Project01: Makers Makin' It, Act I
TNPG: Magical Magnolias
Roster: Nia Lam, Amanda Tan, Naomi Lai, Kishi Wijaya
TARGET SHIP DATE: 2025-01-15

Magnolia Gardens - DESIGN DOCUMENT v.1

Description

Magnolia Gardens is a game about maintaining a garden of flowers. Users may plant flower seeds– each species has slightly different growth requirements. Flowers must be watered each day to grow. Water is infinitely available in the universe of the garden. Once a flower meets the requirement for days watered, it has bloomed and may be "picked" (transferred to the user's inventory).

Another resource, magic power, automatically makes the flower bloom and is obtained from completing minigames of basic and simple arithmetic. Magic power may also be exchanged for flower seeds in the shop page.

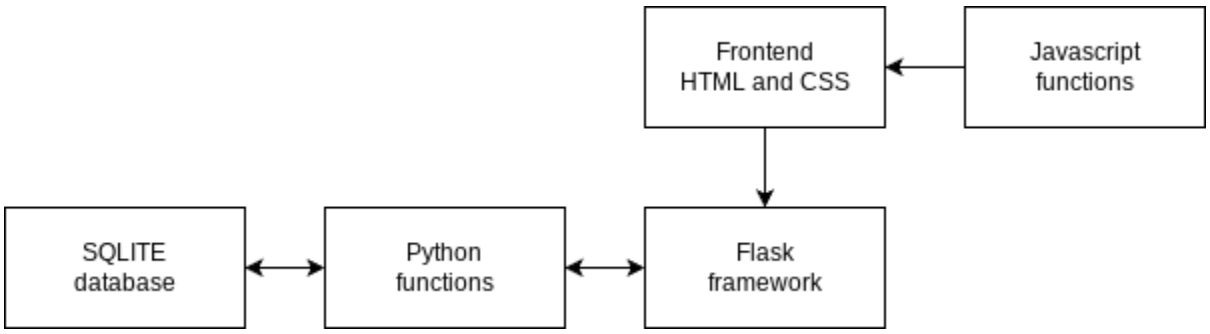
Points called "flower score" will be assigned based on (equal to) the number of days taken to grow the flower. As the user progresses, "flower score" unlocks more flower varieties which will appear in the shop. The game may continue indefinitely, but the "goal" is to unlock the final "Mythical Magnolia."

These interactions take place on a grid-based garden using JavaScript. The user's profile displays minigame statistics and the flower score, as well as "picked" flowers. Users must be logged in to play.

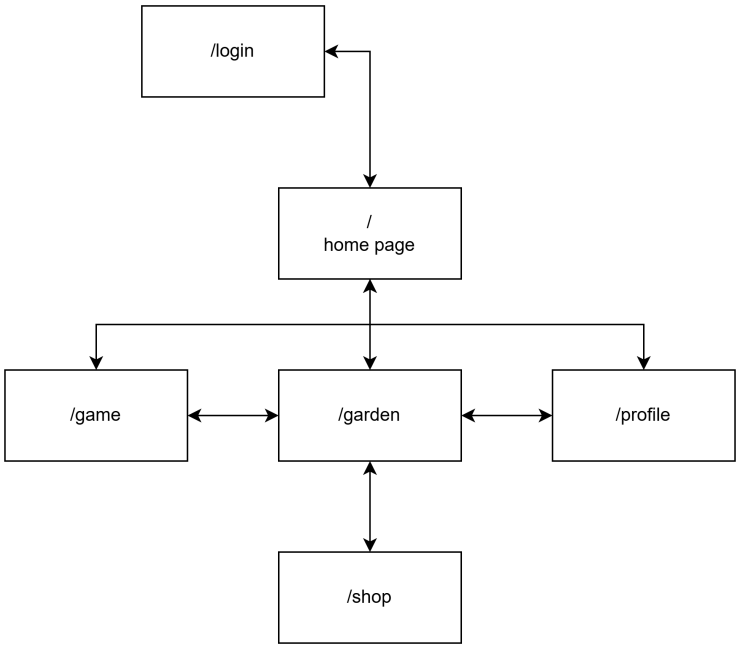
Program Components

1. Flask/Python
    a. Handles user authentication, session management, and redirection
    b. Interact with databases to access garden information
    c. Compares garden information to flower species information
    d. Minigame functionality
        i. Success or failure of game/problems
        ii. Distributing rewards
2. SQLite3
    a. Stores information about user credentials, garden information, and flower species (species resource requirements)

