Node.js - BE developer exercise

Endpoints monitoring service

Create REST API JSON Node.js microservice, which allows to monitor particular http/https urls.

The service should allow to:

1. Create, Edit, Delete monitored URLs and list them for particular user. (CRUD)
2. Monitor URL(s) on background and log status codes + returned payload
3. For each particular monitored URL to be able to list last 10 monitored results.

Data model:

This is our suggestion for the data model, you can edit it as you wish.

MonitoredEndpoint has:
id: (the type is up to you :-))
name : string
url: string
Date of creation: DateTime
Date of last check: DateTime
monitoredInterval: Integer (in seconds)
owner: User

MonitoringResult has:
id: (the type is up to you :-))
Date of check: DateTime
returned http status code: Integer
returned payload: String
monitoredEndpointId: MonitoredEndpoint

User has
id: (the type is up to you :-))
UserName : String
Email : String
AccessToken : UUID like String

For User, we do not expect to have a CRUD, it's fine and enough to have it seeded in database.

It is expected:

```
{
  name: "Applifting",
  email: "info@applifting.cz",
  accessToken: "93f39e2f-80de-4033-99ee-249d92736a25"
},
{
  name: "Batman",
  email: "batman@example.com",
  accessToken: "dcb20f8a-5657-4f1b-9f7f-ce65739b359e"
}
```

In detail:

- Design REST endpoints for management of **MonitoredEndpoints** (if you do not know how, do not hesitate to ask us through github :-))
- Monitor endpoints on background and create **MonitoringResult.**
- Implement endpoint for getting **MonitoringResult**
- Implement microservice created in Node.js framework ideally written in **TypeScript** (if you don't want or you don't know to use typescript, use ES6 or later). For database use: **MySQL**. As REST framework use **Restify**.
- Autentisation do in HTTP header according to your choice, you will get the accessToken in it.
- Autorisation - User can see only MonitoredEndpoints and Result for him/herself only (according to accessToken)
- Do not forget about model validations (what is necessary to validate is up to your decision)
- Write basic tests in **Mocha** or **Jest**
- Push all into public repo at GitHub
- Create readme, where you describe how to start and use the service
- Send us Github link
- Bonus points for: create Dockerfile, add docker-compose and describe, how you start and run it in docker.

Good luck, have fun!