COMP0034 2023/24 Coursework 2 specification (Individual)

Version: 1 released 06/02/2024

Contents

1. Introduction

This document specifies coursework 2 which is worth 50% of the assessment marks available for the module.

In this coursework you will develop and test a application using Python Flask or Plotly Dash and your dataset from COMP0035.

Marks are allocated for:

1. Application code
2. Test code
3. Tools and techniques

2. Getting started

1. Create a GitHub repository using GitHub classroom:

    1. Login to GitHub.com.
    2. Click on the GitHub classroom link for individuals.
    3. Accept the assignment.
    4. If prompted, accept to join the comp0035-ucl organisation.

2. Make your data available in the project.

    You can use any of the following approaches to access the data in your app:

    - dataset .csv/.xlsx file
    - REST API from COMP0034 coursework 1
    - SQLite database

3. Technologies that must be used

1. GitHub for source code control.

Use GitHub, with repositories created in GitHub classroom to allow tutors and PGTAs to gain access. If you can't use GitHub for some reason please contact the course tutor to agree alternative source code control.

2. Python coding environment.

   Use a Python coding environment such as Visual Studio Code or PyCharm Professional. Jupyter notebooks is not accepted for the coursework.

3. Flask python library or Plotly Dash python open source.

   The course will teach Flask and Plotly Dash. You may use Streamlit instead of Dash however this will not be taught in the course. You **must not** use Django, React or any other web framework. You may use additional Python libraries to support your app code such as Flask-SQLAlchemy, Flask-WTF, requests, etc.

4. Use `.pdf` or `.md` format for written content.

   All written evidence must be combined into **a single file**, either PDF or markdown.

4. Coursework content

Application code

Write code to create a web app that makes use of your data.

The minimum for the type of app you are creating is:

- Creates a Flask or Dash app that can be run in development mode.
- The app must include page(s) whose content is generated dynamically (i.e. not static HTML).
  - Dashboards should have at least 2 charts.
  - ML apps should have at least 2 pages, one of which interacts with the ML model.
  - 'Other' apps should have at least 3 pages that interact with data.
- Interacts with the data using one of the following methods:
  - SQLite database using Flask-SQLAlchemy
  - REST API using Python requests
  - .csv/.xlsx using pandas DataFrame
- Responsive design using open source CSS template.

Note: Writing your own CSS will not be considered in the marking so is not recommended. JavaScript is not considered in the marking and should not be needed.

If you have code that is not working as desired, or wish to explain anything related to your app, please add a section called 'Application code' to your comp0034-coursework2.(pdf/.md). For errors/issues please describe:

- what the problem is or how it is affecting your app
- where in the code the issue is
- what steps you took to try and address the issue

Test code

Write integration tests for the app using pytest and selenium webdriver.

Add any supporting evidence (i.e. anything that is not code) to comp0034-coursework2. (pdf/.md).

The minimum required:

- Uses pytest and selenium webdriver (for Chrome)
- A test that verifies the home page URL can be accessed
- Two tests that carry out sequences of actions the user of your app would expect to perform

Use of tools and techniques

Add a section called 'Tools and techniques' to your comp0034-coursework2.(pdf/.md).

Provide evidence that demonstrates effective use of appropriate use of software engineering tools and techniques.

The minimum required:

- Source code control:
  - You must use GitHub for source code control unless otherwise agreed in advance with the course tutor.
  - Add URL to your GitHub repository the evidence.
  - Demonstrate use of source code control throughout the coursework.
  - Appropriate use of .gitignore
- Set-up instructions:
  - Add instructions to set up and run your app and your test code to comp0034-coursework2.(pdf/.md).
  - Provide a `pyproject.toml` (or alternative such as setup.py)
- Dependency management:
  - Provide `requirements.txt` file (or appropriate alternative).

References

Add a section called 'References' to your comp0034-coursework2.(pdf/md).

References should include:

- Acknowledgement of the use of AI
- Attribution for the data set
- Code references
- Books, papers, websites etc

See the Appendix - Referencing for details on how to reference each of the above.

5. Submission

Submit your work on Moodle as a single .zip in the assignment submission. Refer to Moodle for the deadline date and time.

Moodle is used as the submission date/time and to authenticate students. GitHub is not an accepted submission system.

The .zip should expand to the correct folder structure with all required files. Ensure that all files are included in the zip, files that are only provided as URLs will be excluded from marking consideration (since they may be modified after the coursework is submitted).

The submission must include:

- Application code

- Test code
- comp0034-coursework2.pdf or comp0034-coursework2.md. If using markdown make sure any linked files are also included in the submission.

6. Marking

Mark allocation and calculation

An module is expected to take around 150 learning hours.

