# Exercise: Basic CRUD – Meister Task

# Overview

You have been tasked to create a simple Meister Task application. The application should hold **tasks**, which are the main app **entities**. Tasks can change their status:

* **Open**
* **In Progress**
* **Finished**

The app is called MeisterTask, like the popular task management app "**Meister Task**".

The functionality of the application should support **creating**, **listing**, **editing** and **deleting** tasks.

The application should **persist** the data into a **database**.

### Requirements

* **Express** framework
* **Handlebars** view engine
* **Mongoose** ORM
* **MongoDB**

### Data Model

* title – non-empty text
* status – non-empty text, which can hold **either of the following values:**
  + **Open**
  + **In Progress**
  + **Finished**

### Project Skeleton

You will be given the applications’ skeletons, which holds about **90%** of the logic. You’ll be given some **files** (**controllers**, **models** etc.). The files will have **partially implemented logic**, so you’ll need to write some code for the application to **function properly**.

The application’s views will be given to you fully implemented. You only need to include them in your business logic.

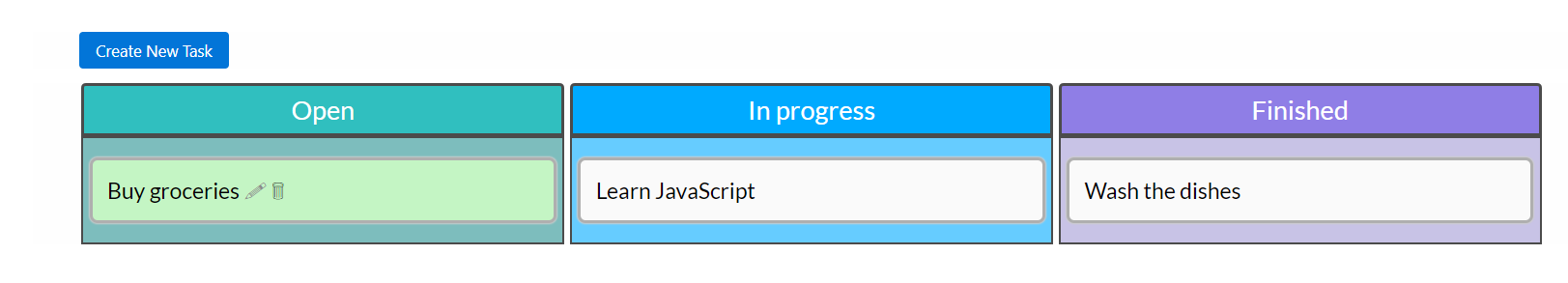
Everything that has been given to you inside the skeleton is **correctly implemented** and if you write your code **correctly**, the application should work just fine. You are free to change anything in the Skeleton on your account.

### User Interface

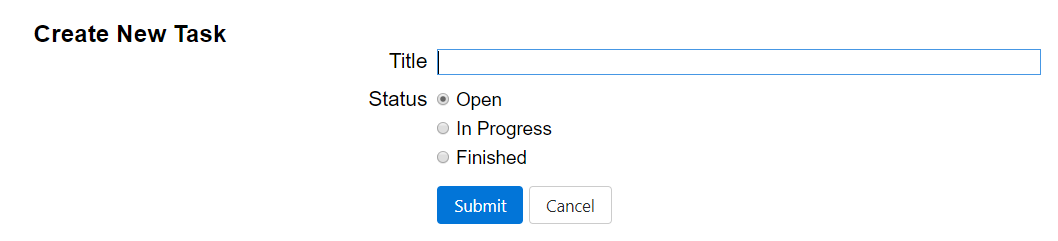
This is the user interface or how the application’s pages should look in their final form (fully implemented). You have several pages, described below:

#### Index Page

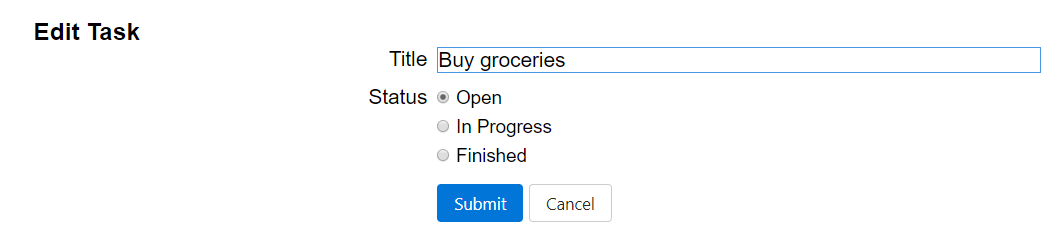
Each of the three columns lists **only the tasks in that status.** When the user hovers over a task, the edit button should point to the /edit/{id} route, and delete button should point to /delete/{id}:



#### Create Page



#### Edit Page



#### Delete Page

