PPP Mode - Ivan Mijacika, Noakai Aronesty, Qina Liu, Justin Morrill
SoftDev pd2
P02 – Snake++ | Design Doc
2022-03-01

Project Description:

Our project is a new and improved Snake Game with a leaderboard, different game modes and settings. The game modes include various speeds, map sizes, obstacles, bosses, and features like border wrapping, peace mode, and poison apples.

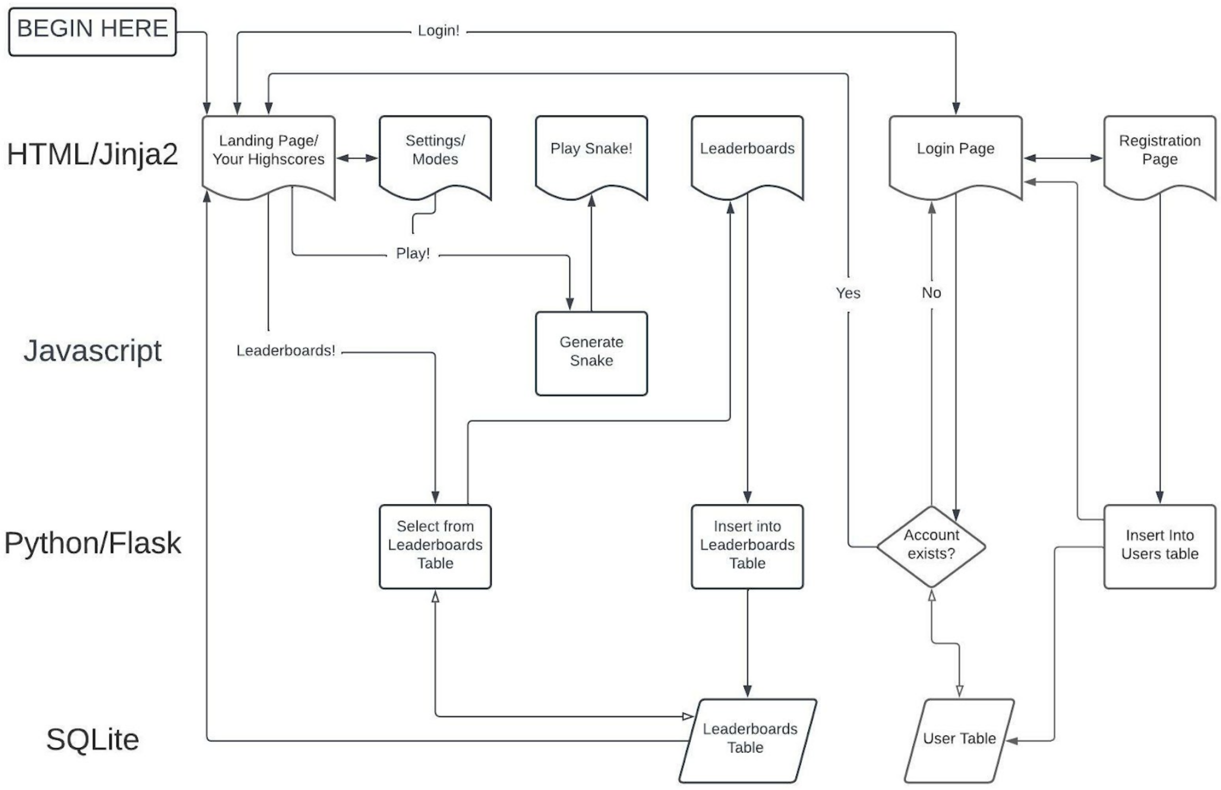
- Speeds: slow, medium, fast, insane
- Map sizes: small (10 x 10), medium (16 x 16), large (24 x 24)
- Obstacles: for every apple you eat, a block appear on the app
- Bosses: moving boss that you shoot lasers at (space for shoot laser), charge up laser by eating apples and shoot in direction of head
    - The laser goes through the entire row/column you are facing
    - If boss touches your tail, you lose that part of the tail; if boss touches your head, you die
    - If your laser touches your tail, you lose that part of your tail
- Border wrapping: hitting the wall does not kill you, it loops you to other side
- Peace mode: can’t die, you can hit wall (border wraps), and can hit yourself
- Jumping: you can jump your snake, allowing you to go over yourself
- Poison apples, a.k.a “stupid mode”: when you eat an apple, you have to answer a math question; if you get it wrong, you lose points, and get a point when you answer right
    - In addition to losing like you can in regular mode (wall, hit tail), if your points go to 0, you also die
    - When you eat an apple, screen pauses and math question pops up

