Assessment Brief

Module title	Web Services and Web Data
Module code	COMP3011/XJCO3011
Assignment title	Professor Rating Service
Assignment type and description	This is a programming assignment. You will implement a RESTful web service for rating professors. You will host your service's website online. You will also write a client application for interacting with the service. Finally, you will write a brief report documenting your work.
Rationale	This coursework will help you develop your skills in: • Designing a relational database for a real-world problem. • Designing different aspects of a RESTful web service including making the right choices of API endpoints, HTTP message types, and message data. • Writing clean, efficient, reliable, and robust server-side code for a RESTful web service. • Writing clean, efficient, reliable, and robust client code for interacting with a web service. • Using open-source web development platforms such as Django. • Writing a brief yet informative technical report.
Word limit and guidance	As part of your submission, you should also submit a brief report. The report should be restricted to a maximum of 4 A4 pages (excluding the title page). It should be typed using Times New Roman font size 11, single line spacing, and default margins.
Weighting	This coursework is worth 30% of the module mark.
Submission deadline	10 March 2025
Submission method	Electronic via Minerva
Feedback provision	Electronic via Minerva
Learning outcomes assessed	• Ability to design a relational database for a real-life problem. • Ability to design RESTful web service and document it. • Ability to write efficient, reliable, and robust server side as well as client-side code for communicating with a web service. • Ability to use a web development platform to achieve the above. • Ability to write a technical report that describes and documents the development of a web applications.
Module leader	Ammar Alsalka

1. Assignment guidance

In this coursework, you will use the Django framework to implement a RESTful web service for rating professors. You will host your service's website on www.pythonanywhere.com (for free). You will also write a simple command line client application for interacting with the API.

The Service

We want to develop a RESTful web service to allow students to rate the teaching of professors in various modules on a scale from 1 to 5. The service must provide all the required API's to allow client applications to offer the functionality described below.

The service maintains data about professors, modules, and the rating of professors by different users (students). Information about modules and professors are manually added by the admin of the service using the admin site. The admin site is automatically created by the Django web development framework.

A module may be:

- Taught by different professors in different academic years.
- Taught by different professors in different semesters.
- Taught by more than one professor at the same time (for example each teaching some part of the module).

Since an academic year spans two calendar years (e.g. 2018-19), an academic year will be given by its first year only (hence 2018-19 is given as 2018).

Users of the service (students) can rate professors but cannot add or change module information. Before they can rate professors, users must register by providing a username, email, and a password. Users can only rate professors when they are logged in to the service. The overall rating of a professor is the average of the professor's rating by all users across all module instances taught by this professor. A module instance is a module taught in a certain year and semester by one or more professors. Any decimal fraction in the average is rounded to the nearest integer.

A client application connected to the service will provide the user with the following options:

Option 1. View a list of all module instances and the professor(s) teaching each of them. Here is an example of a possible client application output for this option.

Code	Name	Year	Semester	Taught by
CD1	Computing for Dummies	2017	1	JE1, Professor J. Excellent VS1, Professor V. Smart
CD1	Computing for Dummies	2018	2	JE1, Professor J. Excellent
PG1	Programming for the Gifted	2017	2	TT1, Professor T. Terrible

Note that modules and professors are given unique identifiers in the web service to avoid any possible mix-up between names.

Option 2. View the rating of all professors. Here is an example of a possible client application output for this option.

```
The rating of Professor J. Excellent (JE1) is *****
The rating of Professor T. Terrible (TT1) is *
The rating of Professor V. Smart (VS1) is **
```

Option 3. View the average rating of a certain professor in a certain module:

The rating of Professor V. Smart (VS1) in module Computing for Dummies (CD1) is ***

Option 4. Rate the teaching of a certain professor in a certain module instance.

Note that all filtering of data and calculations should be done by the server, i.e. the client application does not process incoming data; the application simply displays the data returned by the service in a human readable format.

The Client Application

The client must support the following commands:

register
This is used to allow a user to register to the service using a username, email and a password. When the command is invoked, the program prompts the user to enter the username, email, and password of the new user. The syntax for this command is:

register

login
This command is used to log in to the service. The syntax for this command is:

login *url*

where:

url is the address of the service. Since you will be hosting your web service at www.pythonanywhere.com, this should be something like 'xyz.pyhtonanywhere.com', where xyz is your university username.

Invoking this command will prompt the user to enter a username and password which are then sent to the service for authentication.

logout
This causes the user to logout from the current session. The syntax for this command is:

logout

list
This is used to view a list of all module instances and the professor(s) teaching each of them (Option 1 above). The syntax for this command is:

list

view
This command is used to view the rating of all professors (Option 2 above). The syntax for this command is:

view

average
This command is used to view the average rating of a certain professor in a certain module (Option 3 above). The syntax of the command is:

average *professor_id module_code*

where:
professor_id is the unique id of a professor, and
module_code is the code of a module.

rate
This is used to rate the teaching of a certain professor in a certain module instance (Option 4 above). It has the following syntax:

rate *professor_id module_code year semester rating*

where*:*
professor_id is the unique id of a professor, e.g. JE1,
module_code is the code of a module, e.g. CD1,
year is a teaching year, e.g. 2018,
semester is a semester number, e.g. 2*, and*
rating is a numerical value between 1-5.

