



FIT9136 Algorithm and programming foundation in Python

Assignment 2

Lecturer in Charge: Shirin Ghaffarian Maghool

December 2021

Table of Contents

1. Key Information	3
1.1. Learning outcomes	3
1.2. Do and Do NOT	3
1.3. Marking Criteria	3
1.4. Documentation	4
1.5. Submission	4
2. Getting help	5
2.1. English language skills	5
2.2. Study skills	5
2.3. Things are tough right now	5
2.4. Things in the unit don't make sense	5
2.5. I don't know what I need	5
3. Key tasks (100 marks)	6
3.0. Website business logic description	6
3.1. Task 1 - User class	9
3.2. Task 2 - Course class	11
3.3. Task 3 - Instructor class	13
3.4. Task 4 - Review class	15
3.5. Connections between webpage button and backend functions	16
Important Notes:	23

1. Key Information

Format:	Individual
Due date:	6 th Jan 2022 5:00 pm (AEST)
Weight:	25% of the unit mark

1.1. Learning outcomes

1. design, construct, test and document Python programs;
2. demonstrate on advance topics of python, like classes, objects, visualization;
3. evaluate different algorithms and analyse their complexity;
4. translate problems into algorithms with appropriate implementations by investigating different strategies for the algorithm development

1.2. Do and Do NOT

Do	Do NOT
<ul style="list-style-type: none">• Maintain academic integrity¹• Get support early from this unit and other services in the university• Apply for special consideration for extensions²• Attend an interview (No attendance = 0 for Assignment 2)	<ul style="list-style-type: none">• Leave your assignment in draft mode• Submit late (10% daily penalty applies)³• Submission is not accepted after 7 days of the due date, unless you have special consideration.

1.3. Marking Criteria

Your work will be marked on

Functionality	Correctly working program	80%
Code Architecture and Style	Algorithms, data types, control structures and use of libraries, Variable names, readability, clear logic	10%
Documentation	Program comments, clarity and connection to code	10%

1

<https://www.monash.edu/rlo/research-writing-assignments/referencing-and-academic-integrity/academic-integrity>

² <https://www.monash.edu/exams/changes/special-consideration> (All the Special Consideration should be applied no later than two University working days after the due date of the affected assessment).

³ eg: original mark was 70/100, submitting 2 days late results in 50/100 (10 marks off). This includes weekends

1.4. Documentation

Commenting your code is essential as part of the assessment criteria (refer to Section 1.3.).

You should also include comments at the beginning of your program file, which specify your name, your Student ID, the start date and the last modified date of the program, as well as with a high-level description of the program. In-line comments within the program are also part of the required documentation.

1.5. Submission

You have to submit your assignment via the assignment submission link (i.e., "[Assignment 2 Submission](#)") on the Moodle site by the deadline specified in Section 1, i.e. **6 January 2022 (Thursday) by 5:00 pm**:

- There will be NO hard copy submission required for this assignment.
- You are required to submit your assignment as a .zip file named with your Student ID. For example, if your Student ID is 12345678, you would submit a zipped file named 12345678.zip
- Do not include any unnecessary file in this folder
- Note that marks will be deducted if this requirement is not strictly complied with.
- No submission accepted via email.
- Please note we **cannot mark any work on the FITGitLab Server**, you need to ensure that you submit via Moodle where you need to accept the student declaration. It is your responsibility to **ENSURE** that the submitted files are the correct files. We strongly recommend after uploading a submission, and prior to actually submitting in Moodle, that you download the submission and double-check its contents. Marks will be deducted for any of the requirements that are not complied with.

1.6. Deliverables

Your submission should contain the following documents:

- All files in the "model" folder including course.py, instructor.py, review.py and user.py.

- All files in the “static/img” folder including course_figure{1-6}.png, review_figure{1-2}.png and instructor_figure1. png image files (9 png files in total).
- Marks will be deducted for any of these requirements that are not strictly complied with.

2. Getting help

2.1. English language skills

if you don't feel confident with your English.

- Talk to English Connect: <https://www.monash.edu/english-connect>

2.2. Study skills

If you feel like you just don't have enough time to do everything you need to, maybe you just need a new approach

- Talk to a learning skills advisor:
<https://www.monash.edu/library/skills/contacts>

2.3. Things are tough right now

Everyone needs to talk to someone at some point in their life, no judgement here.

- Talk to a counsellor:
<https://www.monash.edu/health/counselling/appointments>
(friendly, approachable, confidential, free)

2.4. Things in the unit don't make sense

Even if you're not quite sure what to ask about, if you're not sure you won't be alone, it's always better to ask.

- Ask in Ed: <https://edstem.org/courses/4985/discussion/>
- Attend a consultation:
<https://lms.monash.edu/course/view.php?id=90081§ion=3>

2.5. I don't know what I need

Everyone at Monash University is here to help you. If things are tough now they won't magically get better by themselves. Even if you don't exactly know, come and talk with us and we'll figure it out. We can either help you ourselves or at least point you in the right direction.

