# MongoDB Exercises

## Install MondoDB and Mongoose

Open the provided skeleton and create **connection** with **MongoDB** on its default port-27017. If there are any error messages, about deprecated functionalities, try to fix them.

## Implement the following logic in MongoDB playground

### Create mongoose Schemas

In the appropriate folder create two **Schemas** with names tag and image, which you will be using for **creating**, **reading** and **deleting** data. Each **tag** should have **name**, **creation** **date** and array of **images**. Each **image** should have **URL**, **creation date**, **description** and multiple **tags**. Try to add a function in the tag model that turns **tagName** to **lower case.**

### Create Add functionality

In folder **'handlers'**, implement create functionality in **'tagHandler.js'** for creating new **tags using the provided form.** After you have successfully saved a tag in your DB,

the part: <div class='replaceMe'></div>

should be replaced with appropriate string

for each tag: <div class='tag' id=${tag.\_id}">${tag.tagName}</div>.

In folder **' handlers '** implement create functionality in '**imageHandler.js**' for creating new images, that holds a URL to a picture, description, title and array of tags.

|  |
| --- |
| instanodeDb.saveImage({ url: 'https://i.ytimg.com/vi/tntOCGkgt98/maxresdefault.jpg', description: 'such cat much wow', tags: ['cat', 'kitty', 'cute','catstagram'] }) |

### Create Delete image functionality

**Go to point 3.1**

Create functionality that activates upon clicking the delete button in **results** view.

## Implement search functionality

### 3.1 Search with no parameters

The module should return all data from the DB using ‘**results.html**',

replacing: '<div class='replaceMe'></div>'

for each instance of image with:

`<fieldset id => <legend>${image.imageTitle}:</legend>

<img src="${image.imageUrl}">

</img><p>${image.description}<p/>

<button onclick='location.href="/delete?id=${image.\_id}"'class='deleteBtn'>Delete

</button>

</fieldset>`

### 3.2 Search by tag

The module should return all images holding the selected tag sorted by their date of creation in descending order:

|  |
| --- |
| instanodeDb.findByTag('cat') |

### 3.2 Add search by creation date

The module should return all **images** with their tags **between** **two dates**. The two dates are **optional**. For example - if the **'after'** parameter is missing, take into account only the **'before'** one. If **both** are **missing** **do** **not** **filter** by date. Take no more than the provided number of total results. If 'results' is missing, take no more than 10 results:

|  |
| --- |
| instanodeDb.filter({after: minDate, before: maxDate, results: 24}) |

## Deploy

Deploy your application using [Heroku](http://heroku.com) and [Mlab](https://mlab.com).