5. NodeJS homework

Dr. Balázs Simon (sbalazs@iit.bme.hu), BME IIT, 2017

In the following text use your own Neptun code with **lowercase letters** instead of the word "**neptun**".

1 A feladat leírása

The task is to create a system for storing information about movies and the publication of the system as a RESTful web service.

The data structures received or sent by the service is described by the following TypeScript-code:

```
interface IMovie
{
    title: string,
    year: number,
    director: string,
    actor: string[]
}

interface IMovieList
{
    movie: IMovie[]
}

interface IMovieId
{
    id: number
}

interface IMovieIdList
{
    id: number[]
}
```

2 The service

The service must be a NodeJS application. The application must be created based on the 2nd and 3rd chapters of the tutorial. The name of the application is **node_neptun** (where instead of **neptun** use your own Neptun code in **lowercase letters**). Use only the dependencies described in the tutorial, you must not use any other dependencies!

You should make changes only inside the **src** and **app** folders. You can choose implementation language: it can be either TypeScript or JavaScript. In case of TypeScript, you should work in the **src** folder. In case of JavaScript you should work in the **app** folder.

The service has to store state in a MongoDB database between calls:

- name of the MongoDB database: **neptun**
- name of the collection in the database: **Movies**

The base URL of the service must be:

`http://localhost:3000`

The service has the exact same functionality as the REST homework (2nd homework). The messages exchanged by the service is described by the TypeScript code above. When they are serialized they should give the exact same JSON messages as in the REST homework (2nd homework). In this NodeJS homework inly the JSON format has to be supported.

The service has to support the following calls added to the base URL above:

- **`GET /movies`**
  - input HTTP body: empty
  - output HTTP body: **`IMovieList`**
- **`GET /movies/{id}`**
  - input HTTP body: empty
  - output HTTP body: **`IMovie`**
- **`POST /movies`**
  - input HTTP body: **`IMovie`**
  - output HTTP body: **`IMovieId`**
- **`PUT /movies/{id}`**
  - input HTTP body: **`IMovie`**
  - output HTTP body: empty
- **`DELETE /movies/{id}`**
  - input HTTP body: empty
  - output HTTP body: empty
- **`GET /movies/find?year={year}&orderby={field}`**
  - input HTTP body: empty
  - output HTTP body: **`IMovieIdList`**

3 The client

No client is required for this homework. However, creating a client for testing is strongly recommended, but do not upload this client.

4 To be uploaded

A single ZIP file has to be uploaded. Other archive formats must not be used!

The root of the ZIP file must contain one folder:

- **node_neptun:** the NodeJS project of the service

It is not necessary to upload the **node_modules** and **typings** subfolders.

The application must be able to run and function correctly after issuing the following commands:

```
...\node_neptun>npm install
```

```
...\node_neptun>typings install
```

```
...\node_neptun>npm start
```

In case you choose the TypeScript language, make sure to compile all TypeScript code to JavaScript before uploading!

The service must support exactly the JSON format of the REST homework (2nd homework). Don't use any other message formats!

Important: the instructions and naming constraints described in this document must be precisely followed!

Again: use your own Neptun code with **lowercase letters** instead of the word "**neptun**".