3. Key tasks (100 marks)

This assignment is a simple emulation of Udemy data analysis website. You are required to analyse the data retrieved from the Udemy website. A website project based on Python Flask Framework is provided. You only need to add four classes User, Course, Instructor and Review into user.py, course.py, instructor.py and review.py files respectively. Each class handles some functions for running the website.

This is an individual assignment and must be your own work.

The allowed libraries are **os, re, pandas, numpy, matplotlib, random, math**. You will receive mark penalties if you use any other third party libraries. In the **provided project-> lib folder**, some functions and variables in the **helper.py** file can be imported and used in your tasks. For example, if you need to access the course.csv data file, you can add "from lib.helper import course_data_path" in your file. All the path variables in the helper.py should not be changed. Any unexpected changes may cause the website to not run, which will lead to mark penalties.

3.0. Website business logic description

1. **Login page** - Input username and password to login, the username and password will be authenticated by *User.authenticate_user()* function.
2. **Register page** - If no account, you can use the register page to register a new account. It is required to enter username, password and email address when registering. And a register timestamp(unix epoch time) will be generated automatically. All these four values will be sent to the *User.register_user()* function.
3. **Welcome page** - After login, the system will redirect to the welcome page. This page is the home page and only has one reset database button. The reset database button will call the *Course.clear_course_data()*, *Instructor.clear_instructor_data()* and *Review.clear_review_data()* functions to remove all the contents in course.csv, instructor.csv and review.csv files.
4. **Course list page** - In this page, the total number of courses and 20 course objects will be displayed. This total number of courses is obtained by calling *Course.get_total_number_of_courses()* function. All the course objects can be returned based on the page number. By default, the page number is 1. Each page has a maximum of 20 courses. The function *Course.get_courses_by_page()* is used

to get each page of course objects. At the bottom of the webpage, a page number list is shown. This page number list is generated by calling `Course.generate_page_num_list()`.

Next line shows two buttons, that are the Process Course Data button and the Course Analysis Figure button. The Process Course Data button will call `Course.get_course()` function to retrieve all the data from given course data files(this process may take a few minutes). The Course Analysis Figure button will call `Course.generate_course_figure{1-6}()` functions. After running these 6 functions, 6 figures will be saved into `lib.helper.figure_save_path` and all the explanations about each figure will be returned and displayed in a new webpage.

For each item of course, there will be a Details button and a Delete button. Details button will call `Course.get_course_by_course_id()` and redirect to the course details page. The Delete button will call `Course.delete_course_info()` function to remove selected data from the course.csv file.

Udemy Analysis System
Home
Courses
Instructors
Reviews

Course Analysis

Number of Courses: 1111

Process Course Data
Course Analysis Figure

#	Category Title	Subcategory Title	Course Title	Average Rating		
1	Development	Web Development	The Web Developer Bootcamp 2021	4.70	Details	Delete
2	Development	Web Development	Angular - The Complete Guide (2021 Edition)	4.62	Details	Delete
3	Development	Web Development	The Complete 2021 Web Development Bootcamp	4.68	Details	Delete
4	Development	Web Development	The Complete JavaScript Course 2021: From Zero to Expert!	4.71	Details	Delete
5	Development	Web Development	Modern React with Redux	4.68	Details	Delete
6	Development	Web Development	The Complete Web Developer Course 2.0	4.55	Details	Delete
7	Development	Web Development	Build Responsive Real-World Websites with HTML and CSS	4.66	Details	Delete
8	Development	Web Development	The Complete Web Developer in 2021: Zero to Mastery	4.69	Details	Delete
9	Development	Web Development	Vue - The Complete Guide (w/ Router, Vuex, Composition API)	4.75	Details	Delete

- Instructor list page** - In this page, the total number of **unique** instructors will be displayed. This number is generated by calling `Instructor.get_total_number_of_unique_instructors()` function. Because each instructor can teach more than one course, the total number of instructors is less than the total number of courses.






All the instructor objects can be returned based on the page number. By default, the page number is 1. Each page has a maximum of 20 instructors. The function `Instructor.get_instructors_by_page()` is used to get each page of instructor objects. At the bottom of the webpage, a page number list is shown. This page number list is generated by calling `Course.generate_page_num_list()`. For each item of the

instructor, we can see all the courses this instructor teaches by calling the *Instructor.find_courses_by_instructor_id()*.

There are two buttons: Process Instructor Data button and Instructor Analysis Figure button. The Process Instructor button will call *Instructor.get_instructors()*. This function may take a few minutes to finish. The Instructor Analysis Figure button has similar functionality as the course analysis button.

Instructor Analysis

Number of Instructors: 230

Process Instructor Data		Instructor Analysis Figure			
#	Instructor ID	Display Name	Job Title	Image_100x100	
1	4466306	Colt Steele	Developer and Bootcamp Instructor		Teach Courses
2	13952972	Maximilian Schwarzmüller	Professional Web Developer and Instructor		Teach Courses
3	31334738	Dr. Angela Yu	Developer and Lead Instructor		Teach Courses
4	7799204	Jonas Schmedtmann	Web Developer, Designer, and Teacher		Teach Courses
5	5487312	Stephen Grider	Engineering Architect		Teach Courses

6. **Review list page** - In this page, the total number of reviews will be displayed. This number is generated by calling the *Review.get_total_number_of_reviews()* function.

