# **Final Project: Design Doc**

Lactose Kirbies
Nada Hameed (PM), Fang Min Chen, Jasmine Yuen, Talia Hsia
SoftDev
Period 2
Target Ship Date: 2023-06-13 08:00

---

## **General Description:**

Our project will be a website where the user can choose different minigames to play—all of which have some relation to Kirby. You can load, save, and clear game files and your high scores for each game will be displayed on your profile.

Minigame ideas: Kirby… Chess, Pong, Angry Birds, Bejeweled, Candy Crush, Platformer, Lotteria, Flappy Bird, Space Invader….

---

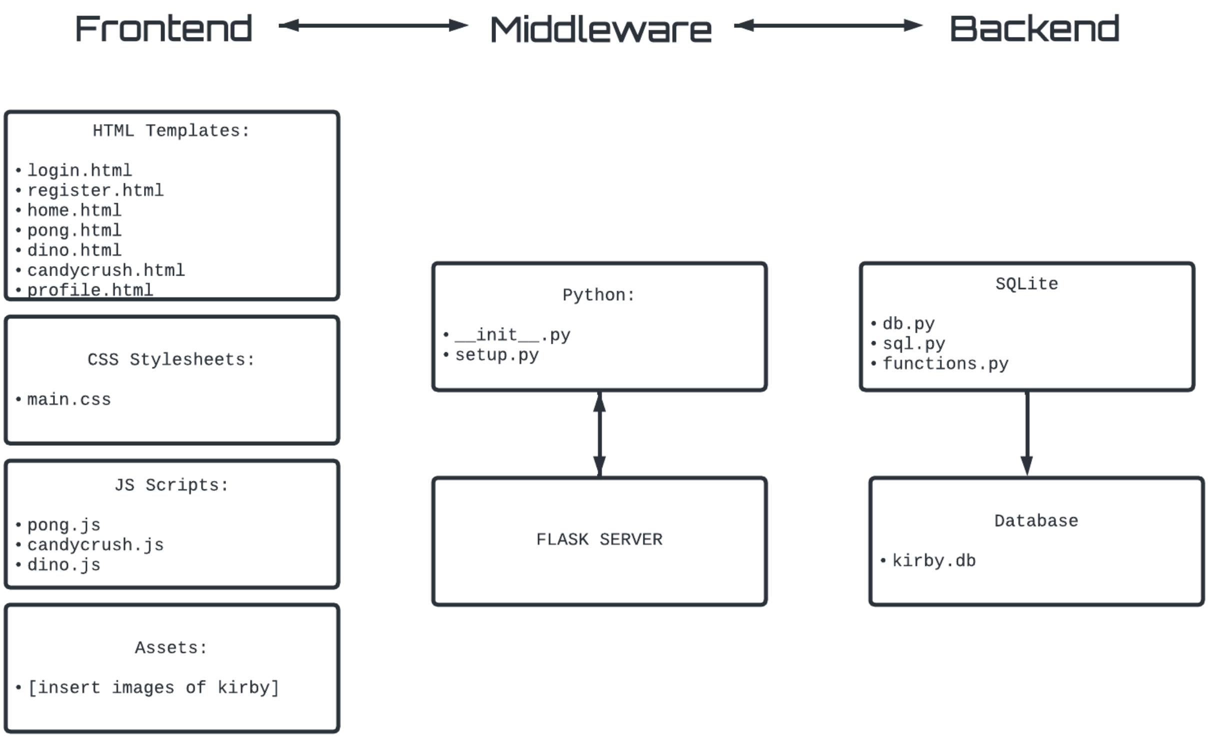
## **Program Components:**

- HTML Templates
    - login.html
    - register.html
    - home.html
    - pong.html
    - dino.html
    - candycrush.html
    - profile.html
- Static Files
    - CSS
        - main.css
    - JS
        - pong.js
        - dino.js
        - candycrush.js
    - Assets
        - Images of Kirby and other necessary things
- Flask
    - __init__.py
    - setup.py
- DB
    - db.py
    - sql.py (connection to sql)
    - functions.py

    - kirby.db

---

**Component Map:**



HTML:

- login.html — login page
- register.html — registration page
- home.html — homepage for all minigames
- pong.html — page for Kirby Pong
- dino.html — page for Kirby Dino
- candycrush.html — page for Candy Crushing Kirby
- profile.html — profile page

CSS Stylesheets:

- main.css — stylesheet in case we want to override any of Bootstrap's functions

JS Scripts:

- pong.js — animations for pong
- candycrush.js — animations for candy crush
- dino.js — animations for dino

Assets:

- folder to keep any assets (images) we need

Python:

- __init__.py — where Flask renders the html templates
- setup.py — any functions we need

SQLite:

- db.py — functions we need to write to database
- sql.py — writing to sqlite shell using python
- functions.py — other functions

Database:

