| **Written Questions** | | **S / US** |
| --- | --- | --- |
| **Q1** | Did you find the requirements outlined for the application to be built and consult with your client (assessor) to confirm that the requirements are correct.  (check correct response)  **Yes  No** |  |
| **Q2** | Did you confirm that there are no applicable legislative or organisation standards or procedures that need to be applied to the application being built in this assessment.  (check correct response)  **Yes  No** |  |
| **Q3** | Determine the best web technology / protocol with GET, POST, PUT, DELETE and other related functionality with which to build your web based Restful API.  Which technology / protocol did you decide on?  HTTPS methods (GET, POST, PUT, DELETE), in my opinion seem to be the best option to develop with and were used for my challenge, they are easy to read and simple in comparison to other methods. |  |
| **Q4** | Provide 2 advantages and 2 limitations of using the HTTP protocol when developing web applications.  **Advantages**   * When an application requires more features, HTTP can download add-ons or plugins and show the necessary information. * Each file via HTTP is downloaded from a separate connection before being terminated. As a result, just one element per webpage may be sent.   **Disadvantages**   * Since HTTP does not employ any encryption techniques, it is possible for someone to change the content. * Any hacker who can intercept the request will be able to see all of the content on the web page. |  |
| **Q5** | Briefly describe the process you used to review and debug your API code as outlined in item 9 of the Task Summary above.  I used a range of tactics, I created a test case and expected output for each end point as a reference before development was begun. If output didn’t reflect initial plan, endpoints were remodelled to resemble initial plan. Using web console to check for error messages helped better track down bugs, console logging calls to see what exactly is getting sent out helps clarify that the correct information is getting sent out and displayed. |  |
| **Q6** | Did you Seek feedback on your work from your assessor.  The feedback demonstration will primarily be done in postman, however at least one API call must be demonstrated in each of 2 browsers and on at least 2 machines, (discuss with your assessor to clarify) Make updates based on the feedback your assessor provides.  (check correct response)  **Yes  No** |  |
| **Q7** | Briefly outline 3 principles of web analysis & design   * **Visual hierarchy**   Which orders the information on a web page from most critical to least important the further scrolled down, its used to get the “main message” delivered as soon as a user view this page.   * **User Friendly**   A user shouldn’t have to put much effort into working your applications, it should be easy and obvious how your app functions, it should be self-explanatory.   * **Minimalistic**   Keeping a design simplistic and easy to read, many applications require forms and users to input data, forms should be direct an clean, all unnecessary components should be hidden in times of action to draw users to the action required. |  |
| **Q8** | Briefly discuss the features and functionalities of 2 different web applications |  |
| **Q9** | Briefly discuss programming control structures and applicable languages  They are instructions that allow a software to "take decisions" and choose one course of action over another. Since it may split up, duplicate code, or skip portions during operation, a programme is often not restricted to a linear series of instructions. |  |
| **Q10** | Briefly discuss 2 code debugging techniques  programme development that is bottom-up and incremental. The software should be developed progressively, with frequent testing performed as each new piece of code is added. If there is a mistake, it most often appears in the last line of code you typed. The last bit of code is minimal with incremental programming, thus the hunt for problems is constrained to discrete areas.  Backtracking, one technique is to start at the problem's origin and work your way back through the code to see how it may have happened. |  |
| **Q11** | Briefly discuss the web application development process |  |
| **Q12** | Briefly discuss the legislative and organisational requirements applicable to creating web applications |  |
| **Q13** | Briefly Discuss web programming concepts including:  **authentication and web security**  Web security is a very important topic in programming because user data has become increasingly valuable over time. Authentication is a typical example of a solution that has been tested and is being used on the web. One well-known illustration is Auth-O, which adds an extra degree of protection by asking users to demonstrate that they are the legal proprietors of their image.  **hypertext transfer protocol (HTTP)**  The Hypertext Transfer Protocol is used to load internet pages through hypertext links (HTTP). Running on top of other network protocol stack layers, HTTP is an application layer protocol used to transmit data between networked devices.  **session management**  The technique of handling several requests from a single user or entity to a web-based application or service safely is known as session management.  **stateless programming**  Stateless programmes are those that don't save any session-related data, such as configuration settings. |  |