All the review objects can be returned based on the page number. By default, the page number is 1. Each page has a maximum of 20 reviews. The function *Review.get_reviews_by_page()* is used to get each page of review objects. At the bottom of the webpage, a page number list is shown. This page number list is generated by calling *Course.generate_page_num_list()*.

There are two buttons: Process Review Data button and Review Analysis Figure button. The Process Review Data button will call *Review.get_reviews()*. This function may take a few hours to finish. The Review Analysis Figure button has similar functionality as the course analysis button.

Review Analysis

Number of Reviews: 684

Process Review Data

Review Analysis Figure

#	Review ID	Review Rating	Review Created Time	Review Modified Time	Review User	Course ID
1	101421312	5.0	2021-11-29T23:03:27-08:00	2021-11-30T15:06:23-08:00	Lee Cheng Yao	625204
2	101421062	5.0	2021-11-29T23:00:24-08:00	2021-11-30T15:06:23-08:00	Stathis Skandalidis	625204
3	101416814	4.5	2021-11-29T21:53:28-08:00	2021-11-30T15:06:23-08:00	Adebisi	625204
4	101416668	4.5	2021-11-29T21:51:17-08:00	2021-11-30T15:06:23-08:00	Eugene Mahaso	625204
5	101416028	4.5	2021-11-29T21:39:36-08:00	2021-11-30T14:03:16-08:00	田中香次	625204
6	101415996	3.5	2021-11-29T21:39:01-08:00	2021-11-30T14:03:16-08:00	Sanket Kasurde	625204
7	101415456	5.0	2021-11-29T21:28:58-08:00	2021-11-30T15:06:23-08:00	Pintu Ray	625204
8	101415370	5.0	2021-11-29T21:26:55-08:00	2021-11-30T15:06:23-08:00	Mohit Kumar Rawat	625204
9	101414874	5.0	2021-11-29T21:18:47-08:00	2021-11-30T15:06:23-08:00	Devin Peters	625204
10	101413488	4.0	2021-11-29T20:52:31-08:00	2021-11-30T15:06:23-08:00	Elliott Brown	625204
11	101412634	5.0	2021-11-29T20:35:32-08:00	2021-11-30T15:06:23-08:00	Hannah Armstrong	625204

3.1. Task 1 - User class

You are required to implement the User class and some functions in the **user.py** file. The User class should have attributes id, username, password, email and register_time.

1. constructor.

Five positional arguments: new_id(int, default value is -1), username(str, default value is ""), password(str, default value is ""), email(str, default value is "empty@gmail.com"), register_time(str, default value is "xx-xx-xxxx")

2. __str__()->str.

Return string format example:

"user id: 12345 | username: xxxxx | password: xxxxx | email: xxx@gmail.com | register_time: xxxxx"

3. authenticate_user()->bool.

Two positional arguments - username and password. This function is used to check whether username and password can be matched with users saved in user.csv data file. If matched, this function will return True, otherwise return False.

4. check_username_exist()->bool.

One positional argument - username. This function is to check whether the given username exists in the user.csv data file. If it exists, return True, otherwise return False.

5. generate_unique_user_id()->str.

This function is used to generate and return a 6 digit unique user id which is not in the user.csv file. You can use a random function from a random library.

6. encrypt_password()->str.

One positional argument - password. For a given password, you are required to encrypt the string. For each character, if it is digit, it needs to be times 10 and plus 5. Two “-” will be added before and after the result. For example, 5 will get “--55--”. If the character is not digit, two “*” will be added before and after the character. For example, “a” will get “**a**”. Finally, all the encrypted text will be put together. For example, given password “aaaaa”, the encrypted result will be “**a****a****a****a****a**”.

7. register_user()->bool.

Four positional arguments - username, password, email, register_time.

- If the username exists in the user.csv file, return False. If register success, return True
- A unique user id is required when registering a new user.
- Register_time will be a unix epoch timestamp (milli seconds) which needs to be converted using **date_conversion()** function. The new user needs to be written into the user.csv file. All the attributes are **separated by three semicolons** - “;;;”. The format example is:
285108;;;aaaaa;;;*a****a****a****a****a*;;;aaaaa@gmail.com;;;2021-11-29 32:32:28.590

8. `date_conversion()-> str.`

One positional argument - register_time. The given register_time will be a unix epoch timestamp (milli seconds) and it needs to be converted to format "year-month-day hour:minute:second.milliseconds". For example, a timestamp 1637549590753 will be converted to str "2021-11-22 13:53:10.753" and returned. The time should be GMT+11 Melbourne timezone. Refer this link <https://www.unixtimestamp.com/index.php> to check how to convert unix epoch time to human readable format. A function called get_day_from_timestamp(timestamp) is provided in the lib.helper file. By using this function, you can convert the timestamp to the day of month. You can import and use this function in the user.py file.

You cannot use any time related libraries here. You are required to implement the conversion by yourself.

