# Databases - Exercise

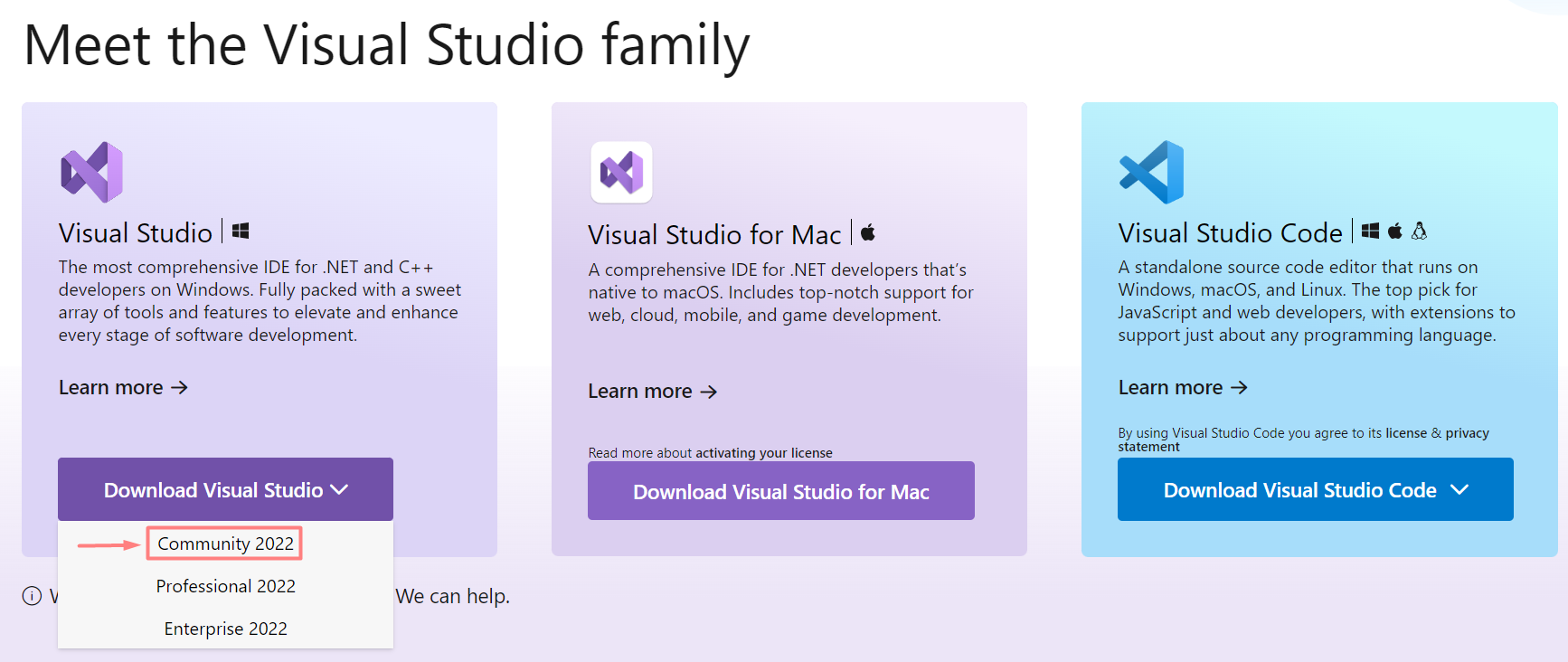
## 1. **Visual Studio Setup**

Steps for Installation and Setup of Visual Studio for C#.

### Step 1:

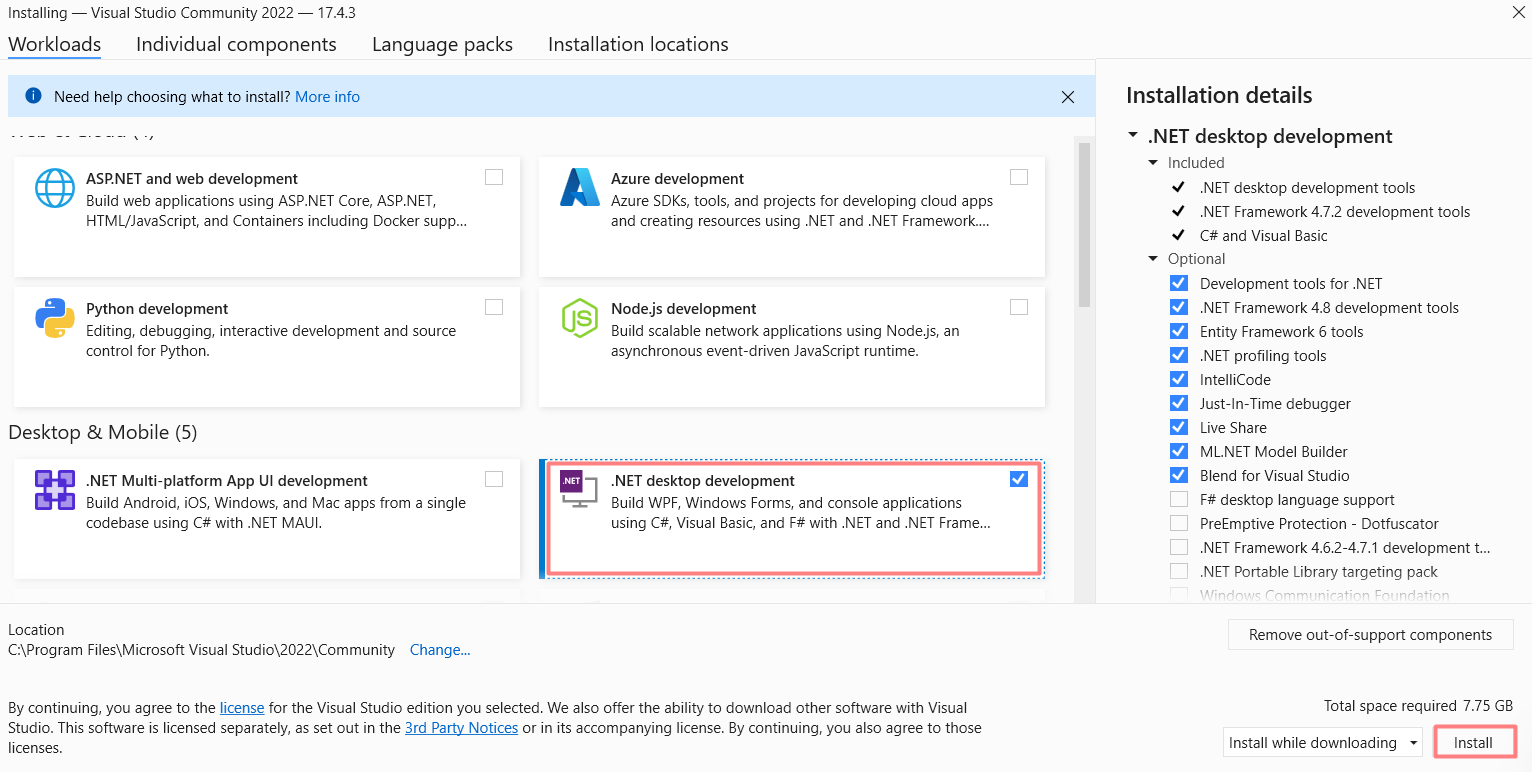
First, there is a need to download Visual Studio. Open [https://**visualstudio.microsoft.com**](https://visualstudio.microsoft.com/).

