

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
|  | Milestone 5: Development Phase (Coding and Testing) | | |  |
|  |  | | |  |
|  | |  |  | |
|  | | John Keen johnkeenishere@gmail.com |  | |
|  | | May 24, 2024—Project Name:Laptop Catalog—CST-452: Senior Project 2—Amr Elchouemi—Document Revision Number: 2 |  | |

# Abstract

This document contains the documentation required for coding and testing code. This document contains the implementation plan which provides a detailed outline for the activities required while implementing the code for this project. An Agile Scrum framework is being used for the development of this project. The document includes Test Cases for that have been performed on the web application. This document also includes a mapping of the functional requirements (i.e. User Stories, Page and Code Module) to the Test Case used to test the web application. It also includes a source code listing where each class is listed with a brief description of all the classes. After the source code listing is a list of unit tests. It outlines whether or not the unit test is still a stub and gives a basic description of what the unit test does.

|  |  |  |
| --- | --- | --- |
|  |  |  |

# Table Of Contents

Cover Page [i](#_Toc167463831)

[Abstract ii](#_Toc167463832)

[Table Of Contents iii](#_Toc167463833)

[Implementation Plan 1](#_Toc167463834)

[Mapping of Functional Requirements 8](#_Toc167463835)

[Test Plan and Test Cases 12](#_Toc167463836)

[Source Code Listing 33](#_Toc167463837)

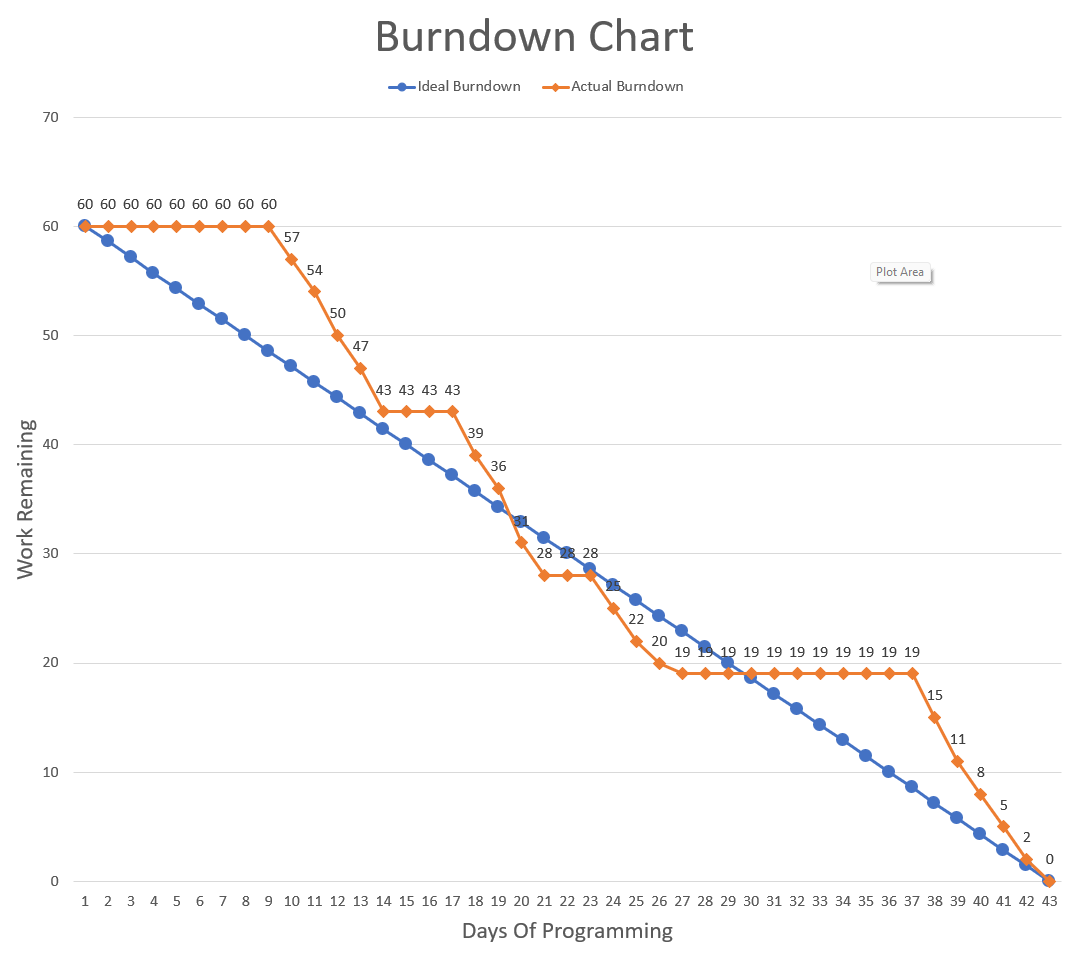
[Unit Testing Documentation 38](#_Toc167463838)