3.2. Task 2 - Course class

You are required to implement the Course class and some functions in the **course.py** file. The Course class should have attributes category_title, subcategory_id, subcategory_title, subcategory_description, subcategory_url, course_id, course_title, course_url, num_of_subscribers, avg_rating and num_of_reviews.

1. constructor.

Eleven positional arguments: category_title(str, default value is ""), subcategory_id(int, default value is -1), subcategory_title(str, default value is ""), subcategory_description(str, default value is ""), subcategory_url(str, default value is ""), course_id(int, default value is -1), course_title(str, default value is ""), course_url(str, default value is ""), num_of_subscribers(int, default value is 0), avg_rating(float, default value is 0.0) and num_of_reviews(int, default value is 0).

2. __str__()->str.

Return string format example:

```
"category: Development | subcategory id:8 | subcategory:Web Development |  
subcategory url:/courses/development/web-development/ | course id:625204 | course  
title:The Web Developer Bootcamp 2021 | course  
url:/course/the-web-developer-bootcamp/ | number of subscriber:713744 | avg  
rating:4.6973553 | number of reviews:215075"
```

3. get_courses() no return.

This function will extract course information from the given course data files. In the source_course_files folder, there are 10 categories of courses. In each category folder, there are some subcategories. Inside each subcategory folder, you can find the course json files. You need to retrieve the category_title from the 10 category folder names and other course info from the json files.

After retrieving the required data, you need to write the info into course.csv file to save all the course data. All the data need to be separated by ",". The required attributes and data format is:

```
"{category_title},{subcategory_id},{subcategory_title},{subcategory_description},{  
{subcategory_url},{course_id},{course_title},{course_url},{num_of_subscribers},{  
{avg_rating},{num_of_reviews}"
```

You can use re or str functions to get the expected data. Any json related libraries or functions cannot be used. Such as library **json** or **pandas.io.json.read_json()**.

4. clear_course_data() no return.

This function will remove all the content in the course.csv file. After calling this function, the course.csv file will become an empty file.

5. generate_page_num_list()->list of int.

Two positional arguments: page and total_pages. This function uses the current page number and total pages to generate a list of integers as viewable page numbers. For

example, the image below shows a default page number list [1,2,3,4,5,6,7,8,9] when the current page number is 1.

Start	...	1	2	3	4	5	6	7	8	9	...	End
-------	-----	---	---	---	---	---	---	---	---	---	-----	-----

If the current page number is less than or equal to 5, the generated page number list is always [1,2,3,4,5,6,7,8,9]. If the current page number is greater than 5 and less than total pages minus 4, the page number list will be integers from current page number minus 4 until current page number plus 4. For example, in the image below, the current page is 8 and the number list becomes [4,5,6,7,8,9,10,11,12].

Start	...	4	5	6	7	8	9	10	11	12	...	End
-------	-----	---	---	---	---	---	---	----	----	----	-----	-----

If the current page is greater than or equal to total pages minus 4, the list of numbers changes to range between total pages minus 8 until total pages.

6. `get_courses_by_page()`->tuple

One positional argument: `page`. The return value is a tuple that contains a list of Course objects and total pages of courses. This function reads the `course.csv` file to retrieve all the course information. With all the course information and the current page number, a list of Course objects will be generated and the total pages will be returned. Each page has at most 20 courses.

For example, if there are 100 courses info in the `course.csv` file and the current page number is 2, then the 21-40 lines course info will be converted to a list with 20 Course objects. The total page number is 5.

7. `delete_course_info()`->bool

One positional argument: `course_id`. The function reads course info from the `course.csv` file and deletes the line that contains `course_id`. Finally, this function returns whether the deletion is successful or not. If the `course_id` cannot find, return False.

8. `get_course_by_course_id()`->tuple

One positional argument: `course_id`. You are required to find the course by given `course_id` and convert the info to a Course object. Then, using the retrieved course info to get the `num_of_subscribers`, `avg_rating` and `num_of_reviews`. Based on these three numbers, generate a comment for this course. If the `num_of_subscribers` greater than 100000 and `avg_rating` greater than 4.5 and `num_of_reviews` greater than 10000, the comment should be "Top Courses". If the `num_of_subscribers` greater than 50000 and `avg_rating` greater than 4.0 and `num_of_reviews` greater than 5000, the comment should be "Top Courses". If the `num_of_subscribers` greater than 10000 and `avg_rating` greater than 3.5 and `num_of_reviews` greater than 1000, the comment should be "Top Courses". The other courses are "General Courses". The Course object and comment will be returned as a tuple.

9. `get_total_number_of_courses()`->int

This function returns the total number of courses in the course.csv file.

10. `generate_course_figure1()`->str

Generate a graph to show the top 15 courses with the most subscribers. (any chart)

11. `generate_course_figure2()`->str

Generate a graph to show the top 15 avg rating(ascending order) of courses that have over 50000 reviews.(any chart)

12. `generate_course_figure3()`->str

Generate a graph to show the all the courses avg rating distribution that has subscribers between 1000 and 100 (scatter chart)

13. `generate_course_figure4()`->str

Generate a graph to show the number of courses for all categories and sort in ascending order (pie chart, offsetting the second largest number of course with "explode")

14. `generate_course_figure5()`->str

Generate a graph to show the top 15 subcategories with the most courses (any chart)

15. `generate_course_figure6()`->str

Generate a graph to show the distribution of courses based on the number of reviews and number of subscribers . (scatter chart)

In all the graphs, if the course title is too long, you need to extract the first 3 words. All the *`generate_course_figure{1-6}`* functions are required to return a string explanation about your understanding of this figure.