Download the **Visual Studio Community** installer (the free Community edition).



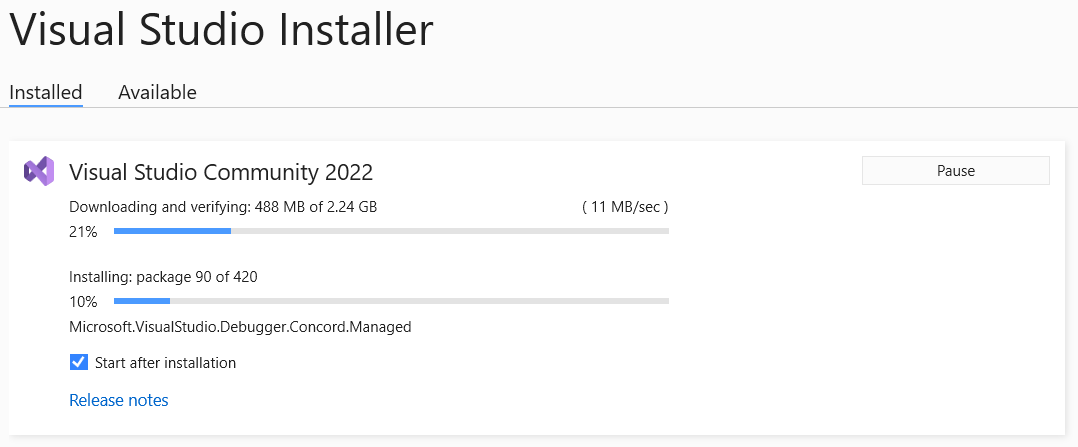
### Step 2:

After downloading the Visual Studio Installer, select the **.NET desktop development** option and then click on **Install**.



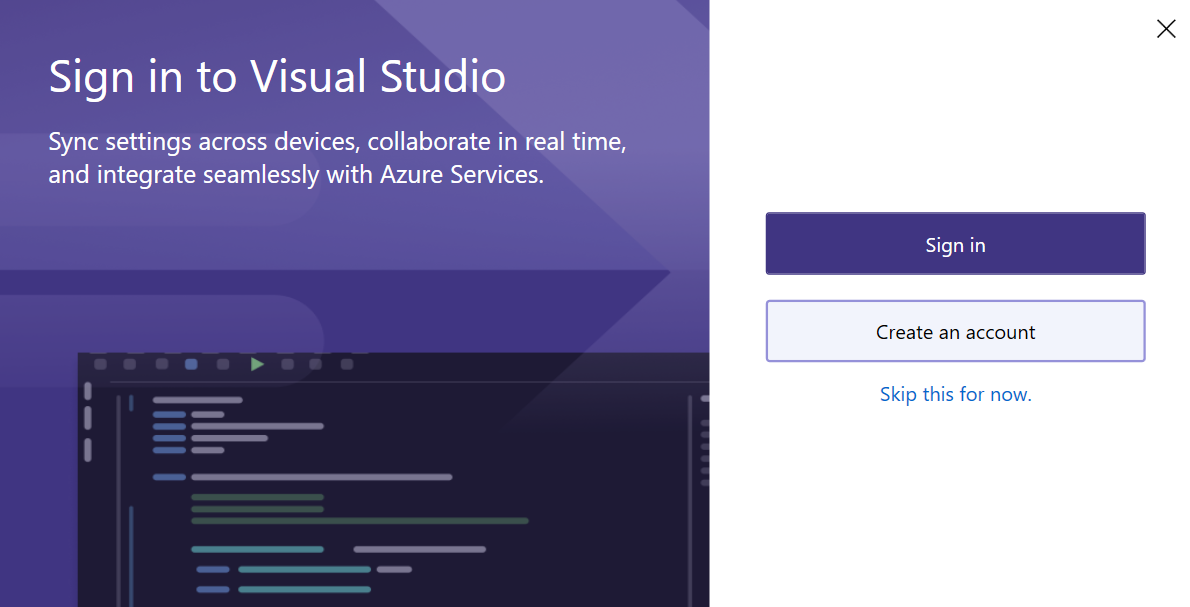
### Step 3:

Now let the **Visual Studio Installer** download the packages and perform the installation.



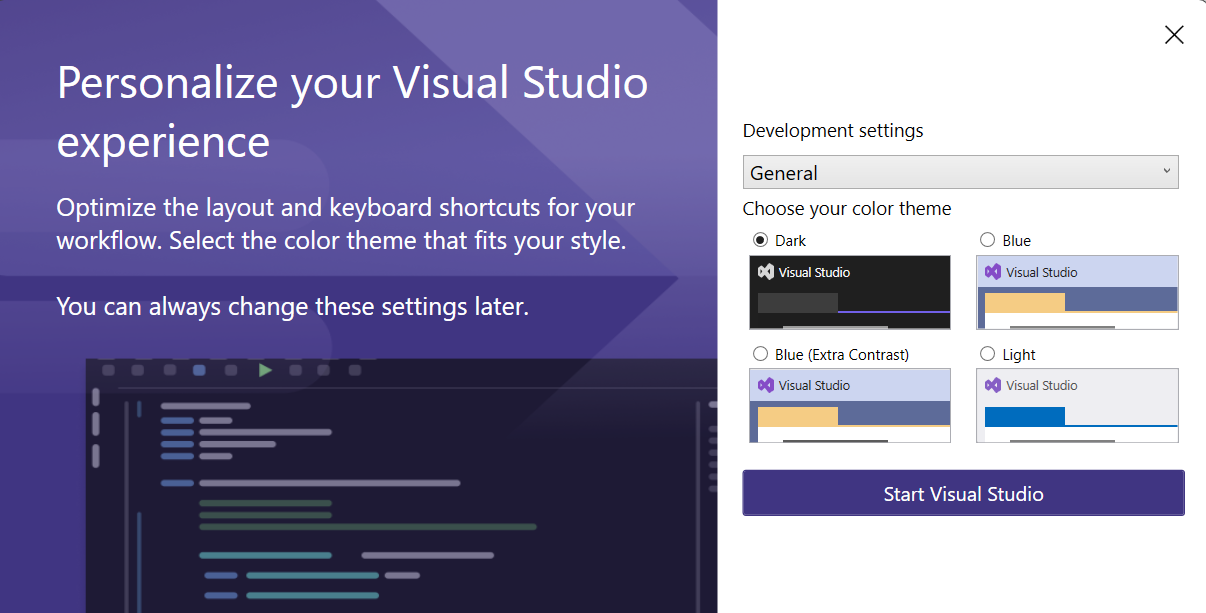
### Step 4:

After the installation is over, begin with the Visual Studio by Signing In to your Microsoft account or Create an account! You can also proceed without Signing In by clicking on the Skip this for now option.



