Assignment 7

Due: Wed 11.12.2019; 18:00h (1 Week)

Goals

After doing these exercises, you will know how to create an API with NodeJS from scratch.
Submit only one code bundle, regardless of how many tasks you finished – each effort counts!

Task 1: Create a Node App with the Express Generator — Difficulty: Easy

Use the express generator as shown in the tutorial, or download the example from GitHub:

https://github.com/mimuc/omm-ws1920/tree/master/tutorials/07-nodejs/examples/OMMExpressApp

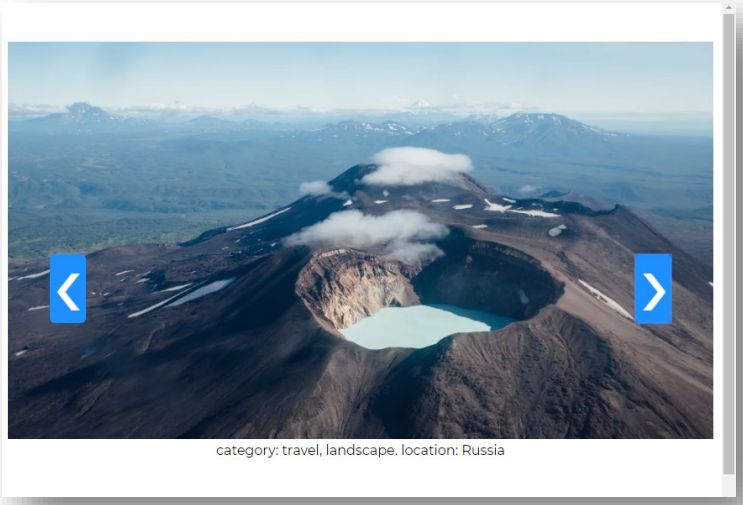
a) Inside the app root directory, run “npm install” to download the necessary dependencies
b) Run “npm start” to launch the app.
c) Go to http://localhost:3000 to verify that the app works.
d) Contact us, if it doesn’t work.

No submission necessary, task continues in Task 2.

Task 2: Slideshow API

Remember the image slideshow from assignment 1? Back then, it only showed some static images. Let’s make it a little bit more dynamic!

In the *assignment-material* folder on GitHub, you will find an improved version of the slideshow website *slide-show.html*. It queries images from an API, allows filtering by category and location, and uploading new images. The webpage frontend is already done, but the backend is missing.



Task 2a: Return predefined images — Difficulty: Easy

In subtask a, let's **start implementing a matching backend API**. To start, **focus on simply fetching existing image URLs**:

- Use the NodeJS Express app of Task 1.
- To fetch a list of image URLs, the webpage's JavaScript code performs a request to http://localhost:3000/images . It expects a response JSON structured like this:

```
[
   {
      "url" : " https://cdn.worldnomads.net/Media/Default/Travel-Safety/colombia/bogota-colombia-skyline.jpg",
      "categories" : ["travel", "cityscape"],
      "location" : "Bogota, Colombia"
   }
]
```

- Such data is contained in the file *imagedatabase.json* . Create a route in your Express app, matching the webpage's request and returning those images.
- Don't care about the filter, upload, … options in the first grey boxes so far!
- **No changes** in the webpage should be done!

Task 2b: Slideshow API – URL uploads — Difficulty: Medium

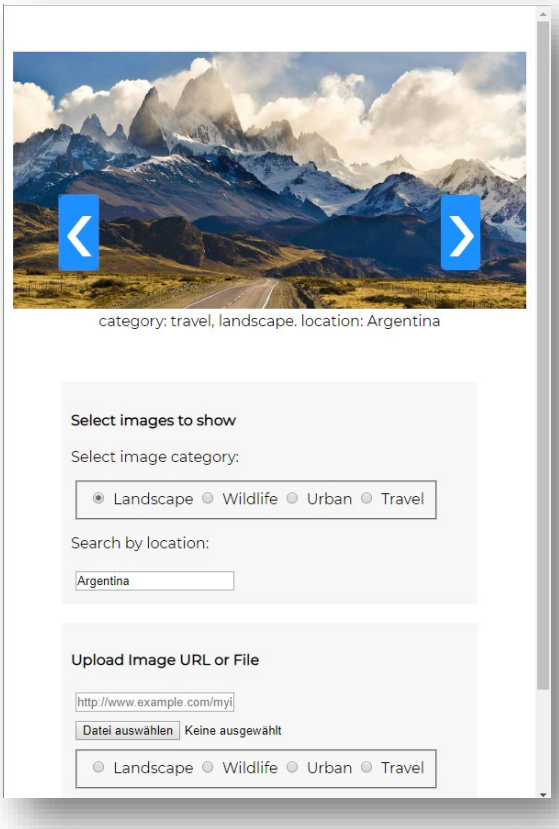
As you might have noticed, the website also allows filtering images and uploading new ones.

In subtask b, you should:

- make **filter features** work (first grey box)
- make the image **URL uploads** work

Appropriate steps might be:

- Read the request parameters in your GET route and filter the returned images
- Create a new route for the URL uploads (don't care about file uploads yet!)
- Depending on how you solved task one, replace the *imagedatabase.json* with some editable "database" (No real database is required, you can simply use a JavaScript variable!)
- Again, any changes in the webpage **must not** be done.



Task2c: Slideshow API – File Uploads

Difficulty: Hard

Adapt your Express app's POST endpoint, so that it can handle **file uploads** from the webpage.

Appropriate steps might be:

- Add a special handling for file uploads in the post route.
- Save the file in a directory within the NodeJS app.
- Serve this directory's content as static resources.
- Create URLs for these static images files, and put them in your image "database".

Tips:

- The NodeJS library express-fileupload might be helpful.
- If you use the Express generator to create empty express apps, the created example app already does some static file hosting (all files in the *public* directory). You could reuse that code or implement it in a similar way.

Submission

Please turn in your solution as ZIP file via Uni2Work. You can form groups of up to three people.

We encourage you to sign up for Slack! All you need is a CIP account and an email address that ends in "@cip.ifi.lmu.de". Ask us if you don't know how to get them.

If you have questions or comments before the submission, please contact one of the tutors. They are on Slack @Aleksa and @Andre, remember that they also want to enjoy their weekends ☺

It also makes sense to ask the question in our #omm-ws1920 channel. Maybe fellow students can help or benefit from the answers, too!