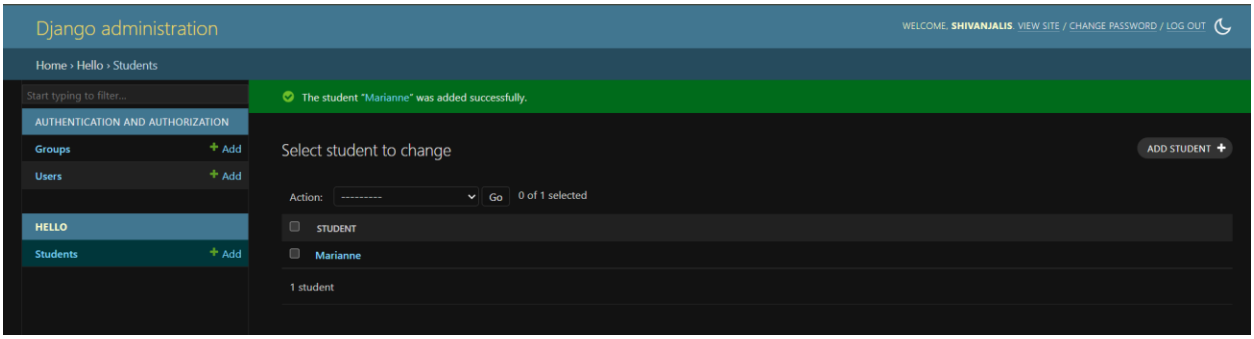
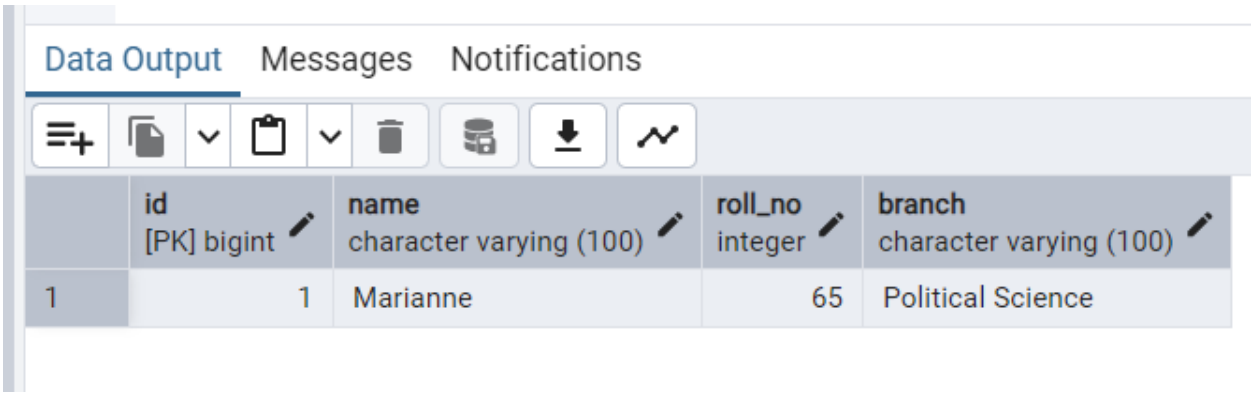


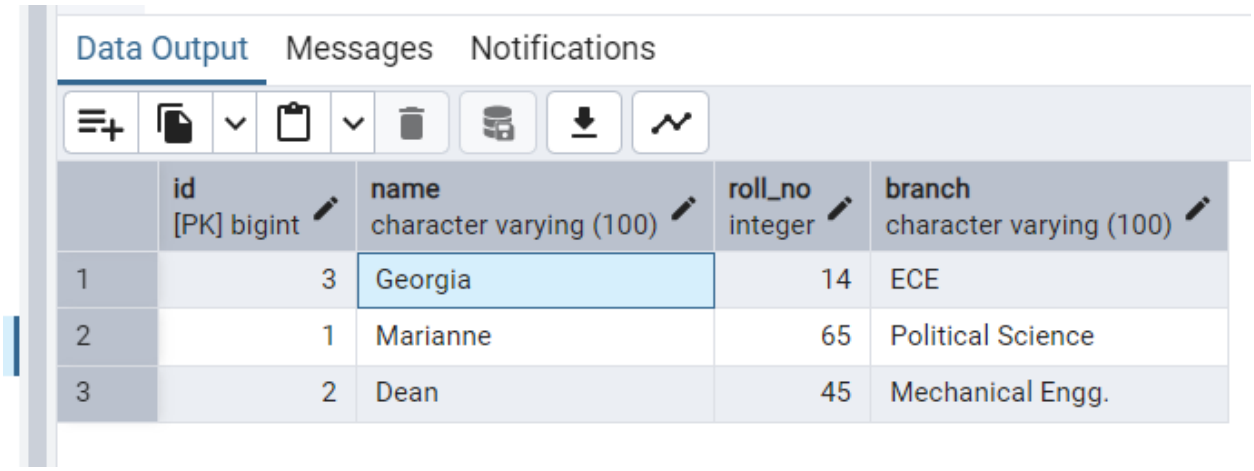
Figure 21: C(create)



Data Output Messages Notifications

	id [PK] bigint	name character varying (100)	roll_no integer	branch character varying (100)
1	1	Marianne	65	Political Science

Figure 22: Entries seen from pgAdmin -> Successful connection b/w Django & PostgresSQL



Data Output Messages Notifications

	id [PK] bigint	name character varying (100)	roll_no integer	branch character varying (100)
1	3	Georgia	14	ECE
2	1	Marianne	65	Political Science
3	2	Dean	45	Mechanical Engg.

