# Dots Summer 2022 Internship

Backend Intern Take-Home Assignment

Thank you for your interest in applying for our summer internship program! This document describes the first phase of the interviews - the take-home assignment. We expect this assignment to take between 2-4 hours and is split into a free response section and a coding section.

The questions asked as part of this assignment have no “right” answer. However, including your rationale can be helpful on more nuanced or complex responses. The free responses are meant to learn a little more about you as a person and how you think about technical challenges. The coding portion is to gauge your hard skills using classes of technologies you’ll use as part of the internship.

*Please submit your responses as a pdf or other text document and also include a link to your code hosted on github or a similar service.*

## Free Response:

- **# Tech Question**
- Emily and Jack are friends that have decided to try making an online puzzle game! They come to you for your help to design parts of their backend capabilities. For now, they ask you to think about designing a system that can **gather**, **store**, and **analyze information** about what players are doing in the game - think metrics and stats that a data scientist might use. Assume you are provided with a list of metrics the game client will send to your endpoints. Your task is to design the backend systems that consume this data.

  What are the major components or “sections” this system needs? What technologies might you use for each of these components? What are some pitfalls and common issues you might encounter? If there are any open questions or unknowns about your design, you should highlight them in your response. You may make assumptions if anything is not explicitly specified, such as scale or cost limits, but please include these assumptions if you make any. Think of this as a thought exercise where you should think of many high level questions and come up with a cohesive high level plan that you might share with teammates, including more questions that need to be investigated. If possible, you should try to keep your response to one page long (~500 words). However this is not a requirement.

- **# Fun Questions**
- What’s the most exciting technical project you’ve worked on and why was it exciting?
- What motivates you to continue working in technology and/or software engineering?
- What do you hope to get out of this internship?

## Coding:

Emily and Jack are still looking to build that puzzle game! This time however, they want you to work on a different part of the backend. They want to build a prototype API that implements features such as **profiles** and **leaderboards**. This game will operate on a server-authoritative model where the server must validate all data sent from the client before storing the data.

You can think of the backend servers as the “source of truth” for the state of the game. For example when the game client finishes playing a level and needs to update the player’s xp, gold, etc, the client will send this information to the server. If the server believes this data to be valid, it will save the data to its database and subsequent client requests asking for this information will get the newly saved data.

**Your task as the backend expert is to build a prototype API using a microservices architecture to support various game functionalities.** This means we expect everything including the actual API code, the docker/docker-compose configs, and the database setup. Ideally, you should also include a README or other form of documentation describing how to run your submission. You are free to choose the exact technologies to implement this solution with.

## Technologies to Use:

- *Language and libraries of your choice*
    - We have a slight preference for Python as that’s what Dots uses, but you will not be penalized for using other languages. Use what you can show your best work with!
- *Database technology of your choice*
- *Docker*

***When submitting your work, please have the files uploaded to Github or similar service and include the link as part of your submission.***

## Prototype API Specification:

You will create a number of endpoints that the game client will be able to use to retrieve and update information. For this prototype, Emily and Jack are asking for the ability to **create new players** and update their profile and stats with information like **username**, **xp**, and **gold**. Additionally, we also want a simple implementation of a **leaderboards** endpoint that returns the top X players based on the parameters supplied to the api.

Emily and Jack are also concerned about the ability to rapidly prototype and scale going forward, so they’re requesting this to all be created using a microservices architecture. At a minimum, you should have an api container and a database container that communicates with each other as well as have the api container exposed to the host network.

Anything that is not explicitly specified in this specification is up to you to decide - *including errors and exceptions*. For example, we provide no guidance on id formats, database schemas, and only mention basic error states. You may also modify the input and output params of the API if it makes sense for your solution.

### Create a New Player

`POST /api/v1/player/`

**Request Body:**

```
{
    "username": "emily"
}
```

**Returns:**

200:

```
{
     "username": "emily",
     "player_id": "123abc"
}
```

4xx:

```
{
     "error_message": "Some kind of unspecified error has occurred!"
}
```

### Retrieve a Player's Stats

`GET /api/player/<player_id>/`

**Returns:**

200:

```
{
    "username": "emily"
    "player_id": "123abc",
    "xp": 9999,
    "gold": 1
}
```

## Update a Player's Stats

`PUT /api/player/<player_id>/`

**Request Body:**

```
{
    "username": "EmilyTheSecond",
    "xp": 10000,
    "gold": 10000
}
```

**Returns:**

200:

```
{
    "username": "EmilyTheSecond",
    "player_id": "123abc",
    "xp": 10000,
    "gold": 10000
}
```

4xx:

```
{
    "error_message": "Some kind of unspecified error has occurred!"
}
```

## Get Top X Players in Leaderboard

```
GET /api/leaderboards?sortby=<gold|xp>&size=<int>
```

This endpoint should return a list of <size> players in descending order. The stat to sort by should always be either "xp" or "gold". Both <sortby> and <size> are required GET parameters.

**Returns:**

200:

```
[
    {
      “username”: “EmilyTheSecond”,
      "player_id": "123abc",
      "xp": 10000,
      "gold": 10000 # Return both xp and gold
    },
    {
      “username”: “Jack”,
      "player_id": "234bcd",
      "xp": 1,
      "gold": 1
    }
]
```

4xx:

```
{
    "error_message": "Some kind of unspecified error has occurred!"
}
```