Name: Matt Ritchie

Teammates: Liam Hurd, Melissa Morrison

Task 1, Repository: https://github.com/l-hurd314/CS320_Group_Project

Task 2, my contributions:

Database: I more-or-less did the database stuff on my own. I did copy in the given example as reference, but the methods used in the actual running of the project are almost entirely handmade.

Major commits:

https://github.com/l-hurd314/CS320_Group_Project/commit/aab961ed5e9e24a1f6807e937a06ae00b039a24b

https://github.com/l-hurd314/CS320_Group_Project/commit/0ebe656d95231624d2ae566c14c890bdcf0540de

https://github.com/l-hurd314/CS320_Group_Project/commit/5b90378a5e18c4d1fa5be7c2c5226e7d0c14a539

https://github.com/l-hurd314/CS320_Group_Project/commit/32e02c4c1fa3b6ff1d9c51643d80c3b62b8b0678

https://github.com/l-hurd314/CS320_Group_Project/commit/9a2e7b32f4d3e7c9a46179bc489c5fe4e34478c6

https://github.com/l-hurd314/CS320_Group_Project/commit/b86fa79cea6de772f0c30278fa27193d081135e1

https://github.com/l-hurd314/CS320_Group_Project/commit/14ea349688410f5b5446e5585dd206dcc25b39d5

https://github.com/l-hurd314/CS320_Group_Project/commit/eb4c4b7c6b70e484a31ce105508ec30ab953f066

https://github.com/l-hurd314/CS320_Group_Project/commit/ef3a9ea4e05d1934228621ba0674bed944650479

JUnits: I wrote essentially all of the Junit tests. This was partly because it was my given task, but mostly because the database stuff is the only bits any of us figured out how to write JUnits for.

Major commits:

https://github.com/l-hurd314/CS320_Group_Project/commit/dcbe1cf149aea22191fe36612dbc4a0ed42ec2c7

https://github.com/l-hurd314/CS320_Group_Project/commit/810b42e9bb5a4a8adedd11d3125eaa5b348c7720

https://github.com/l-hurd314/CS320_Group_Project/commit/416998ebaed0974cb9ca31b606d8050592589678

https://github.com/l-hurd314/CS320_Group_Project/commit/e5ea7bcad55aa7a172cddcb91ba70006a67c6bda

Testing tools: In order to see if the methods I built in DerbyDatabase actually worked before my teammates got it hooked to the front end, I created a bunch of query classes to utilize the methods I wrote and to assert they were outputting the proper data. I also imported and set up the SQLDemo class we were allowed to use for both testing and manual alterations to the running library.

Major commits:

https://github.com/l-hurd314/CS320_Group_Project/commit/b4e3d821a65dcbcbd6a2f00d6ca65440bfb18767

https://github.com/l-hurd314/CS320_Group_Project/commit/c016caaedb1d143c8f06babd9ae8a822bfc23215

https://github.com/l-hurd314/CS320_Group_Project/commit/9a2e7b32f4d3e7c9a46179bc489c5fe4e34478c6

Task 3, reflection:

Our team used a location based approach instead of a feature based approach when giving out tasks. In other words, I did not put in a new feature start to finish, I did the database portion of every new feature. This worked well because we each became specialized in the areas we worked, and could whip up a new feature without too many errors. Unfortunately this fell apart in debugging, where if one of us was stuck, the others would be looking at their code like it was a new language. We had to try to re-teach ourselves the other portions of the project in order to help out. We did this format with the intent of never touching the others' code so as to not muddle up github, but it caused us to be rather unproductive when we weren't sitting next to each other, but very productive when we were.

Next time I work on a team I plan to do a similar setup, with location based tasks, but not exactly. I would push for a task allocation that enforces that we each know enough about the others' portions of the project to construct their parts on their own, even if it's to a rudimentary degree and all it does is transfer a single piece of data up through the chain. This would ensure steady progress and cut down on stepping back and waiting for another teammate to finish hooking things up later.