# Sprint 3 Report

Team: eLation Nation

Project:eLation Mobile Apps

**Project Sponsor**: Innovative Systems LLC

Sprint Duration: November 7, 2012 to December 5, 2012

## Team members:

* Michael Malkowski
* Rachel Pekarek
* Jeremy Warner

**Project Sponsor**: Innovative Systems LLC

## Meetings/Client Interactions

* 11/8/12
  + Scrum Meeting
    - Determined Sprint 3 Goals
* 11/15/12
  + Scrum Meeting
    - Updated Client on status
* 11/20/12
  + Scrum Meeting
    - Updated Client on status
* 11/27/12
  + Scrum Meeting
    - Updated Client on status
* 11/29/12
  + Scrum Meeting
    - Re-defined database schema
* 12/4/12
  + Scrum Meeting
    - Updated Client on status

## Sprint Tasks

Our sprint tasks were divided into the following categories:

* Android
  + Implement all API call functionality in UI – {Mike}
  + Build Technician App – {Mike}
  + Implement PDF Viewer – {Mike}
  + Set up database and build internal data storage structure – {Mike}
* Apple
  + Implement all API call functionality in UI – {Rachel}
  + Build Technician App – {Rachel}
  + Implement PDF Viewer – {Rachel}
  + Set up database and build internal data storage structure – {Rachel}
* API
  + Finish all eLation GET/POST API call handling on Apple and Android – {Jeremy}
  + Build POST API call handling functionality for Apple and Android – {Jeremy}
  + Build all API call handling for Technician app for Apple and Android – {Jeremy}
  + Expand API error checking – {Jeremy}
* Documentation
  + Power Point Presentation – {Mike, Rachel, Jeremy}
  + Software Development Document – {Mike, Rachel, Jeremy}
  + UI Design Master Document – {Mike, Rachel}
  + Sprint 3 Report – {Mike, Rachel, Jeremy}

## Goals Met

This section will detail the goals we were able to meet for this sprint.

* Android
  + Implement all API call functionality in UI – {Mike}
  + Build Technician App – {Mike}
  + Implement PDF Viewer – {Mike}
  + Set up database and build internal data storage structure – {Mike}
* Apple
  + Implement all API call functionality in UI – {Rachel}
    - All available API calls were successfully implemented
  + Build Technician App – {Rachel}
    - Technician App is functional with the exception of the one-time set up process
    - UI may need to be revised
  + Implement PDF Viewer – {Rachel}
    - PDF Viewer working
  + Set up database and build internal data storage structure – {Rachel}
    - Both database and internal data storage structure are set up
    - Still waiting to test database and structure with real data
* API
  + Finish all eLation GET/POST API call handling on Apple and Android – {Jeremy}
    - All available calls are handled. Currently we have one API call that does not give us the correct data back and so is not done.
  + Build POST API call handling functionality for Apple and Android – {Jeremy}
    - Built in ability to handle POST type HTTP requests
  + Build all API call handling for Technician app for Apple and Android – {Jeremy}
    - All available calls are now handled for the Technician API
  + Expand API error checking – {Jeremy}
    - API error checking and handling is standardized and done for Android.
* Documentation
  + Power Point Presentation – {Mike, Rachel, Jeremy}
    - Done
  + Software Development Document – {Mike, Rachel, Jeremy}
    - In progress
  + UI Design Master Document – {Mike, Rachel}
    - Needs to be updated
  + Sprint 3 Report – {Mike, Rachel, Jeremy}
    - In progress

## Goals Not Met

This section will detail the goals we did not meet for this sprint

* Android
* Apple
* API
  + Expand API error checking – {Jeremy}
    - Did not complete standardization and error checking/handling on Apple
* Documentation
  + UI Design Master Document – {Mike, Rachel}
    - We would have liked to have done more on this document. Unfortunately we were unable to find the time to work on this document as it was of a lower priority. It will be addressed further next sprint.

**Research/Code Experiments**

* Rachel and Jeremy found and utilized a Firefox add-on called SQLite Manager that allowed us to make a script for our database
* Rachel and Mike researched ways to add non-native expandable views to Apple and Android applications.

## Prototypes & Features

* Functional UI applications for both Android and iOS
* Real data is implemented in Android’s Current Invoice Summary

## Product Backlog

* Sign in pages using OpenAuth on both platforms
* Implement PDF viewer and storage on both platforms
* Set up Databases (Persistent data)
* Build POST data functions and methods into API calls
* Implement the Log Pages
* CONOPS, Requirements, and Design Documents
* Use API calls to fill UI with real data
* Determine how to handle Previous Invoice and Payment History Pages
* Make iOS app compatible with the iPad
* Check password strengths and show a progress bar
* Valid email verification
* Investigate Trouble Tickets
* Build Unit Tests
* Set up API and UI for Technician App