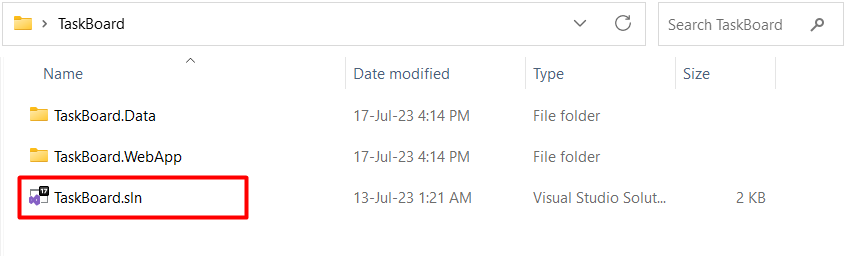
### Step 5:

Choose your color theme and then click on **Start Visual Studio** button.

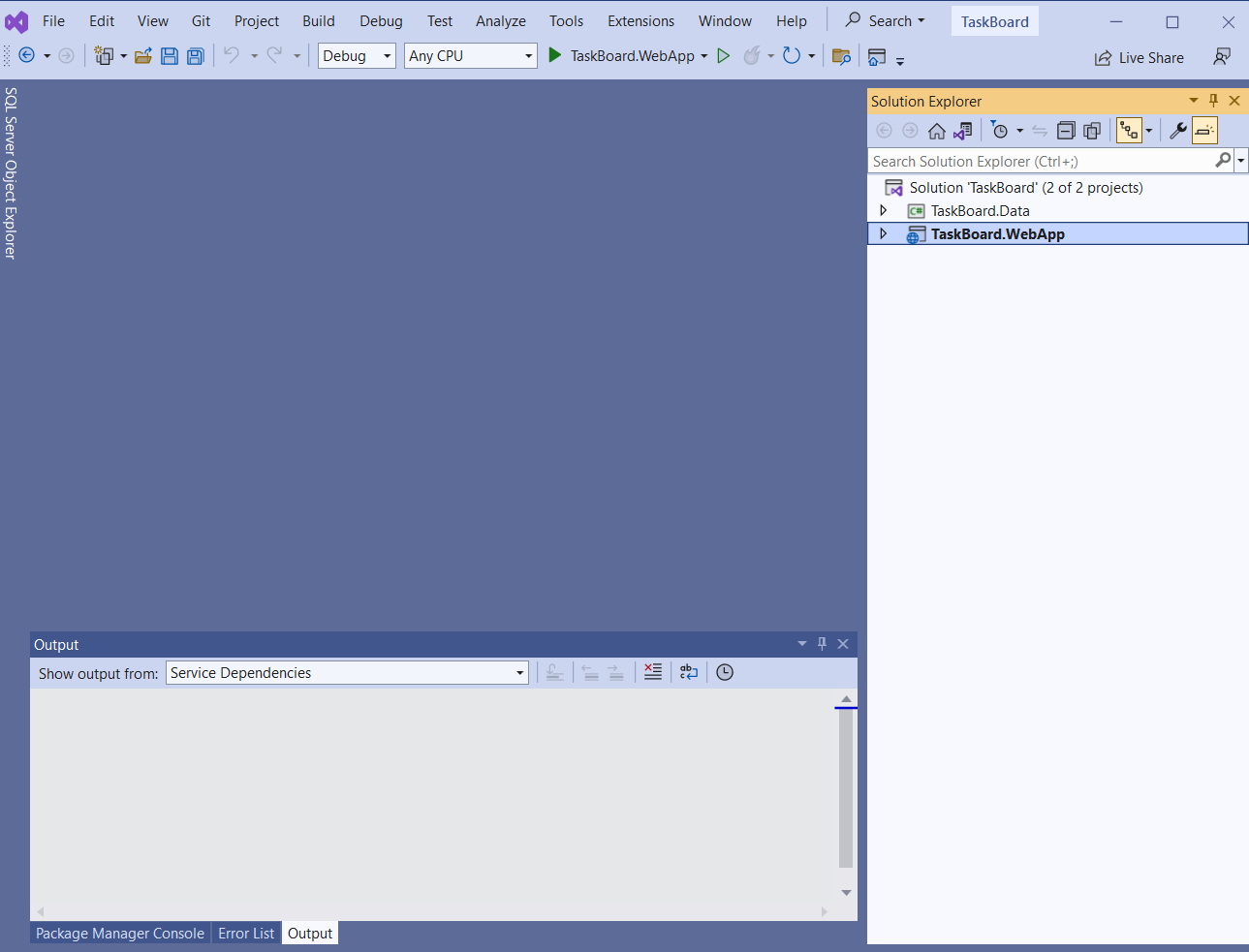


## 2. Running The "TaskBoard" App

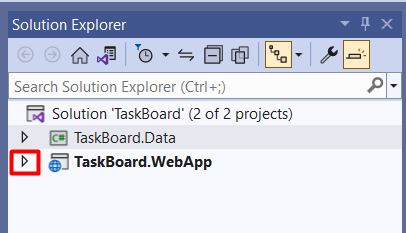
**1.** Navigate to the folder when you extracted the "TaskBoard" from the provided .zip archive and find the TaskBoard.sln. Double click it.

