TM352 TMA01 Sarah Walker B4050530 28/11/2017

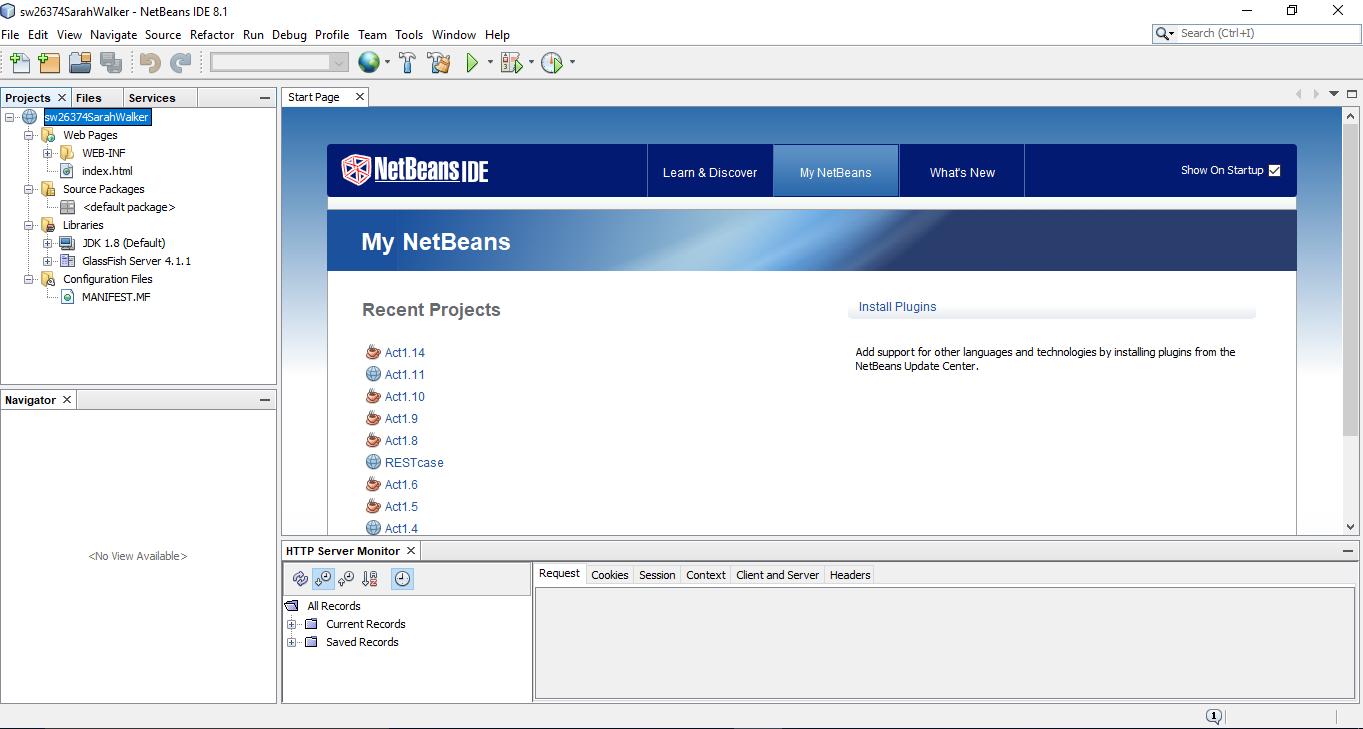
Part 1

A web service is a means by which resources on a server can be requested by a client across a network, such as the internet. This involves a request message being sent from a node in a network, such as a client, software application or browser, to a server. Upon receipt of this request the server provides said resource, and replies with a response message. To allow this communication across a network the messages must be sent in a particular format, adhering to a particular protocol or set of standards, so that messages can be read and understood. Adhering to these standards also ensures that web services are platform independent, and can be used by any number of clients. Web services can be used to provide requested data, write data received in the request message to data storage or to amend data received and return it back via the response message.   
 One method used to exchange such messages is by means of the Simple Object Access Protocol (SOAP). SOAP is a standard which uses Extensible Markup Language (XML) to send and receive messages to and from a SOAP web service. SOAP specifies the structure and the encoding that the messages must be sent in to ensure proper communication. A single end point, or URI, on a server is exposed, which provides a set of operations that can be utilised to perform a task. This set of operations is described by a WSDL file, which also specifies the kind of response that can be expected when performing certain operations. The response from a SOAP web service is sent in XML format.  
 An alternative to approach to SOAP is to use a RESTful service. This method differs from SOAP in that it is implemented using web standards such as HTTP and URI, and existing web protocols, without the need for any additional protocols. Rather than providing a single endpoint and variety of operations at this endpoint, as SOAP does, every resource provided by a REST service has its own unique URI.   
 In my opinion the most appropriate type of web service for Megamax to employ would be a RESTful service. This is due to the fact that it already uses well known standards and protocols in its implementation, meaning it would be simpler and less complex to develop than a SOAP service. REST services are also more flexible as they can return data in other formats than XML, such as JSON, which is much more light-weight, and could mean that the performance of the app is better without the need to parse XML from the response into an object. REST services are also more secure as they can be monitored by firewalls, due to the fact that they operate on HTTP, which is not the case for a SOAP service.