# Implementation Plan

The first week of senior project one was for the project proposal. Coding for this project began over Christmas break. I then used agile scrum practices to develop the concepts and features used for this web application. Coding for the first week of Christmas break consisted of setting up the project and designing the UI. The second week of Christmas break was spent beginning the engineering for the backend and incrementally working on features of the web application. The second week of senior project one for this project was for engineering the requirements for the project. This was a good time to ensure that all of the features that I expected to see in the web application was turned into user stories. The wireframes were not too difficult to create because I knew what I wanted on each page. Designing the use cases, top-down approach and system logical model fit with the work I had previously done. Week three of senior project one was spent developing the final architecture plan. I finished engineering the database as I anticipated I would need it. I created UML diagrams for each class that I had and expected to need. I created collaboration diagrams to show how I imagined code would interact with other code. The last week of senior project one was spent putting all of the pieces together. I finished implementing the payment integration and completing the user stories. This is the week I developed my testing plan and began to test the application. The time that testing takes is not reflected in my implementation calculations. In the first week of Senior project two I tested my site against XSS attack and SQL injection. I found that the application is basically secure but should be penetration tested. I also spent the week looking for bugs. During the second week of senior project two, I created unit tests for each of the controllers that are in addition to the manual tests outlined in the test matrix. I hosted the site on Azure and setup the Stripe payment webhook required for the application to work in the cloud. I had to create a database server and created an identical database to the one on my local machine. I created a user with the Manager role and further setup the application for demonstration purposes. My pursuits were not without errors.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| User Stories | | | | | | |
| ID | Feature | As a(n) <actor> | I would like to <description> | So that I can | Estimate (hours) | Sprint # |
| 1 | Registration | As a user | I would like to be able to register a new account | so that I can login. | 4 | 1 |
| 2 | Login | As a user | I would to be able to login | so that I can access the features of the website. | 0 | 1 |
| 3 | Logout | As a user | I would like to be able to logout | so that I can protect data that may be vital to me or my company. | 0 | 1 |
| 4 | Toggle Feature | As a Laptop Manager | I would like to be able to switch between a table view and a card view | so that I can easily find what I am looking for and ensure that the information and image for the laptop is correct. | 4 | 1 |
| 5 | Laptop Management Search | As a Laptop Manager | I would like to be able to search for laptops | so that I can quickly find a laptop that needs to be edited or deleted. | 1 | 1 |
| 6 | Modal Create Form | As a Laptop Manager | I would like to be able to click a button that loads a create form | so that I can create a laptop entry in the catalog. | 8 | 2 |
| 7 | Modal Edit Form | As a Laptop Manager | I would like to be able to click a button that loads an edit form | so that I can quickly edit a laptop entry. | 8 | 2 |
| 8 | Laptop Record Delete | As a Laptop Manager | I would like to be able to click a delete button | so that I can delete a laptop entry. | 4 | 2 |
| 9 | Management Laptop Details Page | As a Laptop Manager | I would like to be able to click a details button | so that I can navigate to a page with the full details and full description for a laptop entry. | 2 | 3 |
| 10 | Search Results Persistence | As a Laptop Manager | I would like to be able to see the same search results while switching between table and card view | that I can readily find the information I am looking for. | 1 | 3 |
| 11 | Manager Registration | As a Manager | I would like to be able to register a new user | so that I can get an employee started. | 1 | 4 |
| 12 | Role Management | As a Manager | I would like to be able to create or delete user roles | so that I can add roles for new pages that require authorization for a particular role or remove roles that were only needed for pages that have been removed from the web application. | 4 | 5 |
| 13 | Users Role Management | As a Manager | I would like to be able to assign or unassign roles for a user | so that I can manage who has access to particular features of the site. | 4 | 5 |
| 14 | User Role Search | As a Manager | I would like to be able to search for users | so that I can quickly find a particular user. | 1 | 5 |
| 15 | Retailer REST API | As a retailer | I would like to be able to consume a REST API | so that I can display the manufacturers data on my website. | 2 | 6 |
| 16 | Shopping Search | As a Customer | I would like to be able search for a particular laptop | so that I can find a laptop of interest. | 1 | 7 |
| 17 | Shopping Details Page | As a Customer | I would like to be able to click a button that navigates to a full details page of a laptop entry | so that I can learn more about a particular laptop. | 1 | 7 |
| 18 | Products Page Add Cart | As a Customer | I would like to be able to add an item to the cart from the products page | so that I can add more than one item to the cart. | 1 | 7 |
| 19 | Cart Page | As a Customer | I would like to see the items in my cart | so that I can know what I am purchasing. | 3 | 8 |
| 20 | Cart Page Item Management | As a Customer | I would like to be able to add or remove items in the cart | so that I can manage what I am purchasing without having to navigate to a previous page. | 1 | 8 |
| 21 | Checkout | As a Customer | I would like to be able to checkout | so that I can complete my order | 8 | 9 |
| 22 | Greeting | As a Customer | I would like to be greeted when the page loads | so that I know I am on the right page. | 0 | 9 |
| 23 | Confirmation | As a Customer | I would like to see a confirmation of my payment | so that I can know that the payment form was filled out correctly. | 1 | 9 |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  | Total Estimated Hours: | 60 |  |
|  |  |  |  | Actual Hours: | 60 |  |
|  |  |  |  | Percent Complete for Entire Project | 100 |  |
|  |  |  |  | Percent Complete for  This Iteration | 100 |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| User Story Id | User Story | Task | Estimate (hours) | Actual (hours) | Percent Complete |
| 1 | As a user, I would like to be able to register a new account so that I can login. | Setup Identity Framework. Create class based on IdentityUser. Migrate and update database. Update registration form and code behind. Etc... | 4 | 4 | 100 |
| 2 | As a user, I would to be able to login so that I can access the features of the website. | Setup Identity Framework. Etc… | 0 | 0 | 100 |
| 3 | As a user, I would like to be able to logout so that I can protect data that may be vital to me or my company. | Setup Identity Framework. Etc… | 0 | 0 | 100 |
| 4 | As a Laptop Manager, I would like to be able to switch between a table view and a card view so that I can easily find what I am looking for and ensure that the information and image for the laptop is correct. | Create Toggle feature to Asynchronously load a table or grid partial view into CRUD page. | 4 | 4 | 100 |
| 5 | As a Laptop Manager, I would like to be able to search for laptops so that I can quickly find a laptop that needs to be edited or deleted. | Create DAO method for searching. Install form for performing search. | 1 | 1 | 100 |
| 6 | As a Laptop Manger, I would like to be able to click a button that loads a create form so that I can create a laptop entry in the catalog. | Create DAO and database for laptop record creation. Create partial view with modal containing form for submitting laptop data to DAO. Create model for submitting form to DAO. Write validation for form. Etc… | 8 | 8 | 100 |
| 7 | As a Laptop Manager, I would like to be able to click a button that loads an edit form so that I can quickly edit a laptop entry. | Create DAO for laptop record update. Create partial view with modal containing form for submitting laptop update data to DAO. Write JavaScript to load form with values of LaptopModel. Write validation for form. Etc… | 8 | 8 | 100 |
| 8 | As a Laptop Manager, I would like to be able to click a delete button so that I can delete a laptop entry. | Create DAO method and controller action for deleting. Create animation for removing card or row from grid or table. Write logic for determining which partial view is used. | 4 | 4 | 100 |
| 9 | As a Laptop Manager, I would like to be able to click a details button so that I can navigate to a page with the full details and full description for a laptop entry. | Create view for showing details of laptop model. Create action for view. Create details edit form. Write animation for updating details page with updated data. | 2 | 2 | 100 |
| 10 | As a Laptop Manager, I would like to be able to see the same search results while switching between table and card view so that I can readily find the information I am looking for. | Create a data service for holding results of search. Use search results as model for grid and table view. | 1 | 1 | 100 |
| 11 | As a Manager, I would like to be able to register a new user so that I can get an employee started. | Determine where profile menu options are. Write link to registration. Use conditional code to only display link if role is manager. | 1 | 1 | 100 |
| 12 | As a Manager, I would like to be able to create or delete user roles so that I can add roles for new pages that require authorization for a particular role or remove roles that were only needed for pages that have been removed from the web application. | Create UserRoles controller. Inject RoleManager into controller. Learn RoleManager methods. Create Index and Create view. Write action to process delete click for role. Write action to process form submission on Create view. | 4 | 4 | 100 |
| 13 | As a Manager, I would like to be able to assign or unassign roles for a user so that I can manage who has access to particular features of the site. | Create ShowUsers view to display users with link to ManageRole view. Create ManageRole view. Write actions for creating and deleting roles from user. | 4 | 4 | 100 |
| 14 | As a Manager, I would like to be able to search for users so that I can quickly find a particular user. | Create Search form. Write LINQ to render list of ApplicationUsers to ShowUser view. | 1 | 1 | 100 |
| 15 | As a Retailer, I would like to be able to consume a REST API so that I can display the manufacturers data on my website. | Create API Controller and endpoints for API | 2 | 2 | 100 |
| 16 | As a Customer, I would like to be able search for a particular laptop so that I can find a laptop of interest. | Create search form and action to submit form to. Utilize DAO for returning search results. | 1 | 1 | 100 |
| 17 | As a Customer, I would like to be able to click a button that navigates to a full details page of a laptop entry so that I can learn more about a particular laptop. | Create a details view for the ShopController. Add button to laptop card. Create corresponding action that passes model to view. | 1 | 1 | 100 |
| 18 | As a Customer, I would like to be able to add an item to the cart from the products page so that I can add more than one item to the cart. | Create an add to cart button with a corresponding action. | 1 | 1 | 100 |
| 19 | As a Customer, I would like to see the items in my cart so that I can know what I am purchasing. | Create shopping cart service and shopping cart. Create a Cart view and action that returns view. | 3 | 3 | 100 |
| 20 | As a Customer, I would like to be able to add or remove items in the cart so that I can manage what I am purchasing without having to navigate to a previous page. | Create buttons for adding or removing. Create action that will remove from quantity or remove item from cart if quantity is 1. Create action that adds 1 to quantity of item in cart. | 1 | 1 | 100 |
| 21 | As a Customer, I would like to be able to checkout so that I can complete my order | Create purchase and orders relations. Create CheckoutDAO. Create form to accept shipping information. Integrate Stripe. Create actions for handling checkout and payment success. Create webhook for handling PaymentIntent succeeded event. Write checkout fulfillment method in shopping cart service to persist purchase and order data to database once payment intent success. | 8 | 8 | 100 |
| 22 | As a Customer, I would like to see a greetings page so that I know I am on the right page. | Create a simple greeting page. | 0 | 0 | 100 |
| 23 | As a Customer, I would like to see a confirmation of my payment so that I can know that the payment form was filled out correctly. | Create thank you page. | 1 | 1 | 100 |