| **Q14** | Briefly describe the different roles of the following web front end and backend languages, including:  **hypertext markup language (HTML)**  As the fundamental components of a website, HTML specifies which parts will appear and in what sequence. This allows HTML to efficiently define how a browser communicates web page elements to an end user.  **cascading style sheets (CSS)**  CSS styling of the web page  **JavaScript**  JavaScript adds dynamic functionality to the website and controls how it behaves. |  |
| **Q15** | Briefly describe the work performed in web application with reference to its management of statelessness |  |
| **Q16** | Briefly describe organisational procedures applicable to creating web applications. |  |
| **Q17** | Which IDE (Integrated Development Environment) did you use to develop your API?  List 2 things that were good about the environment and 2 things that were bad about it.  I was able to work with many various languages and environments thanks to Visual Studio Code, which also offered capabilities for version control with a direct link to GitHub.  Installing various packages can become tedious and repetitive as well as extensions not working unexpectedly. |  |
| **Q18** | Briefly describe benefits and functions of noSQL database and schema free data persistence, as well as traditional relational data models  NoSQL databases are a relatively recent invention, however they were developed to address the shortcomings of relational databases. NoSQL databases process large amounts of data more quickly, make field and schema changes more easily, and need less maintenance overall. |  |
| **Q19** | Briefly describe methods and different features and functions between scaling out and scaling up (horizontal and vertical)  Horizontal scaling is the process of increasing the number of instances of a machine, prioritising quantity above quality; this does not increase the number of specifications currently in place but rather adds more. However, vertical scaling prioritises processing power over the total number of computers. This implies that although the system it operates on increases in specification, the current code can stay the same. |  |
| **Q20** | Briefly describe language used in required programming language for noSQL applications  often need to connect to the database using tools like Java/TypeScript and add-ons for MongoDB like Mongoose. A separate querying language for noSQL languages is also available. |  |
| **Q21** | Briefly describe partitioning in a noSQL environment and its related terms  NoSQL entities are given an ID or key that serves as a partition-specific indicator. A distinct storage node and its appropriate querying engine are totally dedicated to serving each partition. |  |
| **Q22** | Briefly describe functions and features for time-to-live (TTL) requirements  A packet of data has a predetermined period of time to live before it is discarded from a network. An internet protocol called TTL informs a network router when a packet has been present in a network for too long. As a result, packages can't traverse a network indefinitely. TTL, on the other hand, may control data caching and improve speed. |  |
| **Q23** | Briefly describe authorisation and authentications procedures and levels of responsibility according to client access requirements  When a server has to look into an unknown client's access, authentication is used. Prior to granting access to resources, the server verifies the client's rights and/or degree of responsibility through the authorisation process. Similarly during client access, authentication is utilised when the client has to check out the server seeking access. |  |
| **Q24** | Briefly describe distribution of data storage across partitions  The concept of distributing data across partitions and allocating storage for a table comes from distributing workload around the database's tables. As a strategy, data distribution on its own can result in key skew; conversely, partitioning alone works in direct conjugation with data distribution, allowing for the advantages of both while eliminating the disadvantages of neither. |  |
| **Q25** | Briefly describe debugging and testing methodologies and techniques  The technique of recording data via print or console is frequently encouraged and popular. |  |
| **Q26** | Briefly describe functions and features of sort keys in noSQL storage  Relevant data is gathered together using a sort key. A sort key essentially acts as the 'key' defining a set of data, enabling the data to be changed in relation to it. |  |
| **Q27** | Briefly describe features of transport encryptions, authentication and authorisation  is a protocol for host-to-host communication that determines who is communicating by analysing the ClientHello message. essentially letting the server pick a version and cypher that works |  |
