ENGR 102 – Programming Practice
Mini Project 2
Fall 2018

Tags: Tkinter, GUI Widgets, Layout, Recommendation, Events, Files, Exceptions

To get a moment away from the courses and exams, we hang out somewhere to relax alone or together with our friends. Sometimes we would like to discover new places, try new tastes or watch the landscapes we haven't seen before. We can explore these new places through the recommendations of our friends. In this mini project, you are going to develop a recommendation engine that will help you to choose a new place to hang out by the ratings of other users.

How should it look like?

Your program will have a graphical user interface (GUI) which will function and look like as explained below.

Details about how it should work are also provided below.

1. Initially, your GUI will look like as shown in Figure 1.

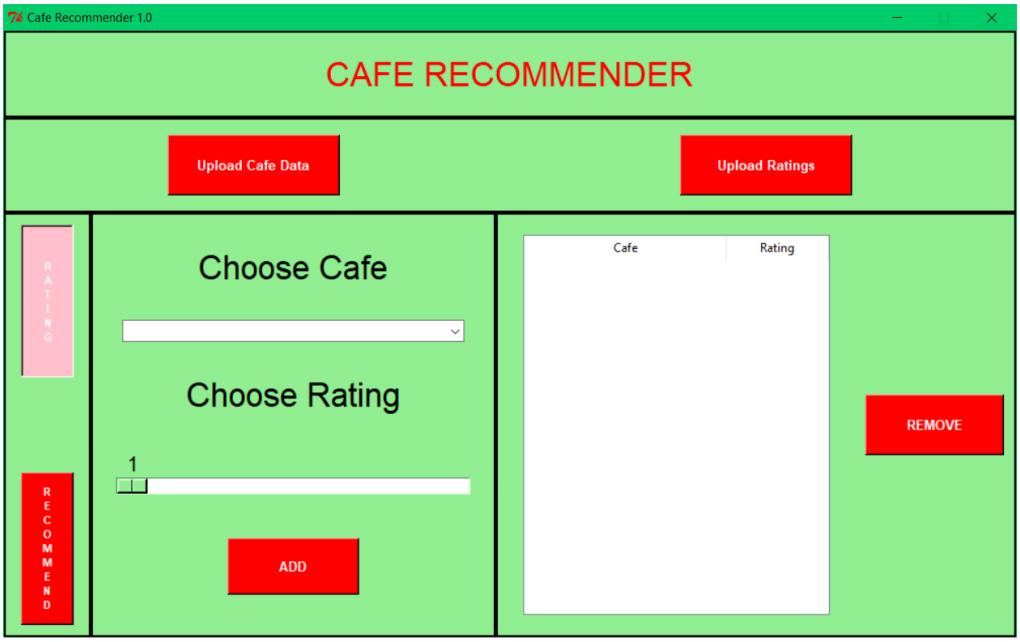

Figure 1. Main GUI window

2. According to the clicked button on the left, the interface will display a different set of widgets. Once the user presses the **Rating** button, the GUI window will look like as shown in Figure 1, whereas if the user presses the **Recommend** button, the GUI window will look like as shown in Figure 2.

Figure 2. GUI window when Recommend Button is pressed.

3. At the beginning, the user will upload cafe list as an excel file and the database file for other user's ratings about that cafes via the provided upload buttons at the top (**Upload Cafe Data** and **Upload Ratings** buttons in Figure 1). Once any of the upload buttons is pressed, a file dialog should be opened (see Figure 3) where the user will select the file. After proper files are selected and uploaded, you should parse those two files. Based on the parsed data, the user will then be able to perform specific tasks as explained in items below.