# Mapping of Functional Requirements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***ID*** | ***Assoc ID*** | ***Functional Requirement / Use Case*** | ***Architectural/Design Document*** | ***Software Module(s)*** | **Test Case Number** |
| 001 | 1.1 | As a user, I would like to be able to register a new account so that I can login. | UI2   ~\Areas\Identity\Pages  \Account\Register.cshtml  ~\Areas\Identity\Pages  \Account\Register.cshtml.cs | InputModel  OnPostAsync(string) | 1 |
| 002 | 2.2 | As a user, I would to be able to login so that I can access the features of the website. | UI3    ~\Areas\Identity\Pages  \Account\Login.cshtml  ~\Areas\Identity\Pages  \Account\Login.cshtml.cs | InputModel  OnPostAsync(string) | 2 |
| 003 | 3.3 | As a user, I would like to be able to logout so that I can protect data that may be vital to me or my company. | UI13    ~\Areas\Identity\Pages  \Account\Logout.cshtml  ~\Areas\Identity\Pages  \Account\Logout.cshtml.cs | OnPost() | 3 |
| 004 | 4.4 | As a Laptop Manager, I would like to be able to switch between a table view and a card view so that I can easily find what I am looking for and ensure that the information and image for the laptop is correct. | UI4  ~\Views\Laptop\Index.cshtml  LaptopController  site.js | /Laptop/Toggle  getResults(string) | 4 |
| 005 | 5.5 | As a Laptop Manager, I would like to be able to search for laptops so that I can quickly find a laptop that needs to be edited or deleted. | UI4  ~\Views\Laptop\Index.cshtml  LaptopController | SearchResults  LaptopDAO All() | 5 |
| 006 | 6.6 | As a Laptop Manger, I would like to be able to click a button that loads a create form so that I can create a laptop entry in the catalog. | UI6   ~\Views\Laptop\Index.cshtml  LaptopController  site.js | CreateLaptop  LaptopDAO Insert() | 6 |
| 007 | 7.7 | As a Laptop Manager, I would like to be able to click a button that loads an edit form so that I can quickly edit a laptop entry. | UI7   ~\Views\Laptop\Index.cshtml  LaptopController  site.js | SaveEditReturnNewCard()  SaveEditReturnNewRow()  SaveEditReturnDetails() | 7 |
| 008 | 8.8 | As a Laptop Manager, I would like to be able to click a delete button so that I can delete a laptop entry. | \_laptopCard.cshtml  \_laptopRow.cshtml  LaptopController | Delete(LaptopModel) | 8 |
| 009 | 9.9 | As a Laptop Manager, I would like to be able to click a details button so that I can navigate to a page with the full details and full description for a laptop entry. | UI5  ~\Views\Laptop\  ShowOneLaptop.cshtml  ~\Views\Shared\  **\_**details.cshtml | ShowOneLaptop(int) | 9 |
| 010 | 10.10 | As a Laptop Manager, I would like to be able to see the same search results while switching between table and card view so that I can readily find the information I am looking for. | LaptopController  DataService | Index()  SearchResults(string)  Toggle()  getResults()  Delete(LaptopModel)  CreateLaptop(LaptopModel) | 10 |
| 011 | 11.11 | As a Manager, I would like to be able to register a new user so that I can get an employee started. | UI8  ~\Areas\Identity\Pages  \Account\Manage\  \_ManageNav.cshtml | /Account/Register | 11 |
| 012 | 12.12 | As a Manager, I would like to be able to create or delete user roles so that I can add roles for new pages that require authorization for a particular role or remove roles that were only needed for pages that have been removed from the web application. | UI9, UI10  ~\UserRoles\Index()  ~\UserRoles\Create()  UserRolesController | SubmitRole(IdentityRole)  Delete(string) | 12 |
| 013 | 13.13 | As a Manager, I would like to be able to assign or unassign roles for a user so that I can manage who has access to particular features of the site. | UI12  ~\Views\UserRoles\  ManageRoles.cshtml  UserRolesController | AddRoleToUser(string,string)  DeleteRoleFromUser(string,string) | 13 |
| 014 | 14.14 | As a Manager, I would like to be able to search for users so that I can quickly find a particular user. | UI11  ~\Views\UserRoles\  ShowUsers.cshtml  ~\Views\UserRoles\  \_searchForm.cshtml  UserRolesController | Search(string) | 14 |
| 015 | 15.15 | As a Retailer, I would like to be able to consume a REST API so that I can display the manufacturers data on my website. | LaptopAPIController | <https://localhost:7085>  /api/  /api/searchresults/13  /api/showonelaptop/1 | 15 |
| 016 | 16.16 | As a Customer, I would like to be able search for a particular laptop so that I can find a laptop of interest. | UI13  ~\Views\Shop\Index()  ~\Views\Shop\  \_searchForm.cshtml  ShopController | SearchResults(string) | 16 |
| 017 | 17.17 | As a Customer, I would like to be able to click a button that navigates to a full details page of a laptop entry so that I can learn more about a particular laptop. | UI14  ~\Views\Shop\  ShowOneLaptop.cshtml  ShopController | ShowOneLaptop(int) | 17 |
| 018 | 18.18 | As a Customer, I would like to be able to add an item to the cart from the products page so that I can add more than one item to the cart. | UI13  ~\Views\Shop\  Cart.cshtml  ShopController  site.js | AddToCart(int) | 18 |
| 019 | 19.19 | As a Customer, I would like to see the items in my cart so that I can know what I am purchasing. | UI15  ~\Views\Shop\  Cart.cshtml  ShopController | Cart() | 19 |
| 020 | 20.20 | As a Customer, I would like to be able to add or remove items in the cart so that I can manage what I am purchasing without having to navigate to a previous page. | UI15  ~\Views\Shop\  Cart.cshtml  ShopController  site.js | AddOneToCart(int)  RemoveFromCart(int) | 20 |
| 021 | 21.21 | As a Customer, I would like to be able to checkout so that I can complete my order | UI16  ~\Views\Shop\  Cart.cshtml  ShopController  ShoppingCartService | /Shop/ :  Checkout(CheckoutModel)  ThankYou()  /Shop/ :  WebhookHandler()  ShoppingCartService:  checkout(string) | 21 |
| 022 | 22.22 | As a Customer, I would like to see a greetings page so that I know I am on the right page. | UI1  ~\Views\Home\  Index.cshtml | Index() | 22 |
| 023 | 23.23 | As a Customer, I would like to see a confirmation of my payment so that I can know that the payment form was filled out correctly. | UI17  ~\Views\Shop\  ThankYou.cshtml  ShopController | ThankYou() | 23 |

# Test Plan and Test Cases

|  |  |  |
| --- | --- | --- |
| **Test Case#** | **Test Title** | **Pass/Fail Status** |
| 1 | Account Registration | Pass |
| 2 | Login | Pass |
| 3 | Logout | Pass |
| 4 | Toggle List and Display | Pass |
| 5 | CRUD Laptop Search | Pass |
| 6 | Create Form Submission | Pass |
| 7 | Edit Form Population and Submission | Pass |
| 8 | Delete Button | Pass |
| 9 | CRUD Laptop Details | Pass |
| 10 | Search Result Persistence | Pass |
| 11 | Manager Registration Access | Pass |
| 12 | Role Management | Pass |
| 13 | Users Role Management | Pass |
| 14 | User Search | Pass |
| 15 | Retailer API | Pass |
| 16 | Shopping Search | Pass |
| 17 | Shopping Laptop Details | Pass |
| 18 | Products Page Add to Cart | Pass |
| 19 | Cart Item Load | Pass |
| 20 | Cart Item Management | Pass |
| 21 | Checkout | Pass |
| 22 | Greeting | Pass |
| 23 | Payment Confirmation | Pass |

##### Test Case #1: Account Registration

Steps:

1. Click Register
2. Fill in form
3. Click Register below form
4. Go to Visual Studio, right click database, choose new query from menu.
5. Put following code in query window:

select \* from dbo.AspNetUsers;

1. Click play in query window
2. Confirm results

Test Data:

test@testing

#Testing41

Test

Testing

(000)000-0000

Testing Town

Tested

00000

111 Maple Street

USA

Expected Result:

[Some Id Hash] Test Testington Testing Town Tested 00000 111 Maple Street USA test@testing.com TEST@TESTING.COM test@testing.com TEST@TESTING.COM 0 [Some Password Hash] [Some Security Stamp] [Some Concurrency Stamp] (000)000-0000 0 0 NULL 1 0

Acutual Result:

02a39de5-67bd-465d-8e61-62db7b3d9ff5 Test Testington Testing Town Tested 00000 111 Maple Street USA test@testing.com TEST@TESTING.COM test@testing.com TEST@TESTING.COM 0 AQAAAAEAACcQAAAAEDaEMEJBpGGHHqf2P2DdKqK7zPAr1QO8PWRfKgZNMSJ6EXPfJPvM76RPpz5jYcZknw== DUFRMWIEP2IZLRS5DH53I7KYRXNBXNKE 7be06e07-60ec-4743-97ae-5c40e1229d7d (000)000-0000 0 0 NULL 1 0

##### Test Case #2: Login

Steps:

1. Click Login
2. Fill in form
3. Click “Log In” button below form
4. Confirm profile is accessible

Test Data:

test@testing.com

#Testing41

Expected Result:

User profile accessible by means of a link provide in the menu as follows:

Hello test@testing.com!

