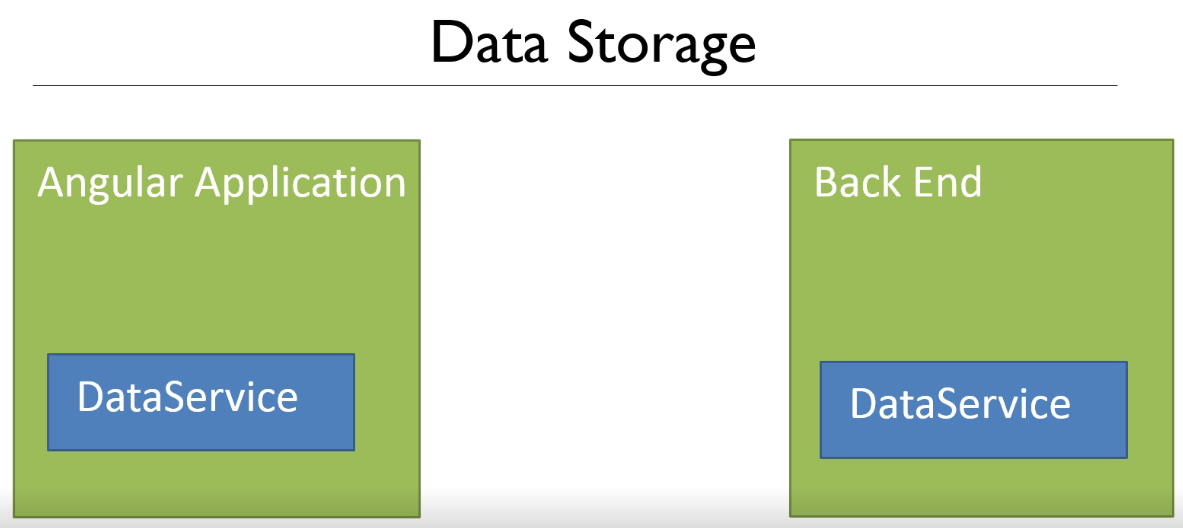
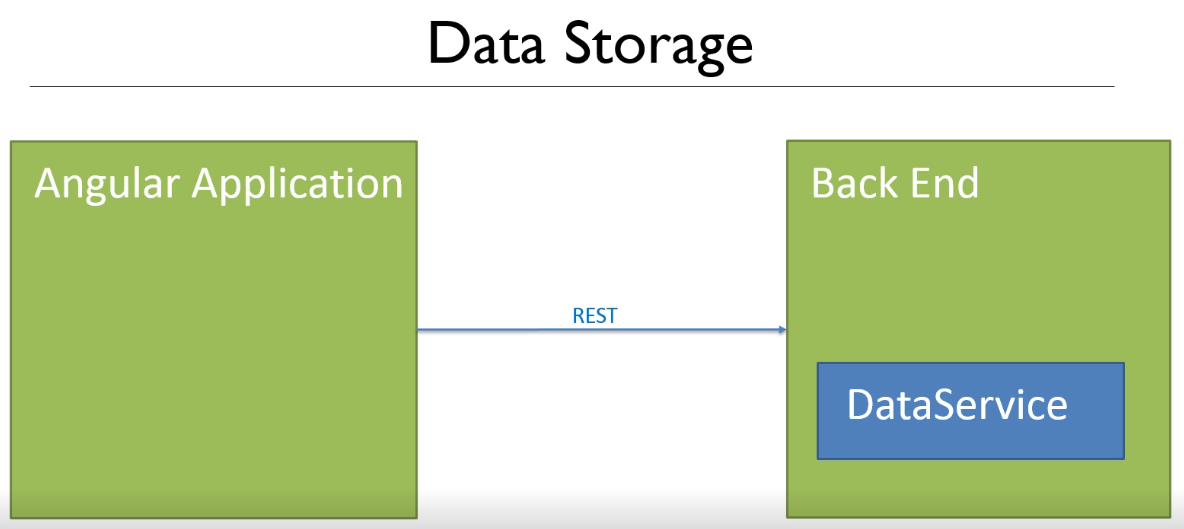
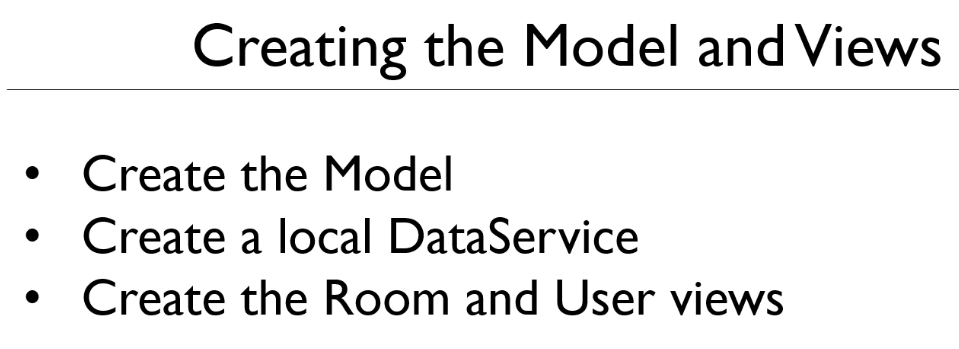
**Models and Views:**



Final Goal:

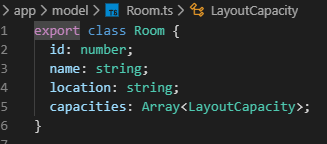




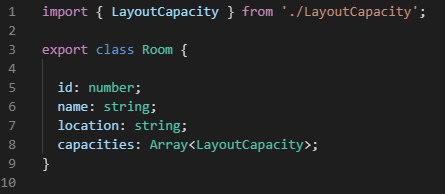
Create the Models

Room.ts class file in the model folder

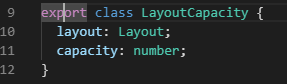




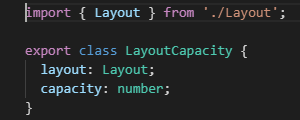
Refactored into a separate class



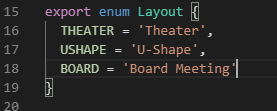
LayoutCapacity.ts class



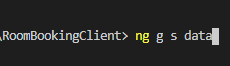
Refactored into a separated class

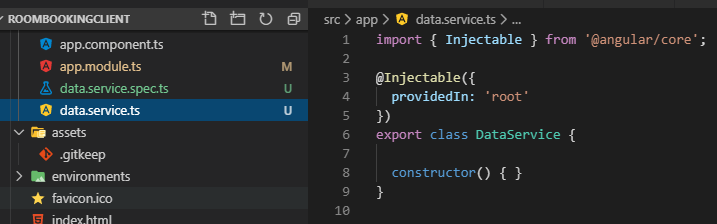


Layout.ts enum



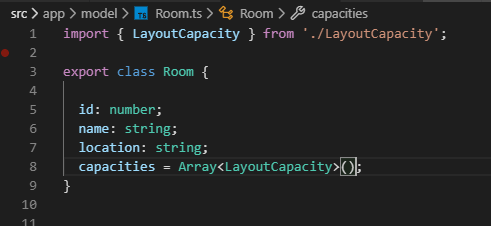
Create the data service



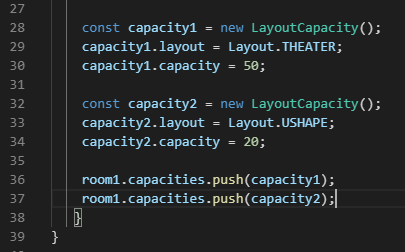


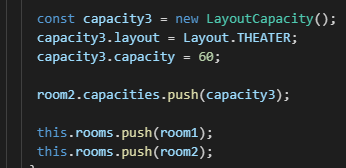


Modify the LayoutCapacity

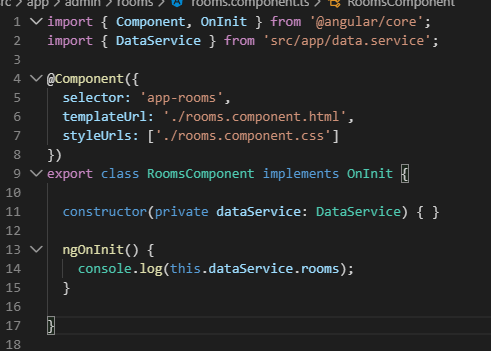


Add capacity

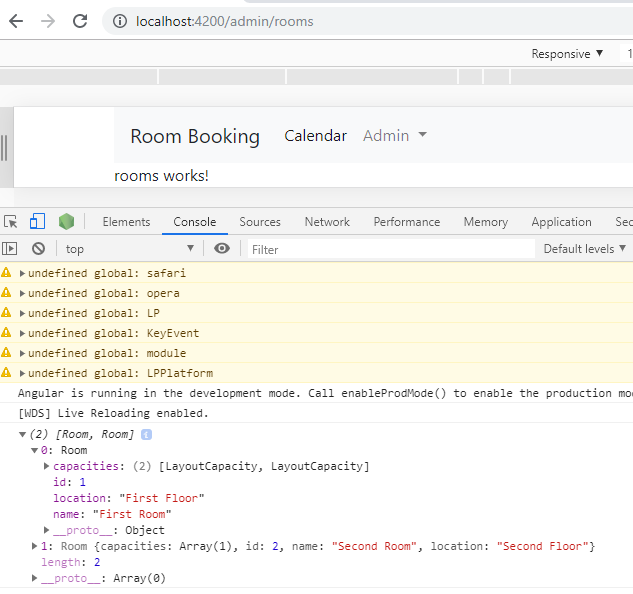




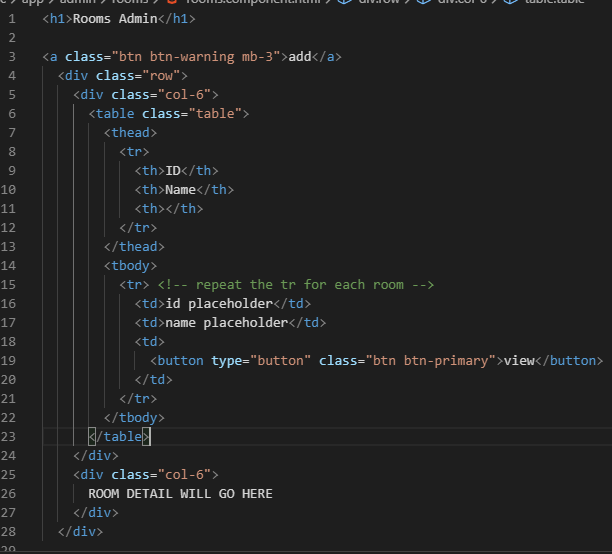
Now, populate the Rooms.component.ts for testing