3.3. Task 3 - Instructor class

You are required to implement the Instructor class and some functions in the **`instructor.py`** file. The Instructor class should have attributes `id`, `display_name`, `job_title`, `image_100x100` and `course_id`.

1. `constructor`.

This function has five positional arguments: `id`(int, default value is -1), `display_name`(str, default value is ""), `job_title`(str, default value is ""), `image_100x100`(str, default value is "") and `course_id`(int, default value is -1).

2. `__str__()`->str.

Return string format example:

"instructor id:2467626 | Anthony Alicea | Software Developer, Architect, and UX Designer | https://img-c.udemycdn.com/user/100x100/2467626_f2e0.jpg".

3. **get_instructors()** no return.

This function will extract instructor information from the given course data files. Similar to the process of retrieving course data, but this function focuses on the instructor data of each course. In each course item, there could be multiple instructors.

After retrieving the required data, you need to write the info into instructor.csv file to save all the instructor data. All the data need to be separated by “;;;”. The required attributes and data format is:

“{course_id};;;{instructor_id};;;{instructor_display_name};;;{instructor_job_title};;;instructor_image_100x100”.

You can use re or str functions to get the expected data. Any json related libraries or functions cannot be used. Such as library **json** or **pandas.io.json.read_json()**.

4. **clear_instructor_data()** no return.

This function will remove all the content in the instructor.csv file. After calling this function, the instructor.csv file will become an empty file.

5. **get_total_number_of_unique_instructors()**->int

This function returns the total number of unique(instructor id is unique) instructors in the instructor.csv file.

6. **find_courses_by_instructor_id()**->tuple

One positional argument: instructor_id. This function reads the instructor.csv file and course.csv file to find all the course information the specified instructor teaches. If this instructor teaches more than 20 courses, only 20 courses will be returned with the total number of courses this instructor teaches. Otherwise, all the courses and the total number will be returned. The return type is a tuple that contains a list of course objects and the total number of courses taught by this instructor.

7. **get_instructors_by_page()**->tuple

One positional argument: page. This function reads the instructor.csv file to retrieve all the instructor information. With all the instructor information and the current page number, a list of Instructor objects and the total pages will be generated. Each page has at most 20 instructors. A tuple contains the list of instructors and total page number will be returned.

This function is similar to the *Course.get_course_by_page()* function.

8. **generate_instructor_figure1()**->str

Generate a graph that shows the top 10 instructors who teach the most courses.(any chart)

In all the graphs, if the instructor display name is too long, you need to extract the first 3 words. The *generate_instructor_figure1()* function is required to return a string explanation about your understanding of this figure.

3.4. Task 4 - Review class

You are required to implement the Review class and some functions in the **review.py** file. The Review class should have attributes `id`, `rating`, `created`, `modified`, `user_title`, `course_id` and `crawable_count`.

1. **constructor.**

This function has seven positional arguments: `id`(int, default value is -1), `rating`(int, default value is 0.0), `created`(str, default value is ""), `modified`(str, default value is ""), `user_title`(str, default value is ""), `course_id`(str, default value is "") and `crawable_count`(str, default value is "").

2. **__str__()**->str.

Return string format example:

```
"review id:101421312 | review rating:5.0 | created time:2021-11-29T23:03:27-08:00 |
modified time:2021-11-30T15:06:23-08:00 | user title:Lee Cheng Yao | course
id:625204"
```

3. **get_reviews()** no return.

This function emulates a simple web crawler to crawl data from the udemy website. The link we are going to use is given below :
"https://www.udemy.com/api-2.0/courses/4106606/reviews/?courseId=4106606&page=1".

You can try to paste this link into a web browser to see the return result. In this link, there are two parts of `course_id`. If we change the `course_id`, the return result will be different. In the course Class, we can get and save all the course info into the `course.csv` file. In this function, we need to get all the `course_id` and use this `course_id` to get the first page of reviews. Since there are too many courses in the `course.csv` file, you only need to get the first 10000 `course_id` and use these `course_ids` and the given url pattern to crawl the reviews. There is a function called `send_request(url)` that is provided in the `lib.helper` file. You can import and use this function in the `review.py` file. This process could last for hours. Make sure you have a computer that can keep running for a long time.

After retrieving the required data, you need to write the info into `review.csv` file to save all the review data. All the data need to be separated by ";;". The required attributes and data format is:

```
{course_id};;{review_id};;{review_rating};;{review_created};;{review_modified};;{review_user_title};;{review_crawable_count}
```

You can use *re* or *str* functions to get the expected data. Any json related libraries or functions cannot be used. Such as library `json` or `pandas.io.json.read_json()`.

4. **clear_review_data()** no return.