**Word count: 469**

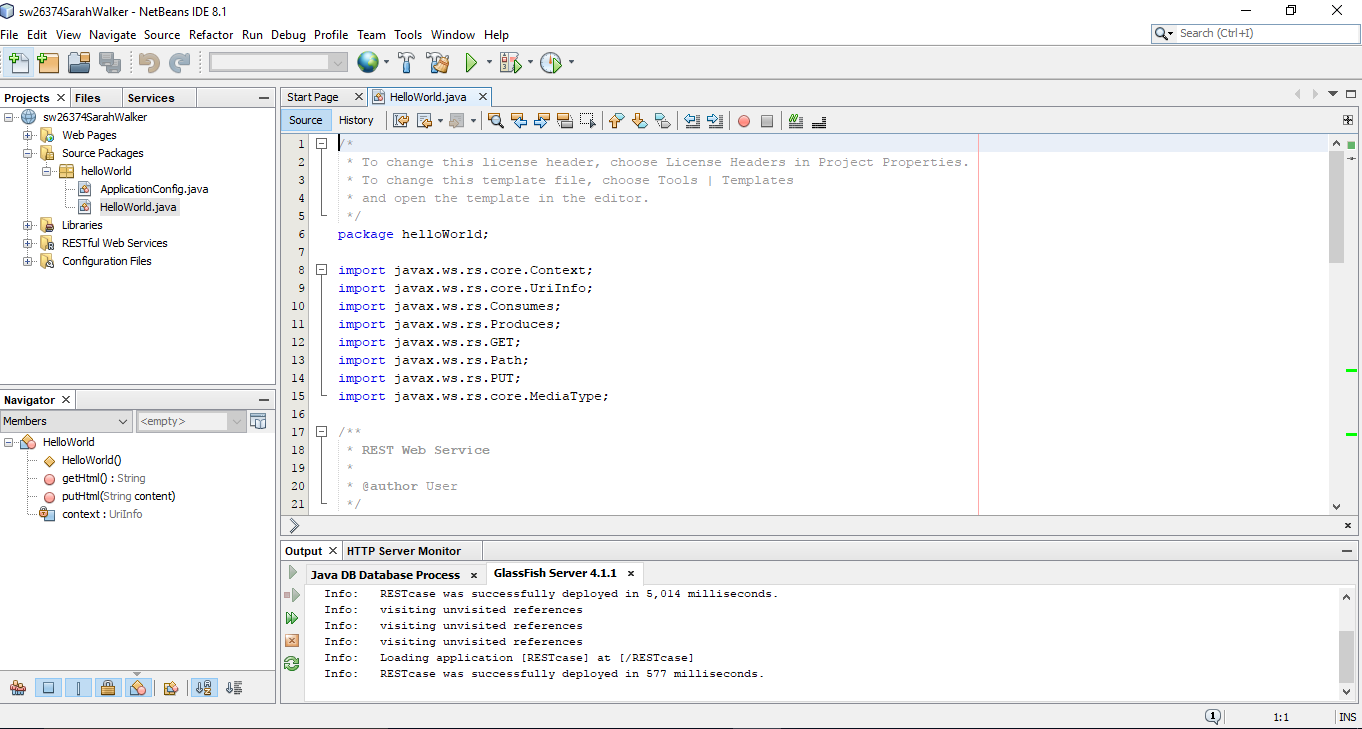
Part 2

i)



