|  |  |  |
| --- | --- | --- |
| **Bachelor of Computing Systems** | | |
| **Course No: ISCG7420** | **Web Application Development** | **Level: 7**  **Credits: 15** |

|  |  |
| --- | --- |
| **Student Name:** | **Student ID:** |
| **Assessment Type: Assignment** | **Weighting: 60%** |
| **Due Date and Time: April 27th 2025 17:00PM** | **Total Marks: 100** |

|  |  |
| --- | --- |
| **Student declaration**  I confirm that:  • This is an original assessment and is entirely my own work.  • The work I am submitting for this assessment is free of plagiarism. I have read and understood the [Academic Integrity Procedure](https://www.unitec.ac.nz/sites/default/files/public/documents/AC%202.8%20Academic%20Integrity%20Procedure_20231025.pdf) (including the key principles about using the use of Generative Artificial Intelligence (GenAI) listed in Section 3.2). I have also read and understood the [Student Disciplinary Statue](https://www.unitec.ac.nz/sites/default/files/public/documents/AC5.0%20Student%20Disciplinary%20Statute%20Aug-2021.pdf).   * Where I have used ideas, tables, diagrams etc of other writers, I have acknowledged the source in every case. | |
| **Students Signature:** | **Date:** |

**Assessment Mapping**

After completing this assessment, the student will have met the following learning outcomes:

|  |  |  |
| --- | --- | --- |
| **Learning Outcome** | **Task A**  **(30 Marks)** | **Task B**  **(70 Marks)** |
| 2. Discuss the tools used in providing web-based applications. | ✓ |  |
| 3. Design and implement a dynamic web application using a range of languages/technologies/tools. |  | ✓ |
| 4. Discuss and evaluate the available data access technologies. | ✓ |  |

**Assessment information/guidelines:**

* This is an individual assignment.
* Read the provided scenario and complete both tasks.
* Task 1: Evaluate two programming languages and two data access technologies. Present your findings in a report of 1000 words [+/- 10%], excluding reference list.
* Ensure all sources are referenced in-text and include a complete reference list at the end of the report, following APA referencing guidelines.
* After the references, include a list of any generative artificial intelligence tools you have used (e.g., Copilot, ChatGPT), along with a brief description of how they were utilised in your work.
* Task 2: Design and implement a Django web application.

**Assessment submission instructions:**

* Complete the cover sheet (page 1 of this document) and attach it to the front of the assessment.
* Submit your task 1 via the Turnitin link on the Moodle course page.
* For Task 2, use the Assignment 1 directory in your class GitHub repository to store your task and add your lecturer as a collaborator. Your lecturer will periodically clone your repository to monitor your progress. On the assignment due date (and up to 72 hours afterward), your lecturer will download the most recent version of your Task 2 submission. It is your responsibility to ensure that GitHub always contains the latest version of your code.
* For Task 2, if you fail to push your most recent changes by the deadline, your lecturer will assess the latest version available on GitHub. Regularly update your files.
* **If you submit the report and code you did not write, you will fail this assignment and the course. If you need help, ask your lecturer well before the deadline rather than resorting to plagiarism.**

**Scenario:**

**Te Whare Rūnanga** **Ltd** is creating a web application for reserving conference rooms. The company has 10 conference rooms. The application should allow users to view available rooms, make reservations, and manage existing reservations.

1. **View Available Rooms**: Users can see a list of all available conference rooms.
2. **Make Reservations**: Users can book a conference room for a specific date and time.
3. **Manage Reservations**: Users can view, edit, or cancel their existing reservations.
4. **User Authentication**: Users must log in to make or manage reservations.
5. **Admin Panel**: Administrators have the following capabilities:
   1. Add, Edit, and Delete Rooms: Manage the details of conference rooms.
   2. Reserve for Users: Make reservations on behalf of users.
   3. Cancel User Reservations: Cancel reservations made by users.
   4. View All Reservations: Oversee all reservations in the system.
   5. Manage User Accounts: Add, edit, or delete user accounts.
6. **Notifications**: Users receive confirmation and reminder notifications for their reservations.