This function will remove all the content in the `review.csv` file. After calling this function, the `review.csv` file will become an empty file.

5. **get_total_number_of_reviews()**->int

This function returns the total number of reviews in the review.csv file.

6. **get_reviews_by_page()**->tuple

One positional argument: page. A tuple contains a list of Review objects and total pages of reviews. This function reads the review.csv file to retrieve all the review information. With all the review information and the current page number, a list of Review objects will be generated and the total pages will be returned. Each page has at most 20 reviews.

This function is similar to the *Course.get_course_by_page()* and *Instructor.get_instructor_by_page()* functions.

7. **generate_review_figure1()**->str

Generate a graph to show the total number of (users, courses, instructors, actual reviews, crawlable reviews, subscribers). (any chart)

8. **generate_review_figure2()**->str

Generate a graph to show the number of users who published reviews [0-5), [5-10), [10-30), >=30, (pie chart)

All the *generate_review_figure{1-2}* functions are required to return a string explanation about your understanding of this figure.

3.5. Connections between webpage button and backend functions

1. **Login page**

Please sign in

Username
Password

Sign in

No account? Please register [here](#)

© 2017–2022

When you click the Sign in button, the function `User.authenticate_user(username, password)` will be called. The sign in can pass only when this function returns True, otherwise, an error message will show.

2. Register page

Register

Username

Password

Email

Register

Back

© 2017–2022

When you click the Register button, the function `User.register_user(username, password, email, register_time)` will be called. The register will be successful only if this function returns `True`. After registering, a new user will be written into the `user.csv` file.

3. Welcome page

Udemy Analysis System

[Home](#) [Courses](#) [Instructors](#) [Reviews](#)

Welcome to Udemy Analysis System

Reset Database

The Reset Database button will call the following methods at the same time:

- `Course.clear_course_data()`
- `Review.clear_review_data()`
- `Instructor.clear_instructor_data()`

The Courses link will call the following methods at the same time:

- *Course.get_courses_by_page(page)*
- *Course.get_total_number_of_courses()*
- *Course.generate_page_num_list(page, total_pages)*.

Then, in the courses page, we can see a list of courses and page numbers in the bottom.

The Instructors link will call the following methods at the same time:

- *Instructor.get_instructors_by_page(page)*
- *Instructor.get_total_number_of_unique_instructors()*
- *Course.generate_page_num_list(page, total_pages)*

The Reviews link will call the following methods at the same time:

- *Review.get_reviews_by_page(page)*
- *Review.get_total_number_of_reviews()*
- *Course.generate_page_num_list(page, total_pages)*.

4. Courses page

Udemy Analysis System

[Home](#) [Courses](#) [Instructors](#) [Reviews](#)

Course Analysis

Number of Courses: 2000

Process Course Data

Course Analysis Figure

#	Category Title	Subcategory Title	Course Title	Average Rating		
1	Development	Web Development	The Web Developer Bootcamp 2021	4.70	Details	Delete
2	Development	Web Development	Angular - The Complete Guide (2021 Edition)	4.62	Details	Delete
3	Development	Web Development	The Complete 2021 Web Development Bootcamp	4.68	Details	Delete
4	Development	Web Development	The Complete JavaScript Course 2021: From Zero to Expert!	4.71	Details	Delete
5	Development	Web Development	Modern React with Redux	4.68	Details	Delete
6	Development	Web Development	The Complete Web Developer Course 2.0	4.55	Details	Delete
7	Development	Web Development	Build Responsive Real-World Websites with HTML and CSS	4.66	Details	Delete
8	Development	Web Development	The Complete Web Developer in 2021: Zero to Mastery	4.69	Details	Delete

			Developer Bootcamp			
12	Development	Web Development	Advanced CSS and Sass: Flexbox, Grid, Animations and More!	4.74	Details	Delete
13	Development	Web Development	Master Microservices with Spring Boot and Spring Cloud	4.50	Details	Delete
14	Development	Web Development	The Complete ASP.NET MVC 5 Course	4.43	Details	Delete
15	Development	Web Development	Modern JavaScript From The Beginning	4.70	Details	Delete
16	Development	Web Development	Learn and Understand NodeJS	4.65	Details	Delete
17	Development	Web Development	React JS- Complete Guide for Frontend Web Development [2021]	4.42	Details	Delete
18	Development	Web Development	Reactive Angular Course (with RxJS)	4.65	Details	Delete
19	Development	Web Development	JSP, Servlet, JSP + Hibernate: A complete guide	4.43	Details	Delete
20	Development	Web Development	Build a Social Network from Scratch: JavaScript PHP + MySQL	4.56	Details	Delete

Start
...
1
2
3
4
5
6
7
8
9
...
End

The Process Course Data button will call *Course.get_courses()*.

The Course Analysis Figure button will call *Course.generate_course_figure{1-6}()* functions. All these 6 functions will not only generate an image but also return the figure explanations.

The Details button will call *Course.get_course_by_course_id(course_id)*. This function will return a course info and a comment for this course.

