Go Backend Developer Assignment - ads.txt

Content sites (like msn.com) have a file called 'ads.txt'. This file describes which advertisers can display ads on the site and has a specific format that allows it to be parsed by machines.

If you browse to https://www.msn.com/ads.txt, for example, you will see that google.com is in the list so that it can display ads on msn.com.

Your task is to develop a RESTful API service that will allow the following:

Single Domain Analysis: Given a domain, return a JSON response with all the advertiser domains in the ads.txt file and the number of times each advertiser appears in the file.

For example, given the domain 'msn.com', you will get the following response:

JSON

```json
{
  "domain": "msn.com",
  "total_advertisers": 189,
  "advertisers": [
    {
      "domain": "google.com",
      "count": 102
    },
    {
      "domain": "appnexus.com",
      "count": 60
    },
    {
      "domain": "rubiconproject.com",
      "count": 27
    }
  ],
  "cached": false,
  "timestamp": "2025-07-13T10:30:45Z"
}
```

Batch Domain Analysis: The API should also support analyzing multiple domains in a single request. This is useful for advertisers who need to verify their presence efficiently in various publisher sites.

Example request for batch domain analysis:

```JSON
POST /api/batch-analysis
{
      "domains": ["msn.com", "cnn.com", "vidazoo.com"]
}
```

Requirements:

- The API should be built using **Go** and follow RESTful principles with proper HTTP status codes and error handling.
- The system must include a **caching mechanism** that can easily swap between storage backends (memory, Redis, file, etc.). Cache parameters should be configurable via environment variables.
- Implement a **rate-limiting feature per second** that you must develop entirely - no external libraries or Go's native rate-limiting packages are allowed.
- The system should be production-ready, as you see it as a developer.
- The code should contain tests that will cover edge cases and help make the development easier.
- Please upload your project to GitHub and send us a link to the repository.

Bonus points (optional):

- Add a Dockerfile and Docker Compose to the project.
- Add a simple CI/CD to the GitHub repository.