CMSC 447 TECHNICAL DOCUMENTATION

Group Members:
Manav Bhatt
Oritsejolomisan Mebaghanje
Jacob Adams
Connor Hohlbein
Sumair Chowdhury

GRAVITY DOG by Maine Coon

Introduction:

Gravity Dog is a game where users navigate through a challenging environment using gravity-defying mechanics. The game will feature a UMBC themed version of Gravity Guy, which utilizes a dog (your choice on the breed) to traverse stages of increasing difficulty. Technical features include: at least three stages, saves of the state of a user, a high score database, simple collectables in the game, and being fully web based. This section of the technical documentation provides a detailed look into the backend architecture of the Gravity Dog game, including the structure of its database models, relationships, and API endpoints used for managing game data and player interactions.

1. **Class Diagram**
    a. UserAccount
        i. Fields: id, username, password
        ii. Relationships: Player
    b. Player
        i. Fields: id, coins, account_id
        ii. Relationships: UserAccount, Outfits, Levels, Runs
    c. Outfit
        i. Fields: id, name, cost, texture
    d. Level
        i. Fields: id, name, texture, difficulty
    e. Run
        i. Fields: id, create_time, points, coins, level_id
    f. Relationships:
        i. Players to Outfits: Many-to-Many (via unlocked_outfit and equipped_outfit)
        ii. Players to Levels: Many-to-Many (via unlocked_level and player_last_level)
        iii. Players to Runs: One-to-Many (via player_run)
2. **Database**
    a. Users
        i. id (Primary Key, Auto-increment)
        ii. username (Unique, Not Nullable)
        iii. password (Not Nullable)
    b. Scores (Linked through Runs)
        i. run_id (Primary Key, Auto-increment)
        ii. player_id (Foreign Key from Player)
        iii. points (Not Nullable)
        iv. coins (Not Nullable)
        v. level_id (Foreign Key from Level)
    c. Levels
        i. id (Primary Key, Auto-increment)
        ii. name (Unique, Not Nullable)
        iii. texture (Not Nullable)
        iv. difficulty (Not Nullable)
    d. Outfits
        i. id (Primary Key, Auto-increment)
        ii. name (Not Nullable)
        iii. cost (Not Nullable)
        iv. texture (Not Nullable)

3. **API Calls**
    a. Endpoint: POST /submit-score
        i. Purpose: Submit the top 5 scores to a remote server.
        ii. URL: https://eope3o6d7z7e2cc.m.pipedream.net
        iii. Request Body:

```
{"data" :
        [
                {       "Group": "Maine Coon"
                        "Title": "Top 5 Scores"
                        "<1st Name>": "<1st score>"
                        "<2nd Name>": "<2nd score>"
                        "<3rd Name>": "<3rd score>"
                        "<4th Name>": "<4th score>"
                        "<5th Name>": "<5th score>"
                }
        ]
}
```

        iv. Response:
            1. Success: 200 OK with a message "Scores submitted successfully."
            2. Failure: Appropriate HTTP status code with error message.
4. **Security Considerations**
    a. Data Security: Use of flask_bcrypt for hashing passwords ensures that user credentials are securely stored.
    b. API Security: Implementing token-based authentication (JWT) for API requests to ensure that only authorized requests are processed.