**Project 3 readme**

<https://git.generalassemb.ly/SEI-66-Online/SEI-66/blob/main/Projects/Project-3/readme.md>

## **Technical Requirements**

* Create the application using at least 2 related models, one of which should be a user
* Include all major CRUD functions for at least one of your models
* Add authentication AND authorization (page protection) to restrict access to appropriate users
  + User must be able to sign up or login
  + Signed in user must be able to change password and logout
  + change password and logout must only be available to logged in users

(https://docs.djangoproject.com/en/4.1/topics/auth/)

* Give feedback to the user after each action, and after form submissions with success/failure (email already exists etc, password doesn't meet requirements)
* Clear forms after submission failure.
* Manage team contributions and collaboration using a standard Git flow on Github
* Layout and style your front-end with clean & well-formatted CSS, with or without a framework (bootstrap/materialize). Put effort into your design!
* Deploy your application online so it's publicly accessible. (We will walk through the setup. Make sure you guys have same directory structure as I have for cat collector. Initialize git when you are inside you project folder, check cat collector for reference.)

### **Stretch Goals**

* Allow user to change website theme, Dark mode etc.
* Include Pagination.
* Utilize 3rd party API's.
* Sent verification email upon registration (Django contains this feature)
* Allow users to upload image files.
* Password reset using an email. (Django contains this feature)

**Necessary Deliverables**

* Projects are due on Friday, 07th Oct 2022 at 10.30am!
* You have to fill the [google sheet](https://docs.google.com/spreadsheets/d/1aomZtNggeeAoQ7wzkJ5slmp-gLpWmdW5/edit#gid=1044180768) with you name, github link and deployed link.
* A working app, hosted somewhere on the internet
* A link to your hosted working app in the URL section of your Github repo
* A git repository hosted on Github, with a link to your hosted project, and frequent commits dating back to the very beginning of the project
* A readme.md file with:
  + An embedded screenshot of the app
  + Explanations of the technologies used
  + A couple paragraphs about the general approach you took
  + Installation instructions for any dependencies
  + Documentation of your application architecture
    - Entity Relationship Diagram (ERD) or Data Model.
    - Your user stories – who are your users, what do they want, and why?
    - Your wireframes – sketches of major views / interfaces in your application
  + Descriptions of any unsolved problems or major hurdles you had to overcome

## **Necessary Deliverables**

* Projects are due on Friday, 07th Oct 2022 at 10.30am!
* You have to fill the [google sheet](https://docs.google.com/spreadsheets/d/1aomZtNggeeAoQ7wzkJ5slmp-gLpWmdW5/edit#gid=1044180768) with you name, github link and deployed link.
* A working app, hosted somewhere on the internet
* A link to your hosted working app in the URL section of your Github repo
* A git repository hosted on Github, with a link to your hosted project, and frequent commits dating back to the very beginning of the project
* A readme.md file with:
  + An embedded screenshot of the app
  + Explanations of the technologies used
  + A couple paragraphs about the general approach you took
  + Installation instructions for any dependencies
  + Documentation of your application architecture
    - Entity Relationship Diagram (ERD) or Data Model.
    - Your user stories – who are your users, what do they want, and why?
    - Your wireframes – sketches of major views / interfaces in your application
  + Descriptions of any unsolved problems or major hurdles you had to overcome

## **Suggested Ways to Start**

* PLAN PLAN PLAN and then PLAN SOME MORE before you write any code make sure that you and your team have thoroughly planned your project. Almost every serious headache you run into later is a result of inadequate planning.
* Don’t hesitate to write throwaway code to solve short-term problems.
* Read the docs for whatever technologies, frameworks, or API’s you use.
* Be consistent with your code style. You're working in teams, but you're only making one app per team. Make sure it looks like a unified effort.
* Commit early, commit often. Don’t be afraid to break something because you can always go back in time to a previous version.
* Keep user stories small and well-defined. Remember – user stories focus on what a user needs, not what development tasks need accomplishing.
* Write code another developer wouldn't have to ask you about. Do your naming conventions make sense? Would another developer be able to look at your app and understand what everything is?
* Make it all well-formatted. Are you indenting, consistently? Can we find the start and end of every div, curly brace, etc.?
* Comment your code. Will someone understand what is going on in each block or function? Even if it's obvious, explaining the what & why means someone else can pick it up and get it.
* Write pseudocode before you write actual code. Thinking through the logic of something helps.

## **Useful Resources**

* [Git Team Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows)
* [Git Team Cheatsheet](https://jameschambers.co/writing/git-team-workflow-cheatsheet/)
* [Django documentation](https://docs.djangoproject.com/en/3.1/)
* [Django REST Framework](https://www.django-rest-framework.org/)
* [PostgreSQL documentation](https://www.postgresql.org/docs/)

—-------------------------------------------------------------------------------------------------------------

# 

# **Project 3 -**

## **Description**

This project was assigned ...

## **Deployment link**

## **Getting Started/Code Installation**

### **Instructions**

## **Timeframe & Working Team**

## **Technologies Used**

## **Brief**

With this project, we were given the brief to

Our project had to:

* BONUS - Allow user to upload a profile photo.
* BONUS - Allow user to upload method by taking photos.