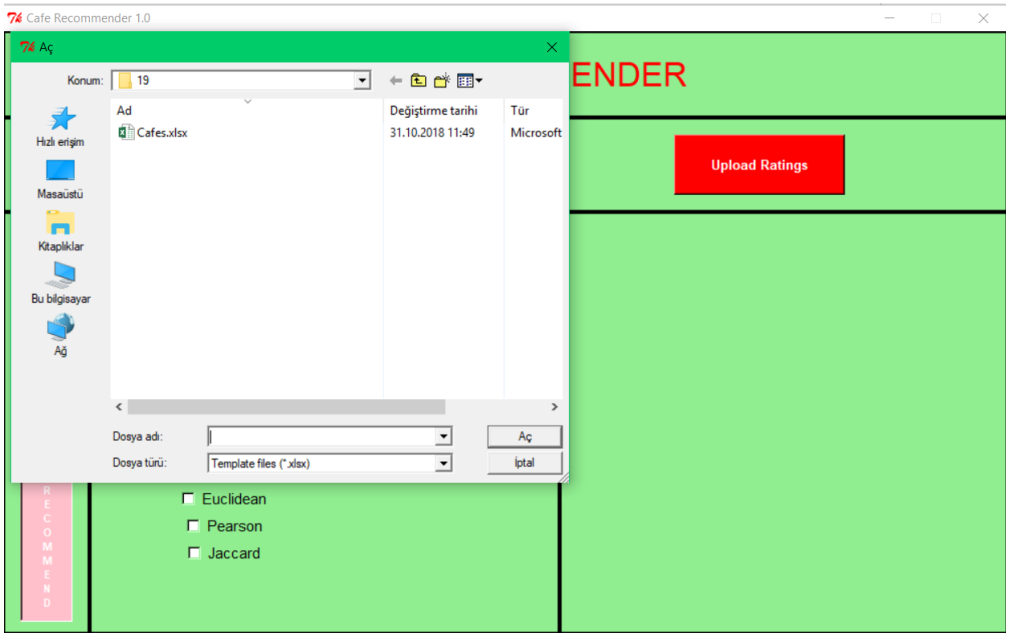

Figure 3. GUI when **Upload Cafe Data** Button is pressed.

4. Once the files are uploaded, the user should be able to see the Cafes at the **ComboBox** in the Rating Page (Figure 4). The user should also be able to select a cafe from the Combobox and select a rating from the Slider which is a **Scale** widget. After selecting the cafe and the rating, if the user presses the **Add** button, him/her selection should be displayed at the **TreeView table**. If the user presses the **Remove** button, him/her selection should be removed from the **TreeView**. (see Figure 4)

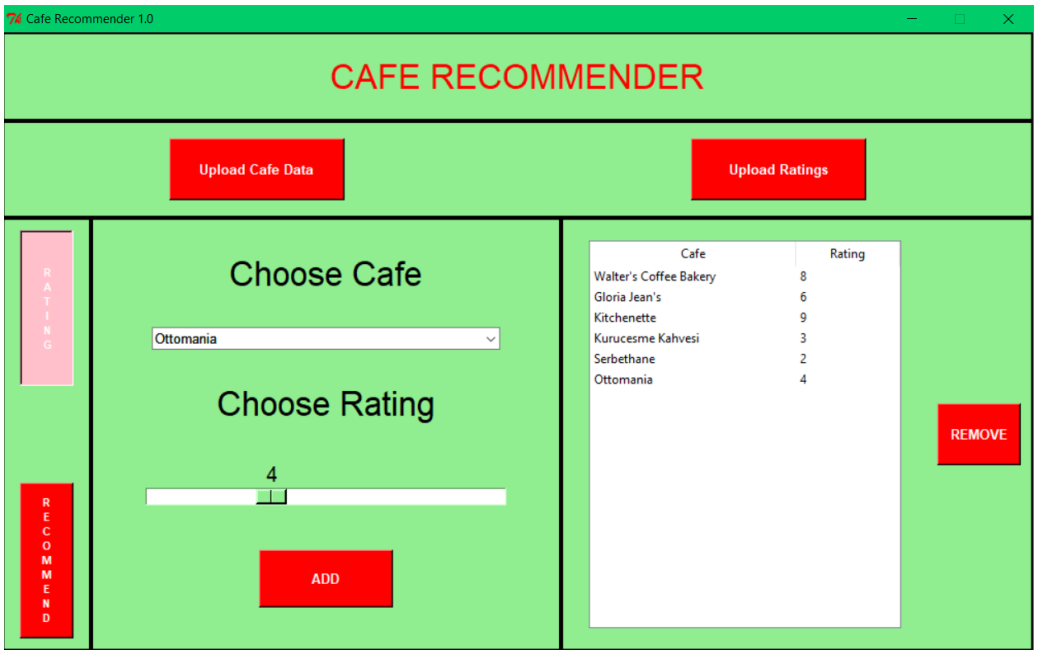

Figure 4. GUI window when user selects a cafe and a rating and adds them to the TreeView.

5. After the rating process ends, the user should be able to press **Recommend** button and get a recommendation based on him/her selection (Figure 5). For the recommender system calculations, the user should provide some settings about the recommendation process on the left side of the Recommend Page (Figure 5). First, the user should provide a recommendation number which is the number of recommendations that he/she wants and a choice of similarity metrics. After those settings are provided, the user should then be able to press **Recommend Similar User** or **Recommend Cafe** buttons. (see Figure 5)

6. When the user presses the **Recommend Similar User** button, the user should see similar users like himself/herself (Figure 5). Moreover, when the user selects a specific user as suggested to be similar and presses the **Get User's Rating** button, he/she should be able to see the ratings of that user in a **TreeView** at the top of which a **Label** should tell us which user's rating we are looking into. (e.g. Merve's Rating in Figure 5).