2. Assessment tasks

You are required to:

1. Design and implement a suitable database for the above service. You must specify all required tables, the fields of each table and their data types, and the relationships between the tables.
2. Implement the database using Django.
3. Design all API's that must be provided by the service to allow a client application to perform the functionality described in Options 1 – 4. For each API, you must specify: the purpose of the API, a URL, an HTTP method, the request data, and the response data.
4. Implement the server-side code of the service using Django.
5. Deploy the service on pythonanywhere.com.
6. Write a command line (command driven) client application to interact with the API. The application should be written in Python 3.x and should be able to send requests to, and process responses from, the web service.
7. Test the service and make sure that it is reliable, robust and responsive.
8. Document your work in a brief report (around 4 pages excluding the title page) that clearly, yet briefly, describes how you implemented the database, the APIs, and the client application. The report should also include brief instructions on how to invoke and use the client application. Please do NOT fill the report with text taken from online sources as we only need to know what you have done yourself. A template for the report can be found on Minerva with this brief.

3. General guidance and study support

To simplify system development, you can divide your work into individual Work Packages (WP) as follows:
WP1: design and implement the database model in Django, activate the admin site, test your model manually and refine if needed.

WP2: design the service and implement the APIs one at a time. In parallel write the client application code to interact with each API and test each part of the service as you go.

WP3: complete and finalize the client application.

WP4: test and debug the entire system on the local host.

WP5: upload the service to pythonanywhere.com and test everything.

4. Assessment criteria and marking process

All aspects of your work will be assessed. Both functional and quality features will be marked. Please see Section 8 below for a detailed breakdown of the marks.

5. Presentation and referencing

The quality of written English will be assessed in this work. As a minimum, you must ensure:

- Paragraphs are used
- There are links between and within paragraphs although these may be ineffective at times
- Word choice and grammar do not seriously undermine the meaning and comprehensibility of the argument
- Word choice and grammar are generally appropriate to an academic text

These are pass/ fail criteria. So irrespective of marks awarded elsewhere, if you do not meet these criteria, you will fail overall.

6. Submission requirements

1. Once you are satisfied that your service is working properly on the local host, open a free account and upload the server code to pythonanywhere.com. Detailed instructions on how to upload a Django project to pythonanywhere.com are available here https://help.pythonanywhere.com/pages/DeployExistingDjangoProject
2. When you register for a free account at www.pythonanywhere.com, please use your university username when prompted for an account name. Hence, if your university username is sc15xyz, your root domain address would be sc15xyz.pythonanywhere.com. This will make it much easier for instructors to link your submission to your website when assessing the coursework.
3. Make sure that you have properly uploaded your server code to pythonanywhere.com and tested it thoroughly.
4. Make sure that your client was written in Python 3.x and fully tested.
5. Nominate the module leader named above as your teacher on pythonanywhere.com. This will enable them to access and view your API code directly on the website. You can do this from the 'teacher' tab in your 'account' page. The username you have to use for this coursework is 'ammarsalka'.
6. Prepare a plain text file called readme.txt containing:
    I. instructions on using the client, i.e. a quick help page with a list of all commands and how to use them.
    II. the name of your pythonanywhere domain.
    III. the password instructors must use to login to the admin account on your service.
    IV. any other information instructors need to know in order to use your client.
7. Put the readme.txt file and your client program in one directory called 'myclient'
8. Bundle your Django project directory (the outer one) and the 'myclient' directory into a single directory. The name of this directory should be your university email address (without the @leeds.ac.uk). **DO NOT BUNDLE THE VIRTUAL ENVIRONMENT FILES** into this directory as this will create a huge folder that will take very long to download.
9. Compress the directory with Zip, and upload to Minerva.
10. Upload the report to Minerva as part of your submission.

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.

Use of Gen AI (Generative Artificial Intelligence):

This assessment is red category. AI tools cannot be used.

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

The database design is valid, scalable, and satisfy other database quality features	(2 marks)
The database has been fully and successfully implemented on Django	(2 marks)
Database tables, fields, and field data are properly displayed on the admin site	(2 marks)
Admin site can be accessed and used to add new data or edit existing one.	(2 marks)
The API's design is valid, scalable, and satisfy all REST constraints	(4 marks)
All APIs have been successfully implemented on the server	(4 marks)
Server-side code is clean, efficient, robust, and satisfy other code quality features	(2 marks)
Server-side code handles all kinds of exceptions successfully	(2 marks)
The client was implemented correctly and according to specification	(3 marks)
The client code is clean, efficient, robust, and satisfy other code quality features	(2 marks)
The client application is robust and properly handles user input errors	(2 marks)
The report is complete, accurate and well written	(3 marks)