|  |  |
| --- | --- |
| **Student Name:** | Sarah Klioris |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Part 6 - Build a cloud application** | | | | | |
| **Task Description**  You are required to create a web application using ASP.net MVC framework. The following requirements need to be met.  **Functional Requirements**   * The application is built with ASP.net MVC framework. * The application provides authentication function. User can sign up / sign in / sign out.   + Use asp.net authentication feature.   + At least one service or web page in the application requires user to login. * Other than the sign up / sign in / sign out web form. You are required to build one additional web form with at least one input field and one submit button.   + Data validation need to be implemented for the input field in order to secure the input.   + Input data must be processed by back-end program; and results must be sent to the front-end and displayed. * Use bootstrap framework to style the front-end webpage.   **Programming Guideline**   * In order to develop maintainable code, make sure your code meet the C# programming guidelines * A copy of internal guideline is provided in Brightspace   **Use of source-code version control tool**   * Maintain your code on a source-code version control server, such github   **Comment your code with XML elements and auto generate internal document using a tool**   * Refer to document <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/xmldoc/xml-documentation-comments>, to comment code. * Use DocFX to generate document.   **Test the application**   * Plan and document test cases according to the requirements. * Test the application and document the results * Iterate design or build until test results meet requirements   **Publish and Deploy**   * Publish and deploy the application to AWS server.   **Use source-code version control**   * Maintain your code on a source-code version control server   You are required to submit the following documents in order to complete this part.   * A completed assessment document, with all checklist ticked and signed by your assessor. * A zip file, file name “Assessment2\_Part6\_Attachments.zip”, including the following   + All of the source codes (i.e. the completed VS project)   + All of the published files   + Any testing document & raw logs if applicable.   + Documentation generated by tools such as DocFX | | | | | |
| 1. Gather requirements from the task description and for the cloud application and list all the requirements? | | | * 1. App is built with ASP.net MVC framework.   2. Application provides authentication on at least one of its webpages.   3. Application contains a web form with at least one input field and one submit button. Data validation is implemented.   4. Input data must be processed by a back-end program then displayed on the front end.   5. Front-end must be styled with Bootstrap. | | |
| 1. List the development environment and tools required to create this cloud application | | | Visual Studio – Development Environment.  ASP.net framework (C#) MVC with individual authentication – Development Package. | | |
| 1. Briefly discuss the purpose of application security in software development and why it is important? | | | Application security ensures data is only seen by those who are meant to see it. Security is important because without it, nothing would be encrypted. For example, online banking wouldn’t work and people’s passwords would be lost. Phone book data and emails would be public to everyone which we don’t want. | | |
| 1. Explain how to configure authentication in asp.net web application and how to use authentication feature to define a custom code access permission. | | | Create a project in the ASP.NET MVC framework then select “Individual User Accounts” for authentication.  Go to the appropriate Controller and write [Authorize] on top of the page you want to protect.  From custom controller PathogenTypesController.cs: | | |
| 1. In the project, how did you plan authentication and authorisation strategy | | | I put [Authorize] above all the View pages in PathogenTypesController.cs. This means you have to be logged in to access any of these pages even if you type in their url. | | |
| 1. Program the cloud application. Provide screenshot of your source code to show that you have develop an appropriate authentication and authorisation strategy for the application | | | In PathogenTypesController.cs: | | |
| 1. For security reason, explain you how did you secure input and output handling? | | | In Model -> PathogenType.cs  I made name be between 2 and 50 characters.  I made URL be between 10 and 150 characters.  You are not allowed to press the submit button unless all your inputs fit the criteria. An error message will appear in the box of your wrong input. This secures input and output handling in one page. | | |
| 1. Provide screenshot to show that secure input & output are handled in your application | | | In PathogenType.cs: | | |
