Back-End Development (BED)

Level 2.1

Academic Year 2025-26 April Semester

SCHOOL OF INFOCOMM TECHNOLOGY

Diploma in Information Technology

ASSIGNMENT

Duration:	**Week 6 to 7, Week 11 to 15 (26 May – 03 Aug 2025)**
Weightage:	40%
Individual/Team/Both:	Both
Deadlines:	Checkpoint 1 Submission: **2359hr Sunday 08 Jun 2025 (Week 7)** Checkpoint 2 Submission: **2359hr Sunday 13 Jul 2025 (Week 12)** Final Submission: **2359hr Sunday 03 Aug 2025 (Week 15)**

Penalty for late submission:

- 10% of the marks will be deducted for each day (inclusive of Saturdays, Sundays and public holidays) after the deadline.
- No submission will be accepted after 2359hr 10 Aug 2025.

There is a total of **11** pages (including this page) in this write-out.

PLAGIARISM WARNING:

If a student is found to have submitted work not done by him/her, he/she will not be awarded any marks for this assignment. Disciplinary action may also be taken.

Similar action will be taken for the student who allows other student(s) to copy his/her work.

OBJECTIVES

On completion of this activity, students will have learned:

- To work as a team to develop a back-end application integrated with a database, and enabling interaction with a front-end application.
- To design and code the functions of the backend application using Node.js and Express technology, implementing RESTful API integrated with a Microsoft SQL Server database to perform CRUD (Create, Read, Update, Delete) operations.

SCOPE

You are to work in **teams of 4 or 5 students** for this assignment based on the **Project Brief provided in Appendix A**.

This assignment consists of both team and individual components. Each team member is expected to contribute ideas and participate in the development of the team component. A peer assessment will be conducted, and any member found to have made little or no contribution may be penalised.

For the individual component, **each student's work must be clearly distinguishable**. No two members should implement the same or very similar functionality. For example, "user sign-up" and "admin adding a user" are considered too similar, even if performed by different roles. Students are advised to base their individual components on different database tables, or on different CRUD operations if using the same table, to ensure uniqueness and avoid similarity.

The scope of both team and individual components includes basic and advanced requirements. Each student must complete the basic features to achieve a pass grade.

Basic Requirements:

Team Component:

- A home page named as ***index.html (back-end code: app.js).***
- A navigation structure which links all features on the front-end application together.
- A credit page (***credit.html***) acknowledges the source of information and media providers.
- A database that provides information support for the back-end application.
- A GitHub repository for sharing source code and files among team members***.***

Individual Component:

- One feature which implements the CRUD (Create, Retrieve, Update, Delete) operations through RESTful API to the database***.***
- Evidence of testing that the features implemented are working.

Advanced Requirements:

Team Component:

- Efficient and effective use of GitHub repository (e.g., using commits, branching and pull requests).
- Preparing API documentation for the functions implemented in the back-end application.
- Implementing unit tests for functions in the back-end application to ensure correctness and reliability.

Individual Component:

- Implementing additional features with CRUD operations
- Incorporating third-party APIs to extend functionalities of the application (Third-party APIs should be invoked from the back-end application).
- Integrating front-end application with the back-end application for user input, interactions and display.
- Error-handling in back-end functions and prompting users with meaningful messages.
- Control the security access rights to relevant functions through user authentication and authorization.

Multimedia materials (e.g. images, music, video, etc.) may be sourced from the Internet. However, you must provide proper citations on the credit page to avoid copyright infringement.

The **back-end application MUST be developed using Node.js and Express framework**. It must include database access and expose RESTful APIs to support interaction with a front-end application.

IMPORTANT NOTES:

Although front-end development and database are necessary for implementation in this assignment, **the primary focus remains on back-end development using Node.js and Express**.

Students from the same team may receive **different grades for team component** if tutors find sufficient evidence of significantly unequal contributions.

The **completeness, polish, complexity, and uniqueness** of the application will influence how well you perform. Marks awarded for features and requirements will depend on how well they are implemented, how comprehensive they are, and their level of complexity. You and your team are encouraged to go beyond the listed requirements and explore additional enhancements where appropriate, in consultation with your tutors.

DELIVERABLES

There are **3 checkpoints** for submission of deliverables for this assignment:

Checkpoint 1 Submission (10%) - Due in Week 7

At this stage, the team should focus on the proposed features and overall design of the application.

Deliverables:

- An **assignment cover page** indicating the Student ID and name of each team member, along with the individual features to be developed by each student.
- A **report** placed after the cover page, containing:
  - A description of the features to be developed, clearly indicating the Student ID and name for each individual feature.
  - Wireframes illustrating the design and placement of the features, including the navigation structure and page layouts of the application.

