



Complete, fold and seal
(see note 2 below)

RODGER

LEE JAMES

12 Pages

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. **Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.**

1	Student Number	1809707
2	Title of Examination Paper	Web Services
3	Module Code	ACCA 7005
4	Date and Time of Examination	13/05/2019 9:00 AM
5	Examination Venue	IQ 310
6	Seat Number	G013
7	Number of Answer Books handed in	

[illegible]

(Gweler yr ochr arall am y fersiwn Cymraeg)

1) Question 1a.

When a web address is entered into a browser, a request is made to a DNS server (Domain Name System) to find an IP address which the name is mapped to. The request may pass through multiple DNS servers until a match is found or potentially no result may be found. Once the IP has been found, it is returned back to the user whereby the user browser then sends a request to the IP, in this case, returning Google.com.

A HTTP ~~data~~ GET request is made ~~at this point~~ which returns the Google homepage. HTTP (Hyper-Text Transfer Protocol) is responsible for ~~passing~~ ^{parsing} HTML (Hyper-Text Markup Language).

TCP/IP (Transmission Control Protocol) is also used to establish a session between the user's machine and the Google endpoint. TCP is the language of the Internet and is the most common medium for nodes on a network to communicate. Alternatives such as UDP also exist.

A request is split up into 'packets'. Each packet contains data, the end IP address, and the next MAC address it needs to reach. Data is split into packets so that if a packet is lost, it would be a lot quicker to retransmit a small portion of the data rather than the whole request again.

As mentioned, each packet contains the IP address at the destination, in this case, Google's IP. But these packets will need to traverse a number of routers on the network before reaching its target. This is why at each step, the packets are updated with the

next endpoint's MAC address so that it knows where to go to reach the endpoint. Each router can make an ARP request to ~~find~~ Translate and IP into a MAC and vice - ~~versa~~ versa.

A network is ~~constantly~~ ^{continuously} connected therefore a request may take multiple routers, longer or shorter as necessary, but will always reach the end goal.

A URL (<http://www.google.com>) is split into separate components when receiving a DNS request. The 'root' is a usually invisible dot (.) at the end of the address. The 'top-level' domain is the end portion (.com, .edu, .ac) which indicates usually the type of organisation. The 'second-level' domain is what a company may register, in ~~this~~ ^{this} case, google. A business may also wish to purchase a sub-domain such as "maps.google.com", whereby 'maps' would be the sub-domain.

If the request to ~~send~~ ^{get} google.com was successful, a 200 HTTP response code will be returned along with the HTML of the webpage, which will be displayed in the browser.

b.

^{Uniform}
URL - Universal resource locator, points to a specific IP address on a DNS server. Such as google.com.

URI - Used in web servers as each object may be given a unique URI as an identifier. May be used to pass data such as account number etc.

URN - A URN ~~will~~ ^{will} contain some information about the resource and the object it pertains to.

2) Question 2a.

HTTP verbs follow the CRUD operations (Create, read, update and delete).

POST - The create operation. Usually used for creating new data to send to a server.

GET - Used to retrieve/read data. This ~~from~~ request is used to retrieve data. For example a URL with the ~~user~~ as an ~~data~~ parameter to view a profile.

PUT - Similar to POST but usually used to update existing data. Such as updating a user's email address in the database.

DELETE - Used when removing data such as deleting an account from a database.

8

b.

Idempotence is important in HTTP as it ensures that each request is specifically used for an appropriate function.

Even though an update could be made through a GET request, this is considered bad practice and can lead to data being snipped.

c.

200 - OK. The request was successful and data was returned.

401 - Unauthorized. No access has been granted.

403 - Forbidden. Content may or may not exist, but no access.

404 - Not Found. The page could not be located, may not exist.

500 - Internal Server error. Usually the server is down, vague response in most cases.

10

d.)

401 is a more specific response than 403, however the two are sometimes used interchangeably. 403 may mean that the content does not exist but both indicate that there is no access granted to the user making the request.

3

3) Question 3

a

RESTful web services are ~~beneficial~~ ^{beneficial} in modern web ~~services~~ applications as they provide a good template for ~~developing~~ development and also provide a lot of functionality for developing object oriented services with easy database mapping.

