

Student API

RESTful API using Flask

By Umutcan Asutlu and Ramon Vilarins

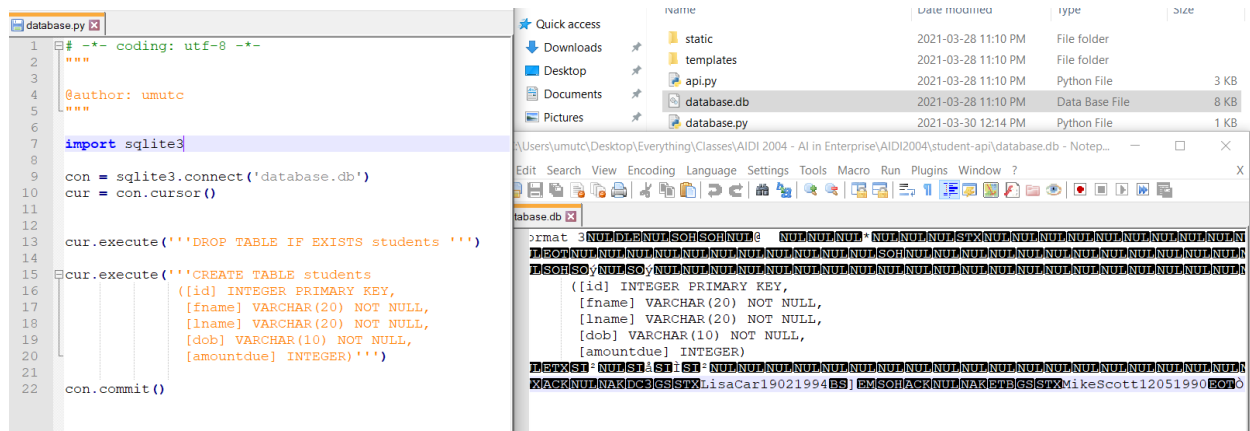
API Directory

We used python code for creating database and API. The flask API's front-end is done by using HTML5 and CSS3. It was supposed to be uploaded to Heroku so Procfile and requirements.txt are generated.

static	2021-03-28 11:10 PM	File folder
templates	2021-03-28 11:10 PM	File folder
api.py	2021-03-28 11:10 PM	Python File
database.db	2021-03-28 11:10 PM	Data Base File
database.py	2021-03-28 11:10 PM	Python File
Division of Workload.docx	2021-03-30 12:06 PM	Microsoft Word D...
Heroku.docx	2021-03-28 11:10 PM	Microsoft Word D...
Procfile	2021-03-28 11:10 PM	File
Report.docx	2021-03-30 12:10 PM	Microsoft Word D...
requirements.txt	2021-03-28 11:10 PM	Text Document
settings.py	2021-03-28 11:10 PM	Python File
students.py	2021-03-28 11:10 PM	Python File

Database

Database is created using Python code and with SQL Alchemy it is connected to API.



API

API has different pages for each CRUD operations. These operations are applied on student object.

```
from students import *

@app.route("/")
def welcomepage():
    return render_template("welcome.html")

@app.route('/allstudents', methods=['GET'])
def getallstudents():
    '''Function to get all the movies in the database'''
    return jsonify({'Students': Student.get_all_students()})

@app.route('/getstudent', methods=['GET', 'POST'])
def get_student():
    return render_template("getstudent.html")

@app.route('/student', methods=['GET', 'POST'])
def getstudent():
    id = request.args.get('id')
    student = Student.get_student(id)
    flash(student[0])
    return render_template("getstudent.html")

@app.route('/newstudent', methods=['GET', 'POST'])
def new_student():
    return render_template("newstudent.html")

@app.route('/studentadded', methods=['GET', 'POST'])
def add_student():
    id = request.args.get('id')
    fname = request.args.get('fname')
    lname = request.args.get('lname')
    dob = request.args.get('dob')
    amountdue = request.args.get('amountdue')
    Student.add_student(id, fname, lname, dob, amountdue)
    flash('Student is added successfully.')
    return render_template("newstudent.html")

@app.route('/updatestudent', methods=['GET', 'POST'])
def update_student():
    id = request.args.get('id')
    fname = request.args.get('fname')
    lname = request.args.get('lname')
    dob = request.args.get('dob')
    amountdue = request.args.get('amountdue')
    Student.update_student(id, fname, lname, dob, amountdue)
    flash('Student is updated successfully.')
    return render_template("newstudent.html")

def delete_student(id):
    Student.delete_student(id)
```

