# 

SIT774

UNIT NAME

Danny Sittrop

219278745

Learning Summary Report

# Self-Assessment Details

The following checklists provide an overview of my self-assessment for this unit.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Pass (D) | Credit (C) | Distinction (B) | High Distinction (A) |
| Self-Assessment |  |  |  | ✓ |

Self-Assessment Statement

# Declaration

I declare that this portfolio is my individual work. I have not copied from any other student’s work or from any other source except where due acknowledgment is made explicitly in the text, nor has any part of this submission been written for me by another person.

Signature: **Danny Sittrop**

## Explain Web Technologies

1. Explain the relevance, purpose, and operation of technologies which underpin web applications.

## Client-side application design and development

1. Design and develop client side components of a web-based application using contemporary tools, languages and frameworks

## Web service design and implementation

1. Design and implement RESTful web services using contemporary tools, language and frameworks.

## Option identification and application deployment

1. Identify options for, and deploy web applications to suitable production environments.

## Achieved outcome justification

1. Justify achieved outcomes through providing relevant evidence and critiquing the quality of that evidence against given criteria.

Update footer with unit title and name. Then delete this box.

# Portfolio Overview

This portfolio includes work that demonstrates that I have achieved all Unit Learning Outcomes for SIT774 Web Technologies and Development to a High Distinction Level.

**Summary of what made it HD:**

* Most of the tasks I went beyond the minimum requirements of the task. This included different functions, more efficient code, and unique design elements.
  + I have successfully completed all tasks to meet the HD requirement
  + Minimum requirements
    - None of my tasks got returned due to incorrect code or functionality
    - Where I did request an extension, sometimes I was able to still submit in time and otherwise always submitted earlier than the extension request due date. The burn down chart shows that I have generally been ahead of schedule.
    - Very often received excellent comments on submission of the assessments
    - Have successfully completed ALL tasks in OnTrack.
  + Beyond distinction:
    - Scrolling banner on my web site
    - Made the login and search functionality operable
    - Focus on code efficiency as well as functionality (e.g. 9.2 Using helper functions, Using loops where possible. 9.1 displaying the actual error instead of a generic message for error 500. Task 3.4 not only did I create a 3-column web page, but also learnt about designing low/high-fidelity wireframes. Individual web site, I adapted some of the template bootstrap to suit my own page. I also created a new logo and branding to make the design more visually appealing. Focused on design as well as the technical implementation. 8.3D I improved on the layout and formatting on the /sales page. Also fixed the source JSON file where the RGB hex value for Black was incorrect.
    - Used comments religiously to hep with readability and de-bugging

**Learning Journey**

* **Where did I start, learn and where did it all take me**
* Since beginning the Masters of Business Analytics, I have become competent in a variety of programming languages, however never done any front-end work. This will enable me to interface between an Analytics model and a user through web design.
* What did I learn:
  + Basics of HTML (elements, tags and structure)
  + Navigation between and on websites
  + Inline styles
  + Inline / Internal / external CSS
  + Image layout and formatting
  + Website layout methods & Layout responsiveness
  + Incorporating Bootstrap
  + Building Forms
  + JavaScript
  + Properties, methods in classes
  + Event handling
  + Understanding the Document Object Model (DOM) and JS Object Notation (JSON)
  + Encapsulated JS (node JS) and Templates
  + Server-side and client-side programming (inc. validation in both)
  + Running local and external server to host web page
  + Increasing engagement with end-user and capturing information from your input (buttons, forms, validations, mailto:, dynamic pages etc)
  + Routing to various dynamic pages
  + Debugging CSS, JS, EJS, HTML
* Where did it take me:
  + I can now confidently build a modern featuring web site
* **Milestones & Hurdles**
  + I probably spent too much time on the initial design of the web page, instead of viewing it as an iterative process. The key was to have the base structure in place, not really concerning too much with the detail. I was also constrained by what I believed I could technically accomplish at the time.
  + When deviating from standard templates on the Bootstrap documentation sometimes required experimentation
  + I often knew logically how the code needed to be constructed, however did not know the exact syntax. Once I had practiced any given syntax enough I could write code more seamlessly.
  + Following the steps exactly to set up server locally or externally.
  + Finding all the relevant reference sites such as W3Schools, Bootstrap or
  + Validation of HTML code
  + Formatting of elements such as containers, rows and col to render correctly
  + Understanding the differences between margin, border, padding and content
  + Understanding the different ways to capture an event in JS