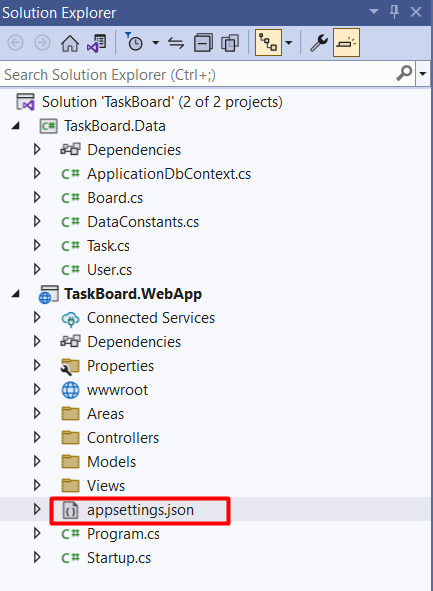


**2.** This will open TaskBoard solution in your Visual Studio. As you can see, on the right side, in the Solution Explorer there are two projects in the solution: TaskBoard.Data and TaskBoard.WebApp

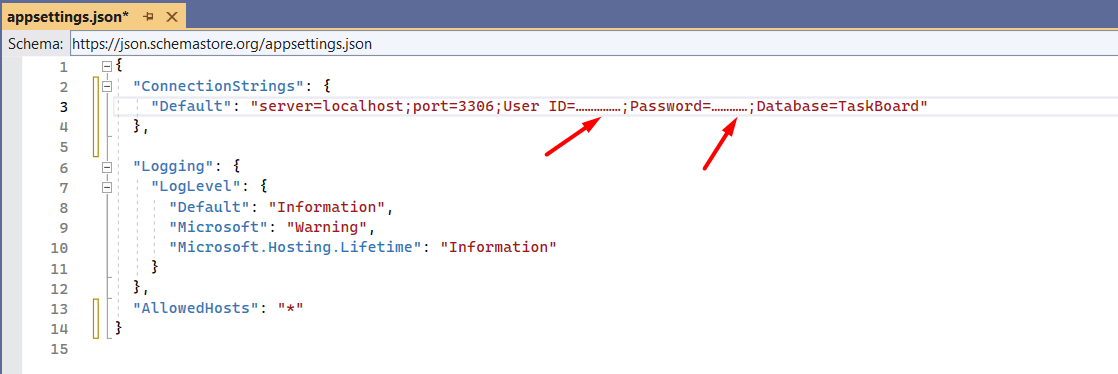


5. Click on the small triangle in front of the TaskBoard.WebApp project in the Solution Explorer.

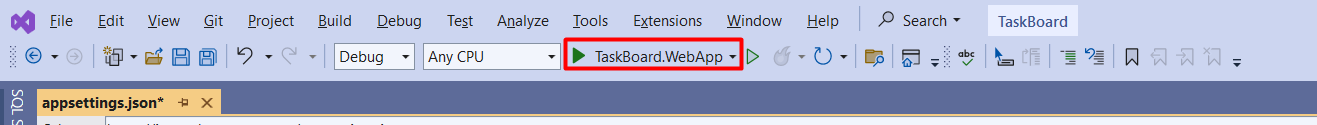
  
**6.** This will open all the files that belong to the TaskBoard.WebApp Project. You will need to find the **appsettings.json** file. Double click to open it.



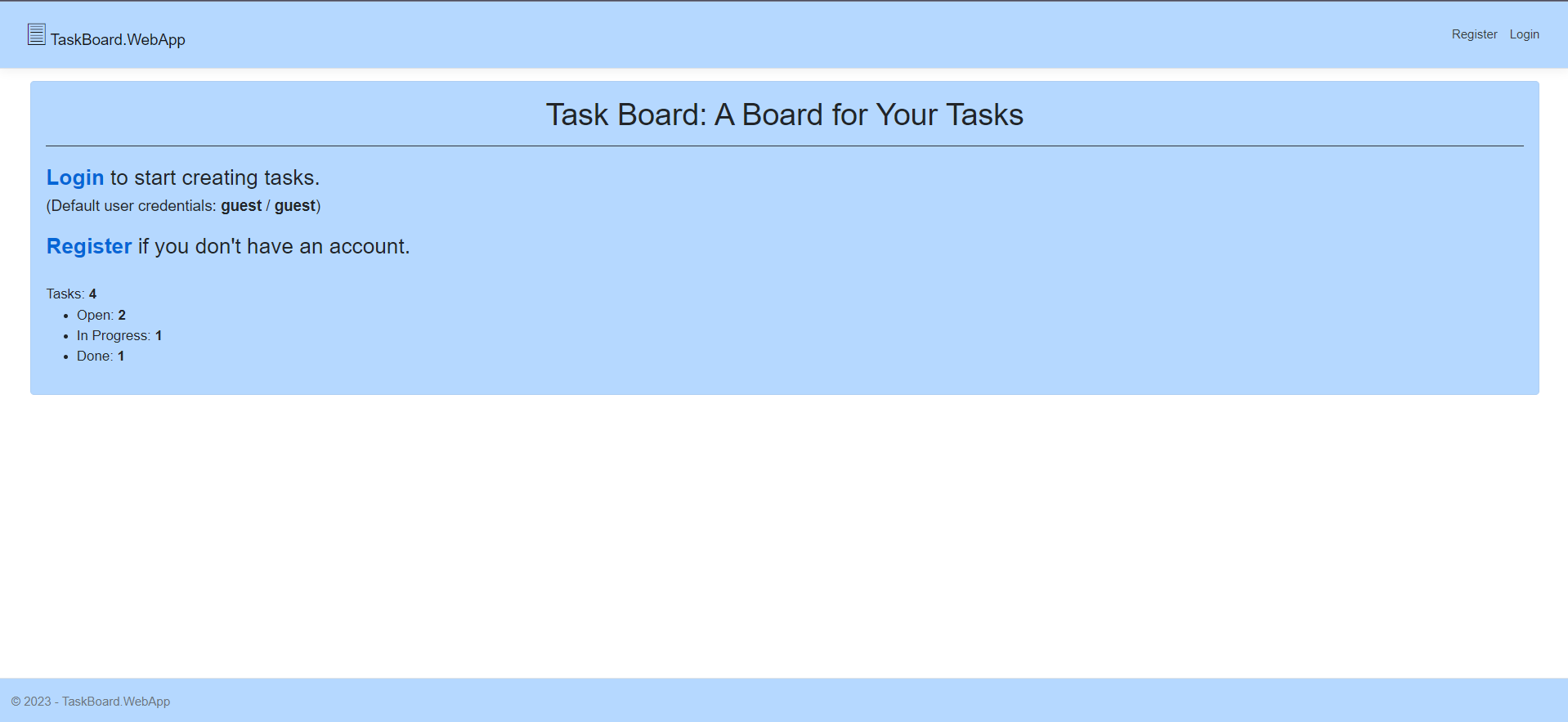
**7.** When opened the json file will look like the picture below. Your task is to change the dots in User ID and password, with those that you created for your MYSQL server.