7. When the user presses the **Recommend Cafe** button, the user should see recommended cafes for him/her (see Figure 6).

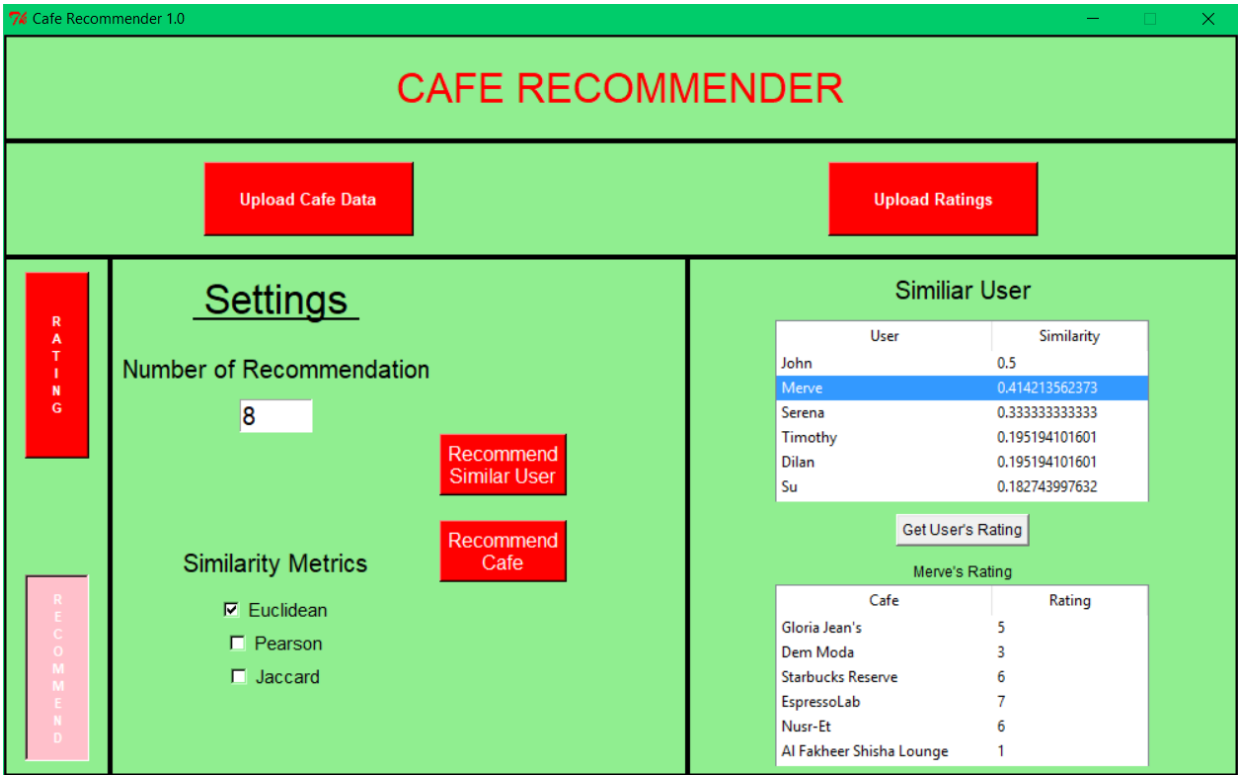

Figure 5. GUI when user get recommendations for similar users by pressing **Recommend Similar User** button.