At the end of the project we then had to deliver a presentation on our project covering the following points:

## **Planning**

### **Task management**

### **Development Overview**

1. Brainstorm:
2. Pick an idea
3. Wireframe and user stories
4. Create repo and starting files (.js, .css., .html)
5. Break the process down by requirements
6. Basic layout
7. Structure logic and begin working on functions
8. Test functions
9. Touch up styling
10. Test site
11. BONUS: Add extras

## **Build/Code Process**

Documented below are the stages we went through in order to build the code for our project.

### **Stage 1: Brainstorming**

### **ERD**

### **Wireframes**

### **User Stories**

## **Challenges**

### **Problem-solving**

## **Wins**

### **Favourite functions**

## **Key Learnings & Takeaways**

## **Bugs**

## **Future Improvements**

—-----------------------------------------------------------------------------------------------------------------------

**1st meet up:**

**Einar & Dan**

Potential Ideas:

**Green social network??**

Profile/Blog page/Hints & tips/Twitter character limit/Link to shops + reviews/Green hints at tips

**Post-it note application**

Effectively an interactive to do list? Users can add a post-it note, move it to in-progress, move it to a completed pile etc. Could link in that image api so if someone has a long list of notes/things to do they can just take a photo and we can take the text from that picture? You could share/assign notes/tasks to other users etc

[**https://apilayer.com/marketplace/image\_to\_text-api**](https://apilayer.com/marketplace/image_to_text-api)

**Revision tool??**

Upload notes and search by word across all pages???

**JOURNEY COLLECTOR**

<https://www.climatiq.io/passenger-vehicle-carbon-emissions>

**\*\*Most APIs dedicated to businesses as opposed to individuals - supply chain and logistics as opposed to weekly food shops etc.\*\***

<https://developers.carboncloud.com/v/0_11_0#/http/models/structures/ingredient-mix>

(opens everything up in json but we are using SQL)

[**https://docs.lune.co/**](https://docs.lune.co/)

<https://www.trycarbonapi.com/#zigzag>

<https://www.thestartupfounder.com/best-apis-to-get-data-of-co2-emissions-in-2022/>

**Food receipt app?**

Photo to text api??

How accurate does data need to be? Surely if we can link some of it to an api?

<https://www.climatiq.io/explorer?category=Food%2FBeverages%2FTobacco&region=GB>

Too over reliant on APIs??

Upload receipt and calculate carbon footprintCarboncloud api.

<https://developers.carboncloud.com/v/0_11_0#/http/models/structures/ingredient-mix>

(opens everything up in json but we are using SQL)

[**https://apilayer.com/marketplace/image\_to\_text-api**](https://apilayer.com/marketplace/image_to_text-api)

[**https://apilayer.com/solution/food-grocery**](https://apilayer.com/solution/food-grocery)

<https://www.thefloopapp.com/> ??? Unable to locate API

Upload receipt and calculate carbon footprint

Carboncloud api.

<https://developers.carboncloud.com/v/0_11_0#/http/models/structures/ingredient-mix>

(opens everything up in json but we are using SQL)

Carbon calculator by recipe/ingredients

<https://carbonfootprintcalculator.cdrc.ac.uk/>

**Einar found below:**

**https://www.greenswapp.com/consumers**

—-------------------------------------------------------------------------------------------------------------

**Recipe app with C02/nutrition figures?**

Photo to text api??

How accurate does data need to be? Surely if we can link some of it to an api?

<https://www.climatiq.io/explorer?category=Food%2FBeverages%2FTobacco&region=GB>

<https://apilayer.com/marketplace/image_to_text-api>

<https://www.foodemissions.com/Calculator>

Image cropping api??

Calculating nutritional info???

Users updating carbon data library?? User specific; footprint per kilo/gram and scale as necessary

Required field for adding ingredients is c02 emissions.

Picture of finished meal.

User could upload food receipt

User can upload photos of ingredients from recipe books

Git - create branches; each focus on a feature, upload to individual branches.

Then joint merge to make sure not overlapping.

Methodology page to explain thinking - to give credibility.

CRUD

C - user creation, image uploads, recipes creation

R - view recipes and C02 figures/nutrition. Ability to save and share.

U - edit recipes, ability for users to select ingredients from list they’ve previously used

D - delete user, delete recipes

<https://www.cleanmetrics.com/ToolsDatabases>

RESOURCES FROM SAAD for authorisation/authentication

<https://dev.to/earthcomfy/django-user-profile-3hik>

<https://www.devhandbook.com/django/user-profile/>

**Wireframe - Dan**

<https://www.pinterest.co.uk/#top>

Agreed on Pinterest style homepage.

<https://iscrolljs.com/> library for pinterest style scrolling

<https://github.com/alvarotrigo/fullPage.js>

<https://github.com/alvarotrigo/fullPage.js/wiki/Extension---Parallax> - effects

**ERD - Einar**

**Einar to collate C02/nutritional value**

**Dan to work on basic files/folders**

TRELLO BOARD - NAG!!!!