***For Pass: you need to indicate how you have demonstrated all Unit Learning Outcomes to an acceptable level.  
For Credit: you need to indicate how you have demonstrated all Unit Learning Outcomes to a good level.  
For Distinction: you need to indicate how you have been able to apply all of the Unit Learning Outcomes in achieving the distinction tasks.  
For High Distinction: you need to indicate how you have been able to extend beyond the material presented in the unit.***

## Summary of all learnings

The objective **"Explain Web Technologies"** focuses on understanding the relevance, purpose, and operation of core web technologies that underpin web applications. Through the various tasks completed, this objective has been met in the following ways:

1. **Understanding Web Page Structure**:
   * Tasks like **1.1P (Structure an HTML tree)** and **1.2P (Serve your own web page)** involved learning how to structure web pages using HTML tags and deploy them on a local server. This provided insight into the foundational elements of web applications, such as headers, footers, and body content.
2. **Creating and Validating Web Pages**:
   * In **2.3P**, I built more complex web pages with navigation, linking, and publishing to a web server, showing a deep understanding of how websites function and how users interact with them.
3. **Using Frameworks and Libraries**:
   * Tasks like **4.3D** involved working with **Bootstrap**, a popular CSS framework. This introduced how pre-built components like buttons, forms, and navigation bars can accelerate web development while ensuring responsiveness across devices.
4. **Dynamic Content and DOM Manipulation**:
   * The **6.xP and 7.xP** tasks involved working with JavaScript to manipulate the **Document Object Model (DOM)**, enabling dynamic content creation, event handling, and form validation. Understanding JavaScript's interaction with HTML elements is crucial for creating interactive, dynamic web applications.
5. **Server-Side Technologies**:
   * In tasks like **8.1P** to **8.3D**, I gained experience in **Node.js** for handling server-side operations such as routing, serving static files, and rendering dynamic web pages using **ejs** (embedded JavaScript templates). This demonstrated the importance of separating frontend and backend functionality in web development.
6. **Working with Databases**:
   * Tasks **10.1P to 10.3D** introduced **SQL databases (sqlite3)** for server-side data storage and retrieval, a critical skill for building full-stack web applications that involve persistent data (e.g., login systems and user-generated content).

Overall, by completing these tasks, I have learned to use key web technologies, including HTML, CSS, JavaScript, server-side programming (Node.js), and database integration, which are essential for building modern, scalable web applications. This shows a comprehensive understanding of how web technologies are applied in real-world scenarios.

The objective "Client-side Application Design and Development" emphasizes designing and developing components for web applications using current tools, languages, and frameworks. The following is a summary explaining how this objective was met through various tasks.

**Building Foundational Skills with HTML, CSS, and Layouts**

* **Tasks 1.xP - 3.xP/D**: These tasks built the foundational skills necessary for structuring and styling web pages. Creating an HTML tree, adding images, hyperlinks, and tables, as well as using internal and external CSS, led to a deeper understanding of how web pages are rendered and styled for various devices and screen sizes. This also involved designing pages with elements like a three-column layout, enhancing visual design through background images, and understanding how cascading styles work.
* **Task 4.xD**: Learning about Bootstrap provided a way to ensure responsive design across devices. Incorporating components like image carousels and advanced features like Navbars, tiles, and tooltips enhanced the website's visual appeal and user experience.

**Interactivity Through Forms and JavaScript**

* **Tasks 5.xP/C**: Designing web forms for gathering user feedback was crucial in understanding client-side data handling. Validation techniques using JavaScript ensured the forms' correctness, and responsive design ensured usability across devices. Complex error handling improved the robustness of form submissions.
* **Tasks 6.xP/C/D and 7.xP/C**: JavaScript played a central role in enhancing interactivity. Creating dynamic lists, implementing loops for content generation, and applying conditional logic for various functionalities like table generation and string manipulation improved the user's interactive experience. Tasks also explored event listeners for user-triggered actions, enabling real-time feedback and enhancing user experience.