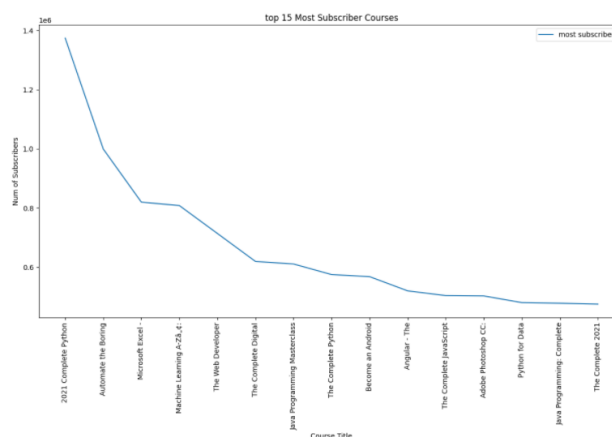
The Delete button will call *Course.delete_course_info(course_id)* function.

5. Course analysis result page

Udemy Analysis System

[Home](#) [Courses](#) [Instructors](#) [Reviews](#)

Course Analysis Result



generate_top_15_most_subscriber_chart explanation

This page shows the generated images and image explanations.

6. Instructors page

Udemy Analysis System











[Home](#) [Courses](#) [Instructors](#) [Reviews](#)

Instructor Analysis

Number of Instructors: 230

Process Instructor Data

Instructor Analysis Figure

#	Instructor ID	Display Name	Job Title	Image_100x100	
1	4466306	Colt Steele	Developer and Bootcamp Instructor		Teach Courses
2	13952972	Maximilian Schwarzmüller	Professional Web Developer and Instructor		Teach Courses
3	31334738	Dr. Angela Yu	Developer and Lead Instructor		Teach Courses
4	7799204	Jonas Schmedtmann	Web Developer, Designer, and Teacher		Teach Courses
15	712832	Mosh Hamedani	Passionate Software Engineer and Best-selling Author		Teach Courses
16	21681922	Brad Traversy	Full Stack Web Developer & Instructor at Traversy Media		Teach Courses
17	2467626	Anthony Alicea	Software Developer, Architect, and UX Designer		Teach Courses
18	98170214	EdYoda Digital University	Visit us at www.edyoda.com		Teach Courses
19	47000136	Qaifi Khan	Learning Enabler		Teach Courses
20	56962440	Mavludin Abdulkadirov			Teach Courses

Start ... 1 2 3 4 5 6 7 8 9 ... End

The Process Instructor Data button calls *Instructor.get_instructors()* function.

The Instructor Analysis Figure button calls *Instructor.generate_instructor_figure1()* function.

The Teach Courses button calls *Instructor.find_courses_by_instructor_id(instructor_id)* function.

7. Instructor teach course list page

Udemy Analysis System

[Home](#) [Courses](#) [Instructors](#) [Reviews](#)

Instructor Teach Course List

Number of Courses: 11

#	Course ID	Course Title	Avg Rating
1	625204	The Web Developer Bootcamp 2021	4.70
2	1218586	The Advanced Web Developer Bootcamp	4.53
3	2320056	The Modern React Bootcamp (Hooks, Context, NextJS, Router)	4.68
4	1968412	The Bootstrap 4 Bootcamp	4.68
5	1350984	The Modern Python 3 Bootcamp	4.69
6	1406344	JavaScript Algorithms and Data Structures Masterclass	4.75
7	2634490	The Modern Javascript Bootcamp Course (2021)	4.75
8	1187016	The Ultimate MySQL Bootcamp: Go from SQL Beginner to Expert	4.63
9	3998050	The Linux Command Line Bootcamp: Beginner To Power User	4.79
10	3792262	The Git & Github Bootcamp	4.82
11	1664822	Bitcoin and Cryptocurrency Bootcamp	4.49

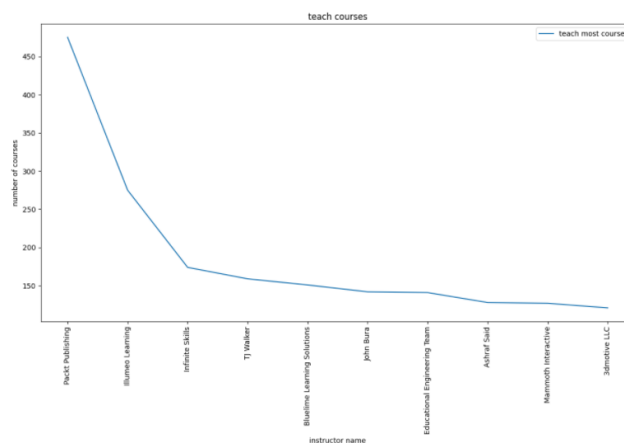
This page shows all the courses this instructor teaches. If this instructor teaches more than 20 courses, only 20 courses will be shown here.

8. Instructor analysis result page

Udemy Analysis System

[Home](#) [Courses](#) [Instructors](#) [Reviews](#)

Instructor Analysis Result



generate_graph_top_10_instructor_teach_most_courses explanation

This page shows the generated images and image explanations.

