COSC2276/2277 Assignment part 3 specification

Deadline	**Monday 10.02.2020 (9 am AEST)**
Face2FaceDemo	**Mon, Tue/ 10 & 11.02.2020** (schedule tba)
% allocated to this assignment	**30**
To be attempted (assignment)	**Individually or a group of 2**
To be submitted via:	**Canvas**

3.1 IMPORTANT:

a. You must submit assignment files before the demo and then attend the face to face demo to get the assignment marked.

b. Demo will happen outside lab. Without a face2face demo, no marks will be awarded. **You must submit your code via Canvas prior to the demo.**

c. You will be marked on the use of GITHUB | code development process | processes used and group dynamics | how you fare during demo. The final marks will be weighted according to your % contribution in group. If you cannot explain your own code during demo, you will receive a ZERO for the whole demo.

d. You will **not** be using your GITHUB for this assignment, you will be invited to be a part of GITHUB account created for you or your group.

3.2 PLAGIARISM:

All assignments will be checked with plagiarism-detection software; any student found to have plagiarised would be subject to disciplinary action. Plagiarism includes:

- CONTRACT CHEATING: paying someone to do your work
- CONTRACT CHEATING: getting someone else to write the test or attend demo
- submitting work that is not your own or submitting text that is not your own
- copying work from/of previous/current semester students
- allowing others to copy your work via email, printouts, social media etc.
- posting assignment questions (in full or partial) on external technical forums
- sending or passing your work to your friends
- posting assignment questions on technical forums to get them solved

A disciplinary action can lead to

- a meeting with the disciplinary committee
- a score of zero for the assignment
- a permanent record of copying in your personal university records and/or
- expulsion from the university, in some severe cases

All plagiarism will be penalised. There are no exceptions and no excuses. You have been warned. For more details please read RMIT's page on Academic Integrity at https://www.rmit.edu.au/students/student-essentials/assessment-and-exams/academic-integrity

3. 3 Scope:

The aim of this assignment is to extend the ASP.NET Core **Internet Banking website** from assignment 2, with .NET Core 3.1 framework, using Microsoft Visual Studio 2019 with a Cloud SQL Server 2019 backend & Angular.

3.4 Overview:

The aim is to extend the website created in assignment 2.

In doing so, you may have to make changes in the database used for assignments 1.

You will add the additional features ASP.NET Core such as: security features, Web API and an Admin portal created.

It is possible to do the whole assignment with ASP.NET Core.

Please read tasks for further details.

You must read the rubrics for more details.

3.5 Business Rules:

Refer to assignment 1 specifications for these.

6 Tasks and Marks allocation:

PLEASE ALSO READ rubric for detail, to see where marks can be lost

Wherever you use a form, you must perform validation, an application without validation is of ZERO commercial value.

Part A: **ASP.NET Core**

a (6 marks) Add the following security -related features to the users' Internet Banking website:

1. auto logout after 1 minute of inactivity on any logged-on page,

2. locking of accounts after 3 unsuccessful login attempts, once the login has been locked you should display some error message to that effect; you should reset the login status after 1 minute so that the user can log back in and,

3. customised error pages, the idea behind this is never to show stack trace or any irrelevant information to the end user. The customised pages should cater to various HTTP response error scenarios.

b (12 marks) Create an ASP.NET Core Web API that will provide the admin the access to the database and the admin portal features. The methods of this well-documented API will be determined by the next specification. *Please read part c.*

In order to receive full marks for this specification you must use a repository pattern, write proper comments to explain the various methods of the API.

The API may be built in the same or a different project as Admin portal.

Part B: can be attempted in **ASP.NET Core** OR **Angular 8**

c (12 marks) Create a separate Admin portal as described below:

- Admin should login via a separate URL as the user website

- You must use a customised secure URL for admin portal

- Admin username and password need not be stored in a database at this stage

- After a successful login (**username = admin, password = admin**), admin should be able to

    1. view transaction history for a user: admin should be able to generate a table displaying the transaction history for a user within a specified time period. Admin should be able to search on transaction histories of all the users in the database.

    2. generate appropriate graphs for the above transaction history to show it in a visual manner. *We have covered something similar in tute sheet 11,* but as rubrics says you will need to display three types of charts to get full marks. Admin should be able to generate charts for all the users.

    3. update/delete/modify user details: these are profile related details.

    4. lock and unlock user account(s), a locked user account automatically unlocks after 1 minute. Once a user account has been locked, you must demo this by means of showing user's inability to log into Internet banking website.

    5. block/unblock a scheduled pay. Once a scheduled pay has been blocked, you must demo it for the corresponding user. The blocked scheduled pays should be displayed as "blocked" in the list of bill pays. The blocked bill pays should not run.

All the above features must use Web API for accessing the database. Admin page should MINIMAL logic code for doing above. It should mostly rely on the API.

Only the pages with EXCELLENT user interface will receive higher marks for this part.

You will also be marked for the effective use of GitHub, web design and HCI principles and code elegance.

3.7 Coding Standards:

- Read the C# coding standard from the following website: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions
- Remember that there are too many to be followed, implementing 6-10 of the standards will be a job well done.
- Do not force yourself to implement every OO feature that you have learnt, use the features wisely and if needed.

3.8 Consultations:

We will hold regular consults for this assignment, so keep posted to the announcements in Canvas.

3.9 Submission Procedure:

Each submission must include a README.txt file containing your full name, Student ID and any other relevant information (if you are working in a group, then please mention the details of your partner).

Zip the entire solution project folder (with README.txt file included) and submit it via Canvas as a single zipped archive.

3.10 Late submissions and Extension-related information:

A penalty of 10% per day of the total marks for each assignment will apply for each day a submission is late, including both weekdays and the weekend. After 5 days, you will receive zero marks for that assignment. Email your lecturer (shekhar.kalra@rmit.edu.au) for extension related queries.