**Improving User Experience with Personalization and Dynamic Content**

* **Tasks 7.4D - 10.4HD**: These tasks focused on developing and improving a custom web project that applies the skills learned. Using banners, cards, Bootstrap components, and JavaScript, the web pages were made more engaging. This personalization of the page based on user interactions and logged-in status demonstrates dynamic and responsive design.
* **Tasks 9.1P/C and 10.xP/C/D/HD**: Server communication was explored through form submissions and response pages, utilizing POST routes to manage user data effectively. Creating a server-side database to store and query user data, such as login information and form submissions, provided a comprehensive understanding of integrating client-side and server-side functionalities. The database-enabled features also allowed for user-specific data to be auto-filled in forms and personalized content to be shown based on user status.

**Application of Modern Tools and Frameworks**

Overall, each task contributed to building a solid foundation in client-side web development using modern tools like HTML, CSS, JavaScript, and frameworks like Bootstrap. The ability to create responsive, interactive web pages that connect to server-side databases showcases a comprehensive understanding of client-side application design and development. The tasks culminated in a full-fledged web application that efficiently handles user input, dynamically renders content, and provides a rich user experience.

The objective "Option Identification and Application Deployment" involves selecting appropriate options for deploying web applications to production environments. Below is a summary that captures how the tasks met this objective.

**Understanding Deployment and Styling Options**

* **Task 2.3P**: This task was pivotal in deploying a web page to an external server, demonstrating a real-world scenario of publishing a web application. It laid the foundation for understanding deployment in practical terms.
* **Tasks 3.xP/D**: These tasks focused on exploring different options for styling and designing web pages, primarily using CSS. By applying internal styles (3.1P) and external styles with fonts (3.2P), different methods of enhancing the visual design were tested. Additionally, using background images, pseudo-classes, and shadows (3.3P) provided a means to deploy more complex design elements. Creating a three-column layout (3.4D) facilitated experimenting with structured page designs, preparing for more advanced deployment options.
* **Task 4.1P**: The incorporation of Bootstrap provided another approach to designing and deploying web applications. It allowed for scalable and consistent designs across devices, facilitating the deployment of more user-friendly applications.

**Enhancing and Improving Web Page Functionality**

* **Task 7.4D**: Improvements to the web page were applied using both CSS and Bootstrap, allowing for a customizable and responsive design. This task synthesized previous learnings to create a more polished, production-ready deployment.
* **Tasks 10.3D and 10.4HD**: These tasks involved the deployment of a functional database to a web application, enhancing back-end functionality. The separation of the database instantiation into createDB.js and the use of session variables through express-session created a dynamic user experience. Task 10.4HD specifically highlighted personalization on a deployed website, offering features such as personalized views, content based on the user's profile, and session data management, demonstrating full-scale deployment in a production-like environment.

**Server-Side Database and Routing**

* **Task 10.xP/D (Database and Routing with SQLite)**: These tasks showed an understanding of server-side integration and database handling, which are critical when deploying full-stack web applications. Separating the database creation from the main application (as done with createDB.js) demonstrated an awareness of maintaining scalable and modular code for deployment.

**Tasks Related to HTML and Web Page Structure**

* **Task 1.1P (Structure a HTML Tree)**: Understanding HTML structure was crucial for web development and deploying clean, organized code that translates well into user interfaces. A clear structure is foundational for deploying web applications that are easy to maintain and extend.
* **Task 1.2P (Serve Your Own Web Page)**: Deploying a locally served web page represents an initial, practical understanding of how to push a web application from development to a functioning state in a local environment.

**Deploying with Focus on Design and Usability**

The tasks as a whole covered a wide range of deployment options, focusing on styling, responsive design, and dynamic user experiences. From deploying structural HTML with validation to enhancing the design using both internal and external CSS, Bootstrap elements, and JavaScript functionalities, all these elements ensured that the applications were ready for deployment to real-world environments. Implementing a database and session management was crucial to deploying a personalized and functional web application, meeting all requirements of the "Option Identification and Application Deployment" objective.