Each coursework should take 20-25 hours.

While not exact, an indication of the effort given the marking weighting is shown below.

App code	60%	12
Test code	30%	6
Tools and techniques	10%	2

Grading criteria

The 'UCL Computer Science: Marking Criteria and Grade Descriptors' will be used to assess the coursework (using only the criteria that are relevant to this coursework). In addition to the standard descriptors, further guidance is given for this coursework in the table below. Given there an be a wid ange of solutions it isn't possible to cover everything you might do to chie a ive level. Th fol win should b d as guidance. Hi her mark ban s assume t t e cri a fro lo er b nds have bee fully ac eved.

App code

As a guide, marking considers aspects such as:

- are the minimum requirements met?
- range of skills/sophistication/challenge evidenced in the resulting application code
- code quality: Python standards, structure and configuration of the application, use of functions and/or classes, DRY and other design principles

Distinction 90+	Exceptional solution and advanced algorithm/technical design.	
Distinction 70-89	Excellent solution, novel and creative approach.	Evidence of challenge and code that goes beyond that covered in the taught materials. Makes good use of the technologies introduced to manipulate data. Excellent code quality maintained throughout; evidence the code meets principles of good design.
Merit 60-69	Good solution, skilled use of concepts, mostly correct and only minor faults.	Evidence of challenge beyond the minimum requirements. Consistently good code quality. App created and configured effectively. Good use of a range of techniques.
High pass 50-59	Reasonable solution, using basic required concepts, several flaws in implementation.	Meets the minimum requirements. Code quality issues. Little/no evidence of challenge beyond the core teaching.
Low pass 40-49	Fundamental technical solution, but mostly incomplete.	There is a working app though this may not meet the minimum requirements. Significant errors in the code and/or issues with code quality. Code that is largely a copy of the course materials with minimal change.

Test code

As a guide, marking considers aspects such as:

- did you meet the minimum requirements of this section?
- evidence of challenge and sophistication in the test code.
- the quality and structure of the test code. This includes code quality, test naming and structure, documentation, appropriate use of assertions, use of fixtures etc.

Distinction 90+	Exceptionally comprehensive testing, extremely thorough approach to testing.	Tests are extensive in their range and scope. There is greater evidence of sophistication in the testing. Exemplary use of supporting tools such as CI.
Distinction 70-89	Very well done test cases.	Tests are more extensive than the minimum AND test code quality is consistently excellent AND tests show a distinctive level of sophistication e.g. in the use of fixtures, parameterised tests, types of tests etc. Effective use of CI is also expected for this level.
Merit 60-69	Solid testing.	Test code quality is consistently good. Tests show some sophistication e.g. in the use of fixtures or the types of tests. Some use of CI may be expected at this level.
High pass 50-59	Basic testing done.	Minimum requirements met; though the test quality may be lower and the tests straightfoward.
Low pass 40-49	Few test cases, but weak execution.	Tests may be incomplete or have substantial issues.

Tools and techniques

a guide only, thinking consider aspects such as:

- Did you meet the minimum requirements of this section?
- Has source code control been used regularly and appropriately?
- Are the dependencies managed using an appropriate technique? Are all the dependencies included?
- Are set up instructions provided? Was the app able to be run after the set-up instructions were followed?

These are not covered by the generic CS grading criteria.

Distinction 90+	Near flawless use of tools and techniques; techniques used in a way that is beyond the course base materials.
Distinction 70-89	Exceptional use of a range of tools and techniques that goes beyond the minimum required.
Merit 60-69	Effective use of source code control e.g. timely use, appropriate messages. Setup includes pyproject.toml (or alternative). Setup instructions complete and clear.
High pass 50-59	Regular use of source code control. All dependencies appropriately included (e.g. in requirements.txt). Setup instructions provided.
Low pass 40-49	Limited use of source code control. Requirements.txt missing substantial number of dependencies. setup instructions may be missing or incomplete.

Appendices

1. Adding challenge

The following are suggestions of things you might do to increase the challenge of your solution. This is not an exhaustive list.

Application code

- A more challenging feature that demonstrates further skills/knowledge. This may be an additional feature though do not try to add many additional features as more features does not mean more marks! Focus on quality rather than quantity.
- (Dash) Use of callbacks and features that allow the user to tailor charts to their preferences
- Error handling

Test code

- Creat[illegible]use of fixt[illegible]es
- [illegible]on [illegible]ion [illegible]f e[illegible]or c[illegible]diti[illegible]ns, edg[illegible]as[illegible]

Tools and techniques

- Evidence of the configuration and use of GitHub Actions to run the tests
- Linting. There are also static and dynamic program analysis tools that do more than linting, you would need to research what these are and then apply them appropriately.
- Evidence that you used AI effectively to improve your code, rather than to just write the code for you e.g.
  - to generate alternatives that you considered
  - to improve the quality or efficiency of your code.

