

# COMP0034 Coursework 1

In this coursework, I created a RESTAPI that uses Flask and the Estates Management Record from HESA (see references). To run the app:

1. Fork this repository: <https://github.com/ucl-comp0035/comp0034-cw1i-4jjnaomi>
2. Clone the resulting repository locally and to your IDE
3. Create and activate a virtual environment
4. Install the requirements using `pip install -r requirements.txt`
5. Run the app `flask --app src run --debug`
6. Open a browser and go to <http://127.0.0.1:5000/>
7. Go to the various URLs outlined in List of URLs
8. Stop the app using CTRL+C

## Application code

The code that creates the REST API is found in the src directory of the project. This Flask app can be run in development mode from the command line of a terminal. The URLs for each route can be seen in Table 1.

### List of URLs

*Table 1 Available urls for flask app and explantion of the REST API methods associated*

Method	URL	Explanation	Example JSON input
N/A	/	Index page	N/A
GET	/hei?page=1 &per_page=10	Obtains all hei entries in the database. The 'page' and 'per_page' variables can be specified by the user	N/A
GET	/hei/<ukprn>	Obtains a specific hei using its UKPRN. /hei/10006840 will obtain the data for The University of Birmingham	N/A
POST	/hei	Adds a new HEI to the database	{ "UKPRN": "12222221", "he_name": "New Random University", "region": "Random Region", "lat": "", "lon": "" }
DELETE	/hei/<ukprn>	Deletes a HEI from database using the provided UKPRN in the URL. /hei/10006840 will delete the data for The University of Birmingham	N/A

PUT	/hei/<ukprn>	Updates an HEI if the ukprn exists or adds a new UKPRN if it doesn't already exist. Requires all the fields of the HEI model	{ "UKPRN": "12222222", "he_name": "New Random University", "region": "Random Region", "lat": "", "lon": "" }
PATCH	/hei/<ukprn>	Updates an HEI that already exist in the database. Doesn't require all fields of the model.	{ "lat": "111", "lon": "111" }
GET	/entry?page=1&per_page=10	Obtains all entries in the database. The 'page' and 'per_page' variables can be specified by the user	N/A
GET	/entry/<id1>	Obtains a specific entry using its id /entry/1 will obtain the data for the first entry	N/A
POST	/entry	Adds a new entry to the database	{ "entry_id": "100000", "academic_year": "20/20", "classification": "dummy",  "category_marker": "dummy", "category": "dummy", "value": "70", "UKPRN": "111111", "he_name": "University of Naomi" }
DELETE	/entry/id1	Deletes an entry from database using the provided id in the URL. /entry/1 will delete the data for the first entry	N/A
PUT	/entry/id1	Updates an entry if the id exists or adds a new id if it doesn't already exist. Requires all the fields of the entry model	{ "entry_id": "100001", "academic_year": "20/20", "classification": "dummy",  "category_marker": "dummy", }

			<pre> "category": "dummy", "value": "70", "UKPRN": "111111", "he_name": "University of Naomi" } </pre>
PATCH	/entry/id1	Updates an entry that already exists in the database. Doesn't require all fields of the model.	<pre> {   "value":   "100000", } </pre>

## Pagination

As can be seen in the table above, the HEI and Entry GET routes have pagination. This is because both of these data tables have more than 100 records. This therefore gives the user the ability to customise the view obtained in the app.

## Test code

### Evidence of tests

As shown in the Figure 1, there were 34 tests and they all passed. The tests can be seen in the tests directory. The tests use parameterisation to access multiple route endpoints. The tests also use patching to mock certain database operations and simulate exceptions for error handling scenarios.