To summarize how the tasks addressed the objective of "Web Service Design and Implementation," the completed activities demonstrated designing and implementing RESTful web services using a variety of modern tools, languages, and frameworks. Below is a breakdown of how each task aligns with this objective:

**HTML & Web Page Deployment (1.2P, 2.3P)**

Tasks involving the structuring and serving of web pages provided the foundation for understanding how RESTful services interact with client requests. Publishing a structured and validated web page created a consistent and navigable structure across multiple pages. These steps were essential for building a seamless interface where RESTful services could later be incorporated.

**Form Creation & User Input Handling (5.1P, 5.2C, 5.3D)**

Creating form pages was central to learning how to gather user input effectively. By using Bootstrap components and improving upon the structure based on requirements, these tasks laid the groundwork for how web services can be designed to accept, validate, and process data. The implementation of these forms was a practical application of REST principles where client-side input was routed to the server for processing.

**Enhancing Client-Side Experience with JavaScript & Bootstrap (7.4D)**

By applying advanced Bootstrap and JavaScript components, custom banners, cards, and interactive elements enhanced the user experience. This contributed to creating a user-friendly interface for RESTful services, making the client-server interaction more intuitive and dynamic.

**Node.js & Server-Side Application Development (8.1P, 8.2C, 8.3D)**

Creating a Node.js project and writing a server application expanded the knowledge on server-side processing. Implementing routing rules and dynamic content with Node.js allowed for multi-route applications, enabling efficient processing of client requests. This was essential in understanding how RESTful APIs work and how different HTTP methods (GET, POST) can be used to handle data exchanges between the client and server.

**Handling Client Requests with POST and Response Pages (9.1P, 9.2C)**

These tasks focused on the server's role in receiving and processing data submitted through forms (POST requests). By implementing response pages, the tasks demonstrated how the server could send feedback based on the client input, and how error handling could be integrated into RESTful service design for better user experience.

**Server-Side Database Integration & Search (10.1P, 10.2C)**

Storing data on a server-side database (sqlite3) and implementing search functionality showed how RESTful services can interact with databases to provide a full range of CRUD (Create, Read, Update, Delete) operations. These skills were crucial for understanding how RESTful APIs connect to and interact with data layers.

**Comprehensive RESTful Service Design with Database Access (10.3D, 10.4HD)**

In these tasks, a fully functional RESTful service was developed, including login and contact form features that interacted with a database. The service personalized user experiences based on their session data and provided pre-filled forms and saved content, demonstrating how to design RESTful services that provide a cohesive user experience. The completion of these tasks represents a full implementation of RESTful principles by ensuring that data retrieval, processing, and display are performed efficiently and securely.

Overall, the tasks combined the development of the front-end, interaction with a back-end server, and integration with a database to build fully functional RESTful web services. Key technologies and frameworks like Node.js, Express, SQLite, and Bootstrap were used to meet contemporary standards in web service design and application deployment.

|  |  |  |  |
| --- | --- | --- | --- |
| Explain Web Technologies: Explain the relevance, purpose, and operation of technologies which underpin web applications. | | | |
| Task | **Name** | **Application (1..5)** | **Comments** |
| 1.1P | Structure a HTML tree | 2 | Learning the structure of a Web Page and the main structural tags. |
| 1.2P | Serve your own web page | 5 | First time to deploy a web page on a local server. |
| 2.3P | Create, validate and publish a structural web page | 5 | Learning how to link and navigate across web pages and publishing to a web server. |
| 4.3D |  |  | Instantiating bootstrap objects |
| 6.1P | Jumpstart JavaScript | 3 | Building our understanding of the DOM (Document object model) to dynamically create lists to be displayed on the website |
| 6.2C | JavaScript Loop | 3 | Building our understanding for handling loops |
| 6.3D | Condition and function | 3 | Performing conditional loops on different DOM objects to create a table. Also bringing in additional math functions. |
| 7.1P | String Objects | 3 | Managing and manipulating string objects, as well as bringing in error handling/data validation |
| 7.2P | Event Handler and dynamic page | 3 | Time types with dynamic HTML via functions. Also event listener of the button element. |
| 7.3C | Form validation | 3 | Form validation via JavaScript |
| 8.1P | Creating a Node Project | 4 | Implementing routing rules (GET) for error handling and new HTML page. |
| 8.2C | Writing a Server Application | 4 | Expanding on routing rules for dynamic web pages |
| 8.3D | Writing a Multi Route Template Server Application | 4 | Routing rules and handling (joining) JSON data, using Encapsulated JavaScript (ejs) files. |
| 9.1P | Send a Response to the Received Data | 4 | Learning about the POST routing rules from user input |
| 9.2C | Create a Response Page | 4 | Server-side form validation which increases web security/stability |
| 10.1P | Server Database for Website Project | 4 | Storing data on a server side database / table using sqlite3 package and running queries against it |
| 10.2C | Search a Database | 4 | Using POST routing to display a custom list based on search terms from the database table. |
| 10.3D | Database Access to your website | 4 | Incorporating all database technologies to manage login and contact form features on the web page. |

