Workshop 4 – Angular, Models and Functions

This workshop is worth 2.5%

You must demonstrate your completed workshop to your tutor in your lab to receive the marks.

In this lab you'll create a new angular project and create a simple login system. This is similar in function to last week's project except this time we will create it using Angular.

Task 1 – Creating an angular project

If you are using your own computer, you will need to install the angular CLI.

Follow the lecture slides to create a new angular project called "week4tut"

ng new week4 (When asked add Routing)

npm install bootstrap --save (Add Bootstrap to your application)

The later command will update angular.json file with link to the bootstrap stylesheet in node_modules/bootstrap

Task 2 – Create two new Components. One should be called login and the other should be called account. Add a link tag to each page so you can navigate between the two components.

Change the file app.component.html and add a menu that links both the login and account pages. You may use the "navbar" class from bootstrap to create a menu (See bootstrap documentation for usage). The following is an example for the file app.component.html

```
<nav class="navbar navbar-default">
  <div class="container-fluid">
    <ul class="nav navbar-nav">
      <li class="active"><a class="nav-link" routerLink="/">Home</a></li>
      <li> <a class="nav-link" routerLink="/login">login</a></li>
      <li><a class="nav-link" routerLink="/account">Account</a></li>
    </ul>
  </div>
</nav>
<div class="container">
<router-outlet></router-outlet>
</div>
```

Task 3 – On the login page create a log-in form that has an email and password input.

Both fields should be part of the pages model.

Google “bootstrap login forms”. You might find some html code already formatted for bootstrap

On the accounts page add a heading “Accounts Page” and an image.

Task 4 – Create a login function as part of angular that takes the form input and checks the values against hard coded (3 users) values. If the inputs don't match an error messages should be shown. If the inputs match than the page should get redirected to the account page.

At this stage, we do not consider the interaction between angular and server side, and you may set the hard coded values as an json array in the class of loginComponent class in login.component.ts file. In later, we will learn how to implement realistic login function, where the hard coded values are stored in the side of NodeJS or MongDB server.

Task 5 – create a new git repository called "AngularLabs" and commit this week's lab to it using the commit message week 4 tutorial.

- Log into Github and create a new repository called “AngularLabs”
- Initialise your local git repository for your application
- Add an alias for origin for your remote repository
- Pull the remote master branch to your local repository
- Add all files to be tracked
- Commit the local changes with the message “Week 4 Tutorial”
- Push your local repository master branch to the remote.