Database CW2
Tim van Ellemeet - tv0366

1. Which tasks were particularly easy/hard or interesting/boring?

Easy:

- It was relatively easy to implement most of methods for section A after the initial method for JDBC actions had been established. However doing them correctly was a different matter.

Difficult

- It was difficult to minimise the number of queries when dealing with requests that required multiple joins and variety of counts. For example when implementing getAdvancedPersonView; instead of doing a query for each count of likes and posts we chose to implement relatively complex algorithm to sift through all the post, topic and likes keeping track of counts which inevitably lead to significant debugging,
- It was often difficult to understand the root cause of an error when creating new API methods; whether it was in the database data, sql query, api function and/or other methods.
- It was hard polishing the API to the expected high standards and resolving all edge cases was pretty tedious.

2. What did you learn during this coursework?

- I learnt how to work with databases in Java, which I later used in Java OO assignments.
- I learnt about prepared statements and how using these avoids SQL injection.
- I learnt how to properly use Try/Catch statements, why they're useful and how to write functions that throw them.
- I greatly improved on my ability to navigate and interface with other APIs and classes. I chose to dig into the source code for methods instead of googling or looking at stack overflow.
- I further improved my debugging capabilities using IntelljiJ debug

3. What is your opinion of JDBC now that you have worked with it for a while?

- Honestly, It's relatively verbose approach to querying databases. Especially with setting queries with lots of prepared statements. However I appreciate that it is very robust and secure, and necessary to stopping SQL injection.
- It would be great if it had features to query the ResultSet itself, this would allow minimal queries to the database be made and also minimise if/else algorithms when cycling through each row in the result set.

4. If you were designing your own database API what would you definitely do the same, or definitely do differently?

- It was interesting to see a Web API built with tightly/statically defined responses, which were first defined as Class Views. This is different to my previous webtech experiences where responses have been loose JSON objects which the F/E interprets. This meant that views were very clear and defined and separated from the actual backend data processing and queries.
- It was also interesting to see the structure of java API where the flow went request in -> server -> handler -> API -> handler
- I would **definitely** use prepared statements in the future to avoid SQL injection.