|  |  |  |  |
| --- | --- | --- | --- |
| Client-side application design and development: Design and develop client side components of a web-based application using contemporary tools, languages and frameworks | | | |
| Task | **Name** | **Application (1..5)** | **Comments** |
| 1.1P | Structure a HTML tree | 5 | How simple page would be rendered in the browser. |
| 1.2P | Serve your own web page | 2 | Being able to see how the web page is to be displayed in browser |
| 2.1P | Add images and hyperlinks into a web page | 5 | Gives the user ability to navigate across objects using hyperlinks in text/images. |
| 2.2P | Create tables in your web pages | 5 | Tables is another standard way of displaying data on a web page. |
| 2.3P | Create, validate and publish a structural web page | 5 | Here we learnt to navigate across different parts of a web page and across pages |
| 3.1P | Create a web page with internal styles | 5 | Using cascading style sheets (CSS) to create custom visualization/design of a webpage. |
| 3.2P | Create a web page with external styles and fonts | 5 | Using external CSS definition allows us to have consistent design across multiple pages. |
| 3.3P | CSS for background image pseudo class and shadows | 5 | Further enhancing visual design learning about background images, |
| 3.4D | Three-column layout web page | 5 | This is the first opportunity to be creative with our website, taking the learnings so far to design and create our own 3-column layout website. |
| 4.1P | Bootstrap Content Page | 5 | Focus on ensuring the design of the web page can be viewed by different devices. Here, we begin to learn bootstrap. |
| 4.2C | Bootstrap Image Carousel | 3 | A image carousel, using Bootstrap, is a very visually appealing method to engage the viewer with the site content. |
| 4.3D | Advanced Bootstrap Components | 5 | Applying features such as a Navbar, Tiles, off-canvas window and tooltips make the overall website more appealing and engaging. It helps the user consume and navigate the website. |
| 5.1P | Create a form web page | 5 | Getting user feedback through a form, ensuring validation of the fields and applying different form types. We also ensured that different devices can view the form (desktop/mobile) |
| 5.2C | Create a form web page from requirements | 5 | With more complex error handling, we were able to provide detailed feedback to the user about how to fill in the form correctly. |
| 5.3D | Website project (Part 1 of 3) | 5 | This initial part of our web site detailed the structure, layout and overall design on the web page/s. It directed us to think from a design perspective prior to writing code. |
| 6.1P | Jumpstart JavaScript | 5 | Our first attempt at creating a list via JavaScript opened the door for surfacing dynamic content to the user. |
| 6.2C | JavaScript Loop | 5 | In this task, users are able to create custom tables (cap schedule) based on user input; expanding the interactivity with the site. |
| 6.3D | Condition and function | 5 | Using different methods to display summary data from raw JSON files. These tools provide a dynamic and scalable method to have engaging interactions on our website. |
| 7.1P | String Objects | 5 | Learning about conditional formatting of strings and having pop-up windows for error handling adds additional depth to the web pages. |
| 7.2P | Event Handler and dynamic page | 5 | Event listener’s enable interactive elements for the user upon actions such as clicks, or hovers. |
| 7.3C | Form validation | 5 | Additional flexibility in error handling is learned here through JavaScript. This way we are not constrained by the Bootstrap features. |
| 7.4D | Improve your website (part 2 of 3) | 5 | Applying on the learnings so far on the custom site, demonstrating custom banners, card, bootstrap and JavaScript elements |
| 9.1P | Send a Response to the Received Data | 3 | The POST route enables us to confirm inputted data from the user and create custom elements based on user inputs. |
| 9.2C | Create a Response Page | 3 | Enhanced error handling allows us to to recommend remedial actions for incorrect form submissions |
| 10.1P | Server Database for Website Project | 3 | Writing data to a server side database allows the user to see previously submitted form submissions. |
| 10.2C | Search a Database | 4 | We now are providing the functionality to search records within the database |
| 10.3D | Database Access to your website | 5 | In this task, I implemented a database for both a sign-in and contact form feature. The sign-in data also prefills the contact form with appropriate user information |
| 10.4HD | Complete your website (part 2 of 3) | 5 | personalise the page based on the user logged in. Things such as outstanding contact form submissions, events registered to, saved blogs/articles for future reading, current member status/benefits etc |