Click on Admin >> Rooms

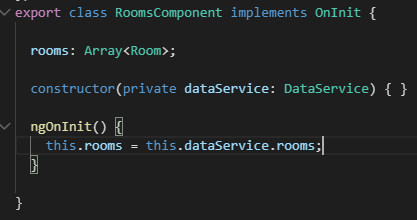


**Binding Data to View and Looping with NgFor**

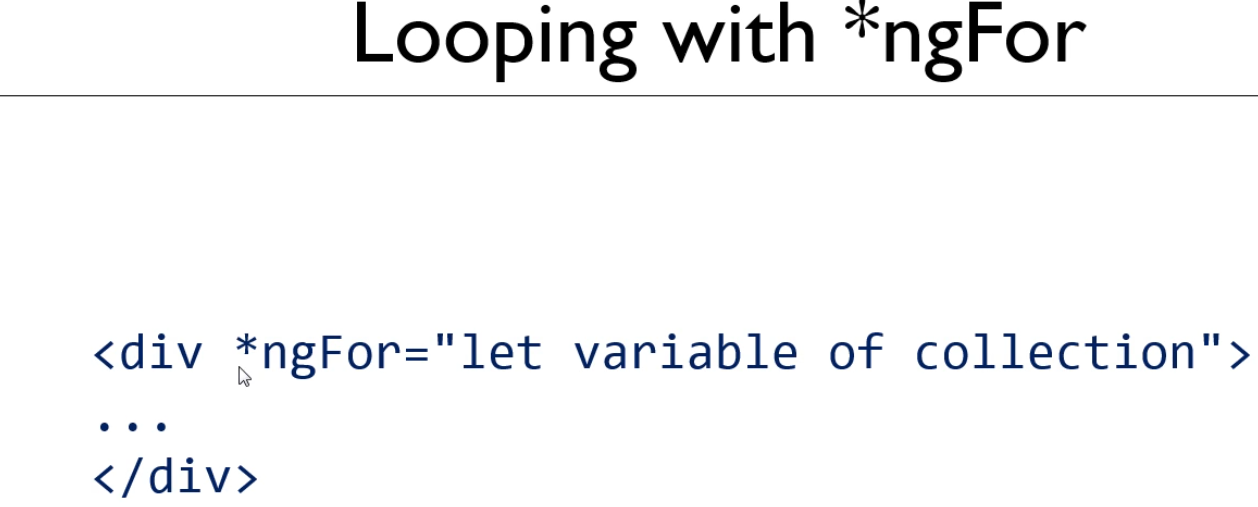


In the rooms.component.ts create a rooms array and populate

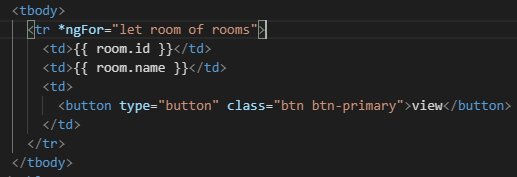
On load or on OnInt()



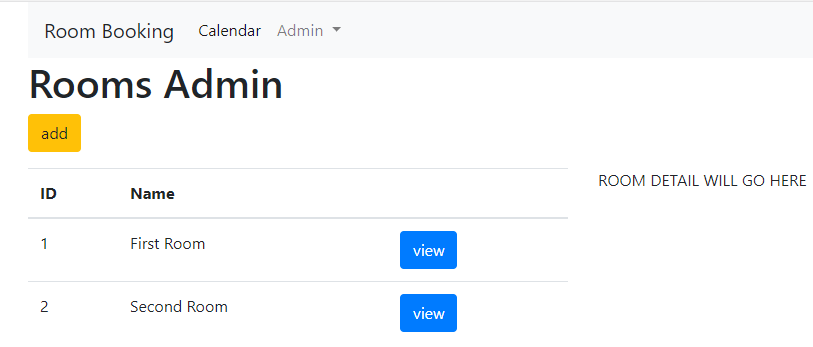
The \* means a structural Directive that will not change the DOM.



In the room.component.html file



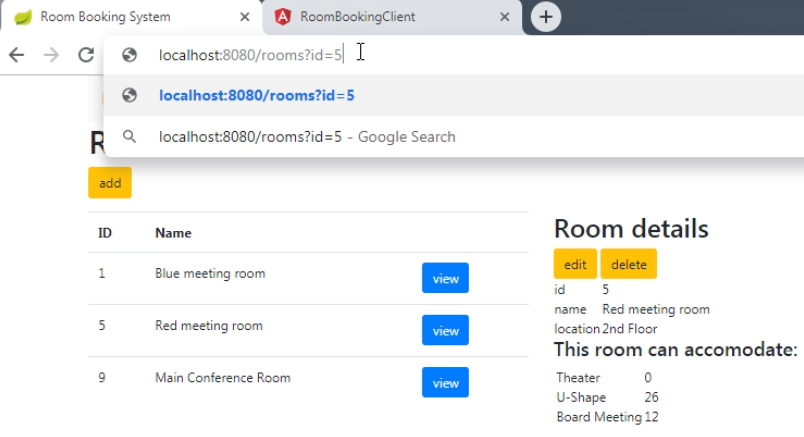
Test it in browser:



**Create a subcomponent:**

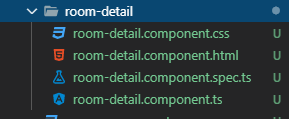
Click on the view, we get the details

In spring-boot it would look like this

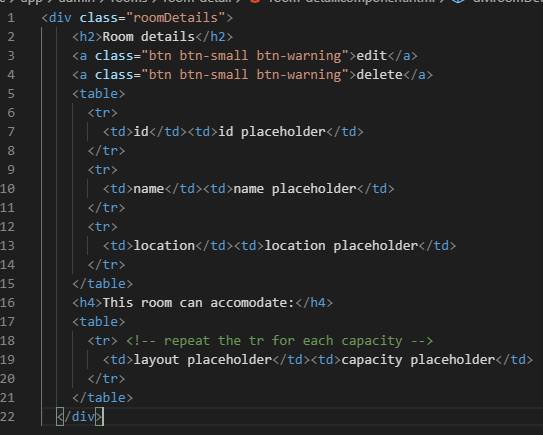


Create a new component for Details and change the url





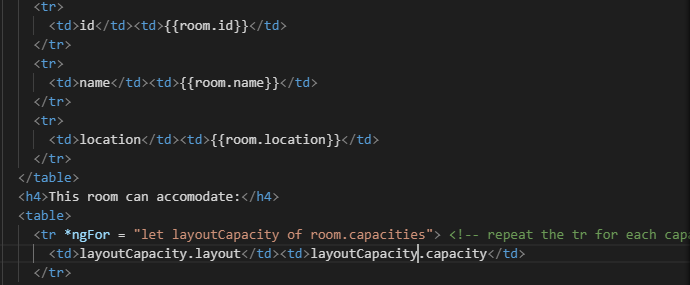
Copy the html from the spring app into the room-detail.component.html file

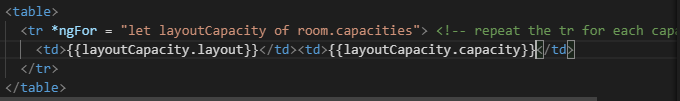


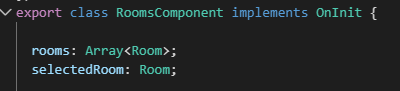
In the class file room-detail.ts file



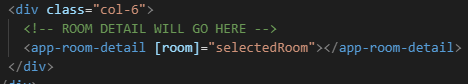
Back in the html fill in the room data:



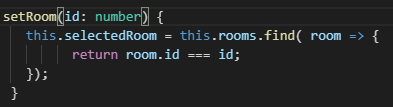




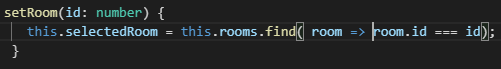
Now use the selectedRoom for the room-details



Let us temporary click event for testing We need a setRoom method



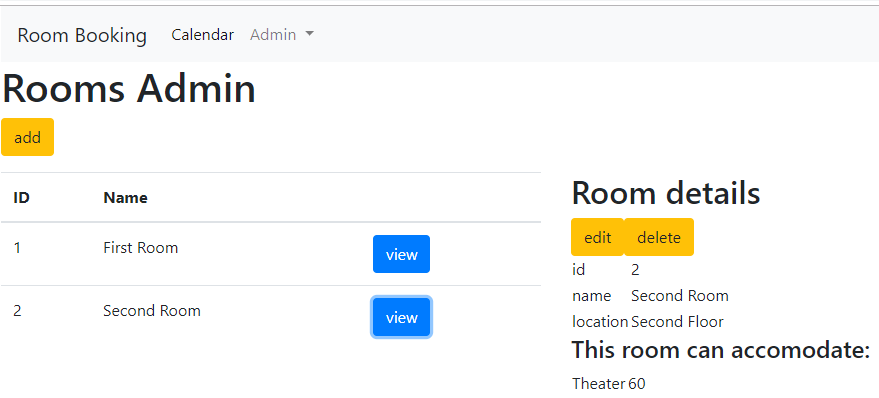
Simplify method by removing the {} and word return