Acutual Result:

User profile accessible by means of a link provide in the menu as follows:

Hello test@testing.com!

##### Test Case #3: Logout

Steps:

1. Click Login, Fill in form, Click “Log In” button below form
2. Ensure that profile is accessible by means of the menu
3. Click Logout
4. Ensure that the profile link is no longer available in the menu
5. Insert the following into the web browser URL:

<https://localhost:7085/Identity/Account/Manage>

1. Ensure that the profile area is not accessible without a password using the previous URL.

Test Data:

test@testing.com

#Testing41

Expected Result:

Profile link is no longer accessible by means of the menu.

Profile area is not accessible by means of a URL.

Acutual Result:

Profile link is no longer accessible by means of the menu.

Profile area is not accessible by means of a URL.

##### Test Case #4: Toggle List and Display

Steps:

1. Login to account that possess the Role of “LaptopManager”
2. Click “Manage Laptops” menu link
3. Click Toggle List/Display
4. Verify that the product viewing area toggles between a table of products and a grid of product cards

Test Data:

test@testing.com

#Testing41

Expected Result:

Toggling between table of products and grid of cards containing laptop information.

Acutual Result:

Toggling between table of products and grid of cards containing laptop information.

##### Test Case #5: CRUD Laptop Search

Steps:

Click Login, Fill in form and login with user that has a Role of “LaptopManager”

Click “Log In” button below form

Click “Manage Laptops”

Click input box below “Search for a laptop” and insert the word “Book” and click search

Ensure that two products are returned

Test Data:

Book

Expected Result:

Two laptops: 13 Inch Mc School Book, 15 Inch Mc Student Book

Acutual Result:

Two laptops: 13 Inch Mc School Book, 15 Inch Mc Student Book

##### Test Case #6: Create Form Submission

Steps:

1. Click Login, Fill in form and login with user that has a Role of “LaptopManager”
2. Click “Log In” button below form
3. Click “Manage Laptops”
4. Click “Create”, Fill in Form, Click “Create” at bottom of modal
5. Ensure that a product appears somewhere on the page with containing the test data

Test Data:

../img/McPro.jpg

Mc Pro Astronaut Edition

Good Laptop

1234.89

Mc Test

90 GB Unified

2 TB

RX 7800 XT

3.4 lbs

Mc OS

Expected Result:

Laptop Record with following fields:

[Id] ../img/McPro.jpg Mc Pro Astronaut Edition Good Laptop 1234.89 Mc Test 90 GB Unified 2 TB Radeon 7800 XT 3.4 lbs Mc OS

Acutual Result:

11002 ../img/McPro.jpg Mc Pro Astronaut Edition Good Laptop 1234.89 Mc Test 90 GB Unified 2 TB Radeon 7800 XT 3.4 lbs Mc OS

##### Test Case #7: Edit Form Population and Submission

Steps:

1. Click Login, Fill in form and login with user that has a Role of “LaptopManager”
2. Click “Log In” button below form
3. Click “Manage Laptops”
4. Click an edit button for one of the displayed laptops
5. Make a note of the Laptop Id and replace the content of any field that is not the Laptop Id or price with the test data
6. Ensure that the changed field contains test data and note edited field
7. Click Save Changes
8. Find product by id and ensure that field was updated
9. Click edit for the same product
10. Change price to 898989.99 and click save
11. Ensure that the price was updated

Test Data:

### THIS IS MODIFIED USING EDIT ###

Expected Result:

Price is: 898989.99

Chosen field is:

### THIS IS MODIFIED USING EDIT ###

Acutual Result:

Price: 898989.99

Weight: ### THIS IS MODIFIED USING EDIT ###

##### Test Case #8: Delete Button

Steps:

1. Click Login, Fill in form and login with user that has a Role of “LaptopManager”
2. Click “Log In” button below form
3. Click “Manage Laptops”
4. Go to Visual Studio, right click database, choose new query from menu.
5. Put following code in the query window:

select \* from dbo.laptop;

1. Choose a laptop record and make note of the id
2. Replace the code in query window with the following code and put the chosen id in place of [id]:

select \* from dbo.laptop where id= [id];

1. Ensure that the record exists
2. Go back to the browser, find the product by id and click the delete button
3. Check in the browser and ensure that the laptop as removed from the page
4. Go to back to the query window in Visual Studio
5. Ensure the following code is in the query window with the chosen id in place of [id]:

select \* from dbo.laptop where id= [id];

1. Execute SQL code and ensure that the record has been deleted

Test Data:

Click delete button with id of 6004

Expected Result:

Laptop entry removed from page in browser.

Laptop relation record removed from database.

Acutual Result:

Laptop entry removed from page in browser.

Laptop relation record removed from database.

##### Test Case #9: CRUD Laptop Details

Steps:

1. Right click database name and choose new query
2. Type the following into the query window and run the SQL code using the green play button

in top left of the query window:

select \* from dbo.laptop;

1. Choose a laptop record that will be used to ensure that the details page is correct
2. Click Login, Fill in form and login with user that has a Role of “LaptopManager”
3. Click “Log In” button below form
4. Click “Manage Laptops”
5. Click “Show Details” button on laptop card or row
6. Ensure that every field of the laptop record is displayed correctly

Test Data:

Click “Show Details” for laptop with id 6003

Expected Result:

Laptop details for the following database laptop record:

6003, ../img/McPro.jpg, 15 Inch Mc Pro, This is one of our high-end laptops. It is lightweight and features a 15 inch screen that makes this laptop lap friendly. This fast and light weight computer will do just about everything you could ever want. A great buy for a work or play. It has a touch screen and comes with a stylus., 1234.12, Mc 4, 90 GB Unified Memory, 2 TB, NVIDIA 3050, 3.2 lbs., Mc OS

Acutual Result:

Laptop Id**:** 6003

Photo**:** ../img/McPro.jpg

Name**:** 15 Inch Mc Pro

Description**:** This is one of our high-end laptops. It is lightweight and features a 15 inch screen that makes this laptop lap friendly. This fast and light weight computer will do just about everything you could ever want. A great buy for a work or play. It has a touch screen and comes with a stylus.

Price**:** $1,234.12

Processor**:** Mc 4

Ram**:** 90 GB Unified Memory

Drive Size**:**2 TB

Graphics Card**:** NVIDIA 3050

Weight**:** 3.2 lbs.