**8.** When done, locate the play button and run the app

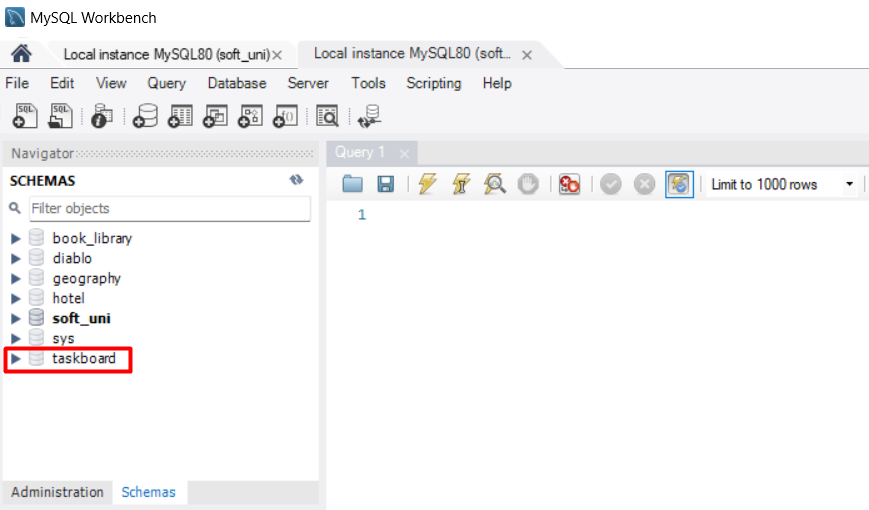


**9.** This will start the "good old" TaskBoard, locally on your machine.

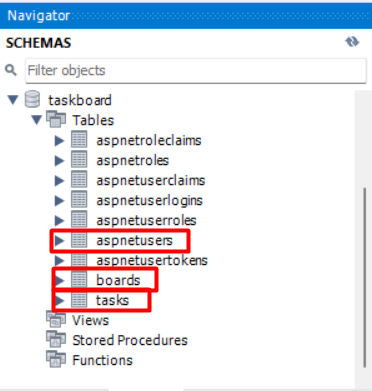


## 3. Browse the "taskboard" database

While the TaskBoard is running, open MySQL Workbench, connect to Server and observe the Databases (Schemas) that you have. You will notice that the taskboard schema is already created.



### 3.1. Explore the tables in the Schema

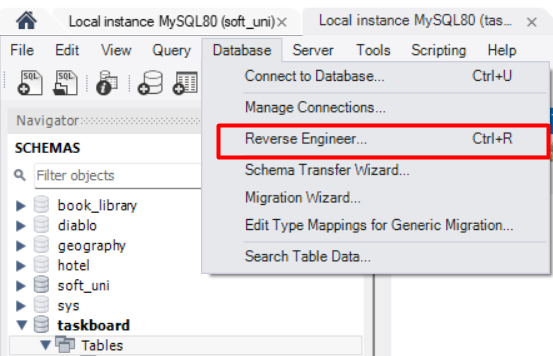
Mostly, we will use aspnetusers, boards, and tasks tables, but feel free to explore the other tables as well.  


Try Schema Inspector (right click on the schema and find it in the context menu)

Try Alter Table (right click on some table and find it in the context menu) to see how the table is structured, what columns it has, what constraints.

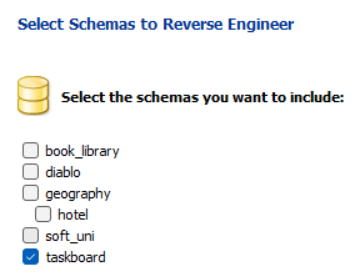
### 3.2. Try the Reverse Engineering Function