| **Q28** | Briefly describe different noSQL data store formats, including:  o key value  o document base  o column based  o graph based  key-value, wide-column, and graph databases are the four main types of noSQL databases. Document databases hold information in specific files, such as JSON or XML, where each file contains fields and values of various sorts. However, objects are each stored in key-value databases with a distinct key. Wide-column databases also contain information in the form of rows, tables, and dynamic columns. Furthermore, nodes and/or edges are the units of data storage in graph databases. The majority of the information is in the first, whereas relational information is in the second. |  |
| **Q29** | Briefly describe different noSQL data types, including:  o numeric  o string  o boolean  o complex  o date time. |  |
| **Q30** | Briefly describe language used in object-oriented (OO) programming  Like its name suggests, object-oriented programming is geared towards objects or classes. In order to later generate more specialised and/or particular data instances, objects and/or classes act as blueprints. Classes are the main categories to which pertinent data will often be attributed. |  |
| **Q31** | Briefly describe HTTP protocol  Resources are retrieved via the HTTP protocol. essentially creating a new document out of the several sub-documents retrieved. Requests are messages sent from the client, while replies are messages delivered from the server in response. |  |
| **Q32** | Briefly describe functions and features of debugging and testing tools  Tools for testing and debugging were developed and finally included to browsers. Network packet visibility, which enables users to view what data is going in and out of the programme, is one of the most typical capabilities.The console, which logs all actions taken during runtime, is another feature. |  |
| **Q33** | Briefly describe principles of model view controller design pattern, conventions and architecture, including:  o scalability  o maintainability  o reusability  The Model View Controller Pattern, often known as MVC, is an architectural design pattern that addresses how the application's front end, or user interface, and backend, which houses all of the application's data, come together. Modern applications use the MVC pattern because it makes it easy to handle the front end and back end as two easily separated components, allowing for scalability while maintaining maintainability. Additionally, this approach enables the recycling and/or reusing of components across an application. |  |
| **Q34** | Briefly describe features, structures, logic and modes of interactions between models, controllers and views, including:  o HTTP Request/Response and redirects  o HTTP request handlers, routes and parameters  o Query strings and key/vale pairs  o model binding  o convention over configuration  o HTML language, templates and dynamic rendering  o view models and data models. |  |
| **Q35** | Briefly describe principles of database management systems applicable to deploying applications to production environments  When using a DBMS, there are numerous factors to take into account, but when it comes to deploying to production, there are several rules that are obviously crucial. One is backup and restoration. A DBMS has to contain procedures to address possible problems with data loss since uptime is becoming more and more crucial to company operations. Similar to this, a DBMS must uphold the data security concept as production settings house actual users whose data is crucial and must be handled as such. |  |
| **Q36** | Briefly describe software development life cycle (SDLC) that may be used in deploying applications to production environments  Testing (uat) and the actual deployment stage of the software development life cycle are aspects of the SDLC that are relevant to deployment to a production environment. The former would be significant after the deployment, whereas the latter would relate to the deployment's actual activity.  inside the SDLC's deployment phase. UAT testing would be finished after a successful deployment, and after that, fixes would be put into place. |  |
| **Q37** | Briefly describe programming language used to create deployment applications  A popular language for writing deployment scripts, YAML is a Mark Up language like HTML that places a focus on readability, clear syntax, and the absence of executable instructions. Instead, programmes like Azure use it as a buffer language where the program's instructions are taken directly from the file. |  |
| **Q38** | Briefly describe Information and Communications Technology (ICT) hardware, software, security protocols and standards and organisational policies relevant to deployment of applications.  When implementing an application in the ICT sector, there are several factors to take into account. Hardware considerations include database and server size, as well as processing speed, if the servers are physically present on site. In terms of software, it's crucial to provide a bug-free user experience while keeping secure code that can't be attacked from the outside. Additionally, it's critical to follow organisational and security best practises when it comes to security regulations; doing so will guarantee a minimum level of quality for your product. Additionally, it's crucial to have employees available before and after a significant production deployment in case problems or defects pop up. |  |