|  |  |  |  |
| --- | --- | --- | --- |
| Option identification and application deployment: Identify options for and deploy web applications to suitable production environments. | | | |
| Task | **Name** | **Application (1..5)** | **Comments** |
| 1.1P | Structure a HTML tree | **2** | How to organize the code for deployment |
| 1.2P | Serve your own web page | **2** | Deploying the web page in the local server environment for the first time. |
| 2.3P | Create, validate and publish a structural web page | 5 | Here, we publish the website to the external web server, showing a realistic deployment of a web site. |
| 3.1P | Create a web page with internal styles | 2 | CSS is a way to provide custom design to web pages to web pages |
| 3.2P | Create a web page with external styles and fonts | 2 | CSS is a way to provide scalable design to multiple web pages |
| 3.3P | CSS for background image pseudo class and shadows | 2 | Providing more options for consistent deployment of more complex design elements |
| 3.4D | Three-column layout web page | 2 | Taking the learnings so far to design and create our own 3-column layout website. |
| 4.1P | Bootstrap Content Page | 2 | Bootstrap is another design option to provide consistent, scalable and flexible web applications. |
| 7.4D | Improve your website (part 2 of 3) | 5 | Applying on the learnings so far on the custom site, demonstrating CSS and bootstrap |
| 10.1P | Server Database for Website Project | 3 | Server-side integration and databases were covered here. |
| 10.2C | Search a Database | 4 | Enhancing functionality of the database interaction of the user. |
| 10.3D | Database Access to your website | 5 | In this design, the createDB.js files instantiates the DB and required tables (which is separate from the web page). I also used a new package express-session for session variables. |
| 10.4HD | Complete your website (part 2 of 3) | 5 | Personalise the page based on the user logged in. Things such as outstanding contact form submissions, events registered to, saved blogs/articles for future reading, current member status/benefits etc |

|  |  |  |  |
| --- | --- | --- | --- |
| Web service design and implementation: Design and implement RESTful web services using contemporary tools, language and frameworks. | | | |
| Task | **Name** | **Application (1..5)** | **Comments** |
| 1.2P | Serve your own web page | 2 | In this task we needed to publish our webpage, ensuring the web page displaying correctly and, in the format, intended. |
| 2.3P | Create, validate and publish a structural web page | 5 | The web site design (structure) was implemented in a consistent way across the four pages, creating a seamless navigation through the information. |
| 5.1P | Create a form web page | 5 | Using bootstrap functions for forms |
| 5.2C | Create a form web page from requirements | 5 | Expanding our capabilities of creating forms. |
| 5.3D | Website project (Part 1 of 3) | 5 | This initial part of our web site detailed the structure, layout and overall design on the web page/s. It directed us to think from a how the client-side functionality will work. |
| 7.4D | Improve your website (part 2 of 3) | 5 | Applying on the learnings so far on the custom site, demonstrating custom banners, card, bootstrap and JavaScript elements |
| 8.1P | Creating a Node Project | 4 | Dynamically loading content on a page. |
| 8.2C | Writing a Server Application | 4 | These features (Technologies) enable a whole new level of user engagement and transparency. |
| 8.3D | Writing a Multi Route Template Server Application | 4 | This is our first exposure to how online transactions could be processed on the web page. |
| 9.1P | Send a Response to the Received Data | 3 | The POST route enables us to confirm inputted data from the user and create custom elements based on user inputs. |
| 9.2C | Create a Response Page | 3 | Enhanced error handling allows us to to recommend remedial actions for incorrect form submissions |
| 10.1P | Server Database for Website Project | 3 | We are incorporating both Client side engagement with server side databases to enhance the overall features of the website. |
| 10.2C | Search a Database | 4 | This feature incorporates the database, SQL query function and a dynamically rendered page (search results). |
| 10.3D | Database Access to your website | 5 | With the use of the technologies, the user is able to login, submit contact forms (with pre-filled information). This sets the stage for a more personalized experience. |
| 10.4HD | Complete your website (part 2 of 3) | 5 | Personalise the page based on the user logged in. Things such as outstanding contact form submissions, events registered to, saved blogs/articles for future reading, current member status/benefits etc |

