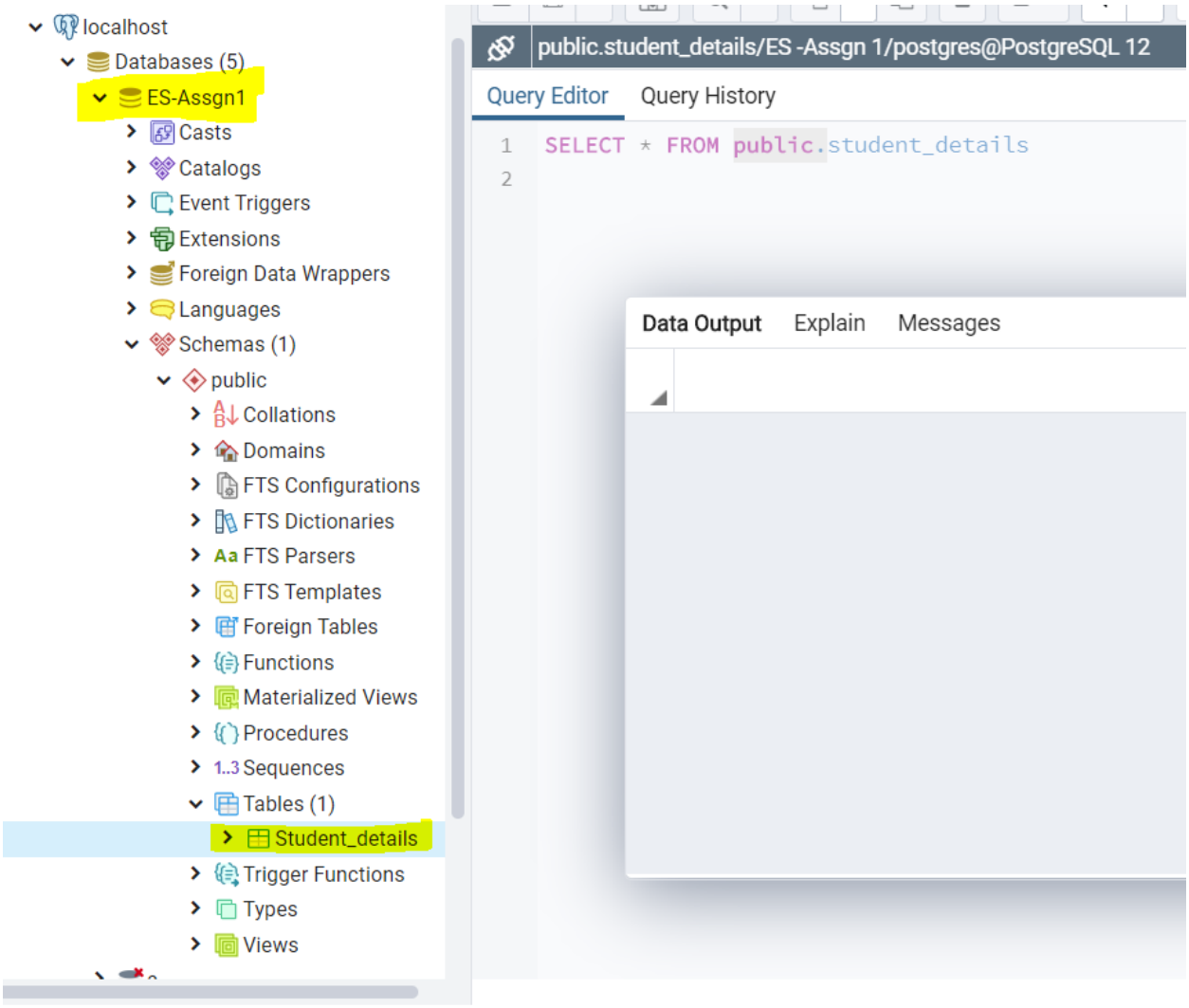
Artificial Intelligence in Enterprise Systems
Assignment #1– Flask

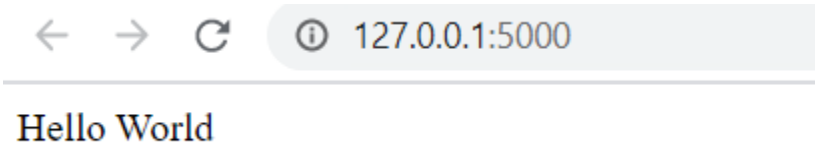
Creating Database in Postgre:

A database ES-Assgn1 and table Student_details



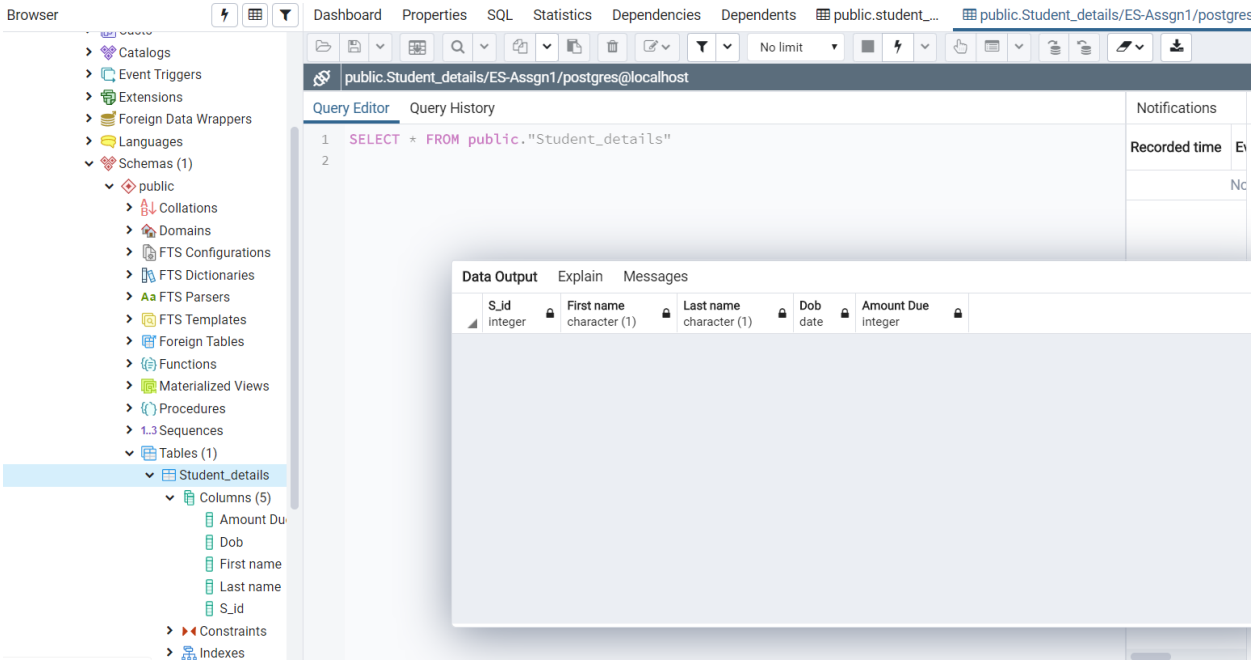
Creating Basic Flask App:

```python
from flask import Flask
app = Flask(__name__)

@app.route('/')


def hello_world():
    return "Hello World"

if __name__ == '__main__':
    app.run()
```



For Create

Columns can be seen before addition as seen below



For add

```python
@app.route('/add', methods=['POST'])
def add_user():
    try:
        json = request.json
        first_name = json['f_name']
        last_name = json['l_name']
        dob = json['dob']
        amount_due = json['due']

        # validate the received values
        if first_name and last_name and dob and amount_due and request.method == 'POST':

            # save edits
            sql = "INSERT INTO student_details(f_name, l_name, dob, due) VALUES(%s, %s, %s, %s)"
            data = (first_name, last_name, dob, amount_due)
            conn = create_conn()
            cursor = conn.cursor()
            cursor.execute(sql, data)
            conn.commit()
            resp = jsonify('User added successfully!')
            resp.status_code = 200
            return resp
        else:
            return not_found()

    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()
```

For update

```
@app.route('/update', methods=['POST'])
def update_user():
    try:
        json = request.json
        student_id = json['s_id']
        first_name = json['f_name']
        last_name = kson['l_name']
        dob = json['dob']
        amount_due = json['due']
        # validate the received values
        if _first_name and _last_name and _dob and _amount_due and _student_id and request.method == 'POST':

            # save edits
            sql = "UPDATE student_details SET f_name=%s, l_name=%s, dob=%s, due=%s WHERE student_id=%s"
            data = (_first_name, _last_name,_dob,_amount_due, _student_id)
            conn = create_conn()
            cursor = conn.cursor()
            cursor.execute(sql, data)
            conn.commit()
            resp = jsonify('User updated successfully!')
            resp.status_code = 200
            return resp
        else:
            return not_found()
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()
```

For delete

```
@app.route('/delete/<int:id>', methods=['DELETE'])
def delete_user(id):
    try:
        conn = create_conn()
        cursor = conn.cursor()
        cursor.execute("DELETE FROM student_details WHERE student_id=%s", (id,))
        conn.commit()
        resp = jsonify('User deleted successfully!')
        resp.status_code = 200
        return resp
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()

if __name__ == "__main__":
    app.run()
```

For viewing all files

```python
@app.route('/select', methods=['GET'])
def users():
    try:
        conn = create_conn()
        cursor = conn.cursor()
        cursor.execute("SELECT * FROM student_detials")
        rows = cursor.fetchall()
        resp = jsonify(rows)
        resp.status_code = 200
        return resp
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()
```

Git Repo:

https://github.com/SriManthra/ES-Assgn-1