BOOTS: SOPHIA C, EMAAN A, JUN JIE L
DESIGN DOCUMENT

TARGET SHIP DATE: 2025-12-22

APP OVERVIEW:

Home Sweet Home is a murder mystery game where the goal is to guess who the murderer is. The website is designed like a house, with interactive features and clues posted throughout each room for the user to explore. To enter the house, the user needs to register and log in. At the end of the exploration, you will need to figure out the murderer to win. Found clues can be tracked on the homepage. After all clues were explored, a question will appear that will ask for who the murderer is.

TASKS/ASSIGNMENTS:

Sophia C. [PM & Frontend Developer]:

- maintain devlog
- create html templates using jinja:
  - startpage.html (home)
  - login.html
  - logout.html
  - register.html
  - map.html
  - bedroom.html
  - kitchen.html
  - livingRoom.html
  - Frontlawn.html
  - settings.html
- write all css
- ensure navigation links work between pages

Emaan A. [Middleware]:

- initialize flask app
  - configure sessions
  - handle APIs
- user registration/login/logout
- Set up routes for html pages
  - /home, /bedroom, /kitchen, /livingroom, /settings, /login, /logout, /create
- requirments.txt

Jun Jie L. [Backend]:

- database organization (tables)
- write SQLite queries for authorization/login/logout

APIS:

411_on_FreeRecipeAPI.md (Recipe Book Kitchen)

411_on_JokeAPI.md (Joke Book on Shelf in Bedroom)

411_on_MoviesAPI.md (Movie Shelf in living room)

411_on_HolidaysAPI.md (Calendar in Kitchen Wall)

411_on_TheCatAPI.md (cat album in living room)

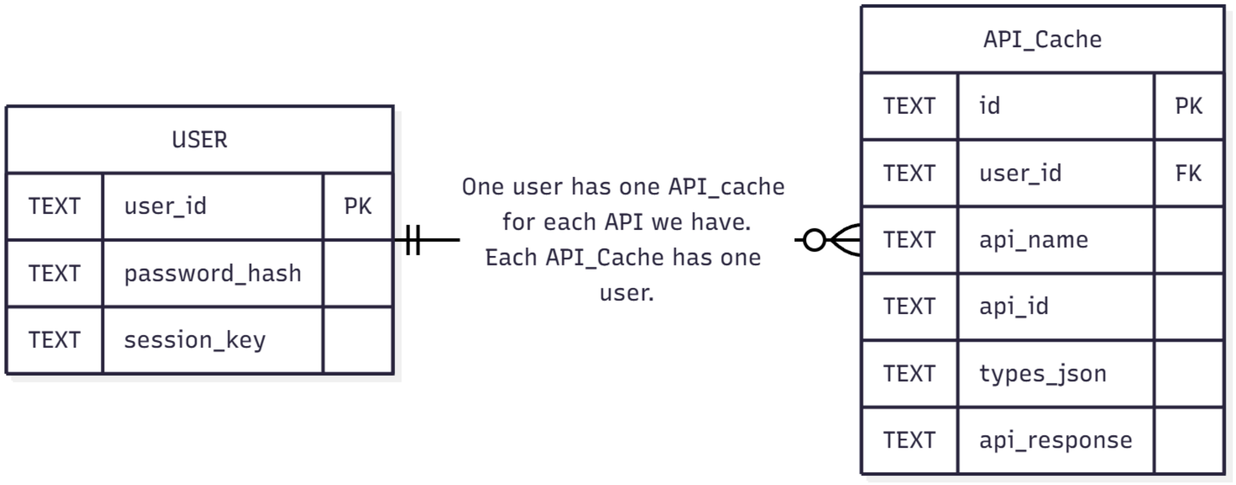
https://api.thecatapi.com/v1/images/search?has_breeds=1

411_on_The_Metropolitan_Museum_of_Art_Collection_API.md (Museum Book on Shelf in Bedroom)

FRONTEND FRAMEWORK:

We are using Foundations Framebook for our Frontend.The features that will be included will be Buttons, Dropdowns, Slider, Switch, Menu, Dropdown Menus, Accordion Menus, a Top Bar, Section Alignment, Coloring, Icons, and other Containers like Cards, Images, and Reveal. We used Foundations because its features suited our styling the best.

DATABASE ORGANIZATION (TABLE1S):