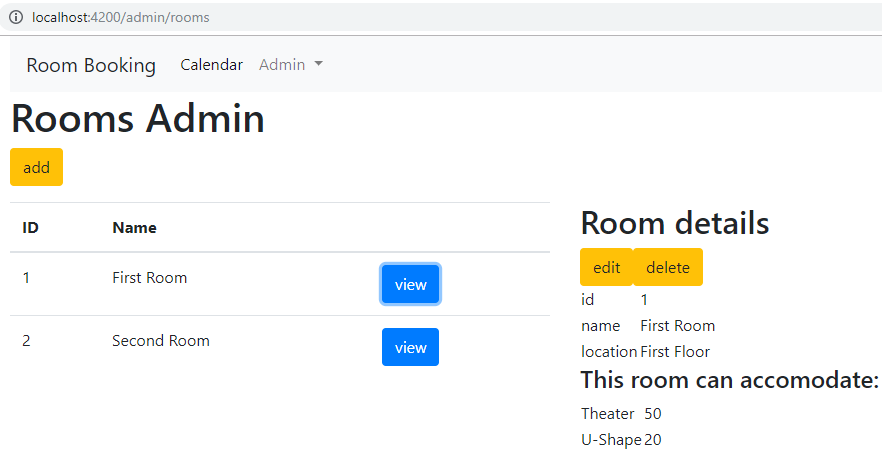


Add the click event to the button

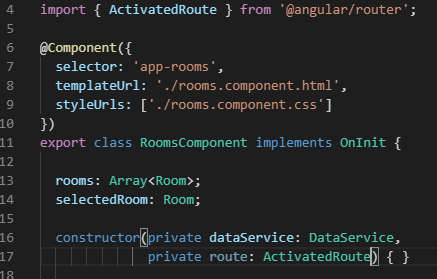


Test the view button click





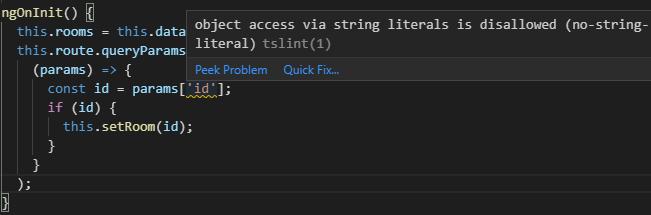
Now, fix the URL so that it changes with the view clicked



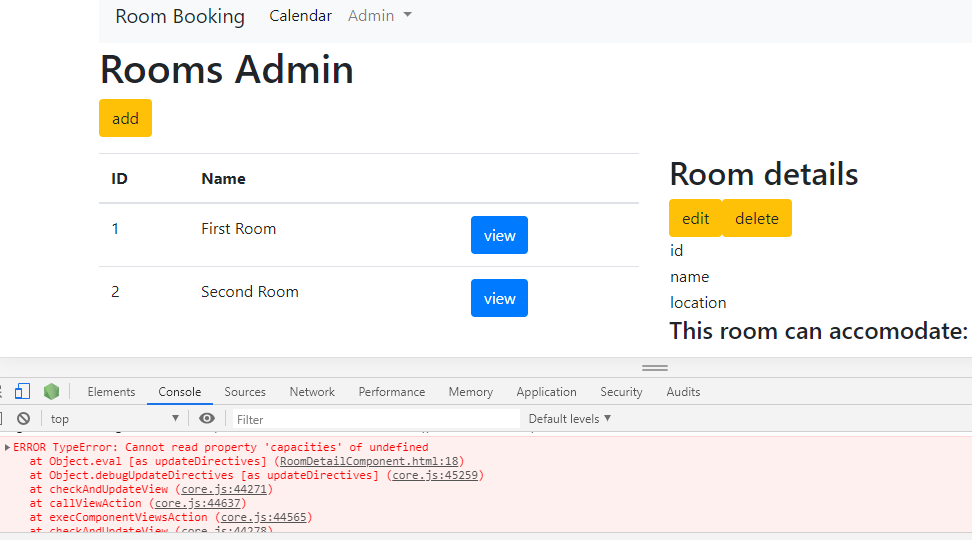
Added the ActivatedRoute to allow inspection of a passed parameter

We can look at the url in query params

Ignore the id error – its not really and error



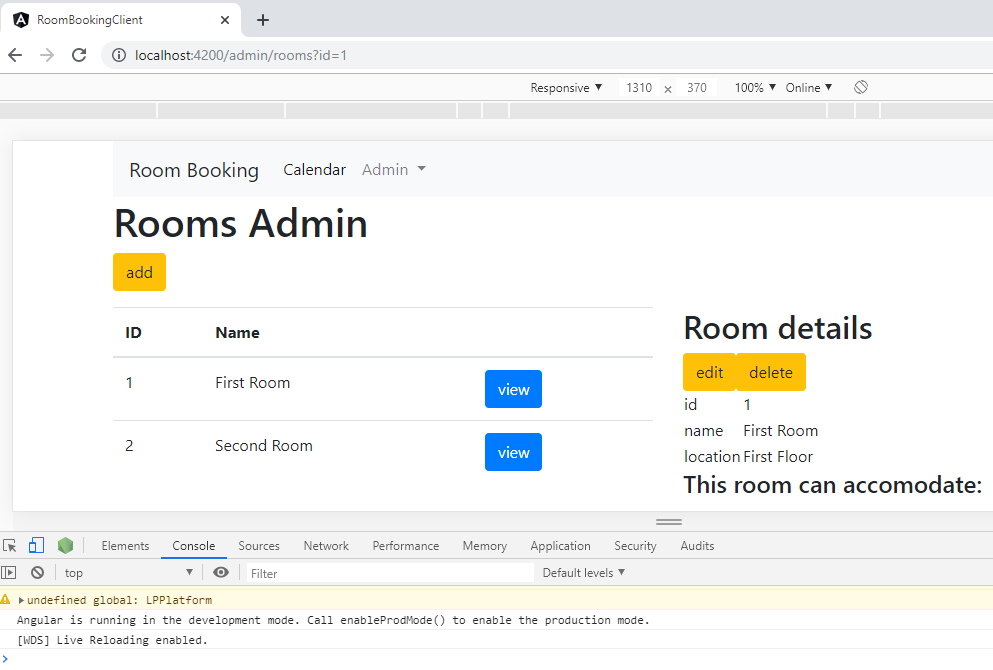
There is a string id versus a number id error



