Milestone 4

1. `REPORT.pdf` (400-600 words): as before. You can continue to propose revisions to the list of features/goals. NOTICE THE GOALS AND STRETCH GOALS ARE VERY SIMILAR AS LAST WEEK
    a. **Changes/updates to your original proposal (if any).**
    b. **A proposal for the list of features/goals that your final product will deliver and be evaluated on. This will be used as part of your grading criteria. Make these features/goals specific enough, so we can measure success. Divide them into two categories (basic and bonus) and give each of them a relative weight that reflects the amount of work required. Categories and weights affect grading; see Grading Criteria below. There should be at least 5 basic features/goals and 2 bonus features/goals.**
        i. **Stretch Goals**
        1. ML recommendation system
            a. 10/10. This goal speaks for itself. We want to implement an effective recommendation system that provides users with new recipes that they would actually cook, which results in more engagement.
        2. Search system for both recipes and users
            a. 9/10. A fast search system is hard to implement. Although it would be helpful for our platform, it will require more work. To start with we can just make this with simple SQL queries. For instance, if I search for "tacos", the search API can just perform a SQL query that finds all recipes including the word "tacos", or something like that. A more advanced approach could involve AI, or integrating with Amazon's Elastic Search service.
        3. Scrape and parse a bunch of online recipes
            a. 10/10. Ideally, this recipe app will be able to build a starting database with current recipes that are already on the internet, essentially building a bank of current recipes. In an ideal case, we would be able to implement some web crawler that automatically finds and parses recipes into the recipe database. These ones will also be categorized automatically (hopefully).
        4. Leaderboard of Users
            a. 6/10 This will require a web page which displays the top 50 or so users ranked by some criterion, such as how many likes they have gotten. Time permitting, we can make that ranking criterion more sophisticated, but it will be fairly simple to make the basic leaderboard. To speed up the leaderboard data fetching, we can just find and cache the top 50 users maybe once a day rather than recomputing it each time a user goes to the leaderboard page.
        5. Bonus Feature: Hosting Flask Server on AWS ElasticBeanstalk
    c. **Summary of progress so far, e.g., components built, tasks completed.**

i. Tasks Completed Before:
    1. Setup PostgreSQL database on AWS RDS
    2. Wrote scripts to set up tables based off completed database schema for application.
    3. Created wireframe in Figma to produce frontend design and frontend logic
    4. Setup Flask API Server
    5. Setup React Web App framework and connect to Flask API
    6. Basic Recipe Creation logic and corresponding frontend page
    7. Implement Feed logic
    8. Implement Search logic
    9. Implement User login/signup logic
ii. Tasks Completed this week
    1. Implement Image logic for posts/profile with AWS (Setup AWS S3 bucket for storing images, add images to create recipe logic**)**
    2. Implemented User data logic after creating profile in settings (to set other user information outside of email/password)
    3. Implemented user session logic and updated previous logic to be user-specific
    4. Improved frontend design for posts, create profile (including tags for posts)
    5. Implemented profile page
    6. Created large dataset of recipes/users from collected dataset
    7. Feed Pagination / load more

d. **This week's end-to-end tasks**
i. Frankie's User Data
    1. Implemented user data logic to fill in the rest of the Users table with other user information. Added logic to ensure that certain fields are required and that the phone number is a real phone number. Can also be accessed at /settings to update the Users entry. Ensured compatibility of previous features with user sessions
ii. Frankie's Create Profile
    1. I also updated the frontend to include tag creation and improve the general design of the create profile page
iii. Keith's Image AWS S3 Bucket
    1. Created S3 bucket to allow us to create and retrieve posts with uploaded images. Also wrote backend code to upload images to the bucket, and frontend code to fetch images from the bucket.
iv. Keith's Feed Pagination
    1. Took the existing feed end-to-end feature, and then made a load more button on the front end with a corresponding API to load more data onto the feed.
v. Louis's User Server-Side Sessions

1. Implemented User session logic with Redis and Flask Cors. Worked with teammates to make everything backwards-compatible and make posts user-specific. Sessions also helped enforce users to be logged in for recipe creations.

vi. Louis's scripts to fill DBs with production-size data
   1. Created two scripts, UploadUsersScripts.py and UploadRecipesScripts.py,
      a. UploadUserScripts creates 1000 random users to fill in Users DB
      b. UploadRecipesScripts uses this dataset https://www.kaggle.com/datasets/pes12017000148/food-ingredients-and-recipe-dataset-with-images, to fill in Recipes, RecipesHasIngredients, and RecipesHasSteps. This dataset has around 10k recipes (rows). In the dataset, there were recipes with images, instructions on how to create it, and ingredients of the recipe. First, we had to process the image through PIL and uploading to S3. Next, we upload the recipe to the Recipes table, which includes pathToImageS3, title, postedByUserId, etc. We chose a random number from (1,1000) for userId. Then we parsed through instructions and ingredients to upload to RecipesHasIngredients and RecipeshasSteps respectively.

vii. Herb's Profile page
   1. Made the profile page, which displays the user information and posts for a given user based on what is stored in the database.
   2. Also improved the design of the posts page

e. **A list of tasks to be completed before the final due date.**
   i. Clean up UI/UX for Posts based on Figma
   ii. Further User logic with preferences
   iii. Add likes, comments, saved logic for posts
   iv. Implement a friends and followers logic
   v. Implement a rating implementation, including cost rating, difficulty rating, and time rating.
   vi. Further feed logic based on tags/categorization for recipes/profile preferences
      1. Stretch goals: Custom feeds based on filters/sorts, machine learning