Operating System**:** Mc OS

##### Test Case #10: Search Result Persistence

Steps:

1. Click Login, Fill in form and login with user that has a Role of “LaptopManager”
2. Click “Log In” button below form
3. Click “Manage Laptops”
4. Type “Book” in search results and verify the number of products displayed
5. Toggle List/Display and ensure that the same results are showing

Test Data:

Book

Expected Result:

Two records with book in title: 13 Inch Mc School Book, 15 inch Mc Student Book

Acutual Result:

Two records with book in title: 13 Inch Mc School Book, 15 inch Mc Student Book

##### Test Case #11: Manager Registration Access

Steps:

1. Click Login, Fill in form and login with user that has a Role of “Manager”
2. Click “Log In” button below form
3. Click the profile link in the menu
4. Click “Register New User”
5. Ensure that the page is the Register page

Test Data:

[johnkeenishere@gmail.com](mailto:johnkeenishere@gmail.com)

#GoodTimes2

Expected Result:

Redirect to registration page

Acutual Result:

Redirect to registration page

##### Test Case #12: Role Management

Steps:

1. Click Login, Fill in form and login with user that has a Role of “Manager”
2. Click “Log In” button below form
3. Click the profile link in the menu
4. Click “Manage Roles”
5. Click “Create New Role” link
6. Type test data into form and click “Create”
7. Verify that the role has been added to the list
8. Go to Visual Studio, right click the database and choose New Query from list
9. In the query window type the following SQL code

select \* from dbo.AspNetRoles;

1. Ensure that test data is in records for AspNetRoles relation of database
2. Go to browser and click “Delete” link for role created with test data
3. Ensure that Role is removed from roles on page
4. Go to the query window and execute this code:

select \* from dbo.AspNetRoles;

1. Ensure that record for Role has been deleted

Test Data:

OrderManager

Expected Result:

Role is created and deleted from database successfully.

Acutual Result:

Role is created and deleted from database successfully.

##### Test Case #13: Users Role Management

Steps:

1. Click Login, Fill in form and login with user that has a Role of “Manager”
2. Click “Log In” button below form
3. Go to the profile page
4. Click on “Manage Users Roles”
5. Search for user
6. Click “Manage Roles” for the user
7. Click “Add” next to role of “Retailers
8. Ensure that the role shows up under “Roles the User Belongs To”
9. Go to Visual Studio, open a query window and execute the following code.

Put the email for the chosen in place of test@testing.com:

select AspNetUsers.Email,AspNetRoles.Name from AspNetUsers

join dbo.AspNetUserRoles

on AspNetUserRoles.UserId=AspNetUsers.Id

join AspNetRoles

on AspNetRoles.Id=AspNetUserRoles.RoleId

where AspNetUsers.Email='test@testing.com';

1. Ensure that the role is in the database for the specified user
2. Go to the browser and click “Delete” next to the role under “Roles the User Belongs To”
3. Ensure that the role is removed from the section “Roles the User Belongs To”
4. Go back to Visual Studio and run the code in step 9 in a query window again
5. Ensure that the role has been removed for the user

Test Data:

test@testing.com

Expected Result:

Role is added then removed from database

Acutual Result:

Role is added then removed from database

##### Test Case #14: User Search

Steps:

1. Click Login, Fill in form and login with user that has a Role of “Manager”
2. Click “Log In” button below form
3. Go to the profile page
4. Click on “Manage Users Roles”
5. Search for user

Test Data:

test@testing

Expected Result:

[UserId] Test Testington test@testing.com (000)000-0000 Testing Town Tested 00000 111 Maple Street USA

Acutual Result:

02a39de5-67bd-465d-8e61-62db7b3d9ff5 Test Testington test@testing.com (000)000-0000 Testing Town Tested 00000 111 Maple Street USA

##### Test Case #15: Retailer API

Steps:

1. Open and sign into Postman
2. Create a new collection with the little plus sign
3. Click on the little ellipsis next to the collection and click “Add Request”
4. Set method to “GET” insert URL for endpoint to test
5. Select Body, JSON and Pretty in the response section
6. Click Send
7. Ensure that status is 200 OK and that the data aligns with expected response
8. Repeat steps for each of the endpoints

Test Data:

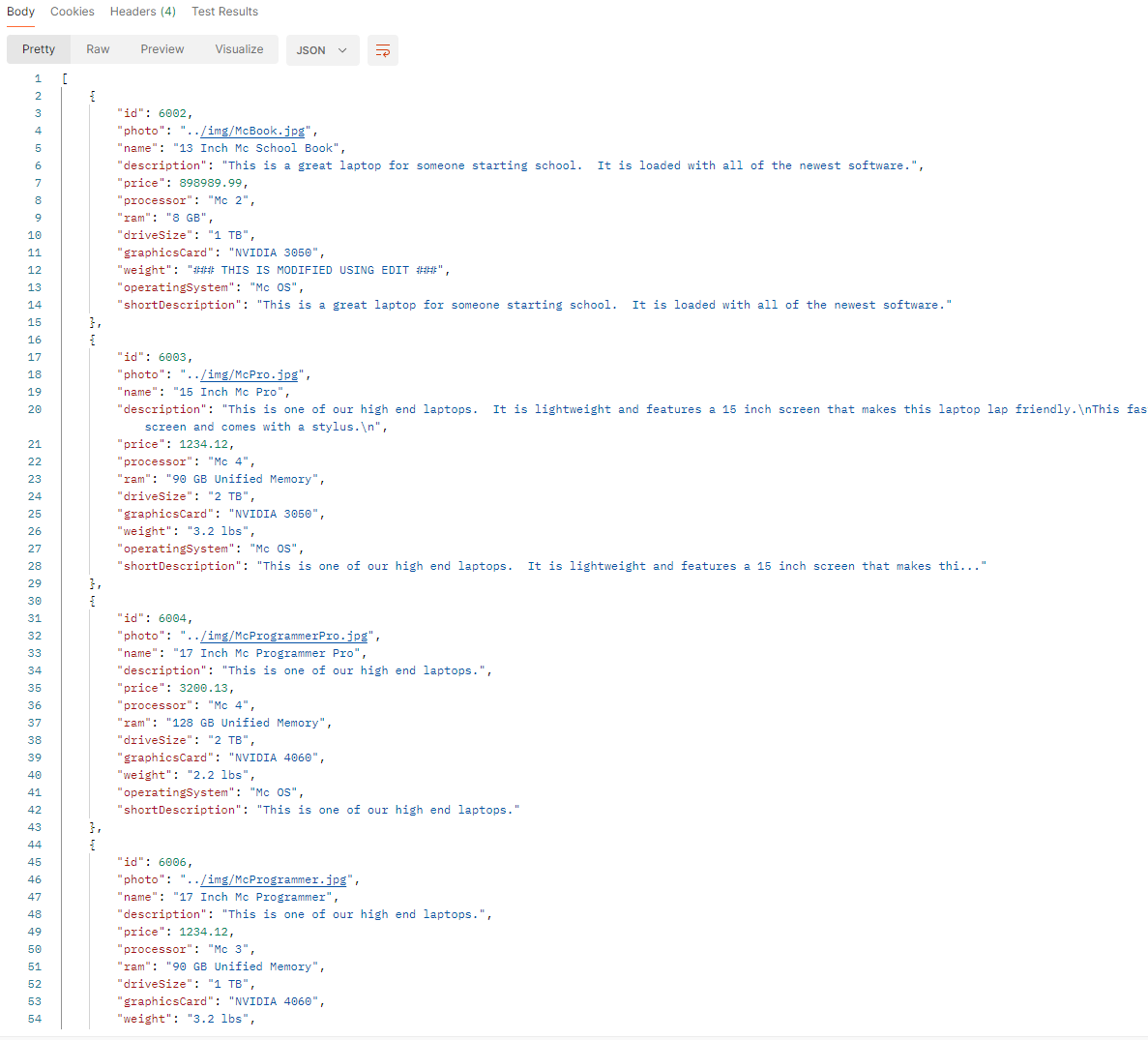
1. <https://localhost:7085/api/>
2. <https://localhost:7085/api/searchresults/13>
3. <https://localhost:7085/api/showonelaptop/6002>