From the main menu, click Database 🡪 Reverse Engineering and follow the steps

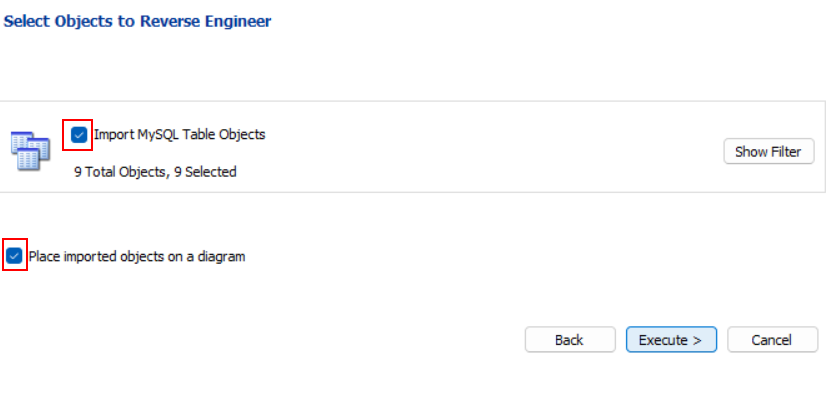


 🡪 NEXT

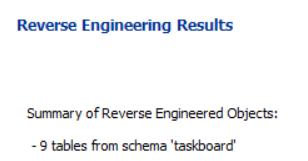
 🡪 NEXT

 🡪 NEXT

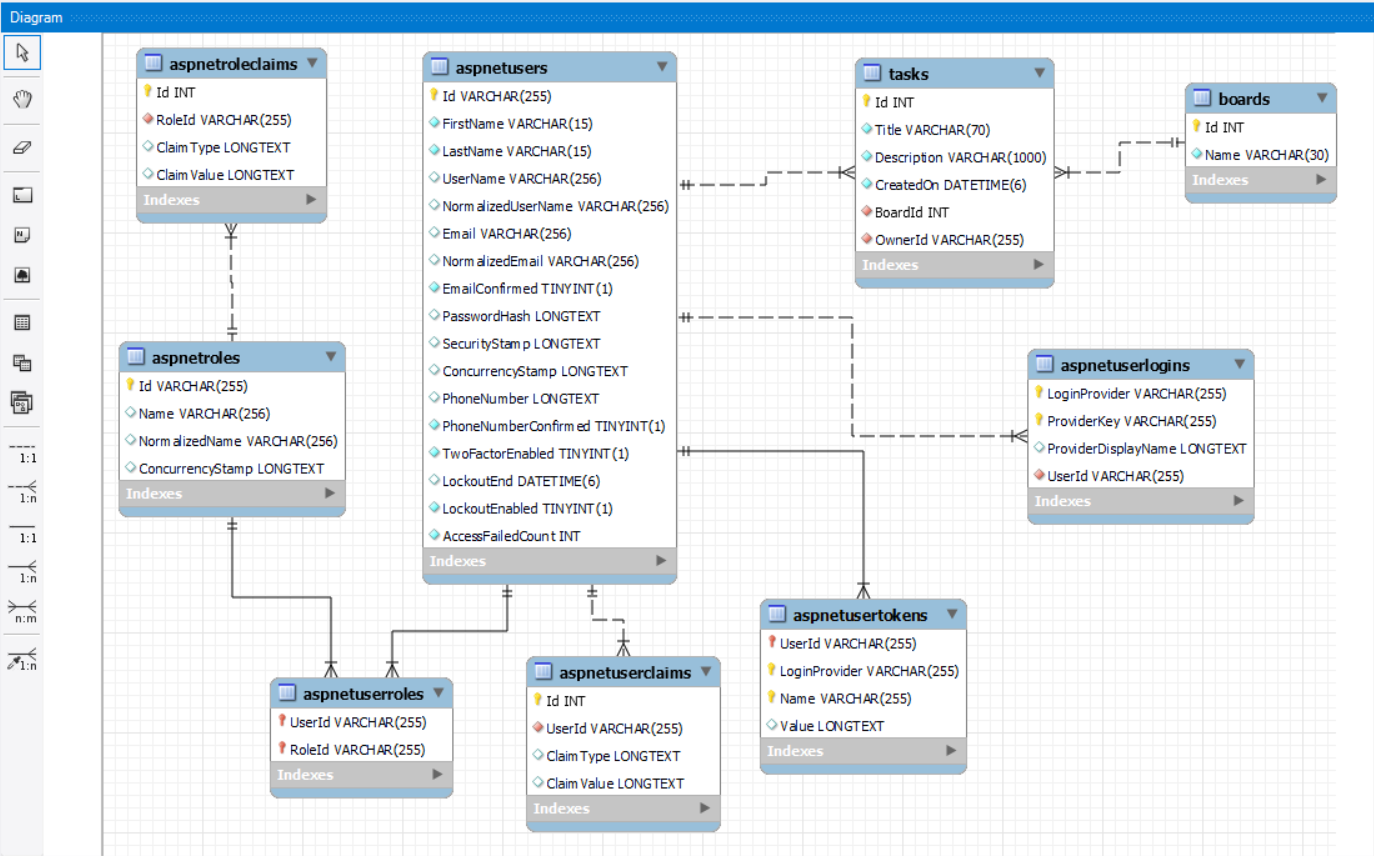
 🡪 NEXT

 🡪 Execute >

 🡪 NEXT

 🡪 Finish

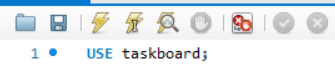
Don’t worry if your diagram doesn’t look quite like the picture, you can arrange each segment by dragging it.



## 4. Working with Taskboard Database

**1.** Create a new SQL tab for executing queries.

**2.** Execute the USE statement to use the taskboard db.

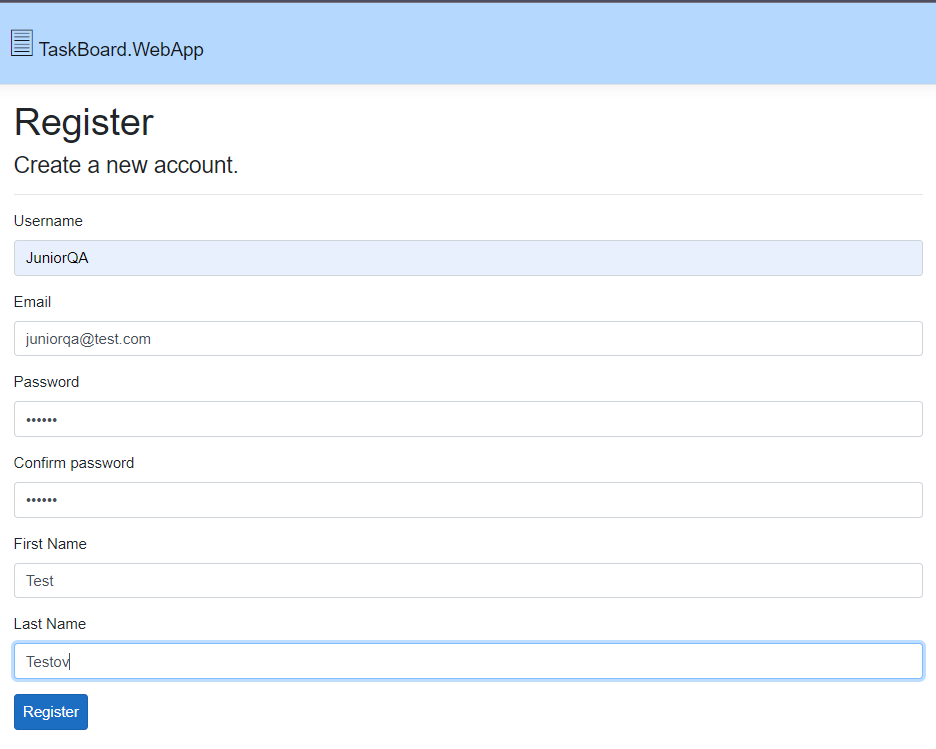


**3.** Execute SELECT all from aspnetusers to see how many users are registered.

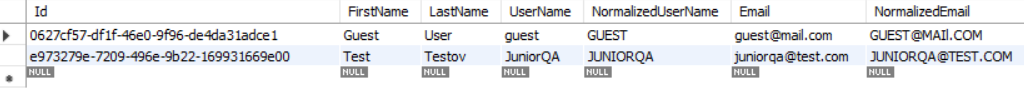


By default, there is only one user. 

**4.** Register a new user via the Web App

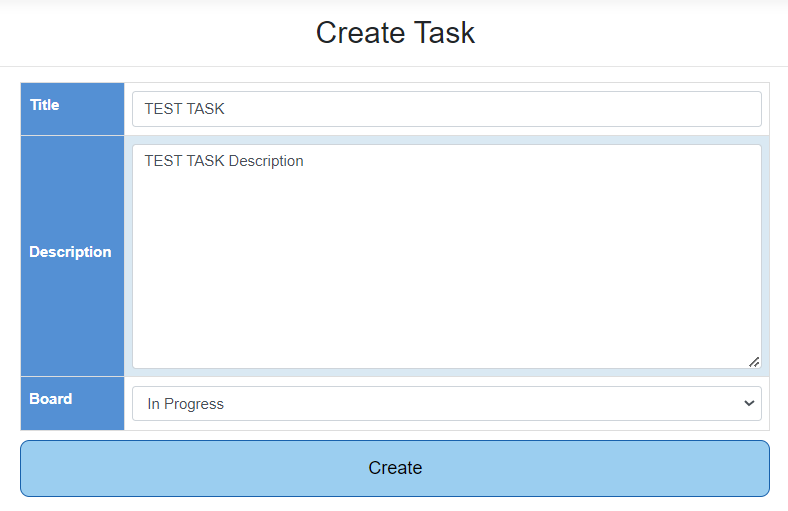


**5.** Go to the database and run the SELECT all statement again. See if the new user is present in the table.

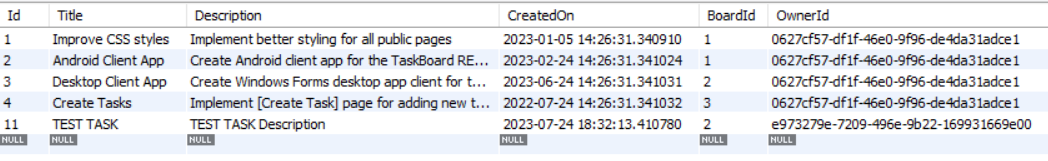


See the Id number of the created user, copy it and paste it somewhere to use it a bit later.

