Assessment Brief

Module title	Web Application Development
Module code	XJCO2011
Assignment title	Assignment 1
Assignment type and description	Programming Assignment You will develop a small web application using Flask which uses a database, forms and some basic Python programming to provide useful functionality to a user. You will also create a simple, accessible design for your website.
Rationale	You will develop your skills in database design, demonstrate your ability to use web forms to input valid data to a database, and combine this with your software development skills to create a cohesive program.
Word limit and guidance	You should spend around 10 hours working on this assignment.
Weighting	40%
Submission deadline	30 November 2025
Submission method	Via Gradescope
Feedback provision	You will receive a marked up rubric and some short comments on your code.
Learning outcomes assessed	LO1 - Use web technologies to implement a web application. LO2 - Improve knowledge of Python in regard to programming skills and web development LO3 - Apply database knowledge and consider architecture in the web application.
Module leader	Amy Brereton

1. Assignment guidance

You will create a simple budget tracker web application using Flask with a SQLAlchemy database.

Users should be able to:

- Add incomes and expenditures
- See a list of all incomes and a list of all expenditures
- Edit an income or expenditure
- See their total income and total expenditure
- Add a savings goal
- See their progress towards this goal
- Edit or delete their goal.

You **are not expected** to add a logic or authentication system.

You should ensure that you **validate** any data entered by the user, both client- and server-side where appropriate.

Your website should have a professional layout which is easy to navigate and complies with the WCAG standards for accessibility. You can use open source CSS libraries such as Bootstrap, but should not rely on them for all of your styling.

You must develop your layout yourself using HTML, CSS and JavaScript if appropriate. You cannot use any software or services to create your layout for you, the code should be written by you.

All JavaScript and CSS should be in its own file(s) in the 'static' folder.

2. Assessment tasks

Homepage

The homepage should have some basic analytics:

- The total value of all expenditures
- The total value of all incomes
- The difference between those two values
- If they have a goal, their progress towards this goal

If there are no incomes/expenditures then an appropriate message should be shown instead.

Incomes and Expenditures

You should create models to represent incomes and expenditures. You *can* do this with a single model, but it may make things more difficult for you- using two separate models is recommended.

They should have:

- A unique name
- An amount

For example, Income(name="June Salary", amount=530.21)

To create a new income or expenditure, the user should be directed to a form which takes their inputs, **validates them** to ensure that sensible values have been used, and adds this data to the database.

The user should be able to see a list of both incomes and expenditures, and should be able to edit or delete them.

Goals

The user should be able to create a single savings goal.

This should have:

- A name (optional)
- A value

Once the user has created their goal, they should be able to edit or delete it. They should be able to see their progress towards their current goal, based on the **difference between their total incomes/expenditures**.

Layout and Navigation

Your website layout should be **professional looking**, **accessible** and **simple to use**. You should ensure that you meet the WCAG standards which are the **legal minimum requirement** for accessibility on public-facing websites in the UK. These standards can be found in full on the w3.org website, and there is a quick reference guide available: https://www.w3.org/WAI/WCAG21/quickref/

If you choose to use any icons or images which are not creative commons, you **must** ensure that you reference these in your code (a comment with the source is acceptable).

3. General guidance and study support

You should use the module website, accessed via Minerva, as your primary source for information.

If you do wish to use any other websites, books or sources, this can introduce some issues as the way that files are set up and names may be different from the module notes.

4. Assessment criteria and marking process

Your submission will be marked based on:

- How much of the required functionality has been implemented
- The accessibility of your website
- The overall design and layout of your website
- The quality and readability of your code

All work will be manually marked against the rubrics given in Section 7 below, and your feedback will be in the form of a marked up rubric and a short comment. Feedback will be provided through Gradescope within 3 weeks of the final submission deadline.

5. Presentation and referencing

Any copyrighted material should be referenced by comment in your code.

6. Submission requirements

You should submit only the files required to run your website and **not your venv**.

