# Project 3: GUI for Book Recommendations

CS 1410

## Background

The purpose of this project is to get experience with event-driven programming with Graphical User Interfaces (GUIs). You will use the **breezypythongui** module, which is based on Python's **tkinter** module, to design interactive windows for the Book Recommendation System from Project 1.

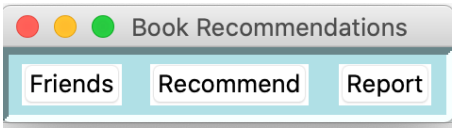
## Requirements

First, make sure your friends and recommend functions from the Book Recommendations project are working properly.

Provide the following options to users, available through simple buttons in a small main window. The corresponding windows/dialogs appear and disappear independently from the main window.

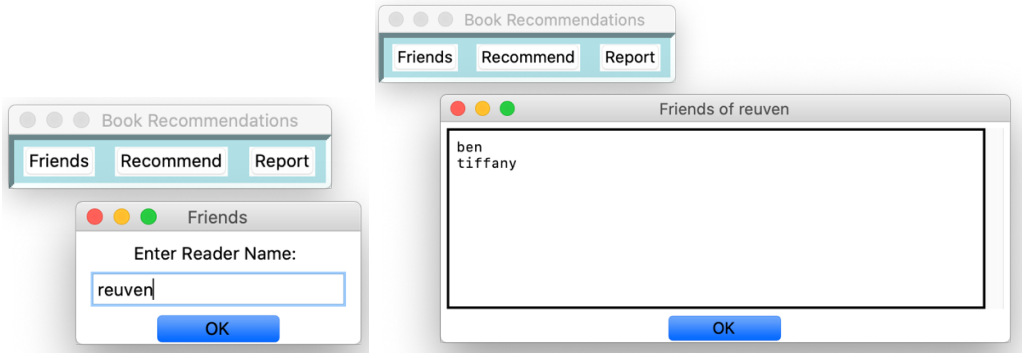
- **Friends**. This option displays a prompter box prompting for a reader name and displays the friends, one per line, in a message box. This calls your **friends** function from the Book Recommendations project.
- **Recommend**. This displays a prompter box prompting for a reader name and then displays recommendations for that user in a message box. Validate that the reader exists in the system. This calls your **recommend** function from the Book Recommendations project.
- **Report**. This option displays a full report with all of the recommendations for all users. The users appear in alphabetical order, and the recommendations are sorted as in Project 1.

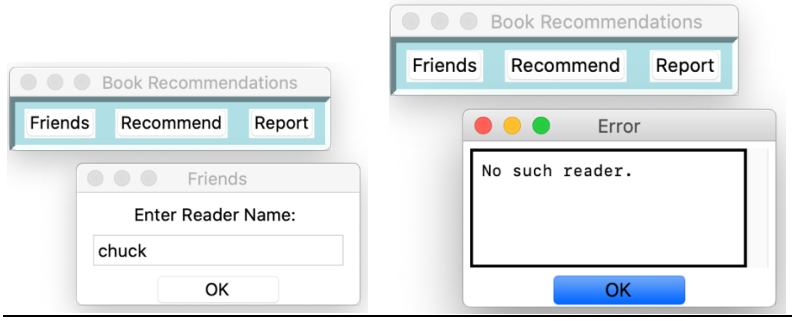
Here is a sample display, using "powder blue" ("#B0E0E6") as the background color:

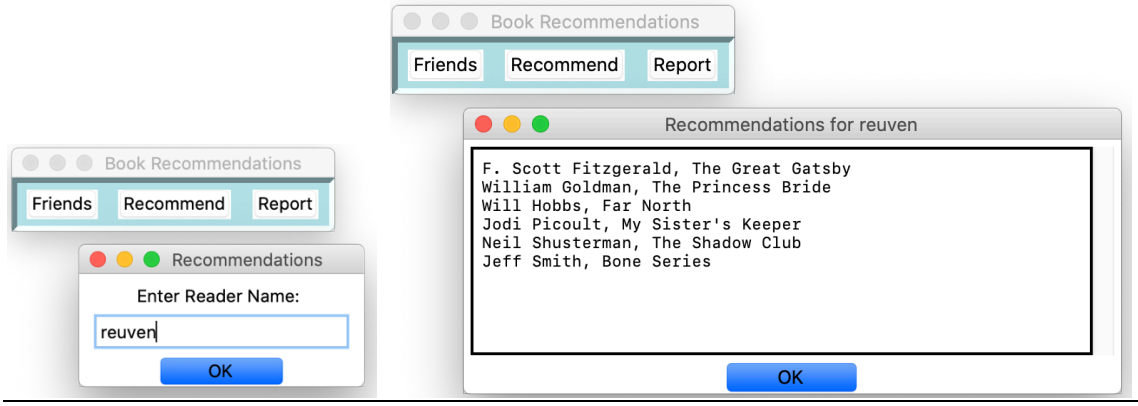


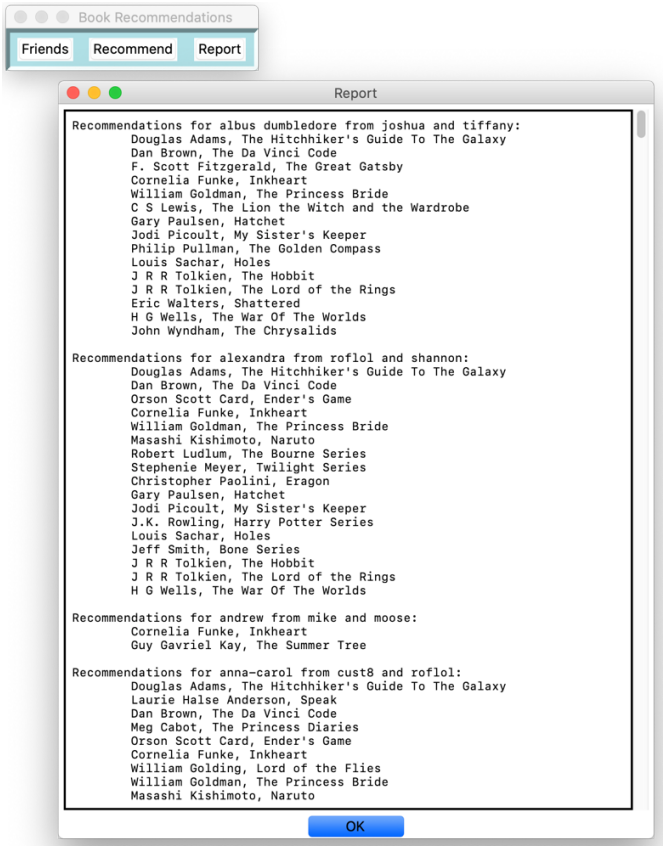

You can find different color codes at https://www.rapidtables.com/web/color/RGB_Color.html.

Here are some suggested sample screens:






## Implementation Notes

Use *breezypythongui.py* to implement your GUI. Make sure to catch invalid names, as shown above.

## FAQs

**Q**. Why did you make this project so easy?
**A**. I was in a good mood. Don't worry—the next project is harder. Plus, some of you may not have had the **friends** and **recommend** functions fully functional in Project 1, so here's your chance to fix that. At least you have to produce a *full report* this time, so that's a new loop you have to write.

**Q**. How do I get the readers in alphabetical order?
**A**. You already know how! Extract the keys from your **ratings** dictionary and call **sorted**.

**Q**. Why do we *import* **bookrecs** instead of just copying the code into a new file for this project?
**A**. Because user interfaces should be *separated* as much as possible from the behind-the-scenes code that processes the data. *Processing* and *presentation* are two different things. The less "coupling" between components, the better. You don't want to copy the code from the **math** module into your code every time you need math functions, right? This is good, modular design.

**Q**. I find GUIs confusing.
**A**. They can be. They certainly are a lot of work. But for this project you only need *one frame* and the GUI methods **addButton**, **prompterBox**, **messageBox**, and **mainloop**. It's only about 40 lines of actual code. Enjoy!

**Q**. Will I ever use **breezypythongui.py** again?
**A**. Probably not. It is unique to this book. It is a very easy way to be introduced to GUIs, though. You can easily pick up **tkinter** later after this.

**Q**. Why did you schedule this project so long after Project 1? I forgot how I did Project 1.
**A**. For a couple of reasons:

1. We needed to cover *classes* first so you could use them here. The book made a mistake switching these topics, IMO.
2. Code **maintenance** is a very important part of software development. It is helpful to learn how to write clear, readable code so you can come back to it later and fix it. So, consider this a "real-world" experience!