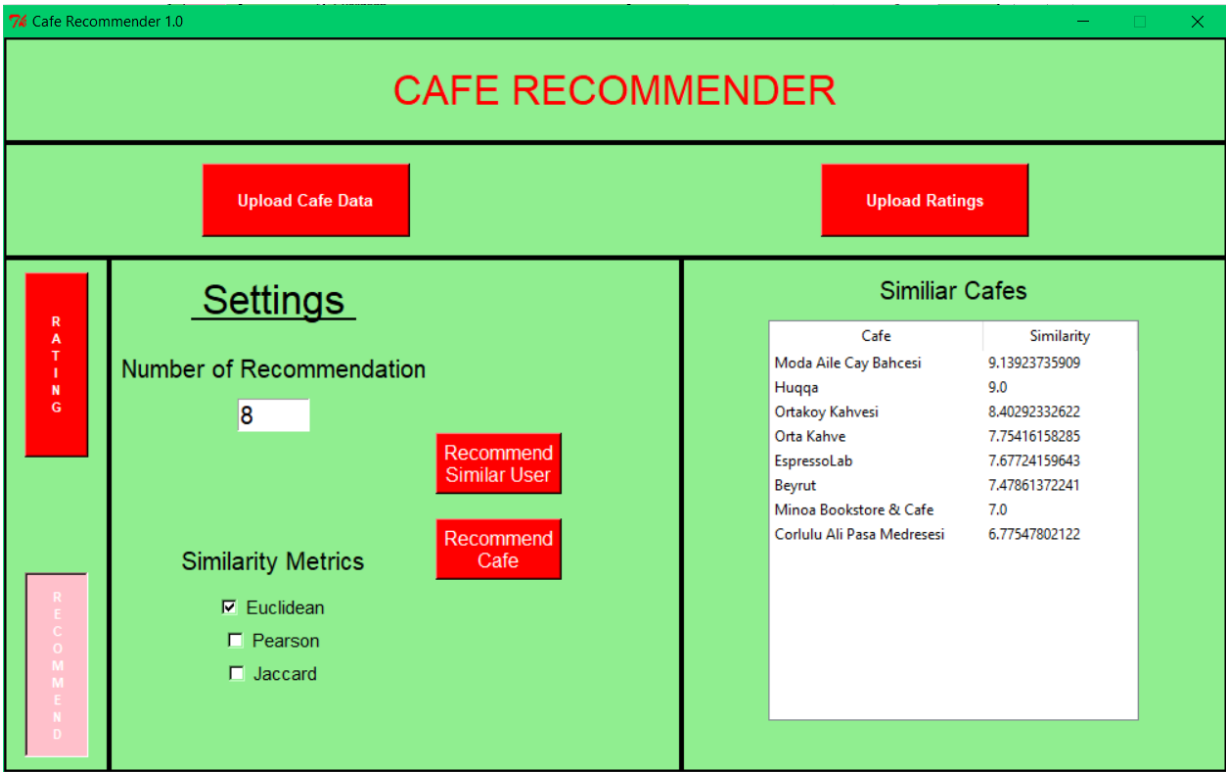

Figure 6. GUI when user get recommendations directly by pressing **Recommend Cafe** button.

How should it work?

1. First, the user should load the list of cafes from the excel file (Cafes.xlsx). The program will open the file dialog from where the file can be selected. The required file (Cafes.xlsx) will be provided to you.
2. At the end of the first step, the data of Cafes.xlsx file should have been loaded. Now the user can see all cafes in the combobox and rate them by selecting a cafe from the ComboBox and selecting a rating from the Slider as a Scale Widget. After the selection, the user may add him/her selection to a TreeView at the right side of the Rating Page by pressing an **Add** button. If the user wants to remove him/her selection, it is also handled by a**Remove** button.
3. After all rating process ends, the user can get recommendations via adjusting some preferred settings. These settings include the number of recommendations to be provided via an **Entry** widget and the choice of similarity metrics via a **Checkbutton** widget.
4. At this recommendation step, you should use the **recommendations.py** file, which is also provided to you.
5. If the user wants to see similar users for getting the recommendation, you should do the computations for **used-based recommendation**. When the user presses the **Recommend Similar User** button, the user should see similar users like himself/herself and the similarity score between him/her and other users in a TreeView at the right top of Recommendation Page (Figure 5). Following that, if the user selects a specific other user and presses **Get User's Rating** Button, he/she should be able to see the ratings of that specific user for cafes in a separate TreeView (Figure 5).
6. If the user wants to see similar cafes for getting the recommendation, you should do the computation for **item-based recommendation**. When the user presses **Recommend Cafe** button, the user should see the similar cafes that himself/herself may like in a TreeView with the possible ratings for that cafe (Figure 6).
7. If the user wants to add new ratings, while he/she is already at the stage of getting recommendations, the user should be able to do this. The user should switch between Rating Page and Recommendation Page without losing any selection, settings or data.

Bonus (10 point)

If you complete the following conditions, you will receive 10 points bonus:

- When the user clicks on buttons at the left (rating or recommend button), the clicked button color should be changed, and its relief style should be changed to "sunken", and the relief of the other button which is unselected should be of style "raised".

- Whenever mouse moves over buttons at the left (rating or recommend button), the button color should change its color to "orange".