REST stands for Representational State Transfer and has a number of key components which must be followed. All items are treated as objects which makes reading and writing data very easy for object oriented programmers.

Methods / functions can easily be created with a HTTP operation (GET, POST etc.) and ~~usually~~ can describe the attributes necessary to run a resource. This also means that there ~~are~~ resources can make database queries through ORM (Object relational Mapping) whereby objects ~~can be~~ and their attributes can be easily mapped to a database.

RESTful applications split the application into several (usually 4) layers. The presentation layer, persistence layer, service layer and data layer. Each of these components should be independent of operation so that changes made in one area will not affect another. This is a strong posture for development.

Data is passed in order through each layer. A GET resource will pass from the presentation layer to the data layer whereby a query is made. The result is then passed from the data layer, through to the presentation layer to be displayed.

REST also uses the idea of caching to speed up performance, reducing unnecessary requests to

the service. Caching requests is better for the user, especially if the service is busy.

5

~~Q.2~~

b.

The JPA is used to access and communicate with a database as from a web service. A

RESTful web service will develop using an object oriented approach, with each of these objects usually ~~represent~~ representing a table in a database.

By using ORM (Object relational mapping), each object and its attributes are mapped to a corresponding table so that information updated through the service can be translated for storage on the database.

When a HTTP request is made, a function is called whereby information may be retrieved or updated in the database. A GET request may use headers to perform the request.

A LAZY request will retrieve data as per the request to save on memory and not retrieve the entire database at once.

Once retrieved, ~~information~~ the data will be translated to a format in a JSON format. This format (JavaScript Object Notation) is used to read objects and extract the data for a function to consume. Similarly, a function may wish to POST/PUT data, which will need to be formatted as JSON so it can be interpreted by a request and build a URI.

Upon any request, the JPA translates

data to and from the database so that
~~the~~ the service and database are
able to communicate.

Mismatch may occur in these requests
whereby a call may fail and the user
should be notified. For example, too much data
for one object may be attempted resulting in
a failure, or even too little data. This is
referred to as object-relational impedance mismatch.

Marshalling occurs to ensure that only one
resource is allowed to happen at any given time,
to avoid starvation or deadlock and data loss
with the database.

12

4) Question 4a.

Cloud computing platforms are greatly beneficial for hosting web services in the modern world. These services, sometimes referred to as 'serverless' mean that developers need to worry less about certain aspects of server-side development.

Companies such as Amazon provide a vast array of cloud services which greatly benefit developers and businesses.

These platforms mean that companies only pay for whatever amount of servers they use. For example, Netflix needs a lot of servers at peak viewing times, but these servers would be wasted during times of significantly less users. The cloud platforms mean that more servers are allocated as the service gets busier and then deallocated when less so. This is greatly beneficial as it means ~~that~~ ^{the} company will save money in the long-run, and only pay for what they use.

The drawback however means that businesses usually become 'tied' to these platforms as they will have little understanding of the technologies used and be unable to develop their own servers in the future. This could mean that they have no choice in paying for those services, no matter how expensive they become.

b.

When implementing a web service, developers must be careful of any potential exploits in the system. There is a risk of data being stolen when using incorrect

HTTP operations. Developers, especially in frontend, need to alleviate the possibility of SQL injections which could manipulate data ~~in~~ in the database.

Reverse methods must only do what is expected and nothing else. PUT/POST should not return any new data directly without a GET request being made.

2

C.

Automation tools greatly facilitate web services by allowing developers to view data easily and quickly. JMeter allows the user to view web service data which can also greatly help in development too.

Tools such as Swagger are great for debugging web services where it's sometimes may take the developer a considerable amount of time to find an error otherwise.

Automation tools allow for multiple requests to be made to test if a service is responding correctly and if it can exceptionally handle many requests.

JMeter allows a user to send many requests at a desired rate to a service to see how well it responds. This is useful in development as any potential problems with many requests can be spotted early. This also simulates how busy a service may be when it goes public, to ensure it won't fail under heavy stress.

These tools provide many benefits to both users and developers and aid greatly in developing and debugging web services.

5