Usage

We tried to deploy it to Heroku but for unknown reasons the app kept crashing even though the deployment was successful.

```
C:\Users\umut\Desktop\student-api\heroku git:remote -a student-api-aidi2004
set git remote heroku to https://git.heroku.com/student-api-aidi2004.git

C:\Users\umut\Desktop\student-api>git push heroku master
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 12 threads
Compressing objects: 100% (20/20), done.
Writing objects: 100% (23/23), 4.74 KiB | 1.58 MiB/s, done.
Total 23 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Building on the Heroku-20 stack
remote: -----> Determining which buildpack to use for this app
remote: -----> Python app detected
remote: -----> Installing python-3.6.13
remote: -----> Installing pip 20.1.1, setuptools 47.1.1 and wheel 0.34.2
remote: -----> Installing SQLite3
remote: -----> Installing requirements with pip
remote: Collecting gunicorn==20.1.0
remote:   Downloading gunicorn-20.1.0.tar.gz (370 kB)
remote:   Building wheels for collected packages: gunicorn
remote:   Building wheel for gunicorn (setup.py): started
remote:   Building wheel for gunicorn (setup.py): finished with status 'done'
remote:   Created wheel for gunicorn: filename=gunicorn-20.1.0-py3-none-any.whl size=78918 sha256=b932ca11013f5b942a8a3f209cca2fa5b12e7e7ebee12d052a36ec946c010da
remote:   Stored in directory: /tmp/pip-ephem-wheel-cache-47h6zway/wheels/9a/86/37/cad4bc71746b420e17c4eb6f5c41cf7b5653c1fddbda27d198
remote: Successfully built gunicorn
remote: Installing collected packages: gunicorn
remote: Successfully installed gunicorn-20.1.0
remote: -----> Discovering process types
remote: Procfile declares types -> heroku, web
remote:
remote: -----> Compressing...
remote: Done: 47.2M
remote: -----> Launching...
remote: Released v3
remote: https://student-api-aidi2004.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/student-api-aidi2004.git
 * [new branch] master -> master

C:\Users\umut\Desktop\student-api>
```

Overview Resources Deploy Metrics Activity Access Settings

Installed add-ons \$0.00/month

Configure Add-ons

There are no add-ons for this app
You can add add-ons to this app and they will show here. [Learn more](#)

Dyno formation \$0.00/month

Configure Dynos

This app is using free dynos

web	gunicorn src.api:app --log-file -	ON
heroku	ps:scale web=1	OFF

Latest activity

All Activity

uasutlu@dcmail.ca: Deployed 7816a9a5
Today at 8:00 PM · v7

uasutlu@dcmail.ca: Build succeeded
Today at 8:00 PM · [View build log](#)

uasutlu@dcmail.ca: Deployed 701490c0
Today at 7:57 PM · v6

uasutlu@dcmail.ca: Build succeeded
Today at 7:57 PM · [View build log](#)

Application error

An error occurred in the application and your page could not be served. If you are the application owner, [check your logs for details](#).
You can do this from the Heroku CLI with the command

