CA2 Individual Report

|  |  |
| --- | --- |
| Name | KAUNG HSET AUNG |
| Student Id | P2340698 |
| Class | DIT/FT/1B/11 |
| Github Repository URL | <https://github.com/ST0503-BED/bed-ca2-drewxkaung> |
| Github Account ID | drewxkaung |

For each competencies, find links to pull requests/commits/files that demonstrate the completion of the requirement. Replace each “**?**” with your Self Rating.

For Self Rating, you may rate yourself accordingly if you feel that you:

1. Have little or **no** understanding. and did not attempt the requirement
2. Have **limited** understanding of the specific competency
3. Have **basic** understanding and only able to replicate examples from tutorials/practicals.
4. Have **adequate** understanding and can extend from what you have learned to fulfil specifications.
5. Have **solid** understanding in the specific competency, able work on the requirement without much references.
6. Have **excellent** understanding and implemented the requirement according to latest industry guidelines, best practices and documentations.

**Important**

1. You are require to provide for each competency:
   * A brief **description**
   * **One or two** of your best implementations with URL **link** to respective repository request/commits/files.  
     **The implementations may come from Section A or B.**
   * You may also provide **screenshots** using POSTMAN to test API test.
2. You are to ensure the hyperlink in this document works. **Failure to do so will result in a 50% deduction of marks.**

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Competencies | Describe What Was Done | Self Rating |
| 1 | Backend Server | *(How did you handles incoming requests, communicates with the database, and provides appropriate responses?)*  MVC architecture. Routes for handling incoming request, Models to communicate with database and Controllers to send user appropriate and informative responses.  Moreover, I updated my backend in accordance with the CA2 needs, new endpoints are also added. Incoming requests from front end are responded with correct data by using fetchmethod from the endpoints I developed. Using if-else statements to handle success-error status.  url: <https://github.com/ST0503-BED/bed-ca2-drewxkaung/commit/2bf94d05a4c88b11f1537d7c14409d9c46c024e2> | 5/5 |
| 2 | Functionality | *(Did your features implemented meets the specified requirements and fulfils its intended purpose?)*  Yes  Requirement says to smoothly authenticate the flow.  I add additional step of authentication. Eg. User must be not only register in user table, but also in student table as well to access the Student-related features    url: <https://github.com/ST0503-BED/bed-ca2-drewxkaung/commit/b05f3c123a962f92a69c3f0bb3d0ebbb752e8e76> | 5/5 |
| 3 | Code Quality | *(How did you organise your code to ensure maintainability, readability and adherence to coding best practices?)*  For maintainability and readability, I label and segregate both controller and model *files* with clear comments, along with the proper indentation. Brief task description is also included to inform the user or future developer about the expected behaviour.  *Controller*    *Model* | 4/5 |
| 4 | Modularity | *(How did you've organized your project to promote code reusability and maintainability?)*  In the places where both successful or unsuccessful when backend response for both comes with { message: #####} I reuse the code. | 3/5 |
| 5 | Error Handling | *(How did you manage errors, provide informative feedback, and handle exceptional situations?)*  Proper structure of controller and response message for each response status are used to give informative feedbacks in terms of backend.    In terms of front end, used if-else statements to show conditional error and success feedback based on response status or data from back end. | 4/5 |
| 6 | Documentation | *(What was written for comments, readme and external documentation?)*  Readme mentioning what front-end pages are there for the user, and briefing instruction on each html page, what are displayed, how can user interact, how the backend will be take care.  url: <https://github.com/ST0503-BED/bed-ca2-drewxkaung/commit/1d88ac41e880f648b9b8d24cbaae2bd8fd189725> | 4/5 |
| 7 | JWT for Authentication | *(How did you generate, validate, and manage user tokens?)*  Generate token upon successful login and store it for further authentication. Added extra step of authentication. Eg. User must be not only register in user table, but also in student table as well to access the Student-related features    url: <https://github.com/ST0503-BED/bed-ca2-drewxkaung/commit/b05f3c123a962f92a69c3f0bb3d0ebbb752e8e76> | 5/5 |
| 8 | BCrypt for Password Hashing | *(How did you securely store and validate user passwords)*  The password user register is hashed with Bcrypt for 10 salt rounds before storing in mysql. And again, bcrypt comparing the hashed password in mysql when user request a log in to authenticate  url: <https://github.com/ST0503-BED/bed-ca2-drewxkaung/commit/34961044a5fb4fe990960cedf77d85e31972ad37> | 5/5 |
| 9 | Frontend to Backend HTTP Requests | *(How did you setup frontend communication with the backend?)*  I used fetchMethod for communication between frontend and backend | 4/5 |
| 10 | HTML & CSS Proficiency | *(Provide example of how you managed to create structured layouts and visually appealing styles)*  Consistent nav-bar across all html pages, and consistent school related functional buttons across all School related pages for better readable layout.  Use of aesthetic color and background for visual appealing  Using CSS files to share consistent layout and styling across the project | 4/5 |
| 11 | Frontend Framework | *(What did you do to enhances the user experience and simplifies complex tasks?)*  User do not need to manually enter confusing id numbers.  URL carries user’s student ID and upon clicking Learn this spell button, event listener carries the Spell ID. Easier for user to use the feature | 4/5 |
| 12 | JavaScript and DOM Manipulation | *(How does your frontend utilise dynamic content and DOM manipulation?)*  My frontend employs JavaScript for DOM manipulation, enabling dynamic updates to HTML elements. We utilize Fetch API to asynchronously fetch and update content without reloading the entire page.  .addEventListener to handle user interactions, ensuring a responsive and interactive user experience. | 4/5 |