(Note: Compress the above files into a single zipped file and submit it to POLITEMall. Only **one team member** should submit the Checkpoint 1 deliverables on behalf of the team.)

Checkpoint 2 Submission (15%) - Due in Week 12

At this stage, the team should have completed the core parts of the proposed features.

Deliverables:

- An **assignment cover page** indicating the Student ID and name of each team member, with the individual features **completed** at this stage, and a link to the GitHub repository containing the source code. (Note: By this stage, each student should have implemented at least one GET function and one POST/PUT/DELETE function.)
- An **Individual video** (not exceeding 5 minutes) from each student, demonstrating the features completed or explaining any challenges encountered if the core features are not yet completed.
- The **back-end application source code** and associated media files organised into appropriate sub-folders.
- **Database creation scripts** and environment files required to set up the back-end database. These must also include sample data.

(Note: Compress the above files into a single zipped file and submit it to POLITEMall. Only **one team member** should submit the Checkpoint 2 deliverables on behalf of the team.)

Final Submission (75%) - Due in Week 15

At this stage, the team should have completed all features and pages proposed during the Checkpoint 1 submission.

Deliverables:

- An **assignment cover page** indicating the Student ID and name of each team member, with the individual features completed by each student, and a link to the team's GitHub repository for the assignment.
- A **group video** (not exceeding 40 minutes) that includes:
  - An introduction to the overall application structure.
  - A demonstration of the features developed as a **team**.
  - A demonstration of each student's **individual** features, with the student's name mentioned when presenting their respective features.
- The **back-end and front-end applications' source code**, along with any associated media files, organised into appropriate sub-folders.
- Database creation scripts and environment files required to set up the back-end database. These files must also include sample data.
- API documentation if the feature is implemented.

(Note: Compress the above files into a single zipped file and submit it to POLITEMall. Only **one team member** should submit the final submission deliverables on behalf of the team.)

IMPORTANT NOTES:

One team member is to submit all deliverables on behalf of the team via the designated **POLITEMall** assignment submission folders for each checkpoint.

Do not submit deliverables by email or messaging to tutors.

All students are strongly advised to **keep a backup copy** of their project files in case of unforeseen circumstances.

Teams are advised to use a **GitHub repository** to facilitate sharing and updating of programming code and files. Tutors should be granted **Collaborator access** to the GitHub repository for monitoring and assessment purposes.

GENERATIVE AI USAGE WARNING:

Submitting assignments or parts of assignments that are substantially generated by Generative AI tools (e.g., AI writing assistants, code generators) and presented as the student's own original work is considered a form of academic misconduct.

If a student is found to have submitted such work, they will not be awarded any marks for this assignment. Disciplinary action may also be taken.

ASSIGNMENT DEMO

Your team will demo the front-end and back-end applications in **Weeks 16 to 17**. Each member is expected to demonstrate the features they have developed and respond to any queries from the tutor. Please note that the demonstration duration will be kept to **40 or 50 minutes** (approximately 10 minutes per student, including Q&A).

After the demo session, each student must submit a **personal reflection** in POLITEMall on the work done and experience gained from the assignment. In addition, students are to provide **peer evaluation** on the contributions of their team members using the template provided. The peer evaluation must be submitted **individually** through POLITEMall.

IMPORTANT NOTES:

Please be **PUNCTUAL** for the demonstration at the timeslot allocated by tutor. Penalty will be applied to the Individual portion of the assignment if student is late without a valid reason.

(To be continued on next page)

MARKING CRITERIA

Team Component: **20%** of Final Submission marks

	Criteria	%
1.	A comprehensive video with excellent explanation and testing of the features/functions implemented, clearly demonstrating each team member's contribution.	25%
2.	Rich and relevant information presented aligns with the assignment theme.	25%
3.	Effective use of GitHub for collaboration, including use of commits, branches and pull requests.	15%
4.	Clear and complete API documentation prepared (e.g., using Swagger) and implementation of unit tests for back-end functions.	15%
5.	A credit page (credit.html) that properly references all external sources of information and media.	10%
6.	A well-organised folder structure for the back-end application.	10%

Individual Component: **80%** of Final Submission marks

	Criteria	%
1.	Clear presentation and ability to answer questions with understanding during the demo session.	25%
2.	Implementation of one feature that performs CRUD (Create, Retrieve, Update, Delete) operations via RESTful API to the database.	15%
3.	Implementation of additional features and/or integration of third-party APIs (invoked from the back-end application).	15%
4.	Proper use of user authentication and authorization to control access to back-end functions.	10%
5.	A well-designed database schema and data that supports the back-end application.	10%
6.	A front-end application that integrates seamlessly with the back-end application for user input, interaction and display.	10%
7.	Good programming practices in Node.js (e.g., naming conventions, source code comments, folder structure).	5%
8.	Effective error-handling in back-end functions with meaningful user error messages.	5%
9.	A thoughtful reflection on personal learning, challenges faced, and contributions made.	5%

PERFORMANCE CRITERIA FOR GRADING

Marks awarded will be based on **source code**, **video**, as well as the **student's degree of understanding** of work done, as assessed during the demonstration.

