TNPG: M'ykolchywiczk
PM: Karen Shekyan (Grape), Sadi Nirloy (Jeevis the S.K.), Gabriel Thompson (Moony), Brian Wang (Mahi)
P5 (FINAL) doc
Softdev Pd 8

Target Ship Date: Tuesday June 13, 2023

Site: AI prompt determinant Gartic Phone (Call of Gart'thulu)
Goal: Players will take turns drawing an image based off a previous image
Stretch: Players will take turns drawing an image based off a prompt generated by an AI from the previous image

Major Considerations:

- **Database (or lack thereof)**: A more .io style pick-up browser game has no need for logins, and saving images outside a file system is unnecessary (and ill advised!). No Mongo, no SQL.
- **Image Manipulation:** Images will be stored as individual pixels in a 2d array, and converted into a jpg for feeding into the AI through Pillow.
- **Interpreting AI prompts:** AI prompts returned are rather bare and require no interpretation or sanitizing.
- **Drawing/Game time:** How long are games expected to go on?
- **Initial prompt:** How is the first prompt determined?
- **Groupings:** groupings will be created through a room code.
- **Room Code Conflict:** Room codes will be created with the unique request.sid provided on a user basis, chosen from the group leader.

FE:

- Bootstrap**:**
    - Why: Our team unanimously agrees that bootstrap looks the best and is easiest to use
    - How: We will be using the Bootstrap grid to organize the web page. This way, our website will be accessible on mobile.
- HTMLs:
    - create.html
    - end.html
    - game.html
    - game_timer.html
    - join.html
    - landing.html

- room.html

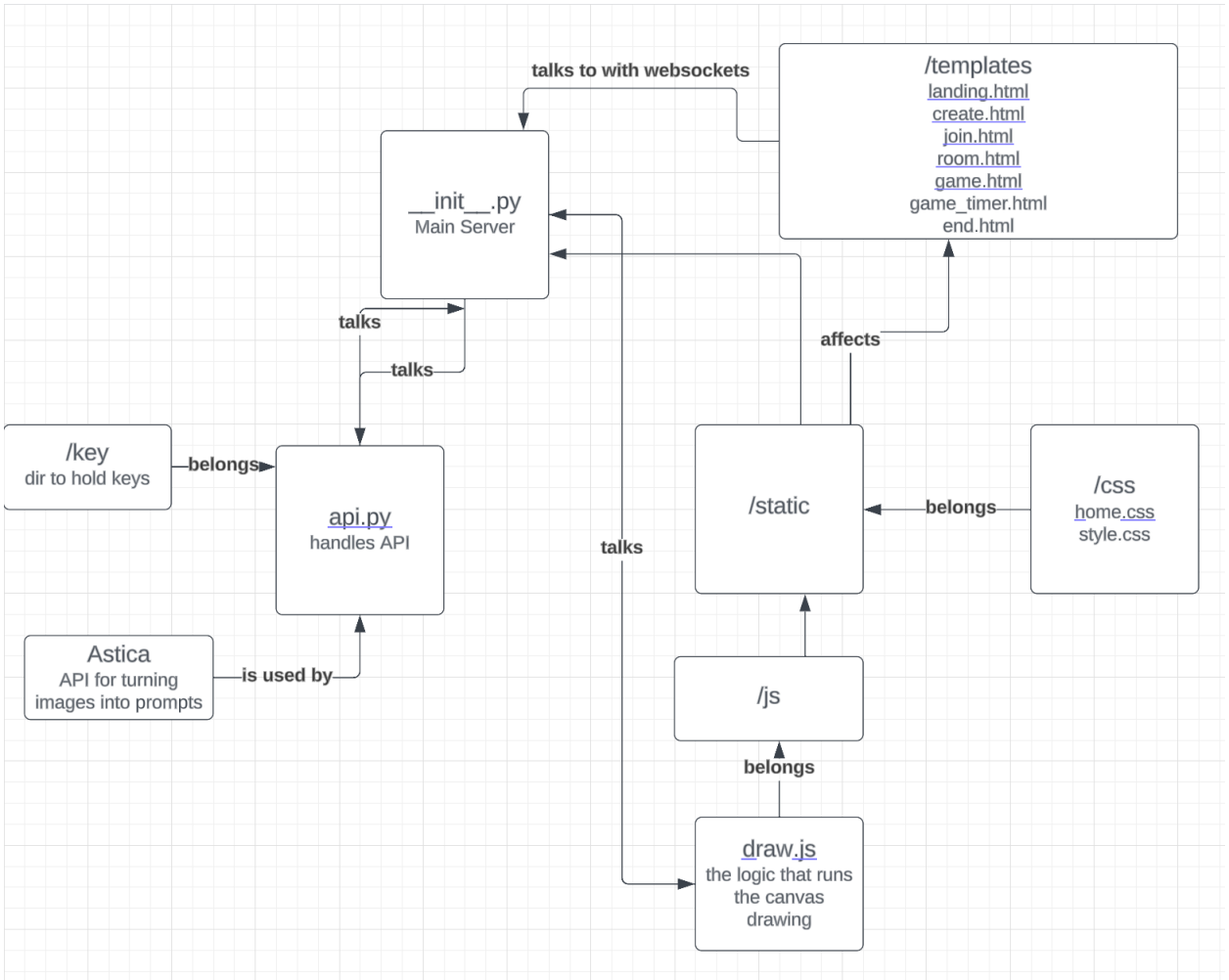
API: Astica (https://www.astica.org/api-docs/asticaVision/): AI image recognition to create a prompt based off an image, will be used to create the next player's prompts
Methods required:

- **TAKES** a jpg image && **RETURNS** a string description of the image

Flask: Using the flask-socketio library to handle "rooming" users together and to take user drawings from the site, turn them into JPEGs, feed them to the API, and return a new prompt.