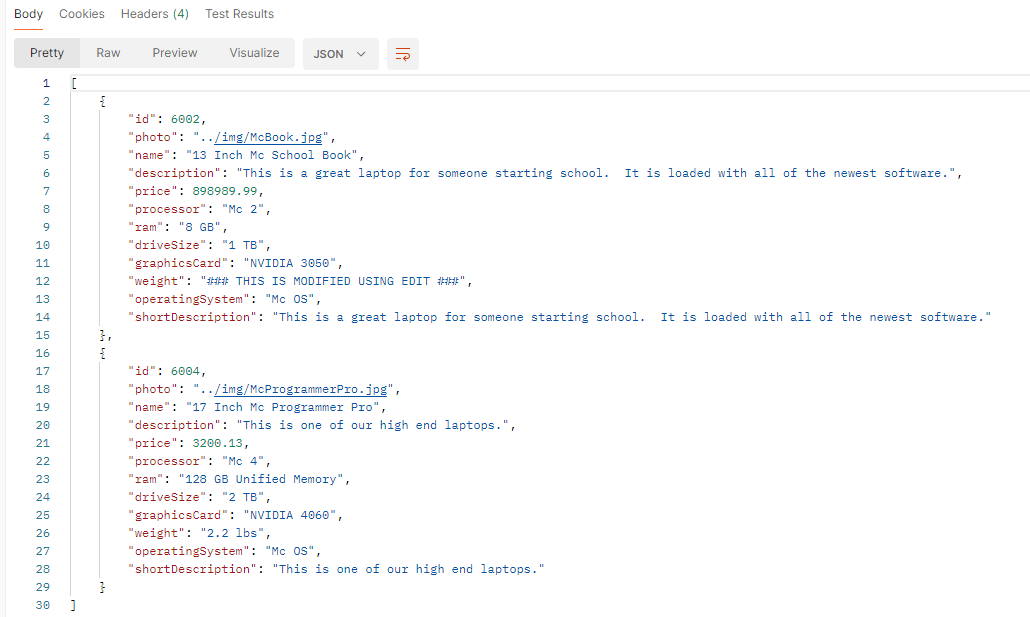
Expected Result:

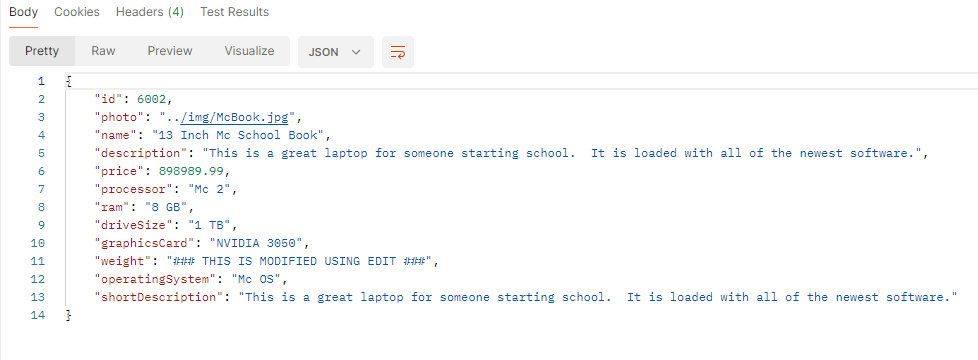
1. All laptop records returned.
2. Laptop records containing the number 13 is returned.
3. Laptop record with id of 6002 is returned.

Acutual Result:

1. All laptop records returned.
2. Laptop record containing the number 13 is returned.
3. Laptop record with id of 6002 is returned.

1. 

2.

3. 

##### Test Case #16: Shopping Search

Steps:

1. Login to any account
2. Click “Shop” link in menu
3. Type in test data
4. Click Search
5. Ensure that the results are as expected

Test Data:

Book

Expected Result:

Two laptop records with “Book” in their name.

Acutual Result:

Two laptop records with “Book” in their name.

##### Test Case #17: Shopping Laptop Details

Steps:

1. Right click database name and choose new query
2. Type the following into the query window and run the SQL code using the green play button

in top left of the query window:

select \* from dbo.laptop;

1. Choose a laptop record that will be used to ensure that the details page is correct
2. Log in to any account
3. Click “Shop” link in menu
4. Click “Show Details” button on laptop card or row
5. Ensure that every field of the laptop record is displayed correctly

Test Data:

Click “Show Details” for laptop with id 6003

Expected Result:

Laptop details for the following database laptop record:

6003, ../img/McPro.jpg, 15 Inch Mc Pro, This is one of our high-end laptops. It is lightweight and features a 15 inch screen that makes this laptop lap friendly. This fast and light weight computer will do just about everything you could ever want. A great buy for a work or play. It has a touch screen and comes with a stylus., 1234.12, Mc 4, 90 GB Unified Memory, 2 TB, NVIDIA 3050, 3.2 lbs., Mc OS

Acutual Result:

Laptop Id**:** 6003

Photo**:** ../img/McPro.jpg

Name**:** 15 Inch Mc Pro

Description**:** This is one of our high-end laptops. It is lightweight and features a 15 inch screen that makes this laptop lap friendly. This fast and light weight computer will do just about everything you could ever want. A great buy for a work or play. It has a touch screen and comes with a stylus.

Price**:** $1,234.12

Processor**:** Mc 4

Ram**:** 90 GB Unified Memory

Drive Size**:**2 TB

Graphics Card**:** NVIDIA 3050

Weight**:** 3.2 lbs.

Operating System**:** Mc OS

##### Test Case #18: Products Page Add to Cart And, Test Case #19: Cart Item Load

Steps:

1. Right click database name and choose new query
2. Type the following into the query window and run the SQL code using the green play button

in top left of the query window:

select \* from dbo.laptop;

1. Choose a laptop record that will be used to ensure that the details page is correct
2. Log in to an account and click “Shop” in the menu
3. Click “Add to Cart” for the chosen product
4. Click “Shopping Cart” button
5. Ensure that the product field in cart matches the laptop record fields

Test Data:

Click “Add to Cart” for laptop with id 6003

Expected Result:

The cart will use a short description instead of a full description.

An item in the cart with a quantity of 1 that matches the laptop fields for the following database laptop record:

6003, ../img/McPro.jpg, 15 Inch Mc Pro, This is one of our high-end laptops. It is lightweight and features a 15 inch screen that makes this laptop lap friendly. This fast and light weight computer will do just about everything you could ever want. A great buy for a work or play. It has a touch screen and comes with a stylus., 1234.12, Mc 4, 90 GB Unified Memory, 2 TB, NVIDIA 3050, 3.2 lbs., Mc OS

Acutual Result:

6003 ../img/McPro.jpg 15 Inch Mc Pro $1,234.12 Mc 4 90 GB Unified Memory 2 TB NVIDIA 3050 3.2 lbs Mc OS This is one of our high end laptops. It is lightweight and features a 15 inch screen that makes thi... 1

##### Test Case #20: Cart Item Management

Steps:

1. Login to Account
2. Click “Shop” in menu
3. Click “Add to Cart” for several items
4. Click “Shopping Cart” button
5. Choose an item and click the “Add” button
6. Ensure that the quantity goes up
7. Click the “Remove” button for the same item and ensure that the quantity decreases
8. Click the “Remove” button for the same item and ensure that the item is removed from the cart.

Test Data:

Laptops with ids of: 6002, 6003 and 6004

Expected Result:

Items increase and decrease quantity correctly.

Item is removed from cart when there is only 1 item in cart and “Remove” button is clicked.

Acutual Result:

Items increase and decrease quantity correctly.

Item is removed from cart when there is only 1 item in cart and “Remove” button is clicked.

