Remaining Unimplemented User Stories

We were not able to complete six user stories, including:

- As a superuser, I would like to be able to query the database using SQL so that I can build custom reports, without having to modify the source code.
- As a manager, I require a method to be able to track changes in the system in order to asses the effectiveness of the team.
- As a radiologist, I want to be able to filter patient information so that I can quickly find and review patient information.
- As a manager, I would like to have a way for multiple radiologists to comment on a single patient's ultrasound, so that there can be a consensus on the treatment options.
- As a manager, I want to be able to resolve potential conflicting patient's information so that a clear consensus regarding the patient can be forwarded to the fieldworkers.
- As a manager, I require a way to approve diagnosis to be send to the field, so that every patient case is reviewed by at least two people.

We were unable to implement these user stories due to lack of time. We consulted our product champions earlier on and discussed which features were most important to them, and asked them to identify which features were less urgent. They identified that the features pertaining to these user stories were features that would be “nice to have”, but not required. As such, we dedicated our time to implementing user stories that were most important, and left the less urgent ones incomplete.

If the project were to continue, we would implement these user stories as follows:

- For the first user story, we would add a pop-up text box that allows admin accounts to type queries in, runs and display the contents of these queries in dynamic Jtables.
- For the second user story, we could add another table into the database that keeps track of changes made to the database, and the time these changes are made. Every time a call is made to the database, we can execute the relevant query, along with an added query that inserts another entry into this table that describes the changes made to the database.
- For the third user story, we could add a text box to the java client that serves as a search bar for patients. Radiologists can enter patient details as the search term, and entries containing this term in the patients tables can be pulled from the database, and displayed on the existing patient information table on the client.
- For the remaining user stories, the existing records table in the database can be broken into two new tables, the record table and the response table. Each entry in the record table will now contain a record ID, together with the patient information, fieldworker comments, and a response ID. Each entry in the response table can then contain a response ID, a record ID, together with annotations and radiologist comments. The response ID in the record table will now correspond to the response for that record that was approved by the administrator, and the record ID in the response table will now correspond to the record that response is addressing. The queries used to insert and retrieve records and responses from the database must then be modified to accommodate these changes, and a new tab must be included on the java client for admin accounts. Here, once a record is selected, all of the responses for that record will be shown, and managers will then be able to select one of these responses. The response ID of this response will then be added to the selected record in the database.