3. HTML/CSS
    a. User interface and showing different pages
    b. Display forms + garden + game + shop
    c. Interacts with javascript
4. JavaScript
    a. Shop interface displaying flower information
        i. New flower seed added to inventory (cannot be seen) if coordinate availability, cost requirements, and progress requirements are met
    b. Garden interface (three options show in selection upon hover of grid space)
        i. Button toggle for planting new flowers
            1. Second hover above initial hover for the three functionalities
            2. Selection between which seed to plant
        ii. Button toggle for growing
            1. Toggling via button for watering flower
            2. Same system for growing with magic power
        iii. Button toggle for picking flowers
            1. Available upon full growth of flower
            2. Removes flowers from garden and places them in profile



Site Map



1. /
    a. Accessible only to logged-in users. Non-logged-in users are redirected to the login/registration page
    b. Displays welcome banner ("Welcome, [username]!")
    c. Displays available plants, navigation links
2. /login
    a. Username and password fields
    b. Registration for first time users
    c. Successful login redirects to the home page
    d. Displays error messages for incorrect login attempts or invalid registrations
3. /garden
    a. Interactive Garden: A grid-based representation of the garden where users can water, use magic on and harvest magical plants using JavaScript for interactivity
    b. Hovering over a plant displays its information (ex. growth stage = days alive)
    c. Allows for growing and picking flowers
4. /shop
    a. Allows user to add new flowers to the garden based on requirements
    b. Shows garden underneath shop options to show space availability
5. /game
    a. Interactive game where user shows off their smarts with basic math questions
    b. Rewards (magic power) will be granted based on completion
6. /profile
    a. Shows users current statistics

i. Minigame rounds (wins/losses)
ii. Total resources
iii. Plants grown to final stage

Database Organization

- Users
  - Usernames, passwords
- Garden Table: user's flowers
  - Type, growth stage
- Flower_base: information about each flower species
  - Type, resource requirements
- User Stats Table: user's "inventory" (their resources)
  - Magic power and points in possession
- Profile
  - Flowers that have reached max growth
- Seeds (inventory)
  - Flower_id matches with id of species in flower_base
  - Stores who owns and how many

1. users

id	username	password
1	user1	TEXT NOT NULL

2. flower_base

id	flower_type	cost	max_growth
1	tulip	1	3
water_req			
3			

3. stats

user	magic power	flower score	
user1	5	5	

4. garden

id	user	flower_type	days_watered
1	user1	tulip	2
grid_row	grid_col	max_growth	
1	1	3	

5. profile (flower info)

flower_type	user	max_growth
tulip	user1	3

6. seeds

user	flower_id	quantity
user1	1	5

APIs

We do not currently plan to use any APIs, though we may consider a mini-game API if we wan to expand on our mini-game.

Front-end Framework

We will use Bootstrap as our front-end framework because of its aesthetics and simplicity in achieving desired appearance. It is intuitive to use.
We plan to utilize Bootstrap's responsive grids to easily format information. We are also interested in customizing Boostrap's default colors and fonts to suit our design.

Task Breakdown

1. Nia Lam: Project Manager + Middleware
    a. Keeping devos and responsibilities in check with reasonable deadlines
    b. Create routing infrastructure for flask
    c. Mystical storyline development
    d. Oversee Prettiness Development
    e. Facilitating communication and aiding with anything as needed
2. Amanda Tan: Frontend + Backend
    a. Writing javascript for growing and picking flowers in garden
    b. Python functioning for updating user statistics
    c. Homepage, login, garden, profile HTML + CSS template development

3. Naomi Lai: Frontend + Backend
    a. Writing minigame functionality
    b. Writing javascript for purchasing new flowers from shop
    c. Shop, (mini) game HTML + CSS template development
    d. Prettiness Development
4. Kishi Wijaya: Database Lead + Artwork
    a. Acquiring images via means
    b. Managing and writing database infrastructure
    c. Creating user sessions and accounts