Implementation Notes / Hints:

- **You need to use at least 3 classes in this project. For instance, you may create one class that represents Users, another one for Ratingsand one more for Data and GUI design.**
- **In order for you to understand better how the GUI should work, please check the GIF image that will be provided with the project documents.**
- You may use **xlrd** module to read Excel files (install it on PyCharm in the same way you did the other modules). Please see the following reference:

  https://www.blog.pythonlibrary.org/2014/04/30/reading-excel-spreadsheets-with-python-and-xlrd/

- For **TreeView**, please see the following reference:

  https://knowpapa.com/ttk-treeview/

- For **ComboBox**, please see the following reference:

  https://stackoverflow.com/questions/17757451/simple-ttk-combobox-demo

- For **Scale/Slider**, please see the following reference:

  https://www.python-course.eu/tkinter_sliders.php

- The following link contains examples of using tkFileDialog.

  https://pythonspot.com/tk-file-dialogs/

- You may use **grid_remove** or **pack_forget** method when implementing Rating/Recommend Page Switching.
  - http://effbot.org/tkinterbook/grid.htm#Tkinter.Grid.grid_remove-method
  - http://effbot.org/tkinterbook/pack.htm#Tkinter.Pack.pack_forget-method

- For the bonus part, you may use Events.
  - http://effbot.org/tkinterbook/tkinter-events-and-bindings.htm

- For the direction of the buttons, you may create a small-width, large-height button with wraplength of 1. Wraplength determines when a button's text should be wrapped into multiple lines.

- Apart from the above, the lecture slides are enough for most of the parts, still feel free to research and implement. Please provide references if you use external sources. TutorialsPoint, StackOverflow and Effbot might be good starting points, but **do not copy codes directly**.

Warnings:

- You **CANNOT** use place for geometry, only grid and pack are allowed.
- Do not talk to your classmates on project topics when you are implementing your projects. Do not show or email your code to others. If you need help, talk to your TAs or myself, not to your classmates. If somebody asks you for help, explain them the lecture slides, but do not explain any project related topic or solution. **Any similarity in your source codes will have serious consequences for both parties**.
- Carefully read the project document, and pay special attention to sentences that involve "should", "should not", "do not", and other underlined/bold font statements.
- If you use code from a resource (web site, book, etc.), make sure that you reference those resource at the top of your source code file in the form of comments. You should give details of which part of your code is from what resource. Failing to do so may result in **plagiarism** investigation. Last but not the least, you need to understand code pieces that you may get some other resources. This is one of the goals of the mini projects.

Even if you work as a group of two students, each member of the team should know every line of the code well. Hence, it is important to understand all the details in your submitted code.

How and when do I submit my project?

- Projects may be done individually or as a small group of two students (doing it individually is **strongly** recommended for best learning experience). If you are doing it as a group, only **one** of the members should submit the project. File name will tell us group members (Please see the next item for file naming details).
- Submit your own code in a single Python file. Name it with your and your partner's first and last names. As an example, if your team members are Deniz Barış and Ahmet Çalışkan, then name your code file as deniz_baris_ahmet_caliskan.py (Do not use any Turkish characters in file name). If you are doing the project alone, then name it with your name and last name similar to the above naming scheme.
    - Those who do not follow the above naming conventions will **get** 10% **off** of their project grade.
- Unlike other MP projects, you are given three weeks instead of two weeks to work on this project. Submit it online on LMS by **17:00 on 27th November, 2018, Tuesday**.

    Late Submission Policy:

    - -10%: Submissions between 17:01 – 18:00 on the due date
    - -20%: Submissions between 18:01 – midnight (00:00) on the due date
    - -30%: Submissions after which are up-to 24 hours late.
    - -50%: Submissions which are up-to 48 hours late.
    - Submission more than 48 hours late will not be accepted.

Grading Criteria?

GUI Design (40)	Reading and Parsing Excel File and Database File and showing them properly in GUI (20)	User-Based Recommendation (20)	Item-Based Recommendation (15)	Appropriate use of Error Handling and Exceptions (5)	BONUS (10)

Your code should be efficient, easy to follow and track. Therefore, from your overall grade, we will deduct points by the specified percentage for the following items:

- Inappropriate/Cryptic variable names and method names (10%)
- Classes and objects are not used properly (30%)
- Insufficient commenting (10%).

Have further questions?:

If you need help with anything, please use the office hours of your TAs and the instructor to get help. **Do not walk in randomly (especially on the last day) into your TAs' or the instructor's offices. Make an appointment first. This is important. Your TAs have other responsibilities. Please respect their personal schedules.**

IMPORTONT NOTES :

Note 1: Plagiarism:

- Zero tolerance
- Cases will be referred to the Ethics Committee
- Both parties (provider and receiver) are responsible
- Process:
    - Automated computerized checks for pre-filtering
    - Human review for confirmation
    - Referral to the Ethics Committee if true positive