Angular project to read C#/MVC/WebAPI/SQL Server backend

Follow the steps to build the Angular app.

1. Create the project named \*angular-to-webapi\* and include routing
2. In app.module.ts:
   1. Import FormsModule and add to appropriate decorator key
   2. Import HttpClientModule and add to appropriate decorator key
3. Create the following folders under the \*app\* folder
   1. Create a folder named \*model\* under \*app\* folder
   2. Create a folder named \*service\* under \*app\* folder
   3. Create a folder named \*user\* under \*app\* folder
4. Generate a \*class\* file named \*user.ts\* in the \*user\* folder
5. Create the class \*User\* in the \*user.ts\* file with the following properties and methods (make sure you export it)
   1. ID: number;
   2. UserName: string;
   3. Password: string;
   4. FirstName: string;
   5. LastName: string;
   6. Phone: string;
   7. Email: string;
   8. IsReviewer: boolean;
   9. IsAdmin: boolean;
   10. Add a constructor to the \*User\* class that initializes all properties
6. Create the service \*user.service.ts\* in the \*app/service\* folder
7. Add an import and appropriate decorator key for \*user.service.ts\* to \*app.module.ts\*.
8. Make the following changes in \*user.service.ts\*:
   1. Add an import for \*HttpClient\* from \*@angular/common/http\*
   2. Add an import for \*Observable\* from \*rxjs/Observable\*
   3. Add an import for \*User\* from \*app/model/user\* (user relative path!)
   4. Inject \*UserService\* into the class as \*UserSvc\*
   5. Create a function in the class named \*list()\* that returns a type of \*Observable<User[]>\*
      1. In the body of the \*list()\* function, make a call to the \*get(..)\* function of the \*UserSrv\* passing the following url 'http://prs.doudsystems.com/Users/List'. Return the result of this function call as \*Observable<User[]>\*
   6. Create another function in the class named \*get(id)\* that takes an id as a parameter and returns a type of \*Observable<User[]>\*
      1. In the body of the \*get(id)\* function, make a call to the \*get(..)\* function of the \*UserSrv\* passing the following url 'http://prs.doudsystems.com/Users/Get/+id’. Return the result of this function call as \*Observable<User>\*
9. Generate the \*user-list\* and \*user-detail\* components in the \*user\* folder
10. Make the following changes to the \*user-list.component.ts\* file.
    1. Import the \*UserService\* service
    2. Import the \*User\* class
    3. Create a property called \*users\* of type \*User[]\*
    4. Inject the \*UserService\* into the component as \*UserSvc\*
    5. In the ngOnInit()
       1. Call the \*list()\* function of the user service and subscribe to the results and place the data returned in the \*users\* property
       2. Display the returned data by calling console.log(this.users)
11. Make the changes to \*user-list.component.html\* to render the data
    1. Add one extra column called \*Action\* and put a link to a route to call the \*user-detail.component\* and pass the Id of the user id.
12. Make the following changes to \*app-routing.module.ts\*:
    1. Import \*user-list.component\* and \*user-detail.component\*
    2. Add the following to the \*routes\* array:
       1. On startup, navigate to the \*/users/list\* route
       2. On route \*/users/list\*, load the \*UserListComponent\*
       3. On route \*/users/detail\*, accept a route parameter named \*id\*, load the \*UserDetailComponent\*
       4. On any other route, load the \*UserListComponent\*
13. Make the following changes to the \*user-detail.component.ts\*
    1. Import \*Router\* and \*ActivatedRoute\* from @angular/router
    2. Import the \*UserService\* service and \*User\* class
    3. Create a property named \*user\* of type \*User\*
    4. Inject the \*UserService\* as \*UserSvc\*, \*Router\* as \*router\*, and \*ActivatedRoute\* as \*route\*
    5. In the ngOnInit()
       1. Create a local variable named \*id\* as a string
       2. Get the router parameter \*id\* by calling: \*this.route.params.subscribe(parm => id = params[‘id’])\*
       3. Use the variable \*id\* to make a call \*get(id)\* call to the UserService subscribing the result and storing it in the property \*user\*
14. With the data being retrieved by both component
    1. Modify the \*user-list.component.html\* to display the list of user records
       1. Include an \*Detail\* like to like to the /users/detail/:id via routerLink
    2. Modify the \*user-detail.component.html\* to display the specific user record based on the user clicking the appropriate \*Detail\* link on each user.