### Coverage

Figure 2 shows the output from the pytest coverage. This shows that I achieved 86% coverage which is good for an app with this amount of code. I looked at the areas where I was missing tests, and this majorly came from the exception block for Marshmallow ValidationErrors. I attempted to write a test this that mocked the ValidationError but obtained the following error: `sqlite3.OperationalError: database table is locked`. ChatGPT told me that this error often arises when using SQLite in testing environments where multiple tests are running concurrently, attempting to access the same database file. To address the error, I could have used an in-memory SQLite database for testing which would've avoided file

```

1 ▶ Run python -m pytest -v --cov=src --cov-report term-missing
2 ===== test session starts =====
3 platform linux -- Python 3.10.13, pytest-8.0.0, pluggy-1.4.0 -- /opt/hostedtoolcache/Python/3.10.13/x64/bin/python
4 cachedir: .pytest_cache
5 rootdir: /home/runner/work/comp0834-cw11-4jjnaomi/comp0834-cw11-4jjnaomi
6 configfile: pyproject.toml
7 testpaths: tests
8 plugins: cov-4.1.0
9 collecting ... collected 34 items
10
11 tests/test_entry_routes.py::test_get_entries PASSED [ 2%]
12 tests/test_entry_routes.py::test_get_entry_endpoints[/entry/755-200-expected_data0] PASSED [ 5%]
13 tests/test_entry_routes.py::test_get_entry_endpoints[/entry/999999-404-expected_data1] PASSED [ 8%]
14 tests/test_entry_routes.py::test_add_entry PASSED [ 11%]
15 tests/test_entry_routes.py::test_delete_entry PASSED [ 14%]
16 tests/test_entry_routes.py::test_delete_nonexistent_entry PASSED [ 17%]
17 tests/test_entry_routes.py::test_patch_entry PASSED [ 20%]
18 tests/test_entry_routes.py::test_patch_nonexistent_entry PASSED [ 23%]
19 tests/test_entry_routes.py::test_put_update_entry PASSED [ 26%]
20 tests/test_entry_routes.py::test_put_new_entry PASSED [ 29%]
21 tests/test_entry_routes.py::test_get_entry_exception PASSED [ 32%]
22 tests/test_entry_routes.py::test_post_entry_exception PASSED [ 35%]
23 tests/test_entry_routes.py::test_patch_entry_exception PASSED [ 38%]
24 tests/test_general.py::test_index_page PASSED [ 41%]
25 tests/test_general.py::test_404_page PASSED [ 44%]
26 tests/test_general.py::test_set_password PASSED [ 47%]
27 tests/test_general.py::test_check_password PASSED [ 50%]
28 tests/test_general.py::test_check_password_false PASSED [ 52%]
29 tests/test_hei_routes.py::test_get_hei_contains_ukprn PASSED [ 55%]
30 tests/test_hei_routes.py::test_get_hei_endpoints[/hei-200-None] PASSED [ 58%]
31 tests/test_hei_routes.py::test_get_hei_endpoints[/hei/10007788-200-expected_data1] PASSED [ 61%]
32 tests/test_hei_routes.py::test_get_hei_endpoints[/hei/12345678-404-expected_data2] PASSED [ 64%]
33 tests/test_hei_routes.py::test_post_hei PASSED [ 67%]
34 tests/test_hei_routes.py::test_post_hei_invalid PASSED [ 70%]
35 tests/test_hei_routes.py::test_delete_hei PASSED [ 73%]
36 tests/test_hei_routes.py::test_delete_hei_nonexistent PASSED [ 76%]
37 tests/test_hei_routes.py::test_patch_hei PASSED [ 79%]
38 tests/test_hei_routes.py::test_patch_hei_nonexistent PASSED [ 82%]
39 tests/test_hei_routes.py::test_patch_hei_invalid PASSED [ 85%]
40 tests/test_hei_routes.py::test_put_new_hei PASSED [ 88%]
41 tests/test_hei_routes.py::test_put_existing_hei PASSED [ 91%]
42 tests/test_hei_routes.py::test_get_hei_exception PASSED [ 94%]
43 tests/test_hei_routes.py::test_post_hei_exception PASSED [ 97%]
44 tests/test_hei_routes.py::test_patch_hei_exception PASSED [100%]

```

Figure 1 Evidence of tests being run for REST API app

locking issues. However, this would require a lot more time and knowledge than what is needed for this course.

```
51 ----- coverage: platform linux, python 3.10.13-final-0 -----
52 Name                        Stmts   Miss Branch BrPart  Cover   Missing
53 -----
54 src/_init_.py                61      5    18      4    89%   59->71, 72->exit, 103, 109-110, 142-143
55 src/controllers.py           194     16    20      4    91%   66-71, 186->190, 196->204, 250-253, 304-307, 356->360, 365->372, 368-371
56 src/error_handlers.py         21     21      2      0     0%   4-74
57 src/models.py                 43      0      0      0   100%
58 src/schemas.py               16      0      0      0   100%
59 -----
60 TOTAL                        335     42    40      8    86%
61
62
63 ===== 34 passed in 14.72s =====
```

Figure 2 Coverage report for tests run for REST API app

The coverage report also showed that I was missing tests for my `add_data_from_csv` function in the directory initialiser. Adding tests for this would be difficult as I would need to overwrite my csv files which would cause issues when running the app while not in testing mode.

