DashClicks Backend Zone

You can put your submission in a new github/gitlab/bitbucket repository and send the link.

PART 1 - Design a NoSQL database schema

Write a single `JSON` "schema" file to represent the items below in a NoSQL database.

Consider what can be nested versus flattened and related in some other way.

- **Field** - description, directions, field photo urls, gps coordinates, etc.
- **Game** - start time, location, etc.
- **Field Availability Slot** - Typically an hour long slot at a field we have access to schedule a game. This is an exact day and time.
- **User** - name, email, etc.
- **Game Signup** - a user played (or will play) a specific game
- **Game Review** - a user reviewed a game (0-5 stars, comment, etc.)
- **Chat Message** - for both 1) games chats and 2) user to user chats

Here is a starting example schema file (you will want a minimum of 1 dummy item in each collection to demonstrate the schema):

```
{
  "fields": {
    "field-id-1" {
      "field_name": "Parc Central",
      ...
    }
  },
  "games": {
    "game-id-1": {
      "field_id": "field-id-1",
      ...
    },
    "game-id-2": {
      ...
```

```
    }
  },
  ...
}
```

PART 2 - Design backend functions

We will write functions to access data from the database you designed in part 1.

1. We want to find users so we can send targeted notifications. We will need to find users based on some criteria by querying the database. These functions should return an array of user ids.
    - **FUNCTION 1**: players who haven't played any games since a given date. Example: all players that haven't played a game since November 18th. (1 Input: a date; returns: array of user ids)
    - **FUNCTION 2**: players that have given an average Game Review rating of below x/5 stars for their previous x games played. For example, all players have given an average rating below 3.2 for their last 5 games. (2 Inputs: avg star rating, num past games; returns: array of user ids)
2. **FUNCTION 3**: A function that returns the number of unutilized Field Availability slots for a given field for a given date range. That means any Field Availability slots that did/do not have a Game scheduled in that slot during the date range. (3 Inputs: a field id, start date, end date; output: integer count)

⌨️ Our backend functions are in Javascript but you can write with pseudo code or in any language style you feel most comfortable.

Part 3 - Describe additional relational schema

As is common in NoSQL databases you may want to add additional redundant relational schema to support your queries and improve efficiency. This is to avoid doing large loops over the data and pulling unneeded data into memory when

searching for relations. For this part you will describe what additional schema you could add to the data from Part 1.

You do not need to modify your functions from Part 2 to use this new schema. You can also assume this relational data will be kept in sync with the "true" data.

```
// example of relational mapping in nosql
...
  "all_games_at_field": {
    "field-id-1": [
      "game-id-1",
      "game-id-2"
    ],
    "field-id-2": [
    ]
  },
...
```