##### Test Case #21: Checkout And, Test Case #23: Payment Confirmation

Steps:

1. Open a command prompt and execute the following:

stripe listen --forward-to https://localhost:7085/webhook

1. Login to Account
2. Click “Shop” in menu
3. Click “Show Details” for a product and ensure that it is the product you want to add to cart
4. Click “Add to Cart” button
5. Click “Shopping Cart” button
6. Click “Checkout”
7. Fill out the shipping address information
8. Click “Pay”
9. Insert test data credit card information
10. Click “Pay”
11. Ensure that the product is displayed on the Thank You page
12. Go to Visual Studio, open a query window for the database and insert the following SQL:

select \* from purchase

join orders

on purchase.PurchaseNumber=orders.PurchaseNumber

where Email='test@testing.com';

1. Ensure that the records in the database match the data that was provided

Test Data:

Product: Laptop with id 6003

Shipping Details:

Senior

Project

Testing Street

Testing City

Testing State

Testing Zip

Testing Country

Pseudo Card Information for Stripe Success:

4242 4242 4242 4242

12/34

567

Senior Project Test

97754

Expected Result:

[PurchaseNumber], [UserId], Senior, Project, test@testing.com, [PaymentIntent], Testing Street, Testing City, Testing State, Testing Zip, Testing Country, 1234.12, [PurchaseDate], [OrderNumber], [PurchaseNumber] 6003 1 unfilled

Acutual Result:

2, 02a39de5-67bd-465d-8e61-62db7b3d9ff5, Senior, Project, test@testing.com, pi\_3OcwNuLnxxUxdnER1qd7t6aK, Testing Street, Testing City, Testing State, Testing Zip, Testing Country, 1234.12, 2024-01-26, 12:52:23.570, 2, 2, 6003, 1, unfilled

##### Test Case #22: Greeting

Steps:

1. Run application
2. Ensure the application says “Welcome to McIntash Laptops”
3. Click “Login”, fill in form and click “Log in”
4. Click “Shop” in the menu
5. Click “Logout”
6. Ensure that the page says “Welcome to McIntash Laptops”

Test Data:

[test@testing.com](mailto:test@testing.com)

#Testing41

Expected Result:

Welcome page displayed upon arrival and logout.

Acutual Result:

Welcome page displayed upon arrival and logout.

# Source Code Listing

|  |  |
| --- | --- |
| Class or File | Description |
| ~\Areas\Identity\Data\ApplicationDbContext.cs | This is used by the entity framework and required by the Identity Framework. This is only used by the Identity Framework for Registration and Login. |
| ~\Areas\Identity\Data\ApplicationUser.cs | This is a custom class that is based on IdentityUser utilized by the Identity Framework. The registration page and ApplicationDbContext have been customized to handle ApplicationUser and its properties instead of IdentityUser. The database has been updated to accomodate the ApplicationUser for the Identity Framework. |
| HomeController.cs | This returns the welcome page upon arrival and logout. |
| LaptopAPIController.cs | This is the REST API and has the endpoints for returning all of the laptop records, laptop records for search results and one particular laptop record. |
| LaptopController.cs | This controller has all of the actions for CRUD operations. Only users with LaptopManager Role can access this controller. |
| ShopController | This controller is accessible to any user. It is used for shopping. It includes actions such as viewing one laptop's details, adding items to cart, removing item from cart, viewing cart and checking out. |
| UserRolesController.cs | This controller has actions for managing the creation and deletion of roles from the database. It also has actions for assigning or unassigning roles to users. |
| CheckoutModel | This is used for transferring address form data to a controller action at checkout. |
| ErrorViewModel | This is used in the event of an error on the HomeController. It comes default with Asp.NET Core MVC projects. |
| LaptopModel | This is used to transfer data regarding records stored in the laptop relation of the database. |
| LaptopModelDTO | This is like the LaptopModel but includes a quantity property. It is used for the shopping cart and keeping track of how many of a item will be purchased. It is used to insert laptop records into the orders relation of the database. |
| OrderModel | This is used to access or update records in the orders relation. |
| PurchaseModel | This is used to access or update records in the purchase relation. |
| CheckoutDAO | This is the Data Access Object Class for checking out. An instance of this is used for checking out and accessing purchases and orders. |
| DataService | This is used to store search results so that the results are the same while switching between a "list" view with a table and a "display" view with a grid of cards. |
| ICheckoutService | This is the interface for the CheckoutDAO. |
| ILaptopDataService | This is the interface for the LaptopDAO. |
| Isecurity | This is the interface for the SecurityService. |
| LaptopDAO | This is the Data Access Object Class for Creating, Reading, Updating or Deleting laptop records. An instance of this is used for checking out and accessing purchases and orders. |
| SecurityService | This is a service that can be used for encrypting vital information. |
| ShoppingCartService | This stores the shopping cart and contains methods for maintaining the cart or fulfilling a purchase once the checkout process is complete. |
| ~\Views\Home\Index.cshtml | This is the view for welcoming users. It is shown upon arrival and after logging out. |
| ~\Views\Home\Privacy.cshtml | This is the view for expressing security and privacy concerns regarding the web application. |
| ~\Views\Laptop\\_createModal.cshtml | This is a partial view with a modal containing a form required for creating a laptop record. |
| ~\Views\Laptop\\_Display.cshtml | This is a partial view that holds a grid containing cards created by iterating over an IEnumerable<LaptopModel>. |
| ~\Views\Laptop\\_editModal.cshtml | This is a partial view containing a form for editing a laptop record. |
| ~\Views\Laptop\\_laptopCard.cshtml | This is a partial view containing a card for a LaptopModel with buttons for viewing details, Updating or Deleting the laptop record from the database. |
| ~\Views\Laptop\\_laptopRow.cshtml | This is a partial view containing a table row for a LaptopModel with buttons for viewing details, Updating or Deleting the laptop record from the database. |
| ~\Views\Laptop\\_List.cshtml | This is a partial view containing a table. The table has rows created by iterating over an IEnumerable<LaptopModel>. |
| ~\Views\Laptop\\_searchForm | This Partial view is used to search laptops for the Manage Laptops section. The results are stored in the dataService instance and passed as a model to the \_List or \_Display partial views. |
| ~\Views\Laptop\Index.cshtml | This is the main view for CRUD operations. It is the view that is returned when the Manage Laptops menu button is clicked on. |
| ~\Views\Laptop\ShowOneLaptop.cshtml | This is a view for viewing the details of a particular laptop record. It uses a LaptopModel. |
| ~\Views\Shared\\_Layout.cshtml | This is the main view that is the structure for the entire web application. |
| ~\Views\Shared\\_LoginPartial.cshtml | This provides links for accessing the profile page, registering, logging in or logging out. |
| ~\Views\Shared\\_navbar.cshtml | This is the NavBar for the entire application. |
| ~\Views\Shared\\_ValidationScriptsPartial.cshtml | This contains client-side validation scripts for logging in and registration. |
| ~\Views\Shared\Error.cshtml | This default partial view is provided for use in the event of an error. |
| ~\Views\Shared\\_details.cshtml | This is a partial view for creating a Description List out of a LaptopModel. |
| ~\Views\Shop\\_checkout.cshtml | This is a partial view containing a modal with a form requesting shipping information pertinent to the purchase relation of the database. |
| ~\Views\Shop\\_Display.cshtml | A partial view used to display a grid of cards based on IEnumerable<LaptopModel> |
| ~\Views\Shop\\_laptopCard.cshtml | A partial view with bootstrap card used with properties of a LaptopModel. Used in \_Display for the shopping section. |
| ~\Views\Shop\\_laptopRowDTO.cshtml | A partial view with a table row filled with properties of a LaptopModelDTO object. This is used to update the cart with a new row when the quantity of a item in the cart is changed. |
| ~\Views\Shop\\_searchForm.cshtml | This is the partial view with a form for searching laptop records in the shopping section. It posts to the SearchResults action of the ShopController. |
| ~\Views\Shop\Cart.cshtml | This is the Cart view which contains a checkout button and a table of rows made with an IEnumerable<LaptopModel>. |
| ~\Views\Shop\Index.cshtml | This is the default view for the ShopController. It contains a grid of cards based on an IEnumerable<LaptopModel> or list returned from the laptopDAO.All() or laptopDAO.SearchLaptops(searchTerm). |
| ~\Views\Shop\ShowOneLaptop.cshtml | This view displays a Data List that is populated with information from a LaptopModel. It has an add to cart button, a back to shop button and a Cart button for navigating to the cart. |
| ~\Views\Shop\ThankYou.cshtml | This page is shown upon completion of the Stripe payment. |
| ~\Views\Shop\PaymentFail.cshtml | Inform user their payment doesn’t have paid status. |
| ~\Views\UserRoles\\_addRoles.cshtml | This partial view displays a table of existing roles in the database with each row for each role containing a hyperlink to add the role to a user. The action of the hyperlink returns the ManageUser view. |
| ~\Views\UserRoles\\_searchForm.cshtml | This partial view is used to search for users in the UserManager by the search criteria and display them on the ShowUsers view. |
| ~\Views\UserRoles\\_userDetails.cshtml | This partial view is used to show the user's details on the ManageRoles view. |
| ~\Views\UserRoles\\_userRoles.cshtml | This partial view shows the roles that exist in the database for the particular user that is being managed. |
| ~\Views\UserRoles\Create.cshtml | This view has a form to input a new role. The form submits to the SubmitRole action. The role is persisted to the database in the SubmitRole action and the Index view is returned. |
| ~\Views\UserRoles\Index.cshtml | This view displays all of the roles currently in the database. Each IdentityRole in the table has a "delete" hyperlink for removing the roll from the database. A create button at the top links to the Create view. |
| ~\Views\UserRoles\ManageRoles.cshtml | This view is where a Manager can assign or unassign roles to a particular user. |
| ~\Views\UserRoles\ShowUsers.cshtml | This is used to show users in the database. Users can be searched and each user is in a row of a table and each row has a link to the ManageRoles view where roles can be assigned or unassigned to a user. |
| appsettings.json | I use this to store the Stripe keys for the application. |
| Program.cs | This file is used to configure and run the application. |
| ~\Areas\Identity\Pages\Account\Register.cshtml | This page contains a form with code behind  that is used for registering a user. |
| ~\Areas\Identity\Pages\Account\Register.cshtml.cs | Code behind for Register |
| ~\Areas\Identity\Pages\Account\Login.cshtml | This page is used for logging in. |
| ~\Areas\Identity\Pages\Account\Login.cshtml.cs | Code behind for Login |
| ~\Areas\Identity\Pages\Account\Logout.cshtml | This page is used for logging out. |
| ~\Areas\Identity\Pages\Account\Logout.cshtml.cs | Code behind for Logout |
| ~\Areas\Identity\Pages\Account\Manage\\_ManageNav.cshtml | Profile navigation |