## Tools and techniques

**Github repository:** <https://github.com/ucl-comp0035/comp0034-cw1i-4jinaomi>

### Continuous integration

This development of this app used a GitHub Actions workflow so that testing was done whenever changes were made to the repository. This meant that I was able to fix issues quickly. Evidence of the workflow being used is shown in Figure 3.

### Linting

In writing the code, a linter was used within Visual Studio Code to ensure good code quality through the identification and reporting of potential errors. Figure 4 shows some of the issues highlighted by pylint with my code.

```
src/_init_.py:12:1: Unnecessary pass statement (W0107) [Unnecessary-pass] (Ln 28, Col 5)
src/_init_.py:62:1: Using open without explicitly specifying an encoding (W1514) [Unspecified-encoding] (Ln 62, Col 14)
src/_init_.py:75:1: Using open without explicitly specifying an encoding (W1514) [Unspecified-encoding] (Ln 75, Col 14)
src/_init_.py:118:1: Unused HEI imported from src.models (W0611) [Unused-import] (Ln 118, Col 5)
src/_init_.py:118:1: Unused Entry imported from src.models (W0611) [Unused-import] (Ln 118, Col 5)
src/_init_.py:118:1: Unused User imported from src.models (W0611) [Unused-import] (Ln 118, Col 5)
src/_init_.py:118:1: Unused SavedChart imported from src.models (W0611) [Unused-import] (Ln 118, Col 5)
src/_init_.py:125:1: Unused controllers imported from src (W0611) [Unused-import] (Ln 125, Col 9)
src/_init_.py:156:1: Import outside toplevel (src.models.HEI, src.models.Entry) (C0415) [Import-outside-toplevel] (Ln 56, Col 5)
src/_init_.py:180:1: Line too long (114/100) (C0301) [Line-too-long] (Ln 80, Col 1)
src/_init_.py:180:1: Import outside toplevel (src.models.HEI, src.models.Entry, src.models.User, src.models.SavedChart) (C0415) [Import-outside-toplevel] (Ln 118, Col 5)
src/_init_.py:180:1: Import outside toplevel (src.controllers) (C0415) [Import-outside-toplevel] (Ln 125, Col 9)
src/controllers.py:160:1: Redefining name 'entry_update' from outer scope (line 160) (W0621) [Redefined-outer-name] (Ln 195, Col 13)
src/controllers.py:334:1: Redefining name 'entry_update' from outer scope (line 334) (W0621) [Redefined-outer-name] (Ln 364, Col 13)
src/controllers.py:3:1: Line too long (107/100) (C0301) [Line-too-long] (Ln 3, Col 1)
src/controllers.py:40:1: Line too long (115/100) (C0301) [Line-too-long] (Ln 40, Col 1)
src/controllers.py:50:1: Line too long (102/100) (C0301) [Line-too-long] (Ln 50, Col 1)
src/controllers.py:114:1: Line too long (105/100) (C0301) [Line-too-long] (Ln 114, Col 1)
src/controllers.py:162:1: Line too long (102/100) (C0301) [Line-too-long] (Ln 162, Col 1)
src/controllers.py:233:1: Line too long (105/100) (C0301) [Line-too-long] (Ln 233, Col 1)
src/test_entry_routes.py:183:1: Unused argument 'mock_execute' (W0613) [Unused-argument] (Ln 183, Col 30)
src/test_entry_routes.py:196:1: Unused argument 'mock_add' (W0613) [Unused-argument] (Ln 196, Col 31)
src/test_entry_routes.py:220:1: Unused argument 'mock_merge' (W0613) [Unused-argument] (Ln 220, Col 32)
src/test_entry_routes.py:27:1: Line too long (103/100) (C0301) [Line-too-long] (Ln 27, Col 1)
src/test_entry_routes.py:192:1: Line too long (101/100) (C0301) [Line-too-long] (Ln 192, Col 1)
src/test_entry_routes.py:216:1: Line too long (101/100) (C0301) [Line-too-long] (Ln 216, Col 1)
src/test_entry_routes.py:240:1: Line too long (101/100) (C0301) [Line-too-long] (Ln 240, Col 1)
src/test_hei_routes.py:208:1: Unused argument 'mock_execute' (W0613) [Unused-argument] (Ln 208, Col 28)
src/test_hei_routes.py:222:1: Unused argument 'mock_add' (W0613) [Unused-argument] (Ln 222, Col 29)
src/test_hei_routes.py:242:1: Unused argument 'mock_merge' (W0613) [Unused-argument] (Ln 242, Col 30)
src/schemas.py:54:1: Line too long (108/100) (C0301) [Line-too-long] (Ln 54, Col 1)
src/schemas.py:55:1: Line too long (110/100) (C0301) [Line-too-long] (Ln 55, Col 1)
```

