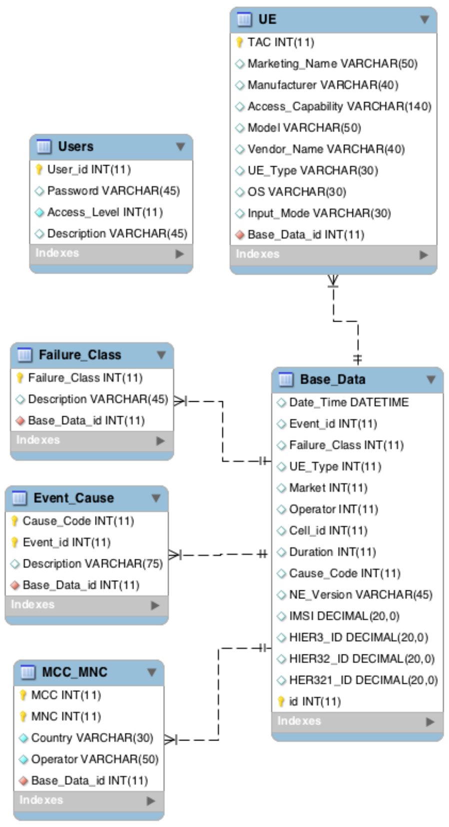
Sprint Two

Group Two

Team Members:

- David Leonard
- Carl McCann
- Sahar Mohamed-Ali
- Ciarán Sweeney
- Dimitar Tsvetkov

Database



The design of the tables and relationships between the tables were based on the data in the excel dataset files.

The tables that we made are as follows basedata, event_cause, failure_class, mcc_mnc, ue and user table. For the base_date table we added in a primary key called id and made it an integer that increments by one once a new row is entered into it. The base_data table has two relationship, one with the event_cause table and one with the failure_class. The failure_class table primary key is Failure_class and this table has one relationship with the base_data table. The event_cause table has a composite primary key which is made up of the fields Cause_Code and Event_id. This table again has a relationship with the base_date table. The mcc_mnc table has again a composite primary key which is made up of the integer fields of MCC and MNC. The ue table primary key is an integer field called Taco. The user table has a primary key called User_id. The user table is used to store all the users and their password which allows them to log into our system. The Access_Level field is an integer which determines the amount of access the user has that logged into the system. If the Access_level was 4 then the user would have network management engineer rights, if it was 3 then the user would have support engineer rights, if it was 2 the user would have service rep rights and if it was 1 then the user would have customer support representative rights.

Since Sprint 1, we have also added join relationships with the other tables for the remaining queries and user stories not covered in Sprint 1.

File Upload

An excel file can be uploaded from the browser via JavaScript, and AJAX. This is done via a POST method and the data is encapsulated as multipart form data. This is passed into the Java code in BaseDataRest where the upload, filename parsing, and saving of file is done.

Initially, an excel file was uploaded, and a WatchService spotted it and passed it on to be converted to a csv file, which is then parsed, cleaned and ready to upload to the database. In the now completed version, the excel file is not changed to a csv file, but parsed using Apache POI. Entities are created from rows in the spreadsheet and persisted via stateless EJB.

The File Upload option is only available to the System Admin. And the whole process takes approximately 80 seconds. No notification is given of the time taken in the front end, but the process always works.

