STARS Web App

CMPT 561 - Homework 2

This is an individual assignment worth 5%. Due date is by 11pm Saturday 14th November 2015. Push your work to Github as you make progress.

1. Requirements

The aim of Support & Tracking of At-Risk Students (STARS) is to manage follow-up actions to help students who are at-risk and improve their success rates. STARS allows the user to keep track of the actions and activities done the course Instructors, the Program Coordinators and the Academic Advisers.

You are required to develop and test a working version of **STARS** using JavaScript (ES2015), TypeScript and jQuery. Data should be stored and managed using the browser's offline storage. The initial data should be loaded using ajax from data files provided. You need to create a modular solution using TypeScript and ES2015 features particularly modules, classes, promises, arrow functions, etc.

You are encouraged to use the base solution provided on GitHub under Examples/Homework2 otherwise you can continue extending your homework 1 but you need to provide similar/better features and user experience.
Your implementation should deliver the following use cases:

- **Login and Logout**
- **List of students at risk** under the care of the current user. Upon login,
  - The Coordinator and the College Adviser by default get all the students belonging to the programs they manage. But they can filter the list of students by program or they can return to the default view.
  - The Instructor by default gets the list of students in the courses that the instructor is teaching in the current semester. But they can filter the list of students by course or they can return to the default view. Note that a student may appear multiple times in the list if they are taking multiple courses with the instructor. Hence, you should include the CourseCode when displaying the list to the instructor.

  From this list, the user can either add a follow-up action or view the list of existing actions. The user can add a follow-up action either for a particular student or for list of selected students.

- **Add Follow-up Action** to allow the user to select a Student or a list of students, enter the Action date (by default this should be set to Today's date) and other Action details as described in the table above. When the action is entered by the instructor, the action CourseCRN should be set the CourseCRN of the selected student.
- **List Actions** to display actions for a particular student by default done by the current user. The user can request getting actions done by a particular adviser type or get all the actions. The user can then get the action details and attachments. They can also add an action.

  From this view, the user can navigate backward and forward through the list of students to get their follow-up actions or they can get the follow-up actions for a particular student without leaving this view.
- **View, Edit and Delete Follow-up Action.**

2. Grading rubric

Criteria	%	Rubric Below will be used
Complete and correct design and implementation of the requirements:		
• Login and Logout	5	
• Students List	25	
• Add Action	20	
• Actions List	22	
• View, Edit and Delete follow-up action	20	
Testing Word documentation with evidence of correct implementation using snapshots illustrating the results of testing.	8	
Total	100	
Copying and/or plagiarism or not being able to explain or answer questions about the implementation	-100%	

Ground Rules

- All assignments **must be your own original work**, not based on the work of other students, online examples/tutorials, or any other material from any other source. Any assignments found to be based on work other than your own will automatically be given a **grade of zero**, and may lead to further disciplinary action as per QU policy.
- All assignments must be submitted electronically to Github. You should push your work to Github as you make progress. Late submission policy: 10 points deduction for each late day and 0 after 3 days.