Figure 4 Evidence of pylint output in VSCode

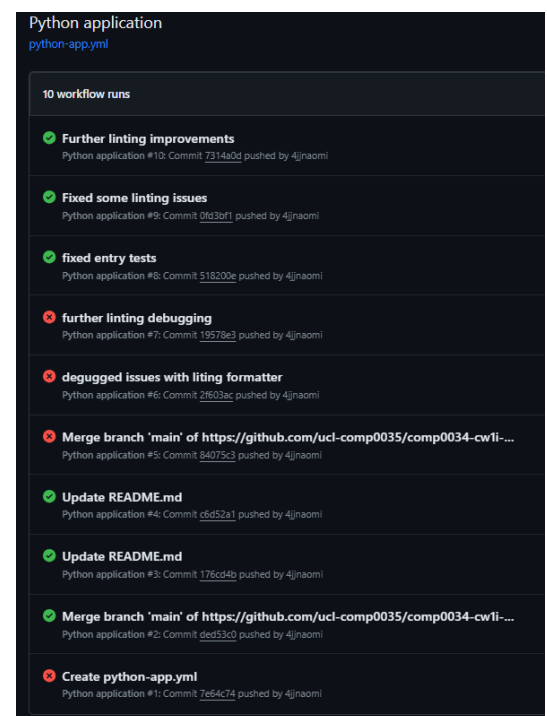


Figure 3 Evidence of GitHub Actions workflow being used

Although pylint in VSCode is useful, some of the errors highlighted are irrelevant. For example, the ‘unnecessary pass statement’ problem was ignored because the use of pass in the block of code stated is required for the logic of the app and to ensure the app didn’t break.

Therefore, only the issues highlighted by flake8 which is integrated in the GitHub Actions workflow was deemed as relevant. In an early commit of the code, there were many code quality issues as shown in Figure 6. Because some of these issues would be very tedious to manually correct (seeing as they were mostly whitespace), autopep8 was downloaded to format the code in VSCode. This was able to fix most of the code quality issues but there were still some that had to be fixed manually such as lines being too long. The final linter results is shown in Figure 5. There are still some code quality issues in the final submission such as my hei\_update and entry\_update functions being too long. However, due to the nature of a Flask app and a route decorator, it is not possible for me to refactor this function into smaller functions as that would not allow for correct route definition.

```
1 ▶ Run # stop the build if there are Python syntax errors or undefined names
0 0
11 ./src/_init_.py:118:5: F401 'src.models.HEI' imported but unused
12 ./src/_init_.py:118:5: F401 'src.models.Entry' imported but unused
13 ./src/_init_.py:118:5: F401 'src.models.User' imported but unused
14 ./src/_init_.py:118:5: F401 'src.models.SavedChart' imported but unused
15 ./src/_init_.py:125:9: F401 'src.controllers' imported but unused
16 ./src/controllers.py:160:1: C901 'hei_update' is too complex (12)
17 ./src/controllers.py:334:1: C901 'entry_update' is too complex (12)
18 2 C901 'hei_update' is too complex (12)
19 5 F401 'src.models.HEI' imported but unused
```

