**ASSESSMENT AND INTERNAL VERIFICATION FRONT SHEET (Individual Criteria)**

**(Note : This version is to be used for an assignment brief issued to students via Classter)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Course**  **Title** | B.Sc. (Hons.) Software DevelopmentB.Sc. (Hons.) Business Analytics | | | Lecturer Name & Surname | Ryan Attard | |
| **Unit Number & Title** | | [ITSFT-506-1616 - Enterprise Programming](https://moodle.mcast.edu.mt/course/view.php?id=3560) | | | | |
| **Assignment Number, Title / Type** | | 3, Building an Enterprise Application using a clean architecture / Home | | | | |
| **Date Set** | | 2/2/2024 | **Deadline Date** | 9/2/2024 | | |
| **Student Name** | **Nick georg** | | **ID Number** | 0203270A | **Class / Group** | Swd 6.2B |

|  |  |
| --- | --- |
| **Assessment** **Criteria** | **Maximum Mark** |
| *KU1.4: Clarify and relate the process models used* | 5 |
| *KU3.1: Describe the management standpoint in implementing enterprise solutions* | 5 |
| *KU4.1: Describe what is meant by cloud services and delivery models* | 5 |
| *AA2.2: Demonstrate software design patterns in specific problems found in enterprise applications* | 7 |
| *AA4.2: Illustrate what methods can be used to upload content onto cloud services* | 7 |
| *SE4.4: Revise and evaluate the content application and its appropriate use on cloud services* | 10 |
| *AA2.3: Select and implement appropriate design patterns to a solution being implemented* | 26 |
| *SE1.3: Construct and Ascertain that enterprise standards fit within an enterprise solution* | 35 |
| **Total Mark** | 100 |

|  |
| --- |
| **Notes to Students:** |
| * This assignment brief has been approved and released by the Internal Verifier through Classter. * Assessment marks and feedback by the lecturer will be available online via Classter (<Http://mcast.classter.com>) following release by the Internal Verifier * Students submitting their assignment on VLE will be requested to confirm online the following statements:   **Student’s declaration prior to handing-in of assignment**   * I certify that the work submitted for this assignment is my own and that I have read and understood the respective Plagiarism Policy   **Student’s declaration on assessment special arrangements**   * I certify that adequate support was given to me during the assignment through the Institute and/or the Inclusive Education Unit. * I declare that I refused the special support offered by the Institute. |

**AA4.2:**

Azure blob storage.

Justification: Azure blob storage can relatively cheaply handle massive amounts of data. It’s usually not great at storing structured data such as databases. However, for this case there aren’t really any relations in the DB and pretty much everything is stored in one table. So blob storage would work well enough.

However, if the DB were more complex, I would choose Azure SQL Database. This would be easier to use and would be able to keep relationships between tables.

Additionally, blob storage is cheaper than Azure SQL DB and since Blob storage allows for the storage of any binary data, if required you could store the image itself too instead of just the path.

**SE4.4:**

**Cache Services:**

**Responses from API/DB (Appointments)**

**Since they can only be added and not removed or updated. It would make sense to chache the responses so that you don’t have to constantly call the data making it technically faster. An additional benefit to this would be cost savings as some services charge per action.**

**CDN:**

**CDN is best used for large files like images, videos, etc**

**CDN Servers will be spread out across the world. Making it so that if a user from Australia requests a Image it doesn’t need to travel all the way from Malta but instead would be pulled from a server closer(either in Australia or a neighbouring region/country)**

**KU4.1**

**3 containers, no.1 for the web part holding all the view, controllers, etc. no.2 for data stuff. And no.3 for the images and other large stored files**

**Reason: Modularity. It lets you keep things separate and neat. Maintainability. Separate containers allow you to work on the indpendaly(ish) allowing you to update/ modify them one at a time.**

**AA2.2**

**1.** **A computer screen shot of text

Description automatically generated**

**2.** **A screenshot of a computer program

Description automatically generated**

**KU3.1:**

**Data Create a new Table/modify old to store the users and store the user ID as a FK to Appointments table**

**Business Logic Modify** ProductsController by adding a function with [Authorize] set at the beginning. This will ensure that only logged in/non-annon users have access to it. Then retrieve the user ID and save it when adding a calendar entry.

**Presentation Modify the view for list of appointments to only show the current/logged in user’s appointments**

**Modify the cshtml so that it also passes the User ID when adding new appointments so the correct one can be retrieved later.**

**Create a new view model for the logged in users to be able to see their appointments**

**KU1.4:**

**Epic 1: User account management**

**User story 1: As a dev I want to create a new table to store relevant user information.**

**User story 2: As a dev I want to**

**Epic 2: Logged in user features.**

**User story 1: As a dev I want only the relevant logged in user to be able to add thing to their calendar**

**User story 2: As a dev I want only the logged in user to be able to see said users calendar data.**