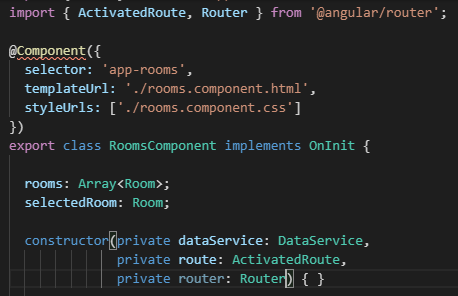
The way around this is to coerce the string id to a number when it is passed

The + plus sign does the trick

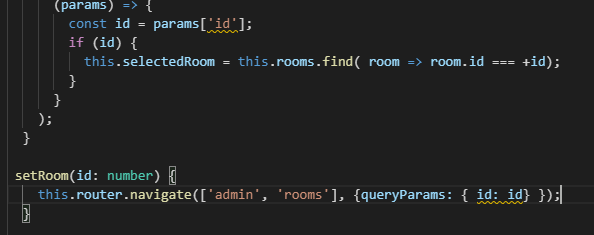


Error is removed

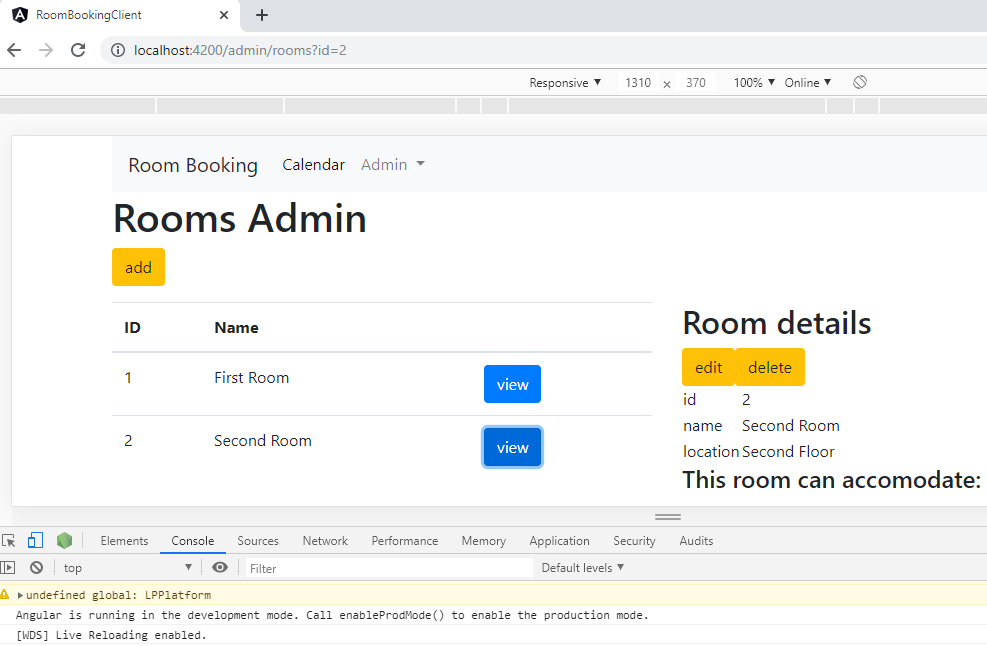
Now, do navigation by adding a router



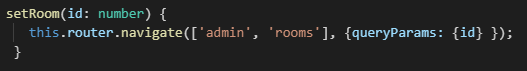
Swap out the call with its definition for id, so the click can perform a url navigation instead.



And the number url changes without error



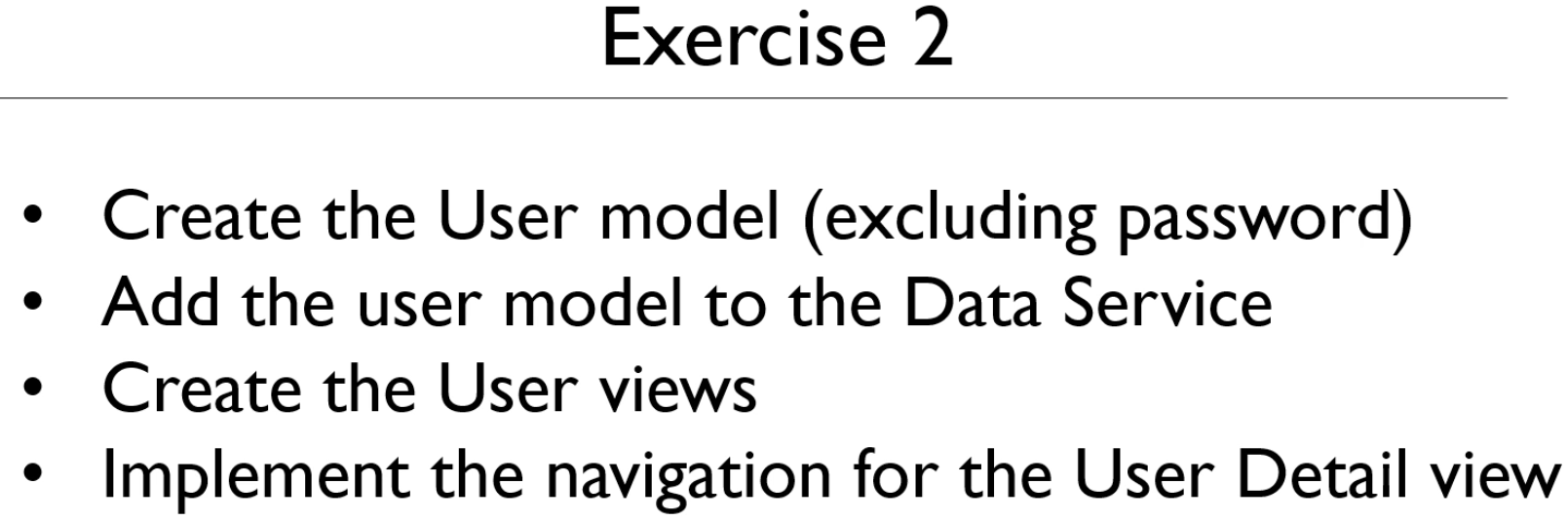
There is a shortcut when the key and value match you do not need the value



