# Express.js Exam Retake – Hotel System

# Exam rules:

* This is a copy of the ' Express.js Exam Retake – Hotel System ‘, for solving this problem you may use the provided sample-code, which contains HTML and CSS that might be useful for completing the following tasks.
* You have 6 hours – from 16:00 to 22:00
* When you are ready, delete the *node\_modules* folder, make sure all dependencies are listed in the *package.json* file and submit your archived project at <https://softuni.bg/trainings/1642/expressjs-fundamentals-may-2017>
* There will be no breaks during the exam but you can go to the toilet or to breathe some fresh air outside **if you are alone**
* If you have any questions for the following description – ask the trainer, he is @ the back of the room
* Have fun! ☺

## Route /

Create a Node.js web server which is capable of returning an “index.html” file containing a welcome message and menu. Add all available routes from below to the menu. Add register, login and logout (only when logged in). List 20 hotels ordered by their date of creation. The newest ones should be on top. Visible to anonymous users.

## Route /add

This route should allow all authorized users to add hotels to the system. Hotels have title, description, location and image URL.

## Route /list

This route should list all hotels ordered by their date of creation. The newest ones should be on top. Visible to anonymous users.

## Route /hotel/{id}/{title}

This route should show the hotel with the provided *id*. The route should show a form allowing authorized users to add reviews to this particular hotel. List all comments on this page too. They should be ordered by their date in ascending order. Visible to anonymous users.

## Route /profile/{username}

This route should show information about the user with the provided *username*. Show all his hotels and all his comments. Visible to authorized users only.

## Add administration

If the user is an administrator, he/she should be able to edit or delete hotels and comments. Add such functionally to the /hotel/{id}/{title} route. Additionally, add administration route /admins/all and admins/add where administrators can add other users as administrators too.

## Add paging - /list?page=2

Add paging on the /list route. Return 20 hotels per page.

## Add likes and views

Add views counter on every hotel. It should be incremented every time the hotel is opened. Show the total number of views on the /list and /post/{id}/{title} pages. Add option for authorized users to like a hotel and show the total number of likes. If the users liked a hotel, they should have the option to dislike it.

## Add categories

Allow the administrators to add and delete categories to the system. Each category name is unique. Then all hotels must be added in one of these categories. It should be a dropdown option for the user. Additionally, add route /list/{category} where only the hotels from that category are shown. Add route /categories where all categories are listed.

## Add anti-spam policy

The administrators should be able to block/unblock users from creating hotels and comments (reading is still allowed).

**OTHER REQUIREMENTS**

* Use Node.js as a web server
* Use MongoDB as a database storage
* Use Express.js as a routing framework
* Follow the standard - <https://github.com/feross/standard>
* You may use whatever frameworks you like