**6. Now, create a new task via the UI.**



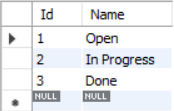
**7.** Execute SELECT all on the task table to see if the new task is successfully created.



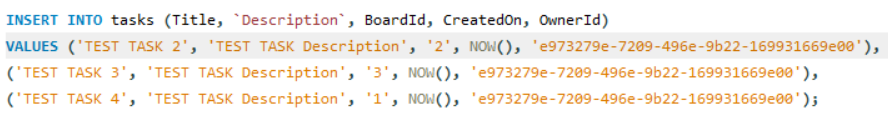
**8.** No let’s try the reverse, to create task in the database and see if it will appear in the UI. In order to do that you will use the INSERT statement, and you will have to populate the following fields – Title, Description, CreatedOn, BoardId, OwnerId.

For CreatedOn – you can use NOW()

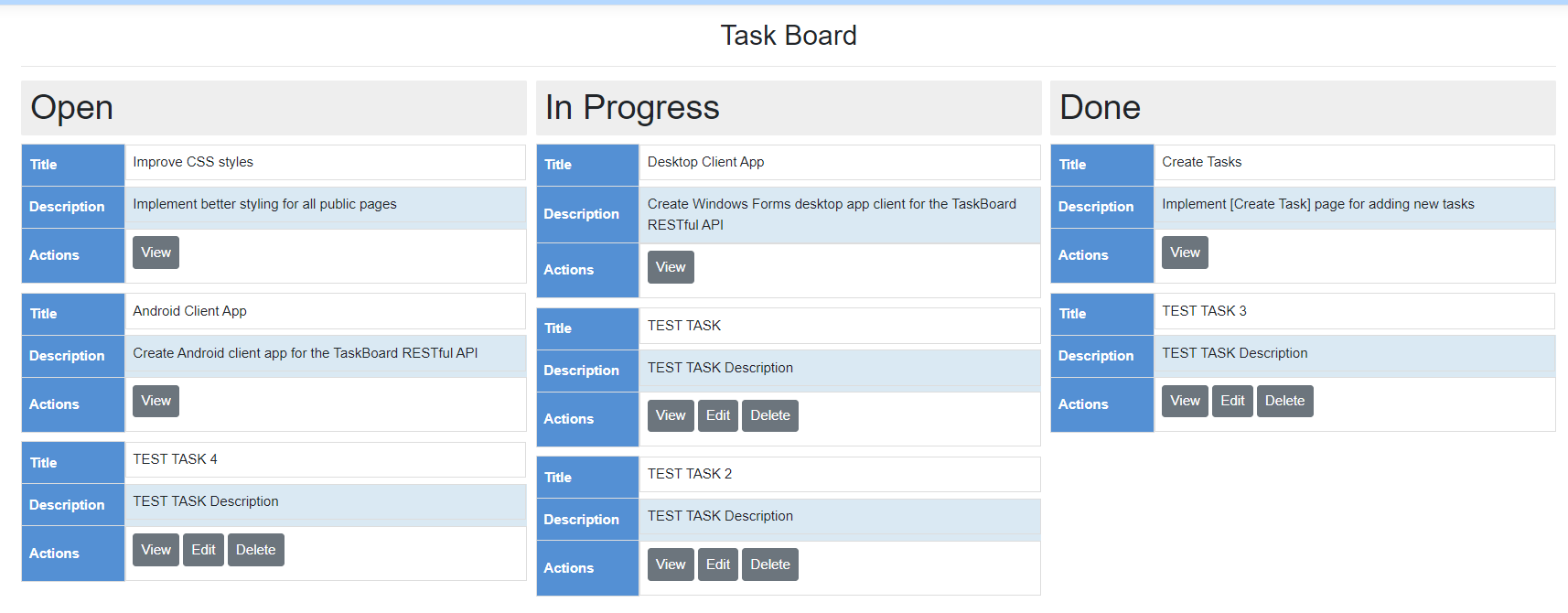
For BoardId – you can run the SELECT all statement on boards table, to find the id of the boards, or just easy guess that Boards Ids are as follows:



For OwnerId you have to use the long Id, that we advised you to store somewhere for later. 😉  
**Actually, let’s create three tasks at the same time**!



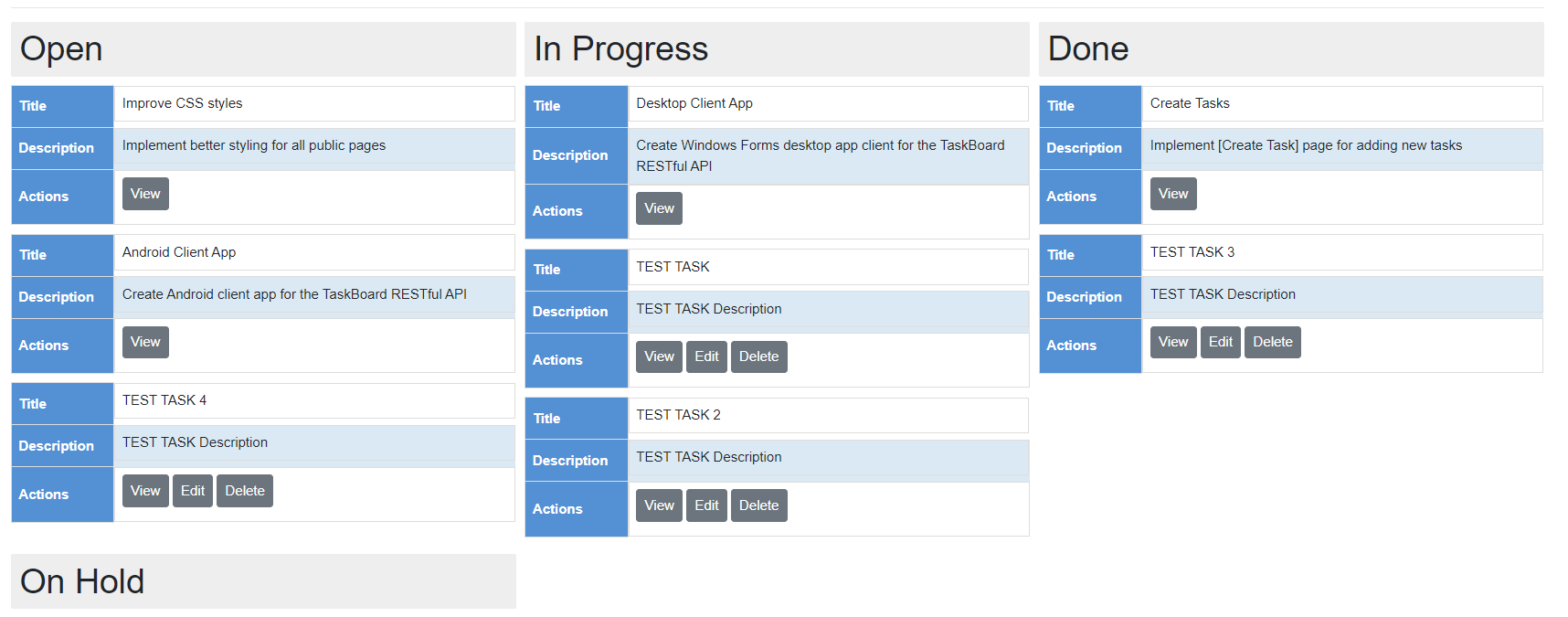
**9.** Refresh the Web App to check in the UI if the tasks are created.

