**Assessment type (þ):**

Questioning (Oral/Written)

Practical Demonstration

3rd Party Report

Other – Project/Portfolio (*please specify)*

**Assessment Resources:**

|  |
| --- |
| You do not need specific resources to answer the questions in this assessment. You may use the learning resources on Blackboard. |

**Assessment Instructions:**

|  |
| --- |
| Answer all questions to the best of your ability. Think about your spelling and grammar. If a question has a minimum or maximum word count, take that into account.  Answer all questions in your own words. If you use external resources, please provide references.  If provided, you should use the provided template to answer your questions. If you have any questions about the questions, please talk to your lecturer.  Do NOT zip the Word document before uploading it. |

**Assessment Instrument:**

Question 0 – Example question (10-20 words)

This is an example question. You should answer the question in the empty box below.

Enter your answer in this box.

### Question 1 – Web design programming (30-80 words)

In your own words, describe web design programming using HTML, CSS, and JavaScript. Clearly explain what each of the three languages contribute to a web design.

Web design programming commonly involves 3 basic languages: HTML: the framework and basic structure of the website, CSS: a common style sheet language used to design the visual elements of the page, and JavaScript: a common scripting language used to add more complex interactive elements to a website.

### Question 2 – Client-side programming (30-80 words)

In your own words, explain the meaning of ‘client-side programming’. Describe its purpose and the type of architecture that may support this paradigm

Client-side, or front-end programming refers to the interface a visitor to the site, or a user of the program would see. Any code running on the client side is using the visitor’s (or user’s) own hardware to run. In web development, frameworks like React.js enable more processes to be run on the client-side, meaning web pages can download the structure of the webpage once and only need to download new posts.

### Question 3 – MVC Architecture (20-60 words)

In your own words, describe the concept of splitting your application into a model, view and controller. Briefly describe if and how it assisted you in completing the final project. If it did not assist, explain why.

By dividing an app into model, view, and controller, the code can become easier to read and test, and the distinction between UI components and data structure components becomes more visible. In the final assessment, this assisted me by allowing me to test individual components before I completed the full application.

### Question 4 – Object-oriented programming (20-60 words)

In your own words, describe the importance of object-oriented design in relation to MVC.   
Provide at least one example where you had to apply an object-oriented concept (e.g., subclassing, polymorphism) to achieve the desired outcome.

Object-oriented design is helpful in encapsulating sections of code to be reused, and MVC is a design pattern for separating the UI from the data. Data and views are often stored as separate classes. In the final assessment I used subclassing to create a base view with a header and styles, and all other views inherited from the base view.

### Question 5 – Templating (1) (20-60 words)

What is the importance of templating in the context of MVC? Briefly explain one templating language or framework and how you used it in the project.

By using templates for the views, you can create pages dynamically by filling in parts of the template with information from the model. In the project I used Django, a Python framework, to create templates of a few views and had them dynamically update depending on the user’s input.

### Question 6 – MVC frameworks (20-60 words)

Name three alternative MVC frameworks that could have been suitable for your project and name a common characteristic that explains why they are suitable.

Ruby on Rails is a Ruby-based framework with a strict approach to naming conventions.

Laravel is a PHP framework which prioritises elegant, developer-friendly syntax.

Express.js is a JavaScript framework focussed on lightweight and minimalist syntax.

The common feature of these frameworks is that they all streamline web development by providing tools to simplify common tasks.

### Question 7 – Large-scale applications (20-60 words)

In your own words, describe what large-scale application development is and how it relates to MVC.

Large-scale application development involves creating complex software systems which handle significant amounts of data or users. MVC is useful for this because by separating the concerns into model, view and controller, you enhance the maintainability and scalability of the application, and can also smooth collaboration between teams who can work on each concern separately.

### Question 8 – Complex data structures (30-80 words)

In your own words, describe procedures to follow for programming with complex data structures. Provide at least one example of a programming paradigm suitable for this context.

When programming with complex data structures, start by understanding the problem domain thoroughly. Design a clear structure for data representation and choose an appropriate programming paradigm. Break down the problem into smaller, manageable tasks and implement algorithms efficiently. Ensure documentation and testing are comprehensive to handle potential complexities. Object-oriented programming (OOP) is a paradigm that is often suitable, allowing encapsulation of data and behaviour.

### Question 9 – Third-party libraries (20-50 words)

In your own words, explain the procedures you would use for using a third-party library for creating standard programming features.

First, research and select a reputable library meeting the project's needs. Study the library's documentation to understand its functionalities. Integrate it into the project using a package manager or similar. Test extensively to ensure compatibility, reliability, and desired functionalities.

### Question 10 – HTTP Protocol (20-50 words)

In your own words, describe each VERB in the HTTP protocol.

GET: Requests data from a specified resource.

POST: Sends data to a server to create or update a resource.

PUT: Updates a resource or creates it if it doesn't exist.

DELETE: Removes the specified resource.

### Question 11 – Debugging: functions and features (20-50 words)

In your own words, identify a framework-based debugging and testing feature that you used. Explain why MVC creates challenges for testing and debugging.

I used Django's TestCase class for unit testing as it provides tools to simulate requests and assert expected behaviours. In MVC, separation of concerns can lead to intricate interactions between models, views, and controllers, making it challenging to trace and isolate bugs during testing.

### Question 12 – MVC Architecture Benefits (20-50 words)

In your own words, explain how MVC contributes to the following: scalability, maintainability, and reusability

Scalability: MVC separates concerns, allowing individual components to scale independently, enhancing the system's scalability.

Maintainability: Clear separation enables easier code maintenance as changes in one component don't heavily impact others, ensuring better maintainability.

Reusability: Modular components in MVC facilitate code reuse across various parts of the application, promoting reusability and reducing redundancy.

### Question 13 – Features, structures, logic, and modes of interaction in MVC

In your own words, write 1 – 2 sentences describing how each of the following features are used in MVC and give an example of how you used these in your project. If you did not use these in your project explain how they *could* be used.

**HTTP Request/Response and redirects**:

Requests trigger controller actions which process data, generate a response, and can redirect to other URLs if needed. Used for navigating the website.

**HTTP request handlers, routes and parameters**

Handlers in controllers manage incoming requests via routes, extracting parameters from URLs or request bodies for processing. Used for navigating the website.

**Query strings and key/value pairs more generally:**

Query strings in URLs or form submissions use key/value pairs to send data to the server (model) for processing. Used in creating data from user input.

**Model binding**:

Automatically maps incoming data from requests to data models for easy manipulation and interaction. Used by Django in its default login page, as well as for the custom classes.

**Convention over configuration:**

MVC frameworks prioritize defaults and naming conventions, reducing explicit configurations and simplifying development. Used when setting up the website and database.

**HTML language, templates and dynamic rendering**:  
  
Templates, written in HTML with embedded dynamic content, generate dynamic views rendered based on data from controllers. Used to model every web page in the site so headers would be consistent.

**View models and data models**:

View models customize data presentation for views, while data models represent application data and its logic within the system. Used in the setup of the website.

# **End of the questions**

# **Thank you!**