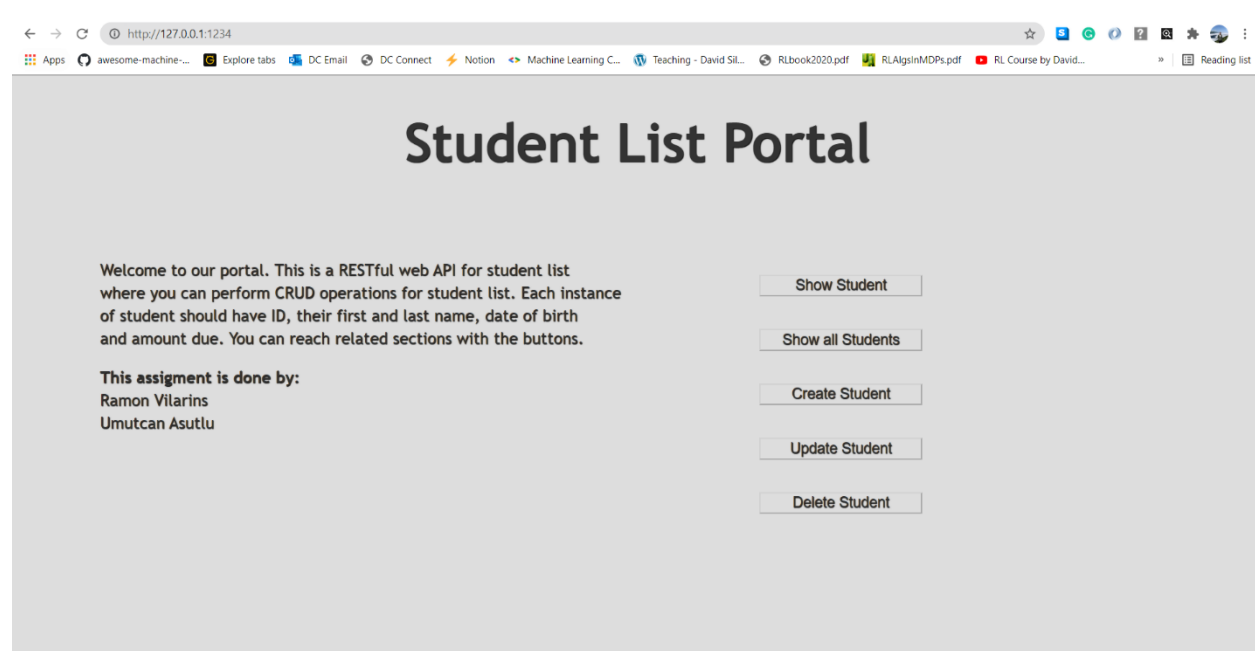
```
heroku logs --tail
```

```
2021-03-28T23:42:56.077872+00:00 app[web.1]:
2021-03-28T23:42:56.077872+00:00 app[web.1]: During handling of the above exception, another exception occurred:
2021-03-28T23:42:56.077872+00:00 app[web.1]:
2021-03-28T23:42:56.077874+00:00 app[web.1]: Traceback (most recent call last):
2021-03-28T23:42:56.077891+00:00 app[web.1]:   File "/app/.heroku/python/bin/gunicorn", line 8, in <module>
2021-03-28T23:42:56.078000+00:00 app[web.1]:     sys.exit(run())
2021-03-28T23:42:56.078002+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/app/wsgilapp.py", line 67, in run
2021-03-28T23:42:56.078125+00:00 app[web.1]:     WSGIApplication("%(prog)s [OPTIONS] [APP_MODULE]").run()
2021-03-28T23:42:56.078125+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/app/base.py", line 231, in run
2021-03-28T23:42:56.078284+00:00 app[web.1]:     super().run()
2021-03-28T23:42:56.078286+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/app/base.py", line 72, in run
2021-03-28T23:42:56.078448+00:00 app[web.1]:     Arbiter(self).run()
2021-03-28T23:42:56.078454+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/arbiter.py", line 229, in run
2021-03-28T23:42:56.078628+00:00 app[web.1]:     self.halt(reason=inst.reason, exit_status=inst.exit_status)
2021-03-28T23:42:56.078633+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/arbiter.py", line 342, in halt
2021-03-28T23:42:56.078862+00:00 app[web.1]:     self.stop()
2021-03-28T23:42:56.078866+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/arbiter.py", line 382, in stop
2021-03-28T23:42:56.079080+00:00 app[web.1]:     sock.close(sockets(self.LISTENERS, unlink))
2021-03-28T23:42:56.079085+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/sock.py", line 210, in close_sockets
2021-03-28T23:42:56.079252+00:00 app[web.1]:     sock.close()
2021-03-28T23:42:56.079256+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/arbiter.py", line 242, in handle_chld
2021-03-28T23:42:56.079428+00:00 app[web.1]:     self.reap_workers()
2021-03-28T23:42:56.079432+00:00 app[web.1]:   File "/app/.heroku/python/lib/python3.6/site-packages/gunicorn/arbiter.py", line 525, in reap_workers
2021-03-28T23:42:56.079620+00:00 app[web.1]:     raise HaltServer(reason, self.WORKER_BOOT_ERROR)
2021-03-28T23:42:56.079835+00:00 app[web.1]:   gunicorn.errors.HaltServer: 
```

Local View

However, it is still possible to use the API on local computer.

```
(base) C:\Users\umutc\Desktop\Everything\Classes\AIDI 2004 - AI in Enterprise\AIDI2004\student-api>python api.py
* Serving Flask app "settings" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with windowsapi reloader
* Debugger is active!
* Debugger PIN: 453-800-125
* Running on http://127.0.0.1:1234/ (Press CTRL+C to quit)
```



Student List Portal

Welcome to our portal. This is a RESTful web API for student list where you can perform CRUD operations for student list. Each instance of student should have ID, their first and last name, date of birth and amount due. You can reach related sections with the buttons.

This assignment is done by:
Ramon Vilarins
Umutcan Asutlu

Show Student

Show all Students

Create Student

Update Student

Delete Student

Notes

It can show student by given ID, it can create, delete, or update any student. The API is fully connected to the database. ID is primary key, but it is defined by the user, it is not auto generated. For show, update and delete the ID must be valid within the database.

To protect API logic and show it could have been used by services like POSTMAN and such, show all students page is prepared in JSON view.

Add new student!

Student is added succesfully.

Student ID:

Student first name:

Student last name:

Student date of birth (DDMMYYYY) :

Amount due:

Add Student

Go Back

Update student!

Enter the ID of the student you want to update:

Student ID:

Enter updated information:

Student first name:

Student last name:

Student date of birth (DDMMYYYY) :

Amount due:

Update Student



```
{
  - Students: [
    - {
      amountdue: 1234,
      dob: "12051990",
      fname: "Mike",
      id: 1,
      lname: "Scott"
    },
    - {
      amountdue: 2141,
      dob: "19021994",
      fname: "Lisa",
      id: 2,
      lname: "Car"
    },
    - {
      amountdue: 3345,
      dob: "03091990",
      fname: "Frank",
      id: 3,
      lname: "Dan"
    },
    - {
      amountdue: 12345,
      dob: "01011980",
      fname: "Ramon",
      id: 1212,
      lname: "Umutcan"
    }
  ]
}
```

Division of Workload

Task	Name
API Architecture	Ramon
API Functions	Umutcan
Flask CRUD Operations	Ramon
Flask Front-End (HTML/CSS)	Umutcan

Database	Umutcan
----------	---------