| 1. Describe the architecture of the web application you build | | | ASP.NET MVC written in C#.  There are four main views to this application: Home (Index), Register, Login and Enter Data.  The Home page is handled by the HomeController.  The Register and Login pages are handled by the AccountController.  The views that can be accessed by the Enter Data page are handled by the PathogenTypesController. They include: Index, Details, Create, Edit and Delete.  Data that is entered through PathogenTypes/Create, is then saved to the PathogenTypes model. This data is then shown on PathogenTypes/Index, and also on Home/Index.    This variable is declared in two controllers, HomeController and PathogenTypesController. On both index pages, it is returned in list form as part of the View() function, like so:  (PathogenTypesController.cs)  (HomeController.cs)  This allows the model data to be handled and shown on both these pages. | | |
| 1. Identify bootstrap component used in your application.   Reference the documentation of the component  Provide screenshot in the source code where the bootstrap component is used. | | | *Screenshot the Bootstrap component used in your front-end page*  *URL to the Bootstrap component online document*   1. <https://getbootstrap.com/docs/4.5/components/carousel/> 2. <https://getbootstrap.com/docs/4.5/layout/grid/> 3. [used default code in \_Layout.cshtml with minor edits]   *Screenshot the code used to generate the above component* | | |
| 1. Provide screenshot & URL link to show that source-code version control has been used.   Such as a github link | | |  | | |
| 1. Program the application and provide screenshot the source codes to show that your code adhere to programming guideline. | | | **Requirement 1: App is built with ASP.net MVC framework.**  See file system    **Requirement 2: Application provides authentication on at least one of its webpages.** In PathogenTypesController:  **Requirement 3: Application contains a web form with at least one input field and one submit button. Data validation is implemented.**  On Views -> PathogenTypes -> Create.cshmtl:  Validation occurs on Models -> PathogenType.cs  **Requirement 4: Input data must be processed by a back-end program then displayed on the front end.**  Data is processed on front end in: Views -> PathogenTypes -> Index.cshtml Views -> Home -> Index.cshtml  PathogenTypes -> Index.cshtml    Home -> Index.cshtml  **Requirement 5: Program is styled with Bootstrap.**  See Question 10. Views -> Home -> Index.cshtml | | |
| 1. Provide screenshot that you have commented the source code with XML elements. | | | PathogenTypesController:    HomeController: | | |
| 1. Provide screenshot that internal documents (i.e. generated by DocFX) are successfully generated.   (You may screenshot the webpage to show that the documents are viewable on web browser) | | | **Home page:**    **Register page (/Account/Register):**  **Login page (/Account/Login):**    **Enter Data page (/PathogenTypes/Index):**  **Create page (/PathogenTypes/Create):** | | |
| 1. Design test cases according to the requirements; and document all the test cases using the template.   “Template1\_Test\_Plan\_&\_Results.docx”  Minimal 3 test cases. | | | See document in project folder. | | |
| 1. Test the application and record test results.   You must follow the test plan finished in previous question, compete the “Actual Test Results” part for each test case, and determine if it is a pass or fail.  You must also capture & store test logs and submit them as “raw log” | | | See document in project folder. | | |
| 1. Are all test results satisfied? If not, iterate design or build until test results meet requirements.   List at least one failed test case, and specify how do you debug the code and fix the issue.  You need to show your assessor that you can use Visual Studio integrated debug tool & proper debug techniques to troubleshoot the application. | | | Error:  *Describe the process you take for troubleshooting*  *Screenshot the how did you use Visual Studio debugger for troubling shooting* | | |
| 1. Publish and deploy the application to AWS server. Provide screenshot to show that your application is running on AWS server.   Provide a URL to the application | | |  | | |
| 1. Provide screenshot of each page of the application. | | | [FROM AWS LINK ON MY OTHER COMPUTER]  Home page (consistently updated):  Enter Data page:  Create page:  Register page:  Log-in page: | | |
| 1. Explain how RSA cryptography works.    1. Discuss Symmetric / Asymmetric Cryptography    2. Discuss data encryption and decryption    3. Discuss public / private key | | |  | | |
| 1. Program using Public/Private Key Encryption. Provide source code screenshot of the following key part of the program    1. Generate public / private key pair    2. Encrypt and decrypt data | | | *Screenshot the code where Key pair are generated*  *Screenshot the encrypt function*  *Screenshot the decrypt function*  *Run the program and screenshot the results, including key pair generation and data encryption & decryption.* | | |
| **Checklist (To be completed by the learner’s facilitator)** | | | | YES | NO |
| 1. Learner has gathered and documented all requirements. | | | |  |  |
| 1. Learner has built the application | | | |  |  |
| 1. Learner has used a source-code version control tool | | | |  |  |
| 1. Learner has commented code with XML element and generated document with tool such DocXF | | | |  |  |
| 1. Test plan has coved all requirements | | | |  |  |
| 1. Test cases have all passed | | | |  |  |
| 1. Application has been published and deployed to cloud server | | | |  |  |
| **Assessor Name** | *Insert Name* | **Assessor Signature** | *Insert Signature* | **Date** | *Insert Date* |