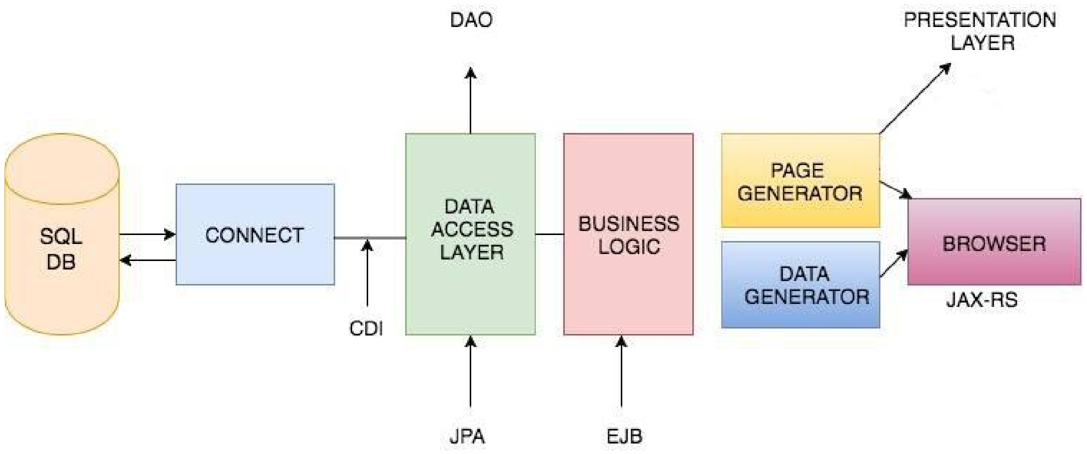
User Login

The Queries are divided by their functionality using language used by the customer - First-line queries, evaluation, investigation, and admin. Each user, when created by the System Admin is given an integer which corresponds to their access level. The Customer Support Rep has only access to their queries, the support engineer has access to his queries and the customer support rep queries, while the network engineer has access to the queries of all three. The system admin, however, only has the ability to upload files and create users.

Using a HttpServlet, that is called via forms in the login screen that call the doPost method, users are logged into the system after a check on the users table. This involves storing the userEntity into a session in the servlet. Logout is done in the same method, but called from the main page. Using if statements to check for a logged in user, the method will either log the user in or out.

The doGet method in the servlet is used to verify users throughout the session, as it passes the userEntity into javascript that loads the username into the navigation bar, and loads queries based on the user’s access level. When no one is logged in, none of the queries are accessible.

Full Slice



The system has implemented a full slice. The server requests data from the database and receives said data. Entities are retrieved from the database by the data access objects using named queries stored in the BaseDataEntity. These were propagated through the system via EJBs, and the REST classes which are called by the front end scripts, and sorted into tables/graphs etc.

Arquillian Testing

Although, our arquillian testing codes are written, we have not been able to get a successful result on the tests. We tried using wildfly and jboss containers, remote, embedded and managed. They were verified with a maven build but our errors ranged from logging errors to not being able to create unique instances of the TestRunnerAdapter.

Queries

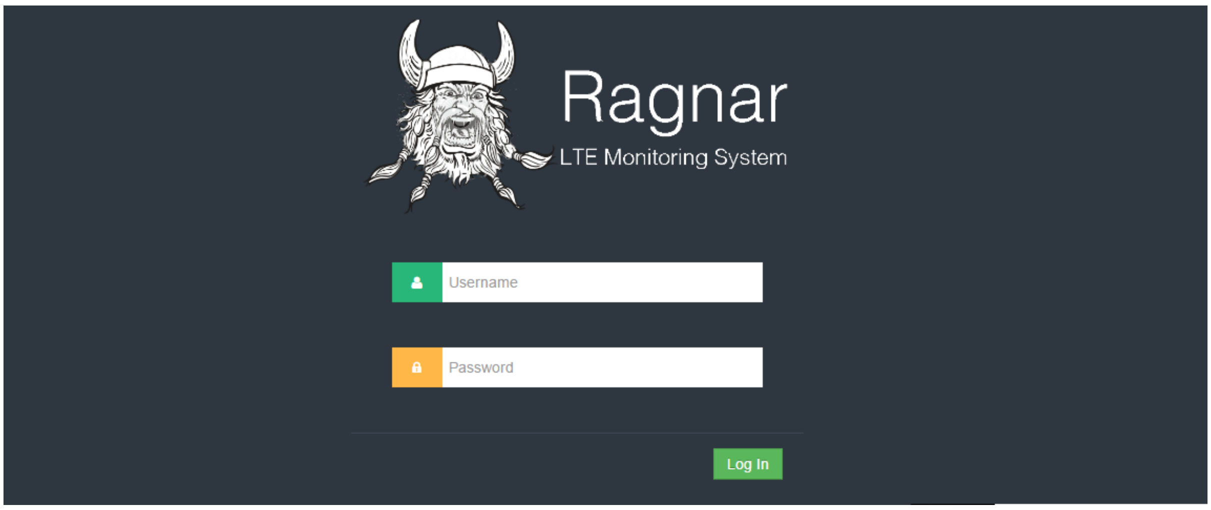
Named Queries were introduced for Sprint 2 to streamline the code, and gather the query logic into one place: BaseDataEntity. This was far superior to the queries being held in each individual DAO method.

