**Team Name:** DIJiTal
**Devs:** Tiffany Chen, Jonathan Quang, Iris Tao, Donia Tung
**Title:** Mario Party

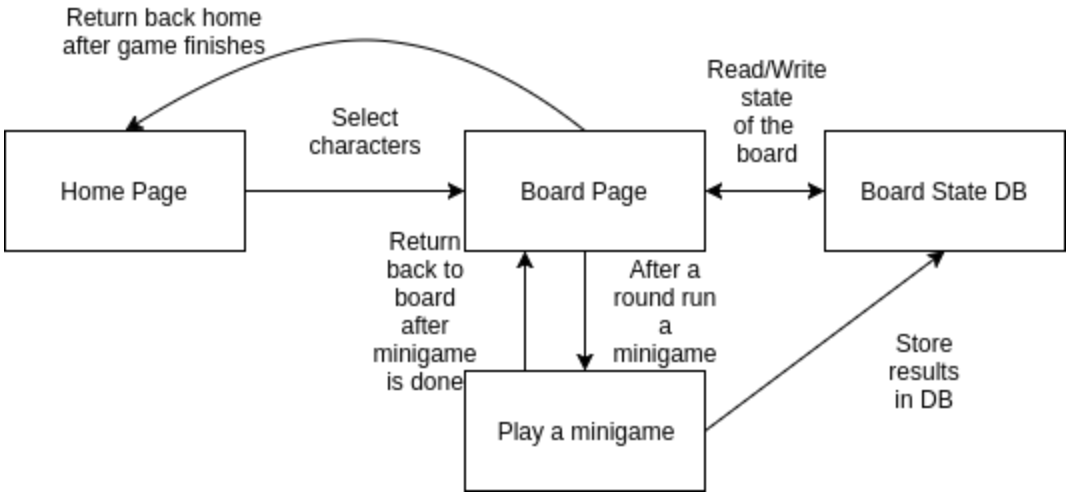
## About

We are creating a simplified version of Mario Party. Two players will play on the same computer for 10 turns (may allow players to choose the amount of turns if we have time). Each player starts with 20 coins. In every turn, each player will roll the die and land on a spot where they will gain/receive a certain amount of coins. After each player traverses the board, a minigame will randomly be selected for them to play. The winner of the minigame will receive 5 (amount subject to change) coins. The player with the most coins at the end wins.

## List of Program Components

- Home page
    - Character Selection
    - Project Description
    - Form where players can enter their gameID and password to continue playing their game
- Board page
    - Has the board with the players traversing it
    - After each turn, a minigame is randomly chosen
- Minigame pages (one per minigame)
    - Allows the players to play the minigame together
- Board database
    - Stores the state of the board
- app.py
    - Flask app
    - Functions:
        - Getters (e.g. getCoins(), getTurnNumber(), etc.) and setters (e.g. setCoins(), setTurnNumber, etc.)
        - Functions for each minigame to work
        - Functions for the board (e.g. function that allows the player to move around the board)

## Relationship Between Components




- Home page
    - Contains character selection form
        - The submit button would be the start new game button, would link to board page
    - Adds a game to the games database, turns = 0, generate a random gamecode to return to the player (to return to later)
    - Contains a form where users can enter a former game ID and password so that players can continue their game
        - If the game was already completed, a message will be flashed on this page saying so
- Board Page
    - Color-coded board of tiles with characters selected on the home page
    - Allows the players to “roll a die”
    - Each colored board tile makes the player gain/lose a certain amount of coins
    - Updates the games database to keep track of what turn the game is on
    - Accesses players database to display leaderboard
    - Once game is over, renders a different template that shows the winner and loser of the game
- Minigame Pages
    - Each minigame has a different page
    - Players will play simultaneously on the keyboard (one player gets aswd and the other gets the arrow keys)
    - Once the game is finished, links back to the board page
    - Updates the player database with coins won during the minigame

## Database Schema (exact names subject to change)

- Board.db
    - Has two tables
        - Games -> stores game ID, how many turns have passed, and a passcode (for returning to the game at a later time)
        - Players -> stores data for each player in each game (gameID, playerID [i.e. Mario or Luigi], coins, player position on the board)