To provide a cohesive document covering how all tasks align with the learning objectives, we will merge the summaries for the four key objectives: **Explaining Web Technologies**, **Client-Side Application Design and Development**, **Option Identification and Application Deployment**, and **Web Service Design and Implementation**.

**1. Explaining Web Technologies**

The objective of understanding web technologies was addressed through a range of tasks that cover the structure, behavior, and operation of key components in web applications.

* **HTML and Web Structure**: Early tasks (1.1P and 1.2P) focused on the core structure of web pages, employing HTML elements to build and deploy a static web page. This included understanding foundational elements such as headers, footers, and the overall layout, establishing how content is organized and delivered.
* **Creating and Validating Complex Pages**: Later, in 2.3P, the development and validation of multi-page websites provided insights into website navigation and how users move across different sections and pages.
* **Using Frameworks and Libraries**: Bootstrap was introduced as a CSS framework (4.3D), demonstrating how pre-built components can be utilized to rapidly build responsive and styled web interfaces. This task also highlighted the importance of consistency in design across devices.
* **Dynamic Content and DOM Manipulation**: JavaScript-based tasks (6.xP, 7.xP) were crucial in manipulating the DOM to create dynamic content and add interactivity, such as form validation, event handling, and updating HTML elements in real-time. This enhanced understanding of how user interactions are handled on the client side.
* **Server-Side Technologies**: Tasks in the 8.x series introduced server-side technologies using Node.js, which underscored the separation of frontend and backend operations. These tasks focused on implementing routing, rendering dynamic web pages, and understanding server-client interactions through ejs templating.
* **Working with Databases**: Finally, in tasks 10.1P to 10.3D, integration with SQLite databases was learned to store, retrieve, and manipulate data on the server side. The inclusion of server-side data storage expanded the functionality to include persistent data such as login systems and user content management.

In summary, through this sequence of tasks, there was a comprehensive understanding and application of technologies ranging from HTML and CSS to server-side development with Node.js and database handling.

**2. Client-Side Application Design and Development**

For the objective focusing on client-side application design and development, tasks covered foundational elements of creating interactive, visually appealing, and responsive web pages.

* **Foundational Skills with HTML, CSS, and Layouts**: Tasks 1.xP to 3.xP/D laid the groundwork for structuring content on a web page. By creating an HTML tree, adding multimedia elements, and applying both internal and external styles, an understanding was built around how to design web pages suitable for different devices and browsers. The design and structure were enhanced by implementing a three-column layout and using background images.
* **Responsive Design with Frameworks**: Learning Bootstrap (tasks 4.xD) offered a way to develop consistent, responsive designs quickly, incorporating interactive elements like carousels, navigation bars, and cards to improve user experience across devices.
* **Interactivity Through JavaScript**: JavaScript was heavily utilized for client-side interactivity in tasks 5.xP/C and 6.xP/C/D, covering the creation of dynamic elements, loops, conditional rendering, and event handlers to facilitate real-time user feedback. Form validation, both client-side and server-side, ensured robust user input handling.
* **Personalization and Dynamic Content**: Advanced tasks (7.4D to 10.4HD) built on personalization techniques by using JavaScript and server-side data. These features allowed dynamic rendering of content based on user status and interactions, such as auto-filling forms and displaying user-specific information.

In sum, through these tasks, modern tools like Bootstrap and JavaScript were applied to create highly interactive, responsive web applications that integrate seamlessly with client-side and server-side components.