All tables were joined properly, and the information returned now from the queries provide much more insight into the errors being monitored. Combined with Bootstrap and Highcharts, these allow quick interpretation of the data at hand.

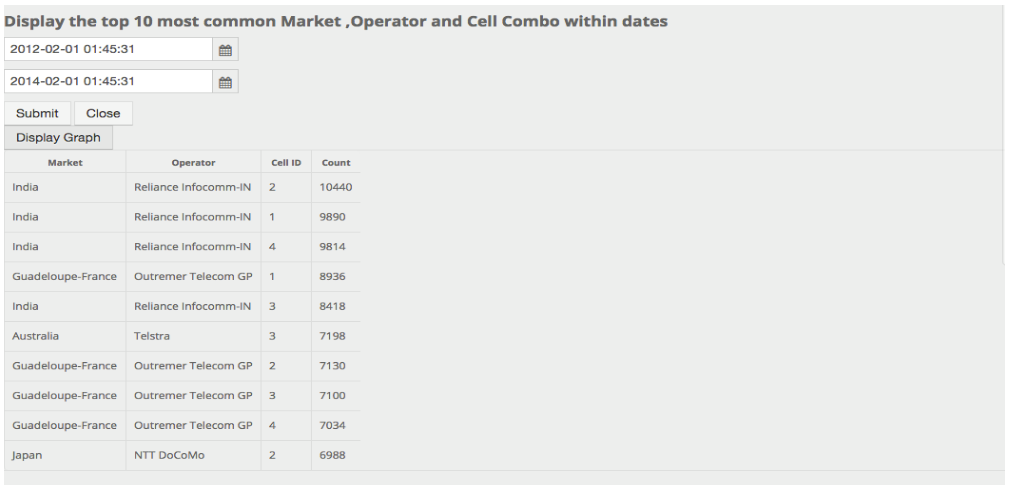
Extra queries were written to facilitate the ease of which data could be picked from the database, filling comboboxes with IMSIs for example, to allow for easier selection.

Graphs
The Network Engineer required graphical representation for their graphs. For this, highcharts was used, the data was taken in as a table for some data-points, and arrays for others.

Bootstrap and Design
One of our main objectives this sprite was to completely change the design of the front end of the system for a more customer-oriented program. Our main goal was to make our site as user-friendly as possible which was our main criticism from sprint one. There are two main pages to this system which is the login page and the main page which was worked from the bootstrap we downloaded. This was coded by taking elements we liked out of the bootstrap example and adding them to our code. This method of coding caused problems as certain features of bootstrap weren't working fully and the whole front end had to be recoded from the start. The main issue was a lot of the icons that came with the bootstrap were not appearing on the page despite having links to it. For the recording we took the bootstrap code and added our code onto it rather than taking code from it. Then we stripped away the unwanted code that was left over.



As you can see from the image from the top this is the login screen, where we also include our logo which we feel make this system very marketable.Once the user enters in their right username and password, they will be brought to the home page. Depending on the access level the user will be restricted to certain elements of the site. The queries are split up into four types on our page which are as follows: first line queries, investigation problems, evaluation and administration. These types of queries are displayed as a drop down menu on the sidebar of the site and once clicked on, shows what queries are in it. Once selected the query is displayed in the centre of the page. This can be seen by the query below being displayed.



