Christine Chen, Naomi Kurian, Ethan Cheung, Owen Zeng
WheelOfFortune
SoftDev
P01 – ArRESTed Development
2025-12-02
TARGET SHIP DATE: 2025-12-22

Design Document

IDEA: A collection of mini-games that allow users to collect different creatures and increase their level through XP accumulation.

PROGRAM COMPONENTS

Flask Application
- Handles routing for all pages
- Manages user session and login/logout
- Connects to database and loads API keys through API helper functions
- Renders pages with Jinja templates

API Module
- Handles communication with external APIs
- Functions which handle errors, key loading, and JSON parsing for each external API

Front-End Framework
- Provides responsive layout and styled components

Pages

Home page
- Only visible for logged in users, if not authenticated, send straight to login page
    - Can be done by running authenticate()
    - Contains a list of all the minigames
- *Navbar* will contain a logout, profile, and rewards button
- *Grid of buttons* that lead to different mini games

Login Page
- Asks for username/password and matches it to what is contained in the *user base*
    - Takes the username and compares it to existing *users*
        - If username does not exist **OR** if username exists but the password does not match the one attached to the corresponding *User*, output 'username or password incorrect'

        - If username exists AND password matches, sends the user back to the *home page* and adds the username to the session
- Contains a button to create an account (sends to registration page)
    - Once account is created, adds the new *user* to the *user base* and signs them in (sending them to *home page* and adding info to the session)

Registration Page

- Username, password, and confirm password form response box.
    - If submit is pressed and it does not pass the comparison check, the page will display a "passwords do not match" message
    - If submit is pressed and the username is the same as another one (case sensitive), the page will display a "username taken" message
    - If submission passes all checks, user will be sent back to the login page and the new user will be added to the user base
- Includes a button (back button or return to login button) that sends the user back to the login page in navbar

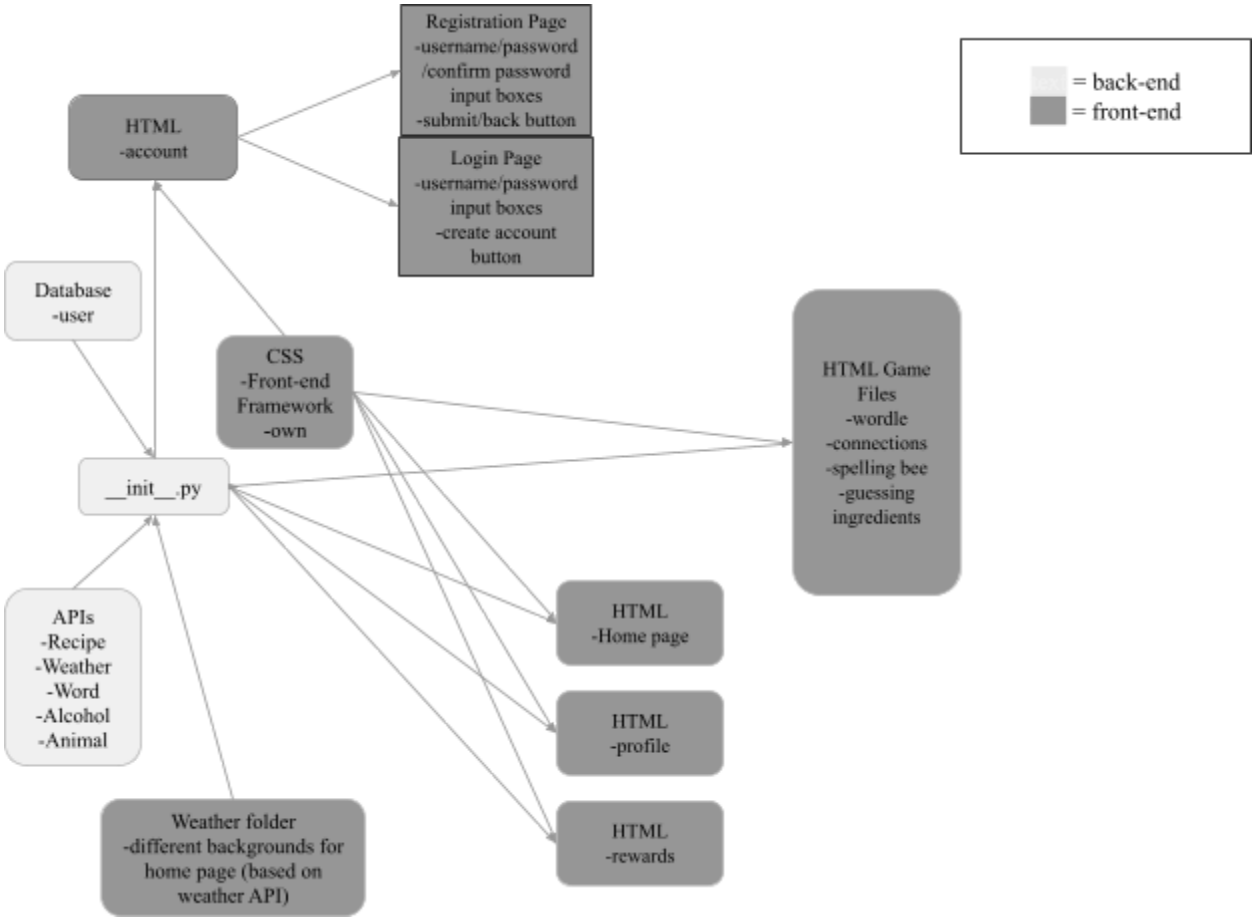
Game Pages

- Different HTML file and corresponding styling for each mini game
    - Games include: spelling bee, wordle, connections, and recipe/ingredient guesser
- Uses corresponding APIs (in corresponding function in __init__.py)
- Navbar containing back button
- Input boxes for users to submit guesses to the game (corresponding function checks accuracy and re-renders HTML file or redirects to home page with award based on performance)