A Grade

- Implements one feature with full CRUD operations successfully.
- Implements multiple additional features with CRUD operations and/or relevant RESTful API functionality using third-party APIs.
- Presents rich, relevant content that aligns closely with the assignment theme.
- Provides complete input validation across all user inputs, with consistent and clear feedback to users.
- Implements robust error-handling code to prevent unexpected program crashes and provide clear user feedback.
- Applies comprehensive security access rights through user authentication and authorization, ensuring secure access control for all relevant functions.
- Seamless integration between the back-end and front-end applications, making the application easy to use with clear, user-friendly navigation across all features.
- Presents a visually polished and consistent home page, credit page and other pages in the front-end application.
- Demonstrates excellent programming practices, including consistent naming conventions, extensive inline comments, and a well-organised folder structure.
- Demonstrates efficient and collaborative use of GitHub (e.g., using frequent commits, branching, pull requests with reviews).
- Provides complete API documentation (e.g., using Swagger) and implements unit tests for back-end functions.
- Provides a comprehensive video with excellent explanation and testing of the features/functions implemented.
- Shows excellent understanding of the work during assignment demonstration and explanation.
- Submits a thoughtful and insightful personal reflection that demonstrates deep understanding of the overall assignment experience and personal growth.

(To be continued on next page)

B Grade

- Implements one feature with full CRUD operations successfully.
- Implements at least one additional feature with CRUD operations and/or relevant RESTful API feature through third-party APIs.
- Presents relevant content that supports the assignment theme.
- Provides input validation for most key user inputs, with some feedback to users.
- Provides good error-handling code to prevent unexpected program crashes.
- Applies sufficient security access rights through user authentication and authorization for key functions.
- Functional integration between the back-end and front-end applications, allowing users to interact with the features.
- Presents a decent home page, credit page and other pages in the front-end application.
- Demonstrates good programming practices, including appropriate comments and a reasonably organised folder structure.
- Demonstrates effective use of GitHub (e.g., using commits and branching).
- Provides a comprehensive video with a good explanation and testing of the features/functions implemented.
- Shows good understanding of the work during assignment demonstration and explanation.
- Submits a meaningful and well-explained personal reflection that shows good understanding of the assignment experience.

C Grade

- Implements one feature with full CRUD operations successfully.
- Provides input validation for some user inputs with limited feedback to users.
- Provides basic user authentication and authorization to protect access to at least one key function.
- Provides basic integration between back-end and front-end applications to allow user input and display of data.
- Includes sufficient comments to help others understand and maintain the code.
- Provides a decent video with sufficient explanation and testing of the features/functions implemented.
- Shows sufficient understanding of the work during assignment demonstration and explanation.
- Submits a personal reflection that shows sufficient understanding of the assignment experience.

(To be continued on next page)

D Grade

- Implements one feature with full CRUD operations successfully.
- Provides some way to capture user input and display data from the database.
- Provides some comments to help others understand the code.
- Provides a decent video with some explanation and testing of the features/functions implemented.
- Shows some understanding of the work during assignment demonstration and explanation.
- Submits a personal reflection that shows some understanding of the assignment experience.

APPENDIX A – PROJECT BRIEF

Background

Established in 1995, Lions Befrienders Service Association (Singapore) is a voluntary welfare organisation dedicated to providing holistic care and support to help seniors age healthily in place with community participation, enabling them to enjoy purposeful and meaningful lives.

Problem Statement

How might we develop a web-based application tailored to the needs of seniors?

While seniors may turn towards technology for support, most digital platforms are not designed with their physical, cognitive, or cultural needs in mind. As a result, these tools often create more barriers than solutions.

Your solution should empower the seniors to confidently navigate daily life, stay socially connected, and manage their personal health.

Your team may consider the following suggested features to implement, including but not limited to:
- Navigating transport systems, public services & local facilities
- Connecting with others through hobbies, chats & virtual events
- Managing medications, health appointments & basic records

End-of-Paper