Components:

- JavaScript
    - Game logic and animations
        - (mode from python, send to python score for db)
    - Leaderboard data matching with selected mode
        - (send mode to py,get data from py)
- Python
    - Send information using flask to create webpage
    - Differentiating between modes
        - (mode data sent to js for game)
    - Leaderboard
        - (get selected mode from js, send db data to js)
    - Log-in and register

- SQlite
  - Set up the database
- HTML/CSS, Jinja2
  - Making webpage look good
  - Functional frontend lists and dynamics
- Bootstrap
  - Front end framework
- Jinja2
  - Create the HTML template
- Flask
  - Create the site that the user will interact with
  - Create a session when the user logs in

Component Map: link



Database Organization:

Users

Username(TEXT)	Password(TEXT)	PfP Number (INT)
<username1>	<password1>	<0-n>

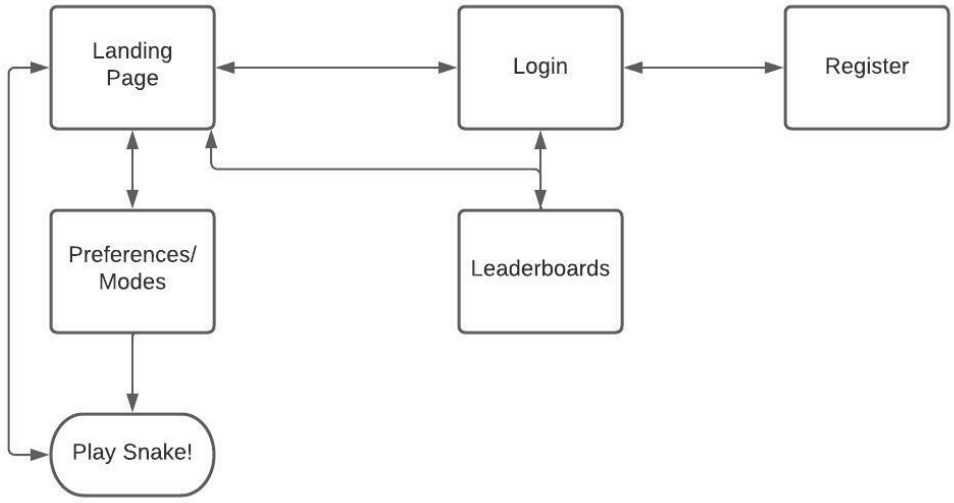
<username2>	<password2>	<0-n>
…	…	…

Profile picture number corresponds to each available profile picture

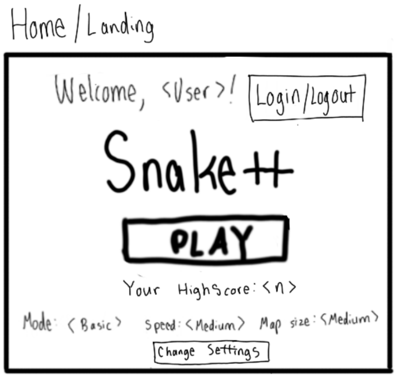
Scores

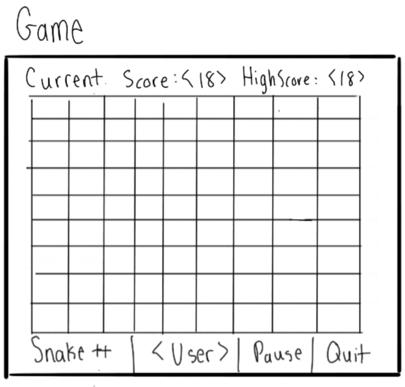
Username(TEXT)	Highscore Mode1(INT)	Highscore Mode2(INT)	Highscore ModeEtc…(INT)
<username1>	<0-n>	<0-n>	<0-n>
<username2>	<0-n>	<0-n>	<0-n>
…	…	…	…