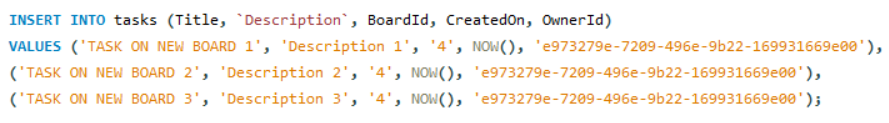


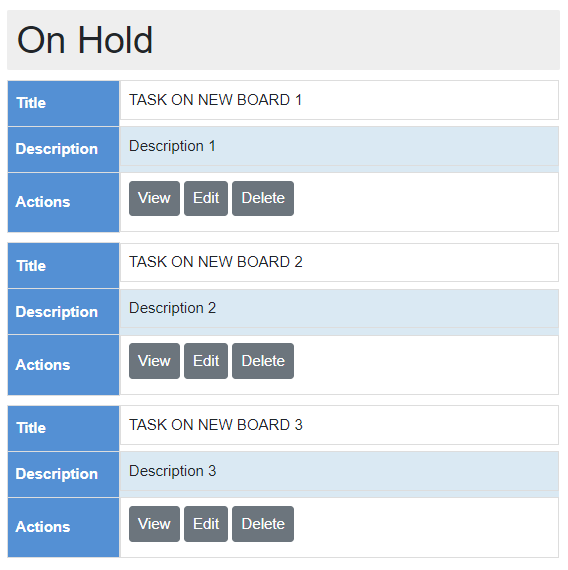
**10.** What will happen if we try to create a new board? We don’t have that option from the UI. Let’s try that.  
Execute again INSERT INTO on boards. You only need the name of the board.



**11.** Refresh the app and Voila! The new board was created.



**12.** Let's add some tasks to it.   


**13.** Refresh and here they are.  


**14.** What will happen if we try to delete a task. Let’s try to delete the last task that we created by name.



Even thou the query is correct; we get an error:  
Error Code: 1175. You are using safe update mode and you tried to update a table without a WHERE that uses a KEY column. To disable safe mode, toggle the option in Preferences -> SQL Editor and reconnect.

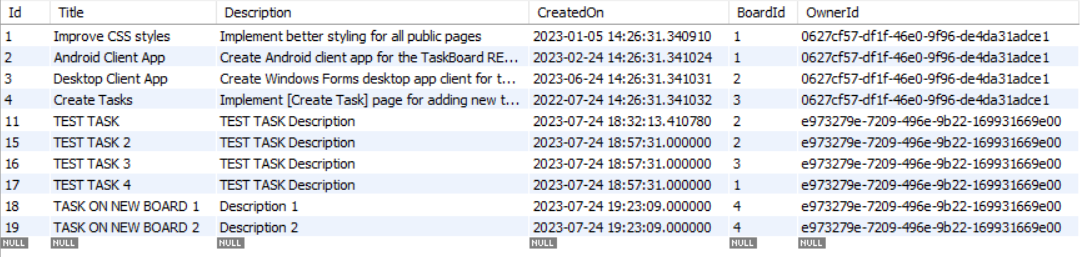
This means that MySQL server is running in "safe update mode," which restricts certain types of dangerous or accidental DELETE or UPDATE statements without a WHERE clause that uses a KEY column to identify specific rows.  
In order to delete the task, we have to use a KEY column, in our case the exact Id, or Where clause with at least two conditions pointing to the exact record. Let’s try both.

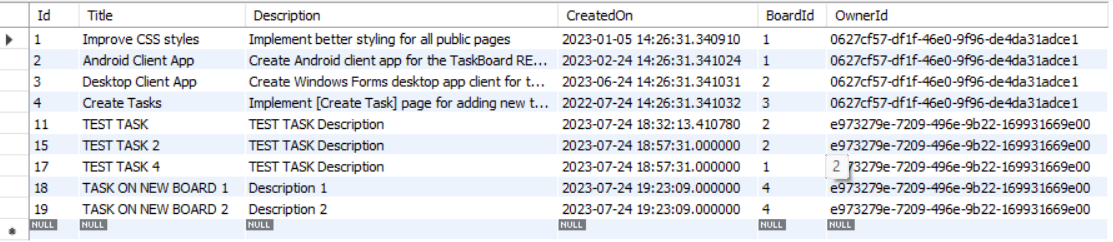
**15.** First, we have to find the Id of the task that we want to delete. Let’s find it using the task’s Title.



**16.** The returned Id in with us is 20, but it might differentiate with you, depending on which task you are searching for. Let’s try to delete it again.



**17.** When we execute SELECT all, as you can see, there is no Task with Id 20.  
  
**18.** Let's try to delete another task, but not by Id, but using the other method with WHERE clause with two conditions, pointing to exact record. Let’s delete TEST TASK 3.

  
And it’s gone!  


**19.** Try a few more:

* Try to filter only Tasks that has the word "task" in the name.
* Compare the result with the same search made in the Web App.
* Try to delete the user and see what happens (hint: you will need his id). See if the tasks created by this user still exist.
* Try to update the name of TEST TASK 4 (hint: The UPDATE statement works similar to DELETE, e.g. you will need the Id of the task or at least two conditions pointing directly to the task you want to delete).