Figure 5 Code quality issues according to flake8 linter in GitHub actions workflow in final commit

```
12 ./src/_init_.py:14:1: E305 expected 2 blank lines after class or function definition,
13 ./src/_init_.py:18:1: E302 expected 2 blank lines, found 1
14 ./src/_init_.py:40:128: E501 line too long (176 > 127 characters)
15 ./src/_init_.py:44:1: E302 expected 2 blank lines, found 1
16 ./src/_init_.py:47:30: W291 trailing whitespace
17 ./src/_init_.py:51:33: E251 unexpected spaces around keyword / parameter equals
18 ./src/_init_.py:51:109: W291 trailing whitespace
19 ./src/_init_.py:68:1: W293 blank line contains whitespace
20 ./src/_init_.py:73:5: F401 'src.models.HEI' imported but unused
21 ./src/_init_.py:73:5: F401 'src.models.Entry' imported but unused
22 ./src/_init_.py:73:5: F401 'src.models.User' imported but unused
23 ./src/_init_.py:73:5: F401 'src.models.SavedChart' imported but unused
24 ./src/_init_.py:80:9: F401 'src.controllers' imported but unused
25 ./src/_init_.py:84:1: E302 expected 2 blank lines, found 1
26 ./src/_init_.py:97:1: W293 blank line contains whitespace
27 ./src/controllers.py:21:1: E265 block comment should start with '#'
28 ./src/controllers.py:23:1: E302 expected 2 blank lines, found 1
29 ./src/controllers.py:25:5: E265 block comment should start with '#'
30 ./src/controllers.py:46:1: E302 expected 2 blank lines, found 1
31 ./src/controllers.py:56:1: E302 expected 2 blank lines, found 1
32 ./src/controllers.py:74:1: E302 expected 2 blank lines, found 1
33 ./src/controllers.py:86:1: E302 expected 2 blank lines, found 1
34 ./src/controllers.py:87:1: C901 'hei_update' is too complex (12)
35 ./src/controllers.py:88:1: W293 blank line contains whitespace
36 ./src/controllers.py:102:1: W293 blank line contains whitespace
37 ./src/controllers.py:105:1: W293 blank line contains whitespace
38 ./src/controllers.py:118:1: W293 blank line contains whitespace
39 ./src/controllers.py:129:1: W293 blank line contains whitespace
40 ./src/controllers.py:137:1: W293 blank line contains whitespace
41 ./src/controllers.py:139:1: E265 block comment should start with '#'
42 ./src/controllers.py:174:1: E302 expected 2 blank lines, found 1
43 ./src/controllers.py:192:1: E302 expected 2 blank lines, found 1
44 ./src/controllers.py:204:1: E302 expected 2 blank lines, found 1
45 ./src/controllers.py:205:1: C901 'entry_update' is too complex (12)
46 ./src/controllers.py:216:1: W293 blank line contains whitespace
47 ./src/controllers.py:219:1: W293 blank line contains whitespace
48 ./src/controllers.py:238:1: W293 blank line contains whitespace
49 ./src/controllers.py:242:1: W293 blank line contains whitespace
50 ./src/controllers.py:247:1: W293 blank line contains whitespace
51 ./src/error_handlers.py:1:1: F401 'flask.make_response' imported but unused
52 ./src/error_handlers.py:27:1: F811 redefinition of unused 'handle_exception' from line 9
53 ./src/error_handlers.py:60:1: E302 expected 2 blank lines, found 1
54 ./src/error_handlers.py:70:38: W292 no newline at end of file
55 ./src/models.py:2:1: F401 'sqlalchemy.Integer' imported but unused
```

Figure 6 Code quality issues according to flake 8 linter in GitHub Actions workflow in early commit

## References

Acknowledgement of the use of AI

In the development of the REST API, AI was used to generate, refactor and debug the code used for the application and the tests. Github Copilot v1.159.0 (Github, <https://github.com/features/copilot>) was downloaded as an extension in Visual Studio Code (the IDE I was using) and was able to automatically generate code based on the code already in my files and modules. Copilot was used to generate application code and test code as well as generate comments and doc strings for some functions. Copilot did not influence my code, the AI was influenced by the code I already had. ChatGPT-3.5 (Open AI, <https://chat.openai.com/>) was also used for debugging and refactoring of code. ChatGPT influenced the quality of my code as it was used to improve its adherence to DRY and good code design principles.

## Dataset

The original dataset has the following details:

Title: Estates management by academic year and HE provider  
Location: UK

Academic years: 2015/16 to 2021/22

Data source: HESA

Data file canonical link: <https://www.hesa.ac.uk/data-and-analysis/estates/data.csv>

Licence: Creative Commons Attribution 4.0 International Licence

The data used in the RESTAPI has been altered from the dataset above.