Make sure that your site is ready-to-run- databases should be initialized but, if possible, empty of any test data, and all your files and folders should be submitted in a single zip file to allow me to run them without modification.

7. Academic misconduct and plagiarism

Leeds students are part of an academic community that shares ideas and develops new ones.

You need to learn how to work with others, how to interpret and present other people's ideas, and how to produce your own independent academic work. It is essential that you can distinguish between other people's work and your own, and correctly acknowledge other people's work.

All students new to the University are expected to complete an online Academic Integrity tutorial and test, and all Leeds students should ensure that they are aware of the principles of Academic integrity.

When you submit work for assessment it is expected that it will meet the University's academic integrity standards.

If you do not understand what these standards are, or how they apply to your work, then please ask the module teaching staff for further guidance.

By submitting this assignment you are confirming that the work is a true expression of your own work and ideas and that you have given credit to others where their work has contributed to yours.

8. Assessment/ marking criteria grid

Marks for Accessibility, Design & Code Quality (50%):

		Outstanding	Very good	Good	Poor
Accessibility (20%)	Alt Text	All images have good alt text descriptions. Any decorative images are correctly tagged for screen readers.	All images have alt text descriptions. Some use of decorative tag on images.	All images have some sort of alt text provided, although this may be insufficient.	No alt text, or alt text left blank.
	Use of colour	Appropriate colours have been chosen to ensure that the website is usable for colour blind users. Colour is not the sole way of conveying meaning to the user. There is a high level of contrast behind all text.	Appropriate colours have been chosen to ensure that the website is usable for colour blind users. There is generally a good level of contrast, although in places this may be lacking.	Some appropriate colours chosen, although there may be some issues with colour blind users. Poor contrast in places.	The site is not usable for people with colour blindness, or there is very poor contrast in many places in the site.
	Keyboard Navigation	The website can be easily navigated using the keyboard alone.	Most of the website can be navigated using keyboard alone.	Parts of the website can be navigated using keyboard alone.	Keyboard navigation has not been implemented.
	HTML	The site is written using valid HTML, with language and all elements such as form fields tagged correctly in the code to ensure that assistive technology can be used.	The site is generally valid, although there may be some places where fields or language have not been tagged.	There are a number of HTML errors, but there has been an effort to tag fields correctly.	Many serious HTML validation errors, and tags such as language missing.

Design & Layout (20%)	Layout & navigation	The layout is consistent and professional looking throughout the site, including an appropriate way of navigating the site.	The layout is consistent and reasonably professional. There is a consistent way of navigating the site in which the links do not move if you change page.	The layout is extremely basic, and lacks a consistent method of navigating the site.	There has been no attempt to implement a layout for the site. Navigation uses plain links.
	Design	The site looks good and there has clearly been a lot of thought and effort put into designing the site.	The site is suitable and some effort has been put into making it look professional.	There has been an effort to add some styling to the website.	The website has no styling used.
10%	Code Quality	All Python code meets PEP8 standard. There are comments provided when needed, and these are not excessive.	There are fewer than 5 PEP8 errors in the code. There are some sensible comments used in the code.	There are fewer than 20 PEP8 errors in the code. There are some comments used.	The code has many PEP8 errors and there has been no attempt to make code readable.

Marks for Functionality (50%):

Functionality	Marks
User can add incomes and expenditures	10
User can see all incomes and expenditures	5
User can edit/delete an income/expenditure	5
User can view their total income and total expenditure	5
User can add a new saving goal	5
User can view progress towards their goal	5
User can edit/delete a goal	5
Total	**40**

	Outstanding	Very good	Good	Poor
Validation (10)	User inputs are validated both client- and server-side to ensure no duplicate data is entered and invalid values are prevented.	User inputs have a good level of client-side validation to prevent invalid data being entered. Users are provided with helpful flash messages or prompts when invalid data is entered.	User inputs have some basic client-side validation such as presence checks. Users receive some sort of message when invalid data is entered.	There has been no attempt to validate data entered by the user.