Figure 23: Adding entries from pgAdmin

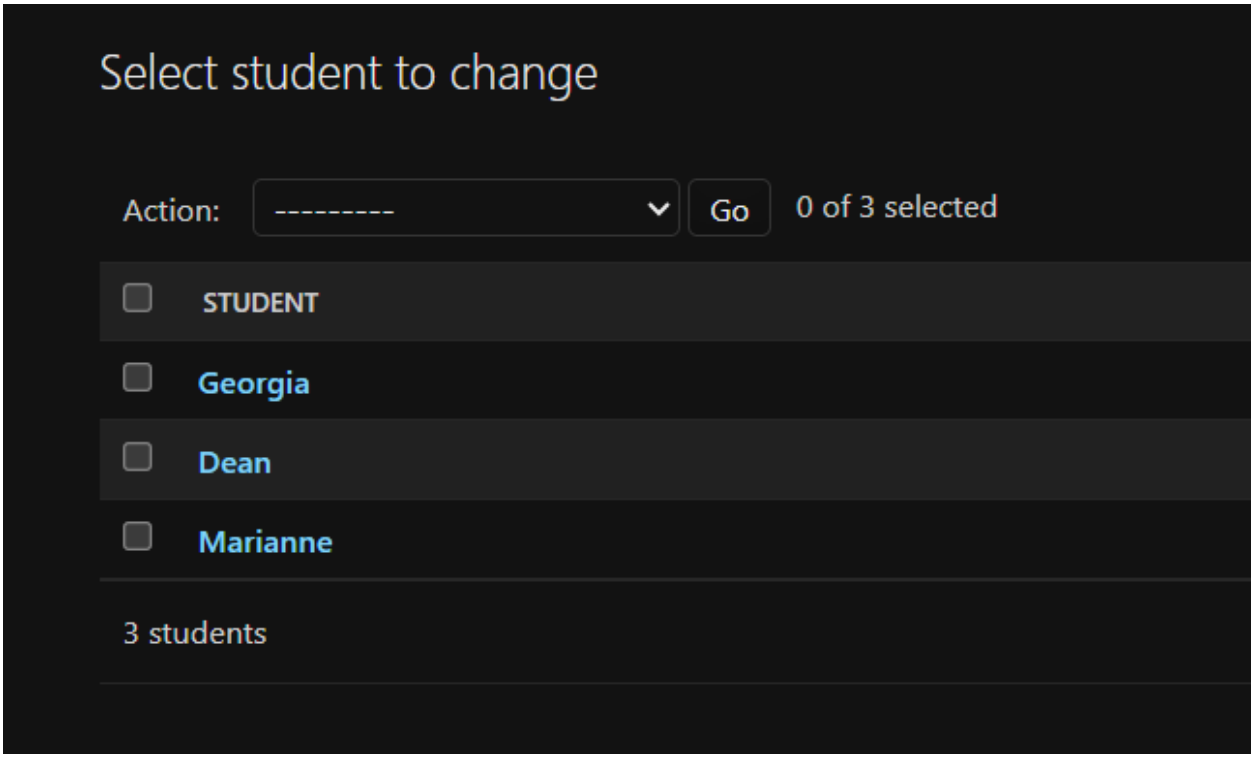


Figure 24: Entries seen from Admin dashboard -> Successful connection b/w Django & PostgresSQL

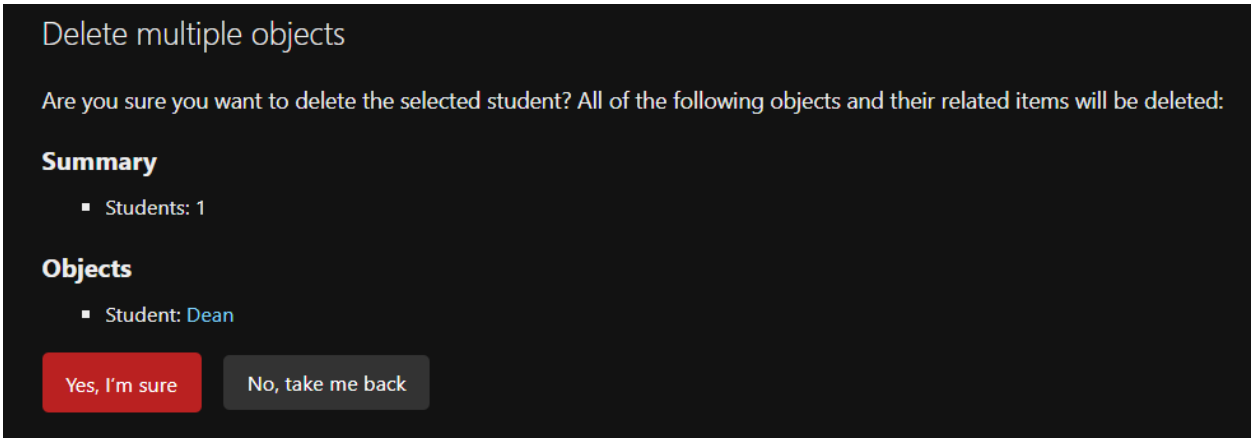


Figure 25: D(delete)

Data Output Messages Notifications

	id [PK] bigint	name character varying (100)	roll_no integer	branch character varying (100)
1	1	Marianne	65	Political Science
2	3	Georgia	14	ECE

Figure 26: Deleted entry removed from pgAdmin too