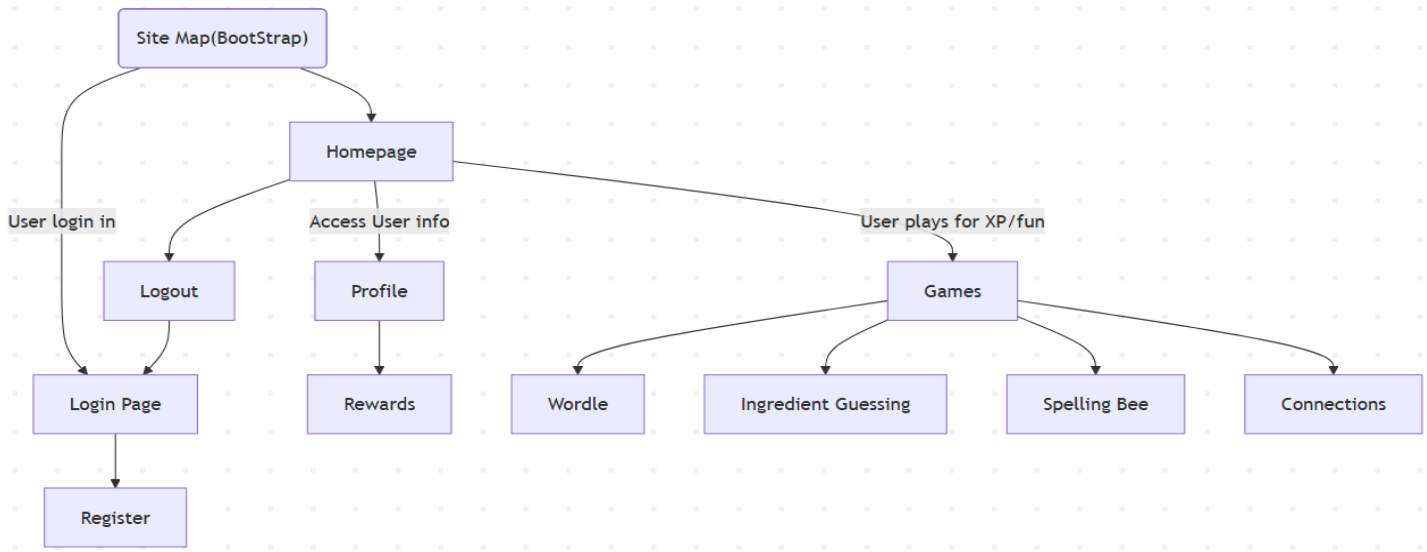
Rewards Page

- Grid with different acquired creatures (stored in database in creatures table)
    - Buttons for leveling up (only works with enough XP)
- Button with pic of egg (new creature)
    - Can only hatch with enough XP (stored in database)
- Displayed XP; user gains eggs when leveling up

COMPONENT MAP



SITE MAP



DATABASE ORGANIZATION

users

INTEGER	user_id	PK
TEXT UNIQUE	username	
TEXT	password	
INTEGER	exp	
INTEGER	level	

creatures

INTEGER	creature_id	PK
INTEGER	user_id	FK
TEXT	name	
TEXT	rarity	
INTEGER	exp	
INTEGER	level	
TEXT	image_path	
TEXT	status (for egg or hatched)	

TASK BREAKDOWN:

Task	Member	Deadline	Status
Implement SQLite database schema and database helpers	Christine	TBD	Incomplete
Set up Flask backend: configure app, routes for all pages, handle GET/POST requests, and render templates	Naomi, Christine	TBD	Incomplete
Implement user registration, login, logout, and sessions	Ethan	TBD	Incomplete

Wordle Logic	Owen	TBD	Incomplete
Connections Logic	Christine	TBD	Incomplete
Spelling Bee Logic	Naomi	TBD	Incomplete
Ingredients-Guessing Logic	Ethan	TBD	Incomplete
Implement Reward System	Owen	TBD	Incomplete
Deal with APIs and their outputs, as well as any logic that may come with using them.	All	TBD	Incomplete
Construct site layout and navigation using Bootstrap	All	TBD	Incomplete

APIs

- Weather API: used for backgrounds based off of weather
- Booze API: used in tandem with recipe API for ingredient guessing
- Recipe API: used in tandem with booze API for ingredient API
- Random word API: used for wordle

FRONT-END FRAMEWORK - BOOTSTRAP

- Grid system and card components will be implemented to present various mini games
- Grid system and card components will be implemented to present various creatures
- Navbar component for convenient navigation across pages
- Forms and buttons for registration/login or other user input
- Image and utility classes used to style images of creatures
- Progress bars to display egg hatching progress, creature level progression, and user level
- Alerts/modals for feedback messages such as invalid login or level up notifications