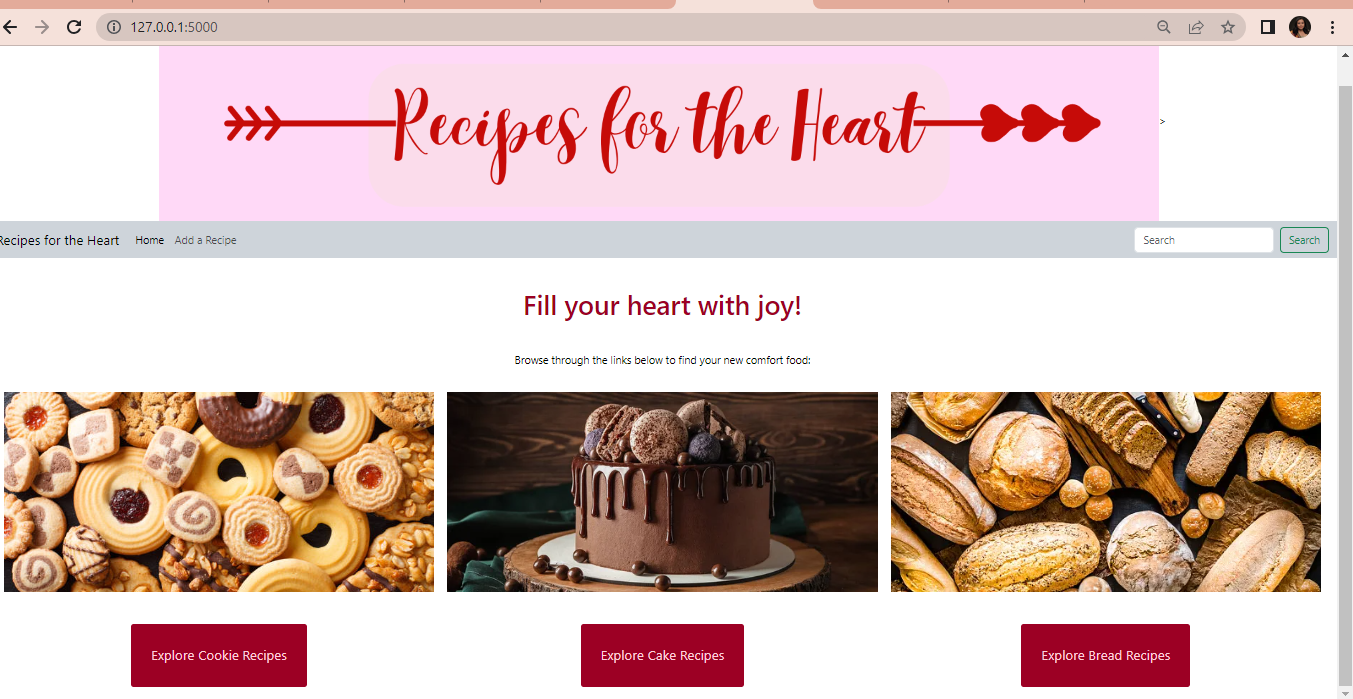
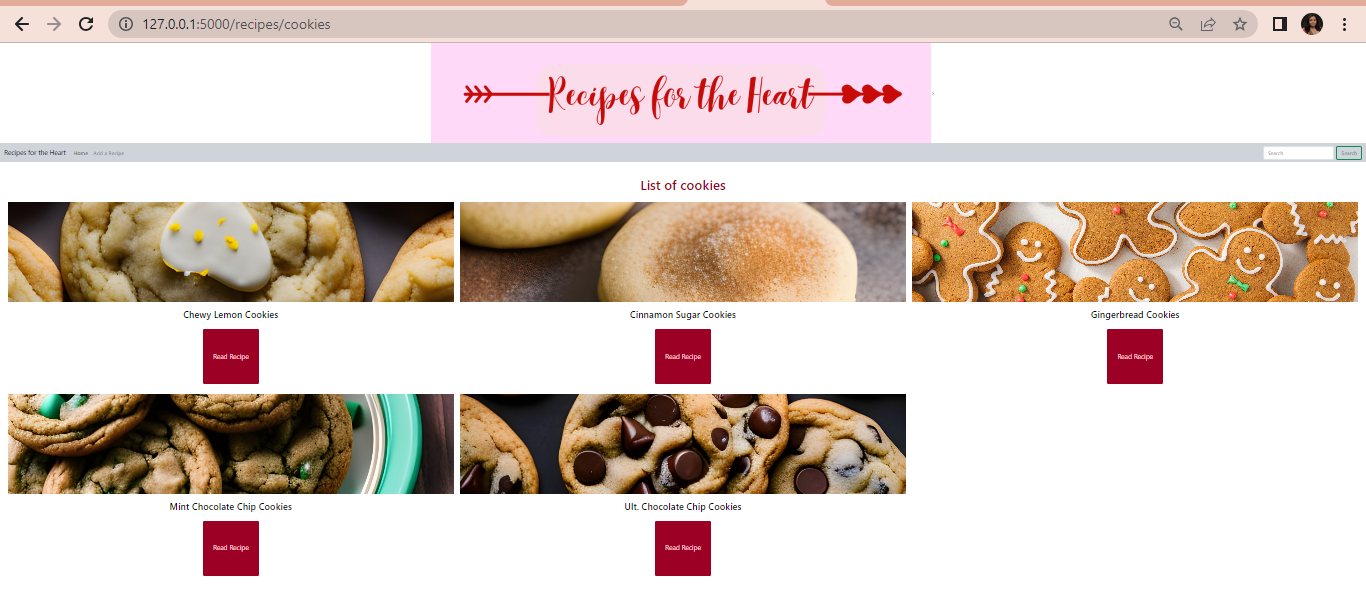
**LIS 161 Project Documentation**