- __init__.py: Skeleton of project, handles the user
- api.py: File that utilizes the aforementioned Astica AI to generate image prompts
- images.py: Contains utilities for converting/storing images

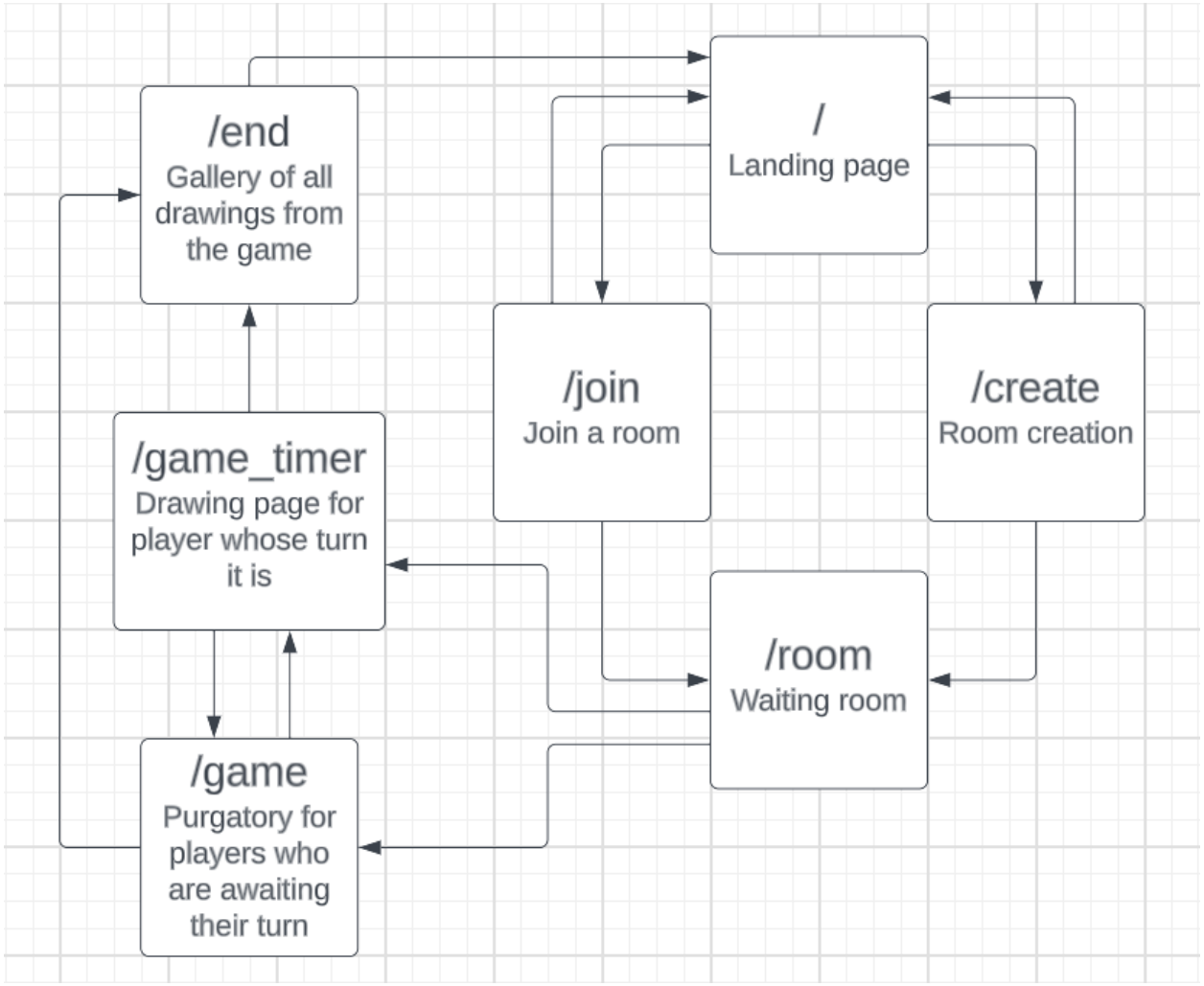
Components:
(The arrow that points to __init__.py comes from the user decides to run __init__.py)



Description of diagram:

- **__init__.py** is the server used to host the HTML. It renders the **templates**. It also imports **api.py** to access the Astica API
- **api.py** is the file that calls the **Astica** API, given the API **key** which is stored as an environment variable
- **HTML** accesses **CSS** and **JS** files to make the game interactive and look pretty

Site:



Breakdown of roles (not strict):

- Flask/Sockets: Sadi and Karen and Gabriel
- API: Brian
- HTML/CSS: Gabriel

Major parts of project:

- Figuring out how to get data to/from the API
- Figuring out how to use flask-socketio

    - First, just getting the module working *at all*. Then, actually getting it to work for our use case. This will probably be pretty tricky because websockets seem complicated
- Figuring out the layout for each webpage, and how it should connect to the back-end