|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | Project Milestone 1 | |
|  |  |  |  |  |
|  |  |  | Project Name: | Biochemical Plate, Assay, and Result Management system |
|  |  |  |
|  | Team: | West-East99 (We99) |
|  | Date: | 23rd Feb 2015 |
| CSCIE-99  Project | Customer: | Peter Henstock |
|  |  |  |  |
|  |  |
|  |  |  | |
|  |  |  |  | |

Contents

1. Document Administration 3

1.1 Document Change History 3

1.2 Document Content Owners 3

1.3 Document Reviewers 4

1.4 Document Approvers 4

2. Project Overview 5

2.1 Project Vision 5

2.2 Current Process Problem being solved 5

3. Assumptions and Constraints 6

3.1 Assumptions 6

3.2 Constraints 6

4. Scope 7

4.1 In Scope 7

4.2 Out of Scope 7

5. Proposed Functionality 8

5.1 Introduction 8

5.2 Overview 8

6. Business (Functional) Requirements 9

6.1 Security Functional Requirements 9

6.1.1 Security Overview 9

6.1.2 Sample Screens / Story Board 9

6.1.3 Open Issues / Questions Security Section 11

6.2 Plate Management Requirements 12

6.2.1 Plate Management Overview 12

6