**User guidance:**

**Steps to take to open the project:**

1. Open “e-shop-Qingyuan-zeng-s316740.sln” with Visual Studio 2019

2. Press Ctrl + F5 or click  to run the project (Note: The project takes a long loading time, it could be up to 5 minutes before you can use it. Also, sometimes it returns “timeout” error after loading, the way I cope with it is by running it again.)

3. Once the project is up and running, you will need to register a new account before you can add items for sale or buy items that other users are selling.

4. Once you complete the registration and login, you will be navigated to the home page.

**Product introduction:**

The ‘Home’ page is where all your selling items are listed. As a new user, you will need to click the “Create Item” node at the navigation bar to create some new items for sale. Once the items are created, they will be listed at ‘Home’ page for you to view and manage, and they will also be listed at ‘Buy’ page for other people to buy.

The “Buy” page lists all the items for sale from all users of this site, but only the user who created the item has the authority to edit it.

The “Create Item” node is where users can create items for sale. The “Item Image” field is not functioning yet, so it would be fine if you left it empty.

Once you logged in, the ‘Register’ will vanish, instead, a dropdown menu with your username showing will appear. That is where you can view and edit your user profile. However, at this stage, you will need to logout and login again to see the changes to the profile.

**Steps I took:**

1. Building environment for an Angular project (installing Node.js, VSCode, Bootstrap, Alertify, etc.).

2. Create “item-card” component to display each item.

3. Create JSON file as temporary data source.

3.Create ‘eshoppingService’ with ‘getAllItems’ function. The service can be imported into components through dependency injection.

4. Create “item-list” component to display a list of items. The component gets item data from the JSON file, and it uses property binding to pass the data to “item-card” component.

5. Define ‘Item’ interface, which can be used as a data type throughout the project.

6. Create ‘Item detail’ component. It shows the details of an item and provides the options to delete or update the item. It uses the Angular Router service to identify which card is clicked, and get the id of the item from the URL.

7. Add ‘putItem’ function in ‘eshoppingService’ file for updating item.

8. Add ‘deleteItem’ function in ‘eshoppingService’ file for deleting item.

9. Add ‘postItem’ function in ‘eshoppingService’ file for add new items.

10. Create “add-item” components with an item form using Angular reactive form approach.

11. Add validators and error reminders to the item form, and import ‘postItem’ method for submitting.

12. Move the data source from a Json file to Entityframework.

13. Create ‘item’ and ‘shoppingCart’ modules and register them in ‘eShopDBContext’ file.

14. Register the context in ‘Startup.cs’, and declare local database server and database name in “appsettings.json” file.

14. Scaffold the modules with API controllers, and make the migration and update.

15. Change all the functions in ‘eshoppingService’ from working with the JSON file to making post, put, delete requests to the API controllers of Entityframework.

16. Create “userService” file, which works with local storage, and it includes CRUD functions for user accounts.

17. Create “shoppingCart” interface, and create “addShoppingCart” function in ‘eshoppingService’

18. Create “identity-register” component with a registration form, and add the validators.

19. Import “addAccount” and “addShoppingCart” functions to “identity-register” component, and define that when a user account is created, a shopping cart that belongs to the account is generated.

20. Create “identity-login” component with a login form, the information that the user enters into the form will be compared with the information of the registered account in the local storage. If it matches, then the login process will be successful.

21. Create a ‘token’ when a login process completes. This token will be used to track current user that is logged in and store the user’s information.

22. Create “identity-detail” component where the current user’s profile can be viewed and information can be updated. This component gets the user info from the ‘token’.

23. The ‘token’ is also used in “item-card” component to ensure that only the creator of the item has the permission to edit it.