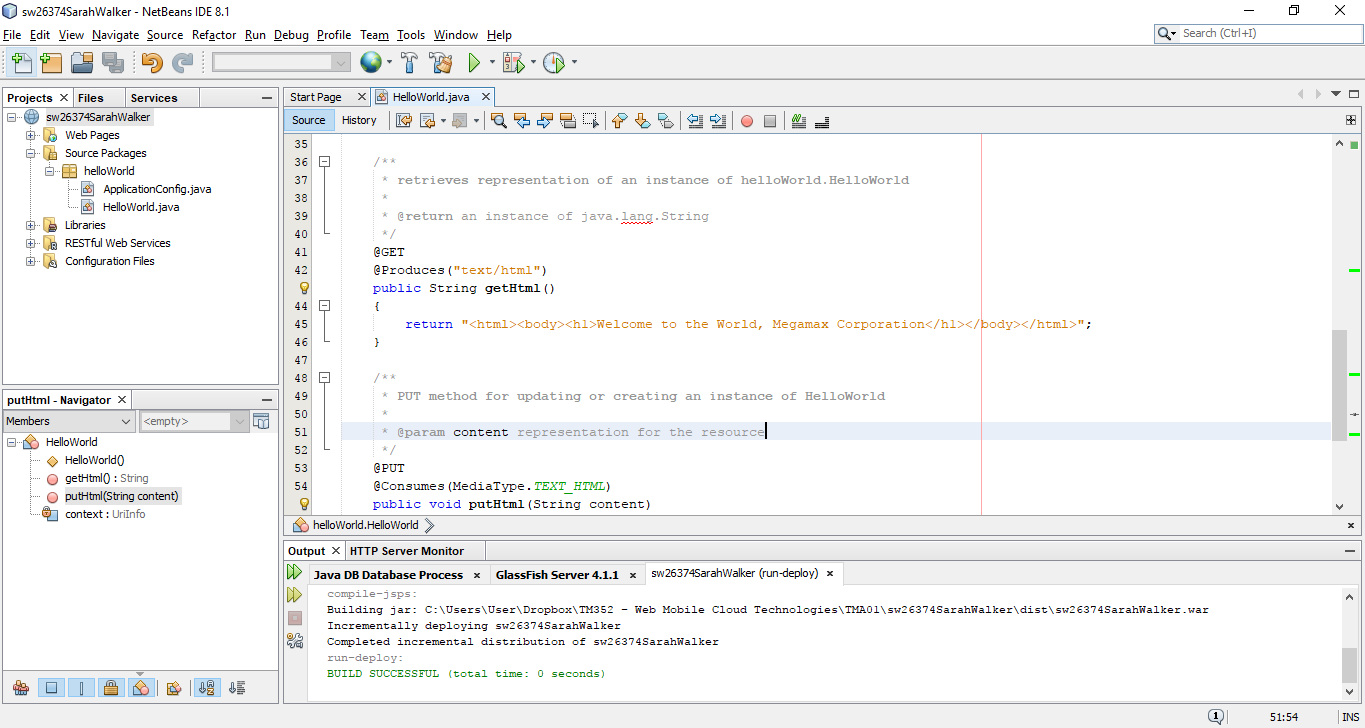
In this snapshot you can see that a new web application project has been created using in NetBeans IDE. The IDE has created all of the necessary filed for a basic web application, and we are ready to add our own web service to the project.

ii)



In this snapshot you can see that a RESTful web service named HelloWorld has been added to the web project, an operation made easy by the NetBeans IDE by simply right-clicking on the project and selecting the required menu option. The IDE has added two files to the project for us, the HelloWorld.java which will contain the code for the REST resources, and the ApplicationConfig.java which among other things specifies the path that will need to be used to access the web service. The path to the service is configurable by changing the ApplicationPath annotation in this file.

iii)



The final snapshot demonstrates how the web service can be tested in NetBeans using the ‘Test RESTful services’ function. A resource called getHTML() has been added to the service, which returns a simple HTML string that will be rendered on the page when the GET method on the service is called. The web application has been deployed using the Glassfish server, which is integrated with NetBeans, and the endpoints specified can then be tested in the browser window that opens. Upon creation, the MIME output type of the web service was set to text/html, and we have provided a method that will return a simple HTML string, but it would also be possible to set a different MIME output type and return data in the form of JSON from the service.   
 Now that the web service has been deployed, we could write a client application that would be able to request a resource from the URI http://localhost:8080/sw26374SarahWalker/webresources/helloWorld , and use the returned response in some way, probably displayed within a web page.

**Word count: 318**

Part 3

When transporting data across a network there are a number of threats that can jeopardisethe integrity, security and safety of the data, and as such security measures should be implemented throughout the various application layers to ensure safety. These measures include good coding practises, such as authentication and authorisation processes, and encryption of sensitive or confidential data during transportation.  
 There are many reasons that an unwanted user might want to access your application, including to steal confidential information, take over computer hardware for nefarious reasons or to attempt to misuse trust in your application to gain information for themselves, to name a few. Good computer security means protecting against these threats to maintain data integrity and confidentiality, and system availability.   
 The security of an application may be put under threat by external parties, but it is also possible for the intended users of the system to create security issues. A means of preventing this is to ensure that users are only authorised to amend the application in acceptable ways. This can be achieved by assigning users to a particular group within the system, ensuring only those in certain roles can perform certain functions. Implementation of a login process ensures that users can be authenticated, and can only perform their permitted functions, as well as ensuring that unauthorised users are unable to gain access to, and manipulate, the application.  
 One method to ensure that data confidentiality is maintained is to encrypt sensitive information for travel across the network. This entails encoding messages using a key to encrypt the message at one end of the communication, and then a key to decrypt the message at its destination. There are many means by which to encrypt data, some more secure than others, but by using a secure protocol for data transmission the encrypting of data is taken care of on our behalf.   
 Using the https protocol to transport data to and from your application will increase the security of its communications, and ensure that sensitive data cannot be intercepted and so remains confidential. Using https also ensures that the client and server are authenticated properly, and we can be sure that we are communicating with the entities we believe we are. https works by communicating across a secure channel, with the server providing a certificate to the client, meaning the client is able to authenticate the server. After this a means by which the message will be encrypted is ascertained, and the message is encrypted using said encryption method. In this way we can be sure that when the Megamax is communicating order and client information across the network, it is sending it to its own server.  
 In conclusion, the issue of web application security is an important subject to consider, but with some assessment of the risks and implementation of good coding practises and secure protocols, is not too difficult to ensure the application, its data and its users remain safe from threats.

**Word count: 492**