**Task 1: Technology Evaluation [30 Marks]**

Instructions:

* Prepare a report analysing the tools and technologies relevant to developing the conference room reservation system.
* Your report should follow a logical structure, including an introduction, main discussion, conclusion, and proper referencing The report should be 1,000 words (+/- 10%), excluding references (APA 7th referencing style). (6 marks)
* You report should address:

1. Tools for developing web-based applications (8 marks)

Analyse the tools for developing the conference room reservation system for Te Whare Rūnanga Ltd. Discuss two of the following:

* PHP: Explain its role in server-side scripting and whether it would suit this project.
* Python: Discuss its use in web frameworks like Django and why Django might be an appropriate choice.
* Node.js: Describe its importance in building scalable network applications and its potential use in this project.
* Other Tools: Consider other relevant tools, such as Ruby on Rails, ASP.NET, etc., and their applicability to this project.

1. Evaluation of Data Access Technologies (8 marks)

Evaluate the data access technologies provided below in the context of the conference room reservation system.

* Query Languages: Discuss SQL and its importance in managing the project’s database.
* Object-Relational Mapping (ORM): Explain how ORMs like Django ORM simplify database interactions and why it is suitable.

Based on your analysis and evaluation in Questions 1 and 2, select a development language and data access technology for the conference room reservation system. Justify your choice.

**Task 2: Web Development [70 Marks]**

Instruction:

Design and implement a dynamic web application for a conference room reservation system using Django and SQLite.

Requirements:

* Develop the application with Django as the primary framework.
* Use SQLite or another appropriate database for data storage.
* Host the completed application on Vercel, ensuring all required packages are configured.
* Upload your source code to GitHub and maintain version control throughout the development process.

Ensure that your implementation follows best practices in web development, including usability and maintainability.

**Assessment Marking Schedule**

Student ID:

|  |  |  |
| --- | --- | --- |
| **Contents/Criteria** | **Marks Allocated** | **Your Marks** |
| **Task 1** |  |  |
| Tools for developing web-based applications | 8 marks |  |
| Evaluation of Data Access Technologies | 8 marks |  |
| Selection of development language and data access technology with justification. | 8 marks |  |
| Report structure and referencing | 6 marks |  |
| **Task 2** |  |  |
| Application functionality | 25 marks |  |
| Django implementation | 20 marks |  |
| Database integration | 10 marks |  |
| Hosting on Vercel | 5 marks |  |
| GitHub Version Control | 5 marks |  |
| Code quality and best practices | 5 marks |  |
| **Total marks** | 100 marks |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task 1 Marking Rubric** | | | | |  |
| **Performance Criteria** | **Outstanding** | **Accomplished** | **Capable** | **Requires development** | **Total** |
| Tools for developing web-based application  (8 marks) | Provides a comprehensive and well-reasoned analysis of two tools. Clearly explains their roles, advantages, and applicability to the project with strong justifications. There may be some minor issue with justification.  7 – 8 marks | Discusses at least two tools with mostly clear explanations and reasonable justifications but may lack depth or application to the project.  5 – 6 marks | Covers at least one tool with basic explanation but lacks depth in analysis, justification, or relevance to the project.  3 – 4 marks | Limited or unclear discussion of tools. Little to no analysis, missing key details or relevance to the project.  0 – 2 marks |  |
| Evaluation of Data Access Technologies  (8 marks) | Provides a detailed evaluation of SQL and ORM. Clearly explains their benefits, limitations, and applicability to the project, using examples. There may be some minor issues with explanation.  7 – 8 marks | Evaluates SQL and ORM with reasonable depth but may have minor gaps in justification, clarity, or examples.  5 – 6 marks | Covers SQL and ORM briefly, but lacks strong reasoning, details, or examples. Some explanations may be unclear or incomplete.  3 – 4 marks | Limited or incorrect evaluation of SQL and ORM. Lacks clarity, relevance, or depth.  0 – 2 marks |  |
| Selection of development language and data access technology with justification.  (8 marks) | Provides a clear, well-structured, and well-reasoned justification for the selected language and data access technology. Uses strong comparisons and evidence to support the choice and directly connects it to the needs of the conference room reservation system.  7 – 8 marks | Justifies the selection with reasonable arguments and some link to the conference room reservation system but lacks depth in comparisons or supporting evidence.  5 – 6 marks | Provides a basic justification with limited reasoning or evidence. Connection to the conference room reservation system is weak or unclear.  3 – 4 marks | Weak or missing justification with poor reasoning or no clear connection to the conference room reservation system.  0 – 2 marks |  |
| Report  (6 marks) | Report is well-structured and clear with logical flow. Writing is free from major grammar or spelling errors. Referencing is correctly formatted and follows the required citation style**.**  5 – 6 marks | Report is mostly well-organised and clear, with minor formatting issues. Some grammar or spelling errors. Referencing is mostly correct but may have minor formatting inconsistencies**.**  4 marks | Report is somewhat disorganised with noticeable grammar, spelling, or formatting issues. Referencing is attempted but contains multiple errors.  3 marks | Report lacks structure and clarity, with significant grammar, spelling, or formatting errors. Referencing is missing or incorrect.  0 – 2 marks |  |
|  | | | | **Total** |  |
| **Feedback** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task 2 Marking Rubric** | | | | |  |
| **Performance Criteria** | **Outstanding** | **Accomplished** | **Capable** | **Requires development** | **Total** |
| Application functionality  (25 marks) | All required functions are implemented and work correctly. Users can view available rooms, make reservations, manage reservations, and receive notifications. No major bugs.  22 – 25 marks | Most required functions are implemented and work correctly, with minor issues.  16 – 21 marks | Some required functions are implemented and work, but with noticeable issues.  10 – 15 marks | Major functions missing or system is largely non – functional.  0 – 9 marks |  |
| **Django** **implementation**  (20 marks) |  |  |  |  |  |
| Models = 6 marks | Models are well-defined with appropriate attributes and correct relationships.  6 marks | Models have mostly appropriate attributes and relationships.  4 – 5 marks | Models contain some appropriate attributes but have issues in relationships.  3 marks | Models are poorly defined, with incorrect or missing relationships.  0 – 2 marks |  |
| Views = 6 marks | Views handle all required functionalities efficiently.  6 marks | Views implement most functionalities correctly with clear logic. 4 – 5 marks | Views handle only some functionalities, with noticeable flaws in logic or implementation.  3 marks | Views fail to handle required functionalities, with major logical errors.  0 – 2marks |  |
| Templates = 8 marks | Templates are well-designed, user-friendly, and correctly implement authentication and permissions. There may be some navigation issues.  7 – 8 marks | Templates are well-structured and implement authentication and permissions with minor issues.  5 – 6 marks | Templates have design issues and authentication or permissions problems.  3 – 4 marks | Templates are poorly designed, lacking authentication and permissions.  0 – 2 marks |  |
| Database integration  (10 marks) | Efficient use of SQLite (or another Database Management System), with well implemented models and relationships. There may be use of local database.  9 – 10 marks | Functional database but with minor inefficiencies or lack of optimisation. Some missing table or data.  7 – 8 marks | Basic database setup with some structural issues.  4 – 6 marks | Poor database implementation, missing tables and data, or incorrect relationships.  0 – 3 marks |  |
| Hosting on Vercel  (5 marks) | Successfully deployed on Vercel with PostgreSQL, requirements.txt, and vercel.json correctly configured.  5 marks | Deployed on Vercel with minor issues in configuration.  4 marks | Deployment attempted but has major problems (e.g., broken features, missing dependencies).  2-3 marks | Not deployed or completely non-functional.   * 1. mark |  |
| GitHub Version Control  (5 marks) | Regular and meaningful commits with clear messages, proper use of branches and merging, follows Git best practices, effective collaboration using pull requests, and a well-structured repository with clear documentation.  5 marks | Frequent commits with generally clear messages, effective but slightly inconsistent branching and merging, mostly follows Git best practices, some use of pull requests, and a mostly organized repository.  4 marks | Commits are somewhat irregular with unclear messages, limited branching with unresolved conflicts, basic Git usage, minimal collaboration, and some organization but lacks proper documentation.  2-3 marks | Infrequent commits with unclear or missing messages, no branching or unresolved conflicts, minimal or incorrect use of Git, no collaboration or pull requests, and a disorganized repository with no documentation.  0-1 mark |  |
| Code quality and best practices  (5 marks) | Clean, well-structured, and efficient code with proper naming conventions, comments, and documentation. Follows best practices for readability, modularity, and maintainability. No syntax errors or warnings.  5 marks | Generally well-structured code with appropriate naming conventions and comments. Minor readability issues or inefficiencies but follows most best practices. Few minor syntax warnings.  4 marks | Code is functional but lacks structure and readability. Some inconsistent naming conventions and missing comments. Best practices are partially followed. May contain some syntax warnings.  2-3 marks | Poorly structured code with unclear naming conventions, minimal or no comments, and does not follow best practices. Contains syntax errors or significant readability issues.  0-1 mark |  |
|  | | | | **Total** |  |
| **Feedback** |  | | | | |