Display the top 10 most common Market ,Operator and Cell Combo within dates

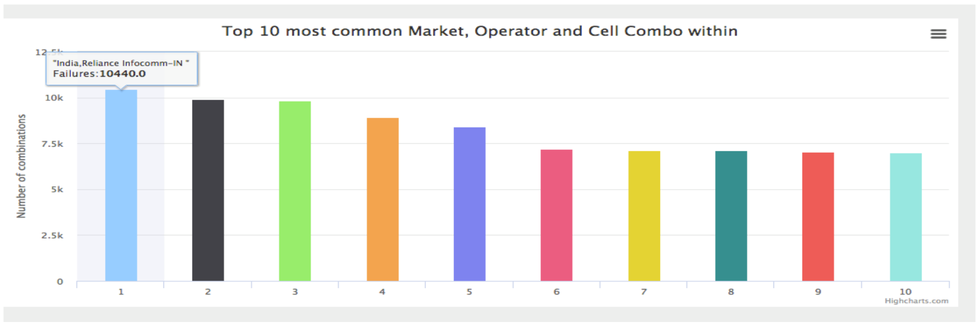
2012-02-01 01:45:31

2014-02-01 01:45:31

Submit Close

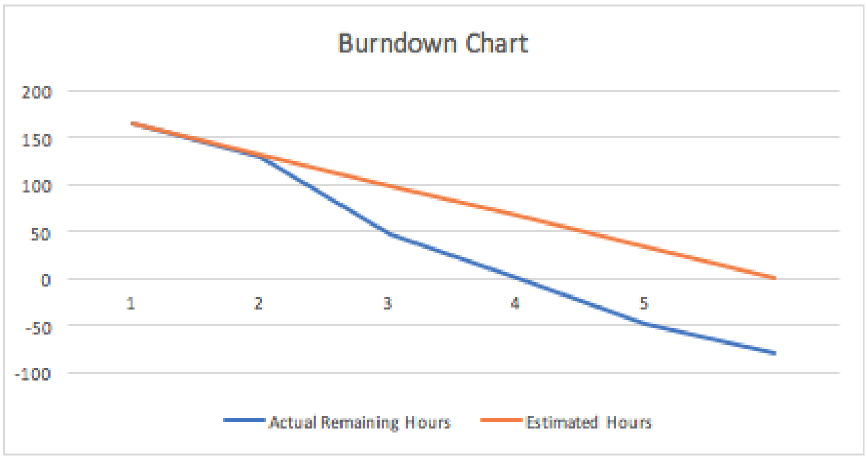
Display Graph

Market	Operator	Cell ID	Count
India	Reliance Infocomm-IN	2	10440
India	Reliance Infocomm-IN	1	9890
India	Reliance Infocomm-IN	4	9814
Guadeloupe-France	Outremer Telecom GP	1	8936
India	Reliance Infocomm-IN	3	8418
Australia	Telstra	3	7198
Guadeloupe-France	Outremer Telecom GP	2	7130
Guadeloupe-France	Outremer Telecom GP	3	7100
Guadeloupe-France	Outremer Telecom GP	4	7034
Japan	NTT DoCoMo	2	6988



Devices and System Used

The IDE used throughout was IntelliJ. The server used was Wildfly 8.2.1.Final. MySQL Workbench version 6.3 CE was used. Version control was managed by Git, and Maven was used to manage the build. Communication was mainly through Whatsapp group. Time management and allocation was managed by excel sheets. Scrum meetings were held once or twice a week and documented in the minutes. The following Burndown chart was generated in an excel sheet used to keep track of time and tasks. The data points coordinate with scrum meetings when time spent on each given task are given as opposed to daily records. As you can see, we have estimated 80 hours less work than the work that was done.


Burndown Chart
Actual Remaining Hours
Estimated Hours