- kirby.db — our database

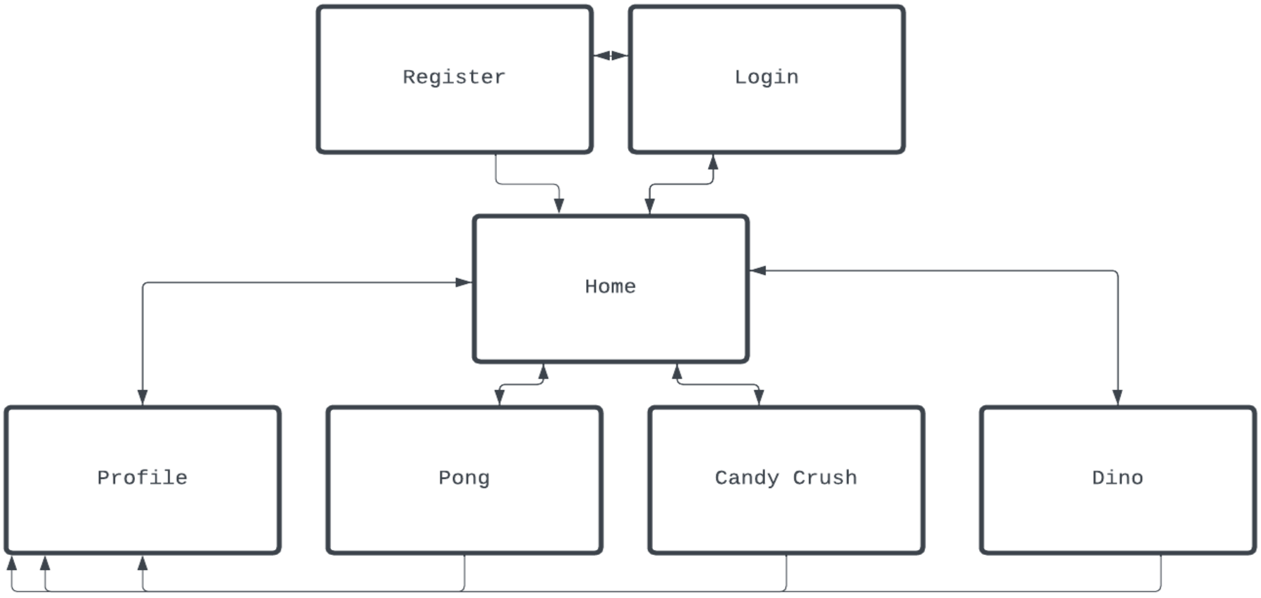
---

**Database Organization:**

| user (str) | pass (str) | pong-high (int) | dino-high (int) | candy-high (int) | all-time (float) |
|---|---|---|---|---|---|
| [username] | [password] | [high score for pong] | [high score for dino] | [high score for candy crush] | [how much time spent playing games in total] |

---

**Sitemap (For Front-End):**



Register Page

- User can register new user, with two inputs confirming password
- Once registered, session will automatically log in with that user

Login Page

- User can login with an *existing* account
- If username or password incorrect, an error message will displayed

Home

- This will have all the playable minigames listed, with thumbnails and descriptions displayed above each game's link, as well as a path to the profile

Pong

- This game will be Kirby Pong
- The user can control a paddle at the bottom of the screen, trying to knock out as many bricks as they can through batting the pong ball towards the bricks
- Each brick broken contributes 2 points to the score — if a specially colored brick is hit, it will contribute 10 points
- The ball is out of play the fourth time the user misses the ball with the paddle

Candy Crush

- This game will be Candy Crush but with different versions of Kirby being the different jellies
- For lack of time, our Candy Crush will not have special combos, but we will have three candy-combos and four candy-combos contribute more than only matching pairs
- Game ends and score is tallied once the number of moves is up — the default number of moves is 15 (subject to change as special candies are broken)

Dino

- This game will be a spinoff of the Chrome Dino game, but Kirby is running instead
- Once the Dino hits certain milestones (how far it's run), the speed will increase and/or there will be more obstacles in play
- Game ends and score is tallied once the Dino hits something

Profile

- User is played, along with a personalized profile picture and header image
- User can upload a status message
- User's total time played across all games is displayed
- User's high scores for each game is displayed

---

**Breakdown of Tasks:**

PM: Nada

Backend

- Database: Fang & Jasmine
- API: Nada & Talia
- Flask: Jasmine & Talia

Frontend

- Framework: Nada & Talia
- JS: Fang & Jasmine

---

**APIs:**

- None (as of this moment) — subject to change if any are recommended to us and we feel they fit the scope of our project

---

**Front-End Framework:**

- Bootstrap
- Bootstrap is simple to use and it allows for a responsive website design that will be useful in our project. We do not need the complexities of Foundation, so we will save ourselves some time on customization by using Bootstrap instead.

---