| **Q39** | Briefly describe design and build an advance UI design.  o In the course of the above the candidate must:  o determine an organisation’s technology, development tools, and UI platform  o apply advanced techniques in order to create a complex user interface (UI). |  |
| **Q40** | Briefly describe UI prototyping techniques and purpose.  The agile model, which regularly iterates and improves the prototype, is a well-liked method for UI prototyping. However, doing UATs with the prototypes and gathering user feedback is a universally accepted procedure. |  |
| **Q41** | Briefly describe data structures applicable to applying intermediate object-oriented language skills  Numbers, booleans, characters, arrays, and structures make up OO's data types. In certain relevant languages, "strings" are used in place of characters. Structures are replaced by objects in object-oriented languages like C++ and Java, where objects serve as data containers. |  |
| **Q42** | Briefly describe object-oriented programming concepts and programming language required to apply intermediate object-oriented language skills  The first of the four concepts of object-oriented programming is inheritance, whereby child classes inherit properties and data from parent classes. Information that is encapsulated in an object but only partially revealed. Additionally, Abstraction restricts the amount of publicly accessible methods by only exposing high level methods. Lastly, polymorphism enables many ways to perform the same function. To use advanced OO abilities, a developer must have a solid grasp of objects, classes, methods, and the concepts. |  |
| **Q43** | Briefly describe process and techniques related to use of a graphical user interface (GUI), to interact with an operator  User patterns, navigation design, and page layout are a few GUI elements that all work together to improve end-user communication and experience. The User Patterns are a collection of guidelines for UX design that get rid of repetition and encourage a shorter browsing experience. |  |
| **Q44** | Briefly describe documenting applications required to apply intermediate object-oriented language skills.  The concept of commenting, which is prevalent in higher level programmers and enables outside users to comprehend what a code-block does, is crucial to object-oriented programming (OO). Comments often describe the operation of an object method or class. The single responsibility principle, which advocates against the usage of generalised classes, also states that a class should only carry out a single function or fulfil a single purpose. |  |
| **Q45** | Briefly describe features and different applications that applies to the HTTP network protocol  The characteristics and use-cases of HTTP network protocols have changed throughout time. A Cookies feature that enables the storage and later reversion of tiny data packets on demand can be implemented using the HTTP network protocol. Furthermore, one well-known use of the HTTP network protocol is in "Cross-Origin Resource Sharing," or CORS, which gives servers a way to offer an overview and indication relevant to origins other than their own and enables external browsers to load resources as a result. |  |
| **Q46** | Briefly describe features and anatomy of REST API HTTP request and response, including HTTP headers and body  When it comes to HTTP method(s) and when they should be used, a REST API adheres to a defined set of criteria. For instance, HTTP GET requests are used to acquire data rather than change or delete it. Additionally, for the same call, the data gathered should remain consistent across time. In addition, if the resource is found, the response must return 200 and provide the pertinent data in xml or json format. When using the http POST method to transmit an object of data to a database, one should get a response code of 201, which stands for "created," along with the location header. |  |
| **Q47** | Briefly describe language used in programming language  As a set of rules and hypotheticals, programming languages can be different. Strings, which refer to text inputs, and ints, which refer to purely numerical inputs accurate to one decimal point, are examples of commonality. Although language is used to connect with the computer when programming, the majority of programming languages also have a compiler that turns code into native machine code. |  |
| **Q48** | Briefly describe HTTP GET, POST, PUT and OPTIONS methods and features of each  While the http POST method is in charge of delivering an object (data) to a certain location or area, which results in the updating of the database, the http GET method is responsible for obtaining data and may thus conduct resource retrieval requests. Furthermore, if the necessary object information is provided, the http PUT technique enables a user to change a specified page. Additionally, http Alternatives describes a target resource's prospective course of action and/or available options. |  |
| **Q49** | Briefly describe CORS.  A server can specify any origins (domains, schemes, or ports) other than its own that a browser should authorise the loading of resources from using CORS. CORS is a method based on HTTP headers. |  |