**3. Option Identification and Application Deployment**

This objective was met through tasks that dealt with identifying and applying the most appropriate methods to deploy web applications effectively.

* **Deployment and Styling**: Deploying web pages to an external server (task 2.3P) was one of the critical practical applications in understanding how to move a project into production. By using both internal (3.1P) and external styles (3.2P), options for enhancing design scalability were explored. Using frameworks like Bootstrap provided responsive, user-friendly options for web design and usability.
* **Improving Functionality with CSS & Bootstrap**: Enhancements to the web application were made through advanced Bootstrap components (7.4D) and CSS features, showcasing the practical application of consistent design across the web pages and flexibility in modifying the web's appearance.
* **Server-Side Database and Session Management**: Database tasks (10.3D, 10.4HD) demonstrated how to implement server-side data storage and retrieval, session management for personalized user interactions, and other full-stack deployment considerations. The code was modularized for scalability, demonstrating best practices for deploying a production-ready application.

These tasks highlighted various deployment options, focusing on delivering an optimal user experience with proper data handling and personalization.

**4. Web Service Design and Implementation**

This objective was met by designing and implementing RESTful web services that facilitated smooth interactions between the client and server.

* **HTML & Web Page Deployment**: Structuring and serving web pages (tasks 1.2P, 2.3P) formed the basis for understanding how client requests are handled and how a RESTful service could later be built upon this structure.
* **Form Creation & Handling User Input**: The creation and validation of forms (tasks 5.1P, 5.2C, 5.3D) using Bootstrap laid the foundation for RESTful service design by showcasing how user input can be managed, validated, and processed through client-server communication.
* **Enhanced User Interface**: Using advanced JavaScript and Bootstrap (7.4D), the client-side interface was designed to be intuitive and engaging. These steps ensured a smooth interaction for the end-user when accessing the RESTful services provided by the application.
* **Node.js & Server-Side Routing**: Creating server-side routing and dynamic content rendering (tasks 8.x series) was fundamental in understanding RESTful APIs and their role in managing data exchanges between the client and server through GET and POST requests.
* **Database Integration & Search Capabilities**: Tasks involving database storage (10.1P, 10.2C) demonstrated the complete range of CRUD operations. Server-side logic, session handling, and querying through RESTful services showcased comprehensive integration between front-end and back-end technologies.
* **Comprehensive RESTful Service Design**: In 10.3D and 10.4HD, a complete RESTful service was developed, incorporating login, contact forms, and personalized user experiences, effectively linking all aspects of the client-server interactions and database management.

Together, these tasks demonstrated the full cycle of designing, implementing, and deploying RESTful services using contemporary tools like Node.js, Express, SQLite, and ejs templates.

# Reflections

## The most important things I learnt:

*Think about what you have learnt in this unit and reflect on what you think were key learning points, tasks, activities, etc. Did you learn what you wanted/expected to learn?*

## I feel I learnt these topics, concepts, and/or tools really well:

What things are you really confident about now?

## I found the following topics particularly challenging:

What was the most challenging part of the unit? Have you mastered those ideas, concepts, or skills now? What did you learn about yourself in how you dealt with these challenges?

## I found the following topics particularly interesting:

What was the most interesting or valuable thing you learnt from this unit? This could be related to the unit concepts, or general things you learnt about yourself.

## I still need to work on the following areas:

University is about developing lifelong learning skills. Given what you have achieved already, what is the next step for you? How will you build upon what you learnt in this unit? This could be related to the unit concepts and skills, or to personal traits you identified as needing further development.

## The things that helped me most were:

What were the most helpful/useful resources? How did they assist you with your learning?

## My progress in this unit was …:

Include a screenshot of your **progress graph** from **OnTrack**, and comment on what happened from your perspective… what does the graph say about how you approached the unit?

## If I did this unit again, I would do the following things differently:

Looking back, what is it that you think you could have done differently to help you achieve the most you could in this unit (both in terms of the unit concepts and skills, and in terms of personal growth). How will you approach learning in the future?

## Other…:

Adjust this heading to add any other reflections you think help you demonstrate what you got out of this unit, and how it has or will help shape you as an IT Professional.