General Project Details

The SwollenCoffee Company has hired your team to develop a new loyalty experience for their 325,000 customer strong base. The SwollenCoffee Company is an ambitious organization that has rapidly expanded their business from a single mom and pop location to a regional powerhouse that encompasses 10 locations in 7 separate cities.

SwollenCoffee would like their new application to become a marketing tool to help encourage repeat customer interactions. With that in mind, SwollenCoffee would like to be able to create and end various loyalty campaigns that are available to customers based on their preferred location. These campaigns will be based either on the number of orders during a time period or on the amount of total purchases during a time period.

As a marketing tool, SwollenCoffee would like to capture a customer's contact information and demographics to use in future campaigns. To assist their management team, SwollenCoffee would like to have access to rich, web-based reporting tools that will help them make data-driven decisions.

The loyalty application should provide an interface for the customer to display their membership id as a QR code that can be scanned by the cashier to tie purchases to the member. The customer should also be able to update the captured contact information and demographics as well as their preferred store from the application.

As the application should be quickly available for use, the customer should be able to authenticate into the application and then a persistent session should remain for authentication without a username/password combination.

Although the SwollenCoffee Company does not have a development team in-house, their Chief Technology Officer has plans to hire a group in the near future and would like to standardize on a modern technology stack for all development initiatives. This technology stack is heavily dependent on Microsoft Azure cloud technologies. The data layer of the application should utilize Microsoft SQL Server and all connection strings should be based on the server/instance paradigm rather than based on the port. Any web services utilized by the loyalty application must be written in .NET with C# be the scripting language. Currently, SwollenCoffee prefers to deploy web services as RESTful endpoints with a serverless approach - Azure Functions. The frontend, presentation layer should be written with HTML, CSS, and Javascript. Since the application must be responsive, SwollenCoffee requests the use of Bootstrap as the presentation framework. Other Javascript libraries such as jQuery and SweetAlerts should be leveraged to provide user functionality and accessibility.

DS3870 Specific Instructions

Each student's score is made up of several parts:

- participation in classroom discussions
- peer evaluations of the effort you put into the project
- points awarded for effectively meeting minimum requirements
- project ranking as compared to your peers

About the ranking component of the project - Your project will be reviewed and ranked by a panel of judges including the instructor. Points will be rewarded based on this ranking with the highest ranked project receiving the most points.

To meet the minimum standards for this project, your team will develop both the Azure functions and the front end to support the following 5 experiences:

- Ability to Create a New Membership
- Ability to Authenticate a Member into the Application
- Display of a QR Code that contains the users' MembershipID
- Update User Information
- Review Purchase History

As this is a ranked grading project, additional experiences and a well designed application will provide a competitive advantage over your classmates. Among the additional experiences that may be used are:

- Ability to authenticate as a different user type (Member vs Administrator)
- Ability to setup loyalty campaigns as an Administrator
- Ability to review sales in a DataTable and charts (Chart.js) as an Administrator
- Ability for a Member user to track loyalty campaign progress

Some words of caution. This project has an accelerated timeline of approximately **22** days. Although the five minimum experiences can easily be developed in this timespan, failure to manage your time properly will result in failing to meet these standards and being awarded an undesirable grade. To assist you with your planning, I have developed a guide below on the efforts you should be producing along the way to meet your deadline. You should strive to work ahead when possible to compensate for any unforeseen issues. During our scheduled class times we will discuss your efforts and the instructor will provide guidance to the class on how to meet the assigned objectives. These classroom exercises will help focus your efforts and provide additional instruction on developing the project.

April 7 - 10: You will be introduced to the project and have an opportunity to discuss it with the instructor of the course. You and your partner should begin outlining responsibilities and discuss each member's expectations. Use the next few days to establish a shared Github repository, discuss your development approach (SPA vs multi-page), scaffold your application,

discuss the process flow, and begin wireframing your experiences. Time spent here will go a long way in making the rest of the project successful.

April 11 -16: This time should be dedicated to developing your front end. Build a new member and authentication experience with the first couple days by referencing the HippoTasks project. For the last part of this period you should develop the main content of the application that a user will navigate to. This section will be responsible for allowing the user to display the QR Code of their membership, update their information, and review their purchase history. The QR Code operation will be the biggest challenge of this period. **HINT**: Use a Javascript library to create the QR Code object on your page.

April 17 - 21: Develop the RESTful web services as an Azure Function project to support your database interactions. Remember a RESTful web service determines what function to perform on the database based upon the HTTP method being called - GET vs POST vs PUT. You will need a minimum of these web services to meet the minimum standard experiences:

/membership

- Create a new member, return a success or error message (POST)
- Update a member, return a success or error message (PUT)
- Return a member's information when provided a SessionID (GET)

/location

- Return an array of locations and their associated information (GET)

/session

- Create a session when passed a correct username and password, return the SessionID or error message (POST)
- Update a session with a new Last Used Date and Time based on a SessionID, return a success or error message (PUT)
- Delete a session with a provided SessionID, return a success or error message (DELETE)
- Return session details when provided a SessionID (GET)

/purchases

- Create a purchase, return a PurchaseID or error message (POST)
- Return an array of purchases when provided a SessionID (GET)

HINT: You can deploy all of these functions out as part of a single project in Visual Studio 2019.

April 22 -27: Integrate the web services with the front end and test. When you have tested and verified functionality, **test more**. If you find you have additional time in your timeline, you can start integrating the optional experiences. The biggest challenge of this section will be the methods related to DELETE and PUT.

April 28: Present your project to the class and panel of judges. This is the time to be proud of the work you have done and the new skills you have gained this term. You will be provided a link at the end of class to provide honest feedback on both your and your partner's efforts.