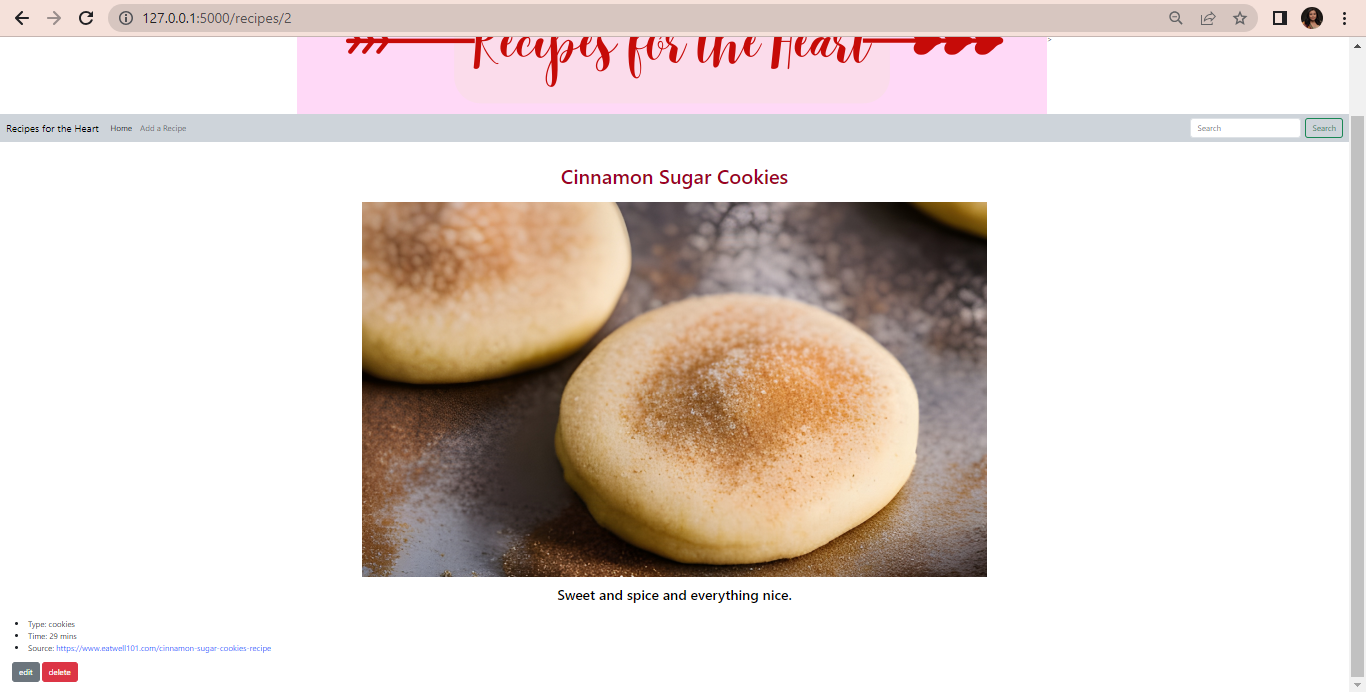
**This is how our web works**

Once you click the link you can access the homepage of our website. You can clearly see the name of our recipe website which is “Recipes for the Heart”. We have three links representing the three major types of pastries we host — you can choose to explore cookie recipes, cake recipes or bread recipes.