9. Reviews page

Udemy Analysis System

[Home](#) [Courses](#) [Instructors](#) [Reviews](#)

Review Analysis

Number of Reviews: 684

Process Review Data

Review Analysis Figure

#	Review ID	Review Rating	Review Created Time	Review Modified Time	Review User	Course ID
1	101421312	5.0	2021-11-29T23:03:27-08:00	2021-11-30T15:06:23-08:00	Lee Cheng Yao	625204
2	101421062	5.0	2021-11-29T23:00:24-08:00	2021-11-30T15:06:23-08:00	Stathis Skandalidis	625204
3	101416814	4.5	2021-11-29T21:53:28-08:00	2021-11-30T15:06:23-08:00	Adebisi	625204
4	101416668	4.5	2021-11-29T21:51:17-08:00	2021-11-30T15:06:23-08:00	Eugene Mahaso	625204
5	101416028	4.5	2021-11-29T21:39:36-08:00	2021-11-30T14:03:16-08:00	田中香次	625204
6	101415996	3.5	2021-11-29T21:39:01-08:00	2021-11-30T14:03:16-08:00	Sanket Kasurde	625204
7	101415456	5.0	2021-11-29T21:28:58-08:00	2021-11-30T15:06:23-08:00	Pintu Ray	625204
8	101415370	5.0	2021-11-29T21:26:55-08:00	2021-11-30T15:06:23-08:00	Mohit Kumar Rawat	625204
9	101414874	5.0	2021-11-29T21:18:47-08:00	2021-11-30T15:06:23-08:00	Devin Peters	625204
10	101413488	4.0	2021-11-29T20:52:31-08:00	2021-11-30T15:06:23-08:00	Elliott Brown	625204
11	101412634	5.0	2021-11-29T20:35:32-08:00	2021-11-30T15:06:23-08:00	Hannah Armstrong	625204
8	101415370	5.0	2021-11-29T21:26:55-08:00	2021-11-30T15:06:23-08:00	Mohit Kumar Rawat	625204
9	101414874	5.0	2021-11-29T21:18:47-08:00	2021-11-30T15:06:23-08:00	Devin Peters	625204
10	101413488	4.0	2021-11-29T20:52:31-08:00	2021-11-30T15:06:23-08:00	Elliott Brown	625204
11	101412634	5.0	2021-11-29T20:35:32-08:00	2021-11-30T15:06:23-08:00	Hannah Armstrong	625204
12	101412074	5.0	2021-11-29T20:25:11-08:00	2021-11-30T15:06:23-08:00	Alejo Hincapie	625204
13	101423964	5.0	2021-11-29T23:44:37-08:00	2021-11-30T15:06:23-08:00	Deepak Bawa	756150
14	101423558	4.0	2021-11-29T23:38:30-08:00	2021-11-30T15:06:23-08:00	Asha Kiran	756150
15	101422028	4.0	2021-11-29T23:15:27-08:00	2021-11-30T15:06:23-08:00	Dave Mercado	756150
16	101421544	5.0	2021-11-29T23:07:01-08:00	2021-11-30T15:06:23-08:00	Mohammed Elsebaey	756150
17	101419426	4.5	2021-11-29T22:35:08-08:00	2021-11-30T15:06:23-08:00	kedar karmarkar	756150
18	101419268	1.0	2021-11-29T22:32:42-08:00	2021-11-30T15:06:23-08:00	Ganesh Deokar	756150
19	101413004	5.0	2021-11-29T20:43:06-08:00	2021-11-30T15:06:23-08:00	Ben Major	756150
20	101412360	5.0	2021-11-29T20:30:25-08:00	2021-11-30T15:06:23-08:00	Hector David Valdez Tirado	756150

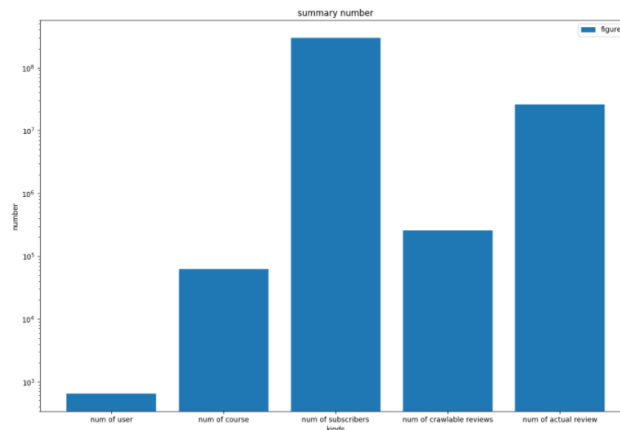
Start ... **1** 2 3 4 5 6 7 8 9 ... End

The Process Review Data button calls *Review.get_reviews()* function.

The Review Analysis Figure button calls *Review.generate_review_figure{1-2}()* functions.

10. Reviews analysis result page

Review Analysis Result



graph_show_total_numbers explanation

This page shows the generated images and image explanations.

Important Notes:

- If any exception happens when running your program, you will lose 50% marks.
- Please refer to the `_demo_{data}.csv` file to see the correct data format
- Your program should also work if you change all the data file paths to demo data file paths.