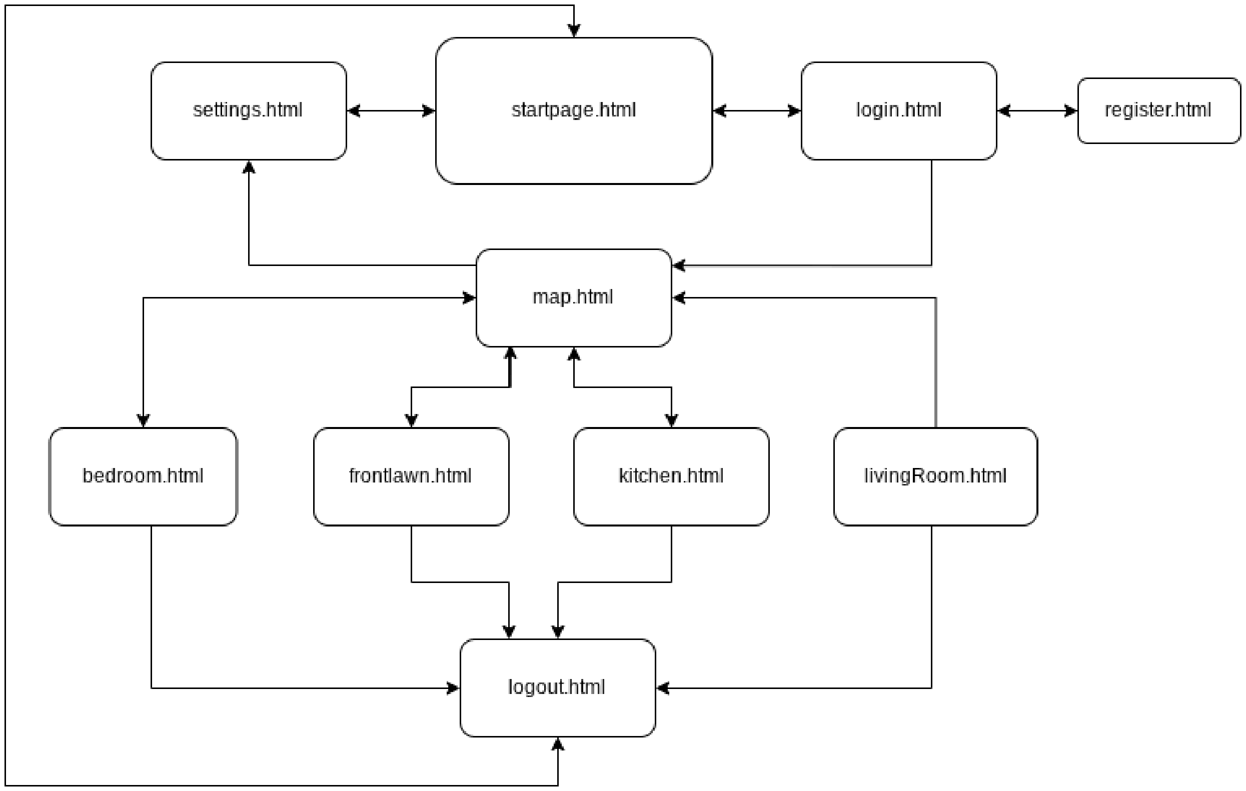
USER		
TEXT	user_id	PK
TEXT	password_hash	
TEXT	session_key	

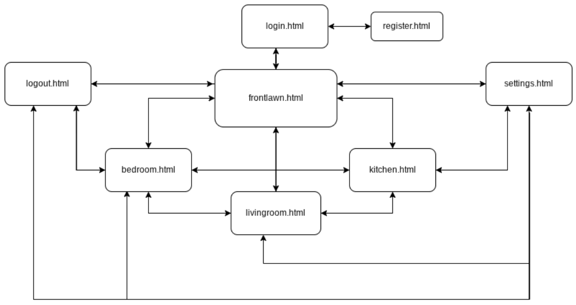
One user has one API_cache for each API we have. Each API_Cache has one user.

API_Cache		
TEXT	id	PK
TEXT	user_id	FK
TEXT	api_name	
TEXT	api_id	
TEXT	types_json	
TEXT	api_response	

(tables not up to date)

SITE MAP


settings.html
startpage.html
login.html
register.html
map.html
bedroom.html
frontlawn.html
kitchen.html
livingRoom.html
logout.html


login.html
register.html
logout.html
frontlawn.html
settings.html
bedroom.html
kitchen.html
livingroom.html

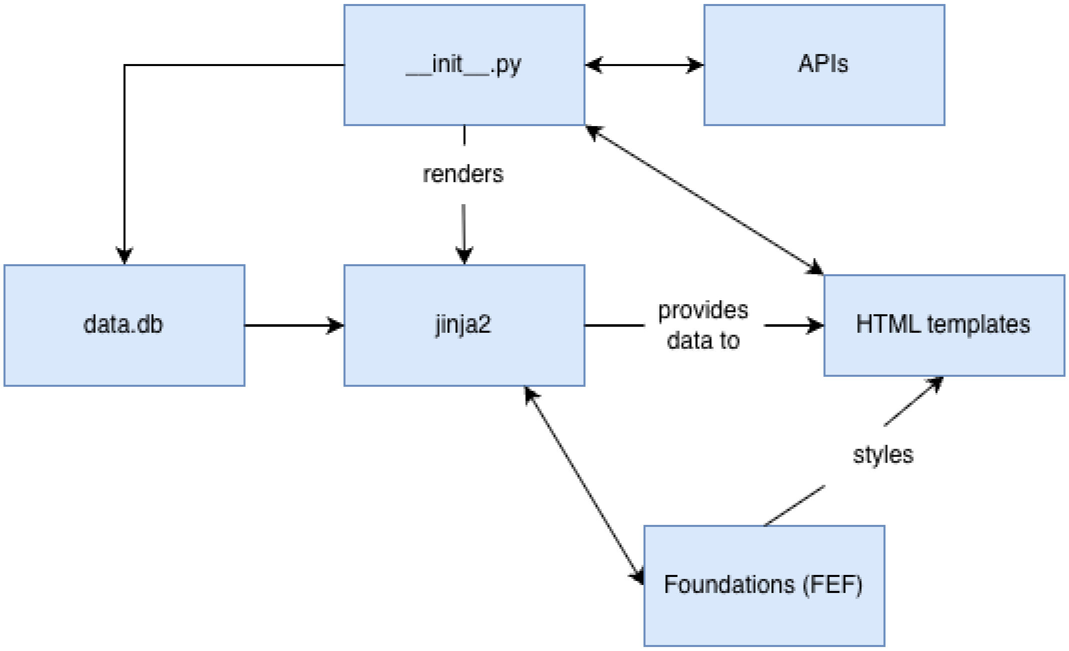
COMPONENT MAP:

__init__.py: configures flask app, makes routes, handles authorization/login, builds database, handles all api calls

frontend framework/css styles html templates

jinja2 reads database and turns data to webpage

data.db: SQLite3 database


__init__.py
APIs
renders
data.db
jinja2
provides data to
HTML templates
styles
Foundations (FEF)