**RESTful Web Services**

The team began with no experience of developing web services and in particular RESTful web services. Initially the web services were auto generated within NetBeans using the functionality provided. This was done by linking the database to NetBeans and then generating web services for every table in the database. This approach was only useful in that it helped the team understand how RESTful web services operated. The downside of this approach was that it required extra libraries added such as Java Persistence API (JPA). This has drawbacks as the libraries are not always supported in servers such as Tomcat.

Once achieving an understanding of RESTful, the RESTful API was completely rewritten because the web services automatically provided by NetBeans did not provide the functionality required by our web and Android applications. Rewriting the API also meant that the API would be compatible within servers such as Tomcat, although the team has opted to use Glassfish Server 4.1 as the team requested to have Glassfish setup on the system provided by the course instructors and then set this up ourselves.

The choice to use web services was taken as it allowed for both the web and Android application to connect to the database through a single source. This ensures that both applications are forced to enter data into the database in the same format. Using web services also extracted the database query implementation out of the web and Android applications thus making much quicker and easier to develop these applications.

The web services are stateless thus every request to the server is dealt with and then forgotten. By doing this the server will not be overloaded with keeping sessions alive for all current requests and therefore many more servers hosting the web services can be added to allow for load balancing if required. Also if a server hosting the web services needed to be taken down then there is no worry about transferring sessions as there are none.

As the web services are designed using RESTful the errors provided map to http error codes such as ‘204 – No Content’. All errors provided when an input fails will be the same for both applications and made providing useful errors to members.