Group Two Project Documentation

Team Members:

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

For this sprint, we focused on completing user stories. We concentrated on the file upload, validation, persistence of objects to the database, and more complex database queries.

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.

Data Import & Validation

DataWatcherAndImportEjb is a singleton bean, that watches the directory where files are uploaded. When it spots a .xls file it calls a method from ExcelHandler to open the file, get the Base Data sheet, which is then passed to the baseDataPersister() method. This iterates over every row, performing some data type validation, and creating objects from them. These objects are then further validated, based on foreign key constraint errors we noticed, nulls, and corrupted columns. A list is returned back to DataWatcherAndImportEjb which loops through it and persists it.

Complex Queries

We first wrote queries based on the provided user stories in mySQL. Once we had these perfected, we translated it to JPQL, and began implementing them. This begins in the DAO, Where we create queries via the Entity Manager. We fetch results from the query and return them. This is propagated through the stack via an EJB, then a REST class, and finally displayed via JavaScript/AJAX.

User Stories

Completed User Stories

- User Story 1 - As the System Administrator I want to be able to receive and import Datasets.
- User Story 3 - As the System Administrator I want to have incoming records checked for consistency (i.e. valid date / time values, valid MNC / MCC combinations, valid Event IDs, Cause Codes etc.) and have erroneous records highlighted and excluded.
- User Story 2 - As the System Administrator I want to be able to assign an Id and password to each User type (i.e. Customer Service Rep., Support Engineer, Network Management Engineer).
- User Story 4 - As Customer Service Rep. I want to display, for a given affected IMSI, the Event ID and Cause Code for any / all failures affecting that IMSI.
- User Story 7 - As a Support Engineer I want to see a list of all IMSIs with call failures during a given time period.
- User Story 9 - As a Network Management Engineer I want to count, for each IMSI, the number of call failures and their total duration during a given time period.
- User Story 21 - As the System Administrator I want data import and update to the Call Failures database to be executed in < 2 minutes for a dataset of 30,000 records.
- User Story 5 - As a Customer Service Rep, I want to count, for a given IMSI, the number of failures they have had during a given time period.
- User Story 11 - As a Network Management Engineer I want to see the Top 10 Market/Operator/Cell ID combinations that had call failures during a time period.
- User Story 6 - As a Customer Service Rep. I want to see, for a given IMSI, all the unique Cause Codes associated with its call failures
- User Story 12 - As a Network Management Engineer I want to see the Top 10 IMSIs that had call failures during a time period.
- User Story 14 - As a Support Engineer I want to display, for a given failure Cause Class, the IMSIs that were affected.
- *User Story 20 - As any type of User I want to have a response time of < 2s for any query I initiate. – Applies to all queries so far.*

We have implemented complete slice for all the above queries.

We focused on queries that did not include joins. In the next Sprint, we will move on to those remaining queries.

User Login

Input	Output	By Who
Dataset	Dataset	System Administrator
Dataset	Incorrect records highlighted and excluded	System Administrator
ID & Password	Changes to user accounts	System Administrator
IMSI	Event ID, Cause Code for all affecting failures	Customer Services Rep
Time period	List of all IMSIs with call failures in the given time period	Support Engineer
Model of phone and time period	Number of call failures it has in the given time period	customer Services Rep
IMSI and time period	Number of call failures and their total duration	Network Management Engineer
Model of phone	All unique failure Event ID and cause code combinations and number of occorunces	Network Management Engineer
Dataset	Updated Call Failures	System Administrator
-	-	Support Engineer
-	-	Network Management Engineer
IMSI and time period	Number of call failures it has in the given time period	Customer Services Rep
Time period	Top 10 Market/Operator/Cell ID combinations that had call failures during a time period	Network Management Engineer
IMSI	Number of call failures and their total duration	Support Engineer
Time period	Top 10 IMSIs that had call failures	Network Management Engineer
Failure Cause Class	IMSIs affected	Support Engineer

Specific queries are allocated to each type of User due to the varying levels of access. For example a Customer Service Representative only uses queries to give a “first-line” response to Customer Help calls. The Support Engineer has a higher level of access and uses queries to help in investigating problems escalated by the Customer Service Representatives, therefore they have access to all Customer Service queries. The Network Management Engineer has access to the Customer Service Representatives queries and the Support Engineers queries. The System Administrator does not have access to any of these queries. They are solely there to import datasets and manage users. For now, we have represented this with a separate html page for each user.
A full slice is done when the user logs in. The user’s password and user ‘s name is compared to the database. If a match is found on both then the user is logged into the site. Depending on the access level he/she will be send to a certain page. The access level is send through the url from page to page. This is not the best way to do it but this will be changed in sprite 2.