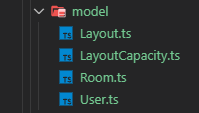
Let us hide the Room details if we have not clicked view



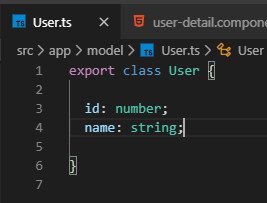
Exercise: Do the USERS object



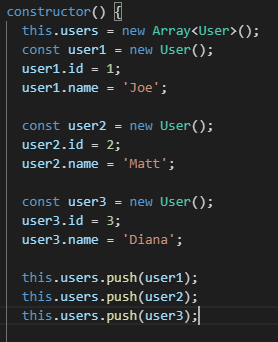
Create the model class User.ts file



In the User.ts file we need 2 properties id and name

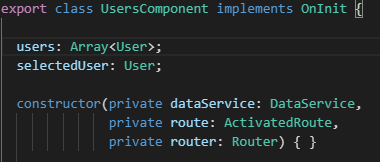


Go to the dataservice to create an array of users



Generate the UserDetail Component

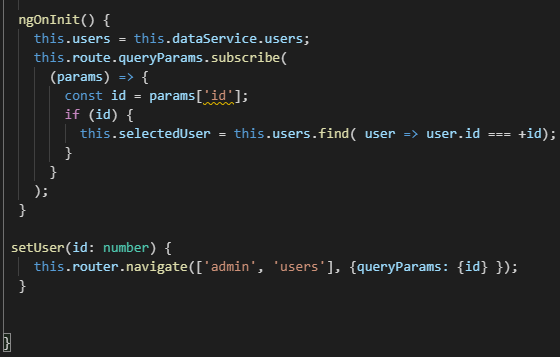
  
Dependency Inject the DataService



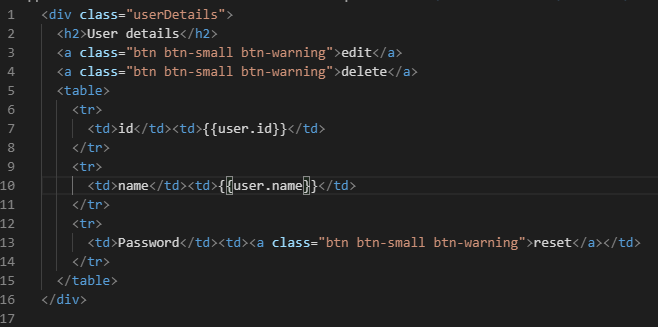
Implement the user.component.html file



Create the setUser method comment out the user-details section until class is implemented



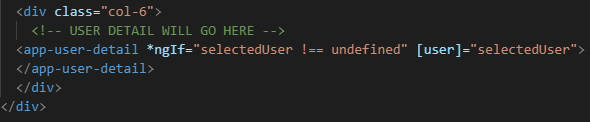
Put the html into the UserDetailComponent html



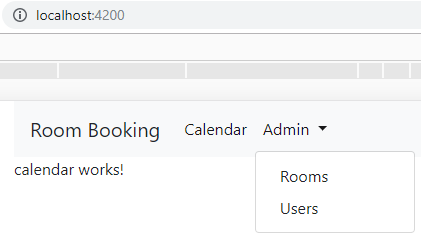
Add the @Input binding to the class file