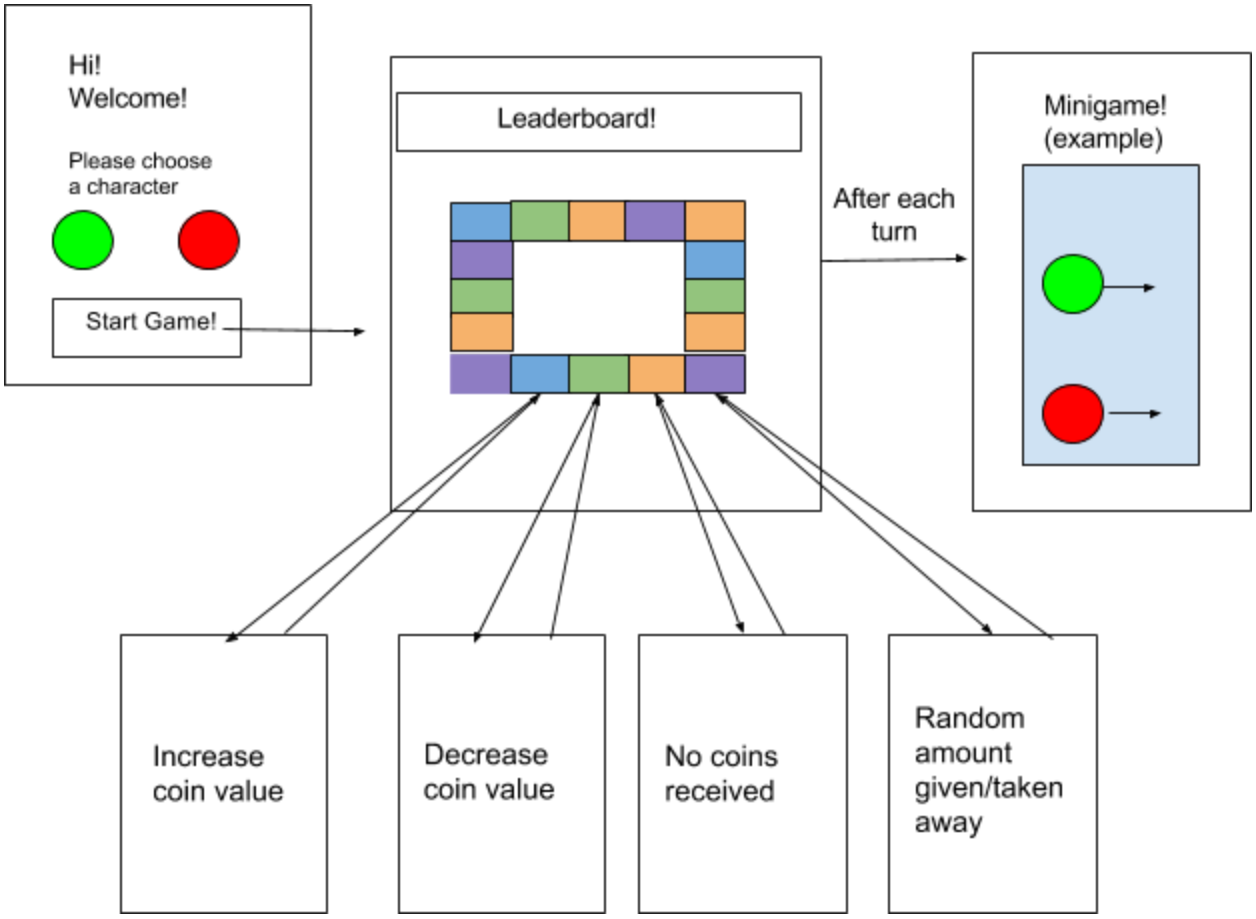
Games

| gameID | turns | passcode |
|---|---|---|
| 1 | 5 | 18329921 |
| 2 | 9 | 38918398 |
| 3 | 7 | 31083108 |
| ... | ... | ... |

Players

| gameID | playerID | coins | Position (in squares from origin) |
|---|---|---|---|
| 1 | 1 | 20 | 0 |
| 1 | 2 | 29 | 5 |
| 2 | 1 | 10 | 8 |
| 2 | 2 | 18 | 2 |
| ... | ... | ... | ... |

## Sitemap for Front-end




## Breakdown of Tasks

Tiffany Chen - project manager, miscellaneous tasks
Jonathan Quang - slot machine minigame and that dinosaur minigame when chrome has no access to internet
Iris Tao - memory match minigame and whack-a-mole minigame
Donia Tung - home and board pages