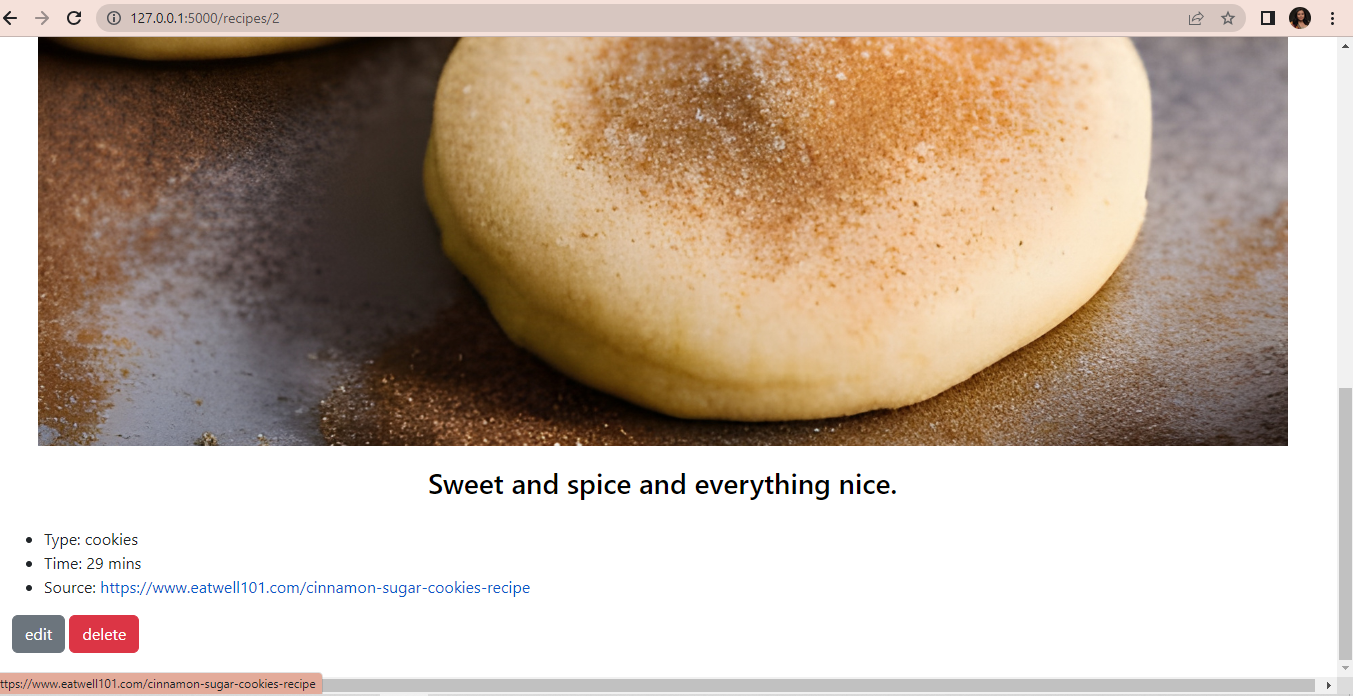


Upon clicking one of the links, you will then be directed to a list of recipes in grid format. For example, if you clicked on ‘Explore Cookie Recipes”, you’ll be directed to a page where you can see different cookie recipes. Each item has a button and a picture, both of which are clickable. 

Upon clicking either the button or the item image, you will be directed to the page listing the details of the item.



Each item’s page reflects the following details: (1) image (2) description of the item (3) their pastry type (4) the time it takes to cook them and (5) the link to the actual recipe page hosted on another website. There is also an “edit” and “delete” button at the bottom.



Our ‘Add a Recipe’ page is where users can add recipes that they love. They are expected to input: type, name (of recipe), time (it takes to cook), link to the image of the final product, and the link to the recipe itself. After inputting the details and clicking the submit button, users will be redirected to the page of the item they added.



Each recipe on our website can be modified by the user. This means that if they made a mistake in the details, or maybe they just want to change the name of the recipe, they can do so by clicking the “edit” button. Clicking that button directs the user to a page similar to add-a-recipe, where they will be freely able to change the pertinent details. Once they hit the ‘update’ button, they are redirected to the page of that item and the changes they made are reflected.



When users click the delete button, the item is deleted and they are redirected to the listing page of that particular recipe type. For example, if they were to delete the item “Valentine Cookies”, all the data about that item will disappear and the user will be directed to the url for the list of cookies.

Our website essentially functions as a directory. We display different recipes and lead the user to the link of the recipes that they are interested in. It is modifiable not only by the developers but the users themselves, so that they may add recipes they’d think other people will be interested in. Users are able to correct the information they input (with the edit button), and in the event that a recipe no longer exists (the source url no longer exists), users will be able to delete the listing or change the source to a similar/better recipe.

**The coding process and deployment**

**Connecting the database**

Routing and connecting the database is the most difficult part of programming. We kept getting a lot of errors, especially with regards to the data not appearing. We figured that the majority of our issues were due to problems in the variable names. For example, recipe should be recipes, and anything calling ID and type should actually be “recipe\_id” and “recipe\_type” in order to be called correctly.

**Designing the website**

Web design is the easiest part of the project. We created a static folder to contain the .css file, and modified the html templates to reflect our design. We linked the css file in the head section of base.html to ensure that the design is inherited across all pages.

**Our deployment process**

The uploaded instructions were not as helpful, so we deviated and deployed it another way.

Instead of using github, we went directly to pythonanywhere and followed the directions below:

* Go to “Web”
* Click “Add a new web app”
* Click “flask” (***not*** manual configuration)
* Choose the latest version (or whichever version is most compatible). We chose python 3.10
* Go to “Files”
* Go to “mysite” folder
* Delete pythonanywhere default file
* Upload: “app.py”, “data.py”, and our database “recipes.db”
* Within, mysite, make new directory: templates
* Upload: all template files from local (index.html, base.html, recipe.html, recipes.html, register.html, update.html)
* Go back to Web and modify WSGI file; change flask\_app to app
* Ensure “source code” and “working directory” have the same link
* Click green reload button
* Click the site link and verify that everything is working smoothly.