gain.pro senior Go engineer assignment

This assignment consists of two parts, a coding project and a small architecture design project.

You should take approx. 5-8 hours to complete the assignment.

After submitting your completed project, we will schedule a meeting in which you present your project and answer some technical questions.

Coding project

Objective

Develop a backend API for a web application: *GopherNet* - a platform for managing rentals of gopher burrows.
Each burrow is updated over time - as they are dug deeper by the Gopher renters.
Each burrow has specific attributes:

- Depth (`D`, *in meters*), changing each day as the Gophers dig deeper at a discrete rate of `D*0.9 cm/minute` The depth only increases if the burrow is occupied.
- Width (*in meters*), a constant that doesn't change.
- Occupancy (*boolean*), whether the burrow is currently occupied or not. One burrow per gopher - no room sharing.
- Burrow age (`A`, *in minutes*), with each burrow lasting exactly 25 days before collapsing.

The API server should keep the burrow listings up-to-date and allow Gopher users to rent them.

State should be maintained while the server is running - burrow depth, age, occupancy etc. You do not need to maintain state when restarting the server (although bonus points if you do)

Non-specified implementation details are up to you.

Include a *README* that explains how to build and run the server. The *README* should also contain example `curl` commands for testing the server using HTTP requests.

Requirements

1. *Load Burrows:* Load a JSON file detailing the burrows' initial states. This file should be stored in the repo as `data/initial.json`

```
[
    {
        "name": "The Underground Palace",
        "depth": 2.5,
        "width": 1.2,
        "occupied": true,
        "age": 10
    },
    {
        "name": "Tunnel of Mystery",
        "depth": 1.8,
        "width": 1.1,
        "occupied": false,
        "age": 30
    },
    {
        "name": "The Molehole",
        "depth": 3.0,
        "width": 1.3,
        "occupied": true,
        "age": 50
    },
    {
        "name": "The Deep Den",
        "depth": 2.2,
        "width": 1.2,
        "occupied": false,
        "age": 40
    },
    {
        "name": "Surface Level Statis",
        "depth": 0,
        "width": 1.3,
        "occupied": true,
        "age": 5
    }
]
```

2. *Update Status:* Update the state of each burrow every minute as a background task

3. *Rent Burrow:* Implement an HTTP REST endpoint to handle requests for renting a burrow. If a burrow is available (not occupied and hasn't collapsed), the burrow will be rented and it’s status will be updated. Otherwise, return an appropriate error message.
4. *Burrow Status:* Provide a REST endpoint to fetch the current status of the burrows
5. *Reporting:* Create a report summarising the total depth and number of available burrows, as well as the largest and smallest burrows - by volume. You can assume the burrows are cylindrical (where width is the diameter). This report should be outputted as a plain text file every 10 minutes.

Please focus on delivering on the 5 requirements before adding any bonus features. Some ideas for bonus features:

- Data persistence (between runs, ie when restarting the server)
- Logging
- Configurable initial data file using flags
- Additional endpoints that make sense given the context

Deliverable

You should upload your solution to a private github repo and share it with our team with the following github usernames:

- brianleenen
- fritzkeyzer
- jasperkuperus
- nielskrijger

Architecture project

Objective

This assignment aims to evaluate your understanding and proficiency in cloud architecture. You should demonstrate your ability to design a robust, scalable, and efficient cloud solution for a hypothetical burrow advertising platform, *GopherNet*.

GopherNet is an emerging digital platform that facilitates advertising for gopher burrows, serving thousands of users daily. It has multiple components, including user registration and authentication, search functionality and burrow ad placement. *GopherNet* is looking to transition from it's current traditional server setup to a scalable and efficient cloud-based infrastructure.

Design an architecture to support the *GopherNet platform.* Your design should use cloud services from your preferred cloud provider (eg, AWS, GCP)
Your solution should ensure high availability, scalability, and cost-effectiveness.

Task 1

Draw a detailed diagram of your proposed cloud architecture. The diagram should clearly illustrate how different components of the architecture interact with each other. You may use any diagramming tool of your choice.

Task 2

Write a paragraph for the following topics:

- How to manage and provision *production*, *test* and *dev* environments
- Basic explanation of how deployments will work
- High-level explanation of the local development workflow

Task 3

In a few small paragraphs - discuss how your cloud architecture could handle a sudden surge in traffic, which might occur due to a promotional event. Explain how your design maintains high performance and availability during these peak times.

Deliverable

Your final submission should include your architecture diagram and written explanations. Please package these into a single PDF document.

We look forward to your creative solutions. Good luck!