Sr. Back-End Engineer Technical Challenge

Your mission

Build a **fast** and **cost-efficient** API that allows to search songs released between a range of dates using the data from the Release Service. It should follow this specification:

URL: `/releases`
Method: `GET`
Query Arguments:
- `from`: Lower bound of the date range in YYYY-MM-DD format (**required**)
- `until`: Upper bound of the date range in YYYY-MM-DD format (**required**)
- `artist`: Name of the artist for filtering releases (**optional**)

Response Schema:

```
{
   "type": "array",
   "items": [
     {
       "type": "object",
       "properties": {
         "released_at": {"type": "string"},
         "songs": {
           "type": "array",
           "items": [
             {
               "type": "object",
               "properties": {
                "artist": {"type": "string"},
                 "name": {"type": "string"}
               },
             }
           ]
         }
       }
     }
   ]
 }
```

Example

URL: `/releases`
Method: `GET`
Query Arguments:

- `from=2021-01-02`
- `until=2021-01-05`
- `artist=VetLove`

Response:

```
[
 {
     "released_at": "2021-01-04",
     "songs": [
         {"artist":"VetLove", "name":"In the Air Tonight - WooGy Remix"}
     ]
 },
 {
     "released_at": "2021-01-05",
     "songs": [
         {"artist":"VetLove", "name":"In the Air Tonight - The Distance & Igi Remix"
         },
         {"artist": "VetLove", "name": "Play That Game - Desib-L Remix"}
     ]
 }
]
```

Requirements

- It should be implemented using the latest version of Golang.
- The dependencies should be handled using Go Modules.
- There must be a README that explains how to set up and install the application.
- The code must be in the GitHub repository assigned to you.

Considerations

Consider this test as not only as an assessment of your skills as a developer but as a way for us to get to know you as a potential teammate. With that in mind, please feel free to reach out to us while you are working on the challenge when you have thoughts or questions that will help you complete the challenge to the best of your ability. You will NOT be penalized for doing this, we want this to be interactive!

Release Service

Endpoints

This service offers two ways of requesting the same data: daily or monthly.

	Daily	**Monthly**
Data Returned	All songs released in the **day** specified.	All songs released in the **month** specified.
Endpoint	`v1/songs/`**`daily`**	`v1/songs/`**`monthly`**
Date format	YYYY-MM-DD	YYYY-MM
Costs per call	1 DataDeck Coin	25 DataDeck Coins
Response Reusability	30 days from when the request was made	30 days from when the request was made

Documentation

The key to authenticate into the service is `ec093dd5-bbe3-4d8e-bdac-314b40afb796`. Bear in mind that this should be treated as a secret value.

URL: `https://de-challenge.ltvco.com/v1/songs/daily`
Method: `GET`
Query Arguments:

- **api_key**: Key used to authenticate into the service (**required**)
- **released_at**: Day to filter songs by the time they were released, in YYYY-MM-DD format (**required**)

URL: https://de-challenge.ltvco.com/v1/songs/monthly
Method: GET
Query Arguments:

- **api_key**: Key used to authenticate into the service (**required**)
- **released_at**: Month to filter songs by the time they were released, in YYYY-MM format (**required**)

Successful Response:

```json
[
 {
   "song_id": "a7d8feae-cac5-40c2-8272-53b4089636c7",
   "released_at": "2021-01-21",
   "duration": "3m22s",
   "artist": "Weezer",
   "name": "All My Favorite Songs",
   "stats": {
     "last_played_at": 3371937363724866642,
     "times_played": 98621,
     "global_rank": 87
   }
 }
]
```

Error Response:

```json
{
 "error": "error message"
}
```