# Unit Testing Documentation

##### HomeControllerTests:

| **Name** | **Is Stub** | **Description** |
| --- | --- | --- |
| HomeController\_Index\_ReturnViewResult | No | Tests the Index method of the HomeController and checks to ensure that it correctly returns a ViewResult. |
| HomeController\_Privacy\_ReturnViewResult | No | Tests if the Privacy method returns a ViewResult. |
| HomeController\_Error\_ReturnSuccessViewResult | No | Tests the Error action and ensures that the a ViewResult is returned. |

##### LaptopAPIControllerTests:

| **Name** | **Is Stub** | **Description** |
| --- | --- | --- |
| LaptopAPIControllerTests\_Index\_ReturnActionResultOfIEnumerableLaptopModel | No | Tests if the Index method returns an ActionResult of IEnumerable of LaptopModel |
| LaptopAPIControllerTests\_SearchResults\_ReturnActionResultOfIEnumerableLaptopModel | No | Tests if the SearchResults method returns an ActionResult of IEnumerable of LaptopModel when a search term is provided |
| LaptopAPIControllerTests\_ShowOneLaptop\_ReturnActionResultOfLaptopModel | No | Tests if the ShowOneLaptop method returns an ActionResult of a single LaptopModel when an ID is provided |

##### LaptopControllerTests:

| **Name** | **Is Stub** | **Description** |
| --- | --- | --- |
| LaptopController\_Index\_ReturnViewResult | No | Tests if the Index method returns a ViewResult, with a setup of fake LaptopModel instances. |
| LaptopController\_SearchResults\_ViewResult | No | Tests if the SearchResults method returns a ViewResult for a given search term. |
| LaptopController\_Toggle\_ReturnPartialViewResult | No | Tests that a partial is returned for the card partial view if session variable “list” is set to “false”. Also tests if a row partial view (a row in a table) is returned if session variable “list” is set to “true” |
| LaptopController\_getResults\_ReturnPartialViewResult | No | Tests the method that returns a partial view of either a table or grid of cards. |
| LaptopController\_ShowOneLaptop\_ReturnJsonResult | No | Tests if the ShowOneLaptop method returns a ViewResult for a given laptop ID. |
| LaptopController\_ShowOneLaptopJSON\_ReturnJsonResult | No | Tests if the ShowOneLaptopJSON method returns a JsonResult for a given laptop ID. |
| LaptopController\_Delete\_ReturnBool | No | Intended to test if the Delete method returns a boolean value. |
| LaptopController\_SaveEditReturnNewCard\_ReturnPartialViewResult | No | Intended to test if the SaveEditReturnNewCard method returns a PartialViewResult. |
| LaptopController\_SaveEditReturnNewRow\_ReturnPartialViewResult | No | Intended to test if the SaveEditReturnNewRow method returns a PartialViewResult. |
| LaptopController\_SaveEditReturnDetails\_ReturnPartialViewResult | No | Tests if the SaveEditReturnDetails method returns a PartialViewResult after updating a LaptopModel. |
| LaptopController\_CreateLaptop\_ReturnPartialViewResult | No | Tests CreateLaptop action and asserts that a partial view is returned. |

##### ShopControllerTests:

| **Name** | **Is Stub** | **Description** |
| --- | --- | --- |
| ShopController\_Index\_ReturnViewResult | No | Tests if the Index method returns a ViewResult. |
| ShopController\_SearchResults\_ReturnViewResult | No | Tests if the SearchResults method returns a ViewResult for a given search term, with a setup of fake LaptopModel instances. |
| ShopController\_Cart\_ReturnViewResult | No | Tests if the Cart method returns a ViewResult. |
| ShopController\_AddToCart\_ReturnJsonResult | Yes | Intended to test if the AddToCart method returns a JsonResult for a given laptop ID. ShoppingCart service is protected and static which makes this code currently untestable. |
| ShopController\_AddOneToCart\_ReturnViewResult | Yes | Intended to test if the AddOneToCart method returns a ViewResult for a given laptop ID. ShoppingCart service is protected and static which makes this code currently untestable. |
| ShopController\_RemoveFromCart\_ReturnPartialViewResult | Yes | Intended to test if the RemoveFromCart method returns a PartialViewResult. ShoppingCart service is protected and static which makes this code currently untestable. |
| ShopController\_ShowOneLaptop\_ReturnViewResult | No | Tests if the ShowOneLaptop method returns a ViewResult for a given laptop ID. |
| ShopController\_getTotalJson\_ReturnJsonResult | No | Tests if the getTotalJson method returns a JsonResult. |
| ShopController\_checkout\_ReturnPartialViewResult | Yes | Intended to test if the checkout method returns a PartialViewResult. ShoppingCart service is protected and static which makes this code currently untestable. |
| ShopController\_ThankYou\_ReturnViewResult | Yes | Intended to test if the ThankYou method returns a ViewResult. ShoppingCart service is protected and static which makes this code currently untestable. |

##### UserRolesControllerTests:

| **Name** | **Is Stub** | **Description** |
| --- | --- | --- |
| UserRolesController\_Index\_ReturnViewResult | No | Tests if the Index method returns a ViewResult. |
| UserRolesController\_Create\_ReturnViewResult | No | Tests if the Create method returns a ViewResult. |
| UserRolesController\_SubmitRole\_ReturnViewResult | No | Tests if the SubmitRole method returns a ViewResult after checking if a role exists. |
| UserRolesController\_Delete\_ReturnViewResult | No | Tests if the Delete method returns a ViewResult for a given role ID. |
| UserRolesController\_ShowUsers\_ReturnViewResult | No | Tests if the ShowUsers method returns a ViewResult. |
| UserRolesController\_ManageRoles\_ReturnViewResult | No | Tests if the ManageRoles method returns a ViewResult for a given email, with a setup of a fake ApplicationUser instance. |
| UserRolesController\_AddRoleToUser\_ReturnViewResult | No | Tests if the AddRoleToUser method returns a ViewResult when adding a role to a user, with a setup of a fake ApplicationUser instance. |
| UserRolesController\_DeleteRoleFromUser\_ReturnViewResult | No | Tests if the DeleteRoleFromUser method returns a ViewResult when deleting a role from a user. |
| UserRolesController\_Search\_ReturnViewResult | No | Tests if the Search method returns a ViewResult for a given search term. |