  To demonstrate this please include the prompts used (if applicable) and the code before/after AI.

Note: If you use other tools/techniques please provide evidence incorporated into your comp0034-coursework2.(pdf/md). For example, include screenshots or image formats in your file. A hyperlink to an external file is not acceptable for proof of submission.

2. Unresolved bugs and code issues

There may be problems with your code that you were not able to address in the timescales of the project. This is not unusual either for the coursework or in real life.

If this occurs, add the following to the comp0034-coursework1.(pdf/md):

- what the problem is or how it is affecting your app
- where in the code the issue is

- what steps you took to try and address the issue

3. Referencing

Acknowledgement of the use of AI

If you did not use AI then clearly state this. Otherwise, you must include the following details:

- Name and version of the generative AI system used; e.g. ChatGPT-3.5
- Publisher (company that made the AI system); e.g. OpenAI
- URL of the AI system.
- Brief description (single sentence) of context in which the tool was used.
- Statement of how the AI influenced your code.

This is based on UCL student guidance on the use of AI. |

Reference your dataset

Include any details to acknowledge your dataset (attribution) to comply with any license condition required for your data set (given in the data set link in COMP0035 Moodle).

Book, journal, web article etc. references

It is unlikely to you will have any but if you do then include them.

For book, journal, and web article references, follow the referencing style you use in your own department (e.g. Harvard, APA). Different departments in UCL use different styles.

Code references

How to cite code

UCL has no single method for referencing code. For the purposes of this coursework use the following.

- Give the author, URL and the date of retrieval. If you adapted the code, indicate "Adapted from" or "Based on".
- Include citations within your code using code comments close to the affected code.

An example:

```
def play_sound(button_text):
    # Adapted from code from 'remdog' on the Plotly Community Forum at
    # https://community.plotly.com/t/linking-scatter-plot-elements-to-audio-
files/11908/6
    # Accessed 01/02/21
    ...some code here...
```

When to cite code

You must cite external sources; including where you copy and then adapt the code. External sources include:

- user forums e.g. stack overflow
- open source repositories e.g. GitHub repos where an explicit open source license is stated

- communities associated with a library e.g. plotly forum

It is not appropriate to copy from other students on the course, past or present.

For this coursework you do not need to cite the following external sources:

- course teaching materials
- official library documentation and tutorials of the python libraries and frameworks

4. Support and guidance

The assessments start in week 1 (coursework 1) and week 5 (coursework 2) of the course. You are expected to make progress each week.

1. **Moodle activities (online)**: There are weekly activities and checkpoints in Moodle to guide your progress.
2. **Weekly lab/tutorial sessions (in person)**: Module staff will be available to provide support and guidance in weekly lab/tutorial sessions. Use these sessions to complete activities and then work on your coursework.
3. **Tutor office hours (online)**: The tutor will be available during a weekly module office hour (see Moodle).
4. **Moodle Q&A (online)**: You can ask questions at any time in the Moodle Q&A forum where anyone in the course is able to reply. When posting questions about your code, please post the URL to your repository. This enables the tutor and PGTAs to see your code which is usually essential for solving technical problems! Other students won't be able to access your repository from this URL so long as your repository is private and is within the ucl-comp0035 organisation.

5. Changes to the coursework submission date

Coursework submission dates are set centrally in Computer Science and not by the course tutor.

These have been planned carefully to fit with the teaching schedule.

Requests to change these for all students must be made to the course tutor who will then forward the request to the relevant authority in the computer science department. Requests must be supported with appropriate evidence of the need for change.

6. SoRA and EC

If you have an approved extension resulting from a SoRA or EC, these will be carefully checked by the teaching and learning team before marks are entered in portico.

However, the deadlines in Moodle may not be modified for individuals, so it may appear on Moodle that your submission is late even though you have an approved extension to cover the period.

7. Late submission penalties

Late submission rules apply to all assessments.

Any penalties for late submission are applied by the computer science teaching and learning team when marks are entered in portico.

The mark you see in the Moodle gradebook may not be your final mark as it is the mark before moderation or any penalty has been applied.

8. Data sets

Only use the ethics approved data sets.

The data sets allocated to students on this course have been approved by the Computer Science Ethics Committee and signed by the Head of Department for use in this course. These data sources comply with GDPR and UCL ethics and data protection policies. You must not use any other data set for the coursework.

The approved data sets do not require data protection registration and since you will not be working with participants in your project there is no requirement to complete any further training relating to data protection and GDPR.