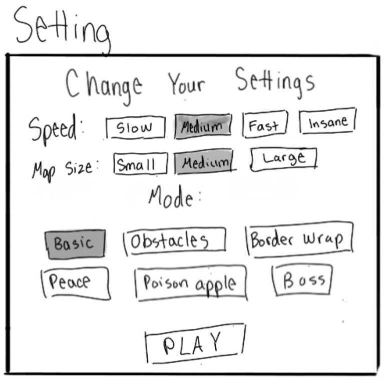
Site Map: link

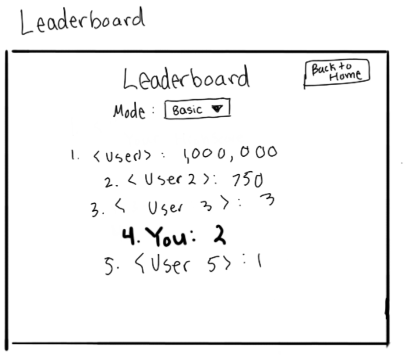


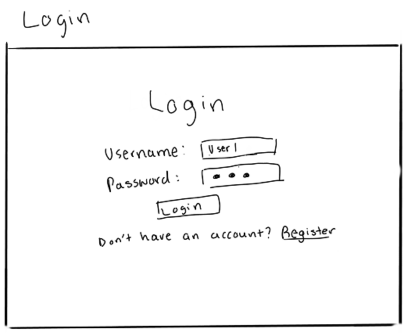
Templates:

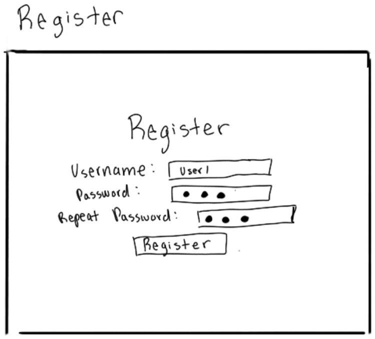












Frontend Framework:

We plan on using Bootstrap, utilizing the easy to import javascript and the clean and easy to use classes in order to make sure the information is properly laid out.

Good Bootstrap Features:

- Containers!
    - Put everything in containers: they're magical.
- Alerts are a nice type of container that (may not be used for intended purpose here but) look very nice and allow access to other colors for containers.
    - Some listed items on main pages will likely be in Alerts, as well as actual alerts, like login errors and such.
- Dropdowns are easy to create, look nice and are user-friendly.
    - Settings and/or modes perhaps?
- Buttons have a lot of support classes for aesthetics and links can be made to look like buttons
    - Navigation between pages
    - Form submission
- Display headings are special text classes that allow for much larger and better headings
    - Title, TNPG or other large headings
- Forms in bootstrap are supported nicely and come with good-looking default classes.

    - Settings/modes and other submissions
- Tables and auto formatted (but controllable) columns
    - Positioning different page elements in rows
- Lots of *text utility* like sizing, spacing, margins , **font weight** and `more`.

Task Division:

Ivan Mijacika (PM): frontend devo

- Frontend: jinja2 and html
- Flask and website main functions
- Leaderboards

Noakai Aronesty: frontend devo

- Frontend: bootstrap, css
- Authentication functions
- Javascript and snake game

Justin Morrill: backend devo

- SQLite and db
- Leaderboards
- Flask and website utility functions

Qina Liu: backend devo

- SQLite and db
- Javascript and snake game
- Settings/game mode differentiation

TARGET SHIP DATE: Monday, 3/21/22

Notes:

Mode ideas:

- Ouroboros: your goal is to eat your own tail. The longer you are when you do so the more points it is worth
- https://www.quora.com/What-are-the-most-interesting-variants-of-the-game-of-Snake