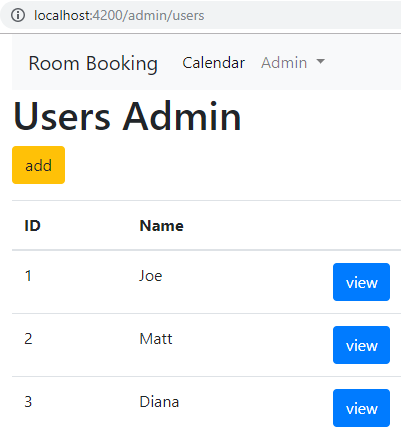


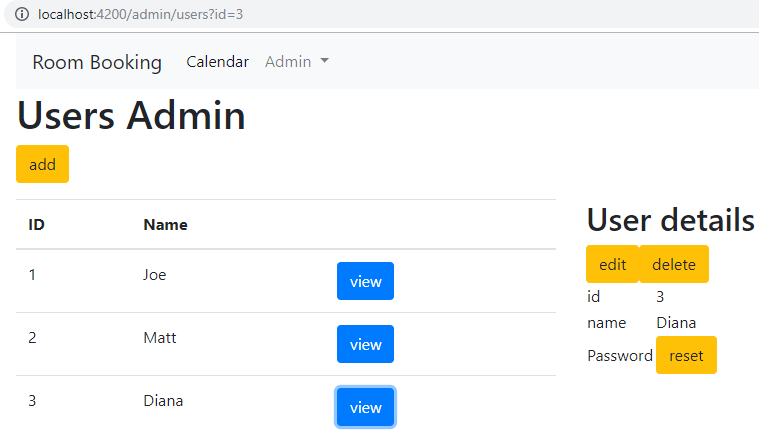
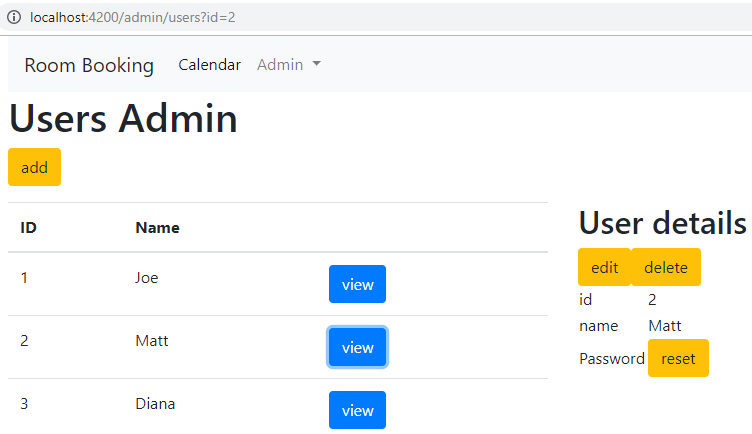
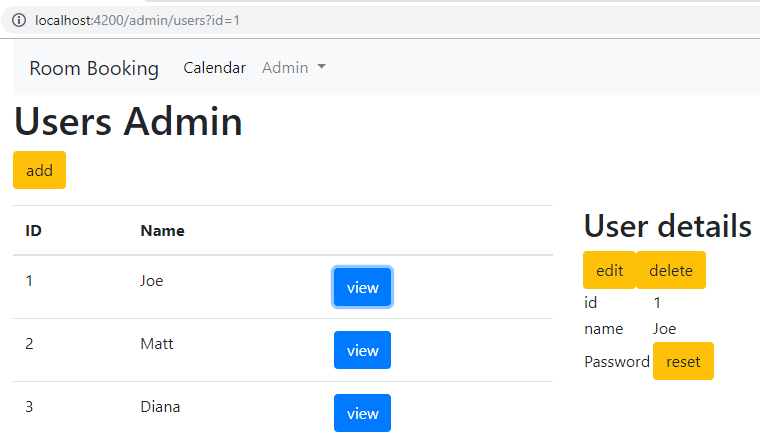
Implement the Details html code in the user.component.html file



Compile with ng serve and test it



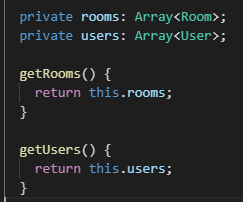




**Why use observables?**

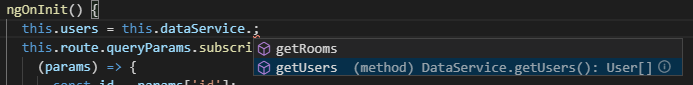
**Fix the dataservice to use getter methods to obtain the data:**

**Setup Encapsulation: make properties private and add accessors**



**Now, replace the rooms with getRooms() and users with getUsers() to implement the change**

**Erase users, type dot Ctrl+Space bar and the code suggestion appears**



**Tab down to getUsers and press Enter.**

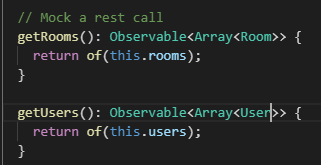


**Look for the this.dataService.rooms**

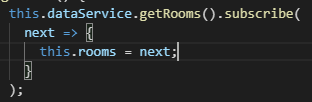


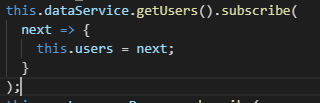
**Rest calls use the Observer design pattern**

**Mock up the subscription is returned instead of an array**



**Now implement the subscriptions**

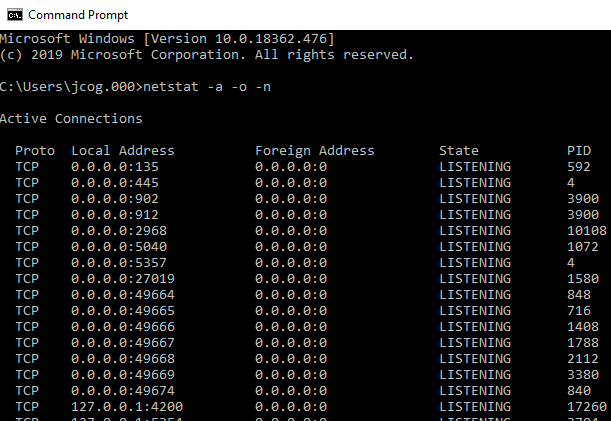




**Check for breakage**



Indicates a browser closed still running the server – solution kill the process at 4200



**The PID is 17260 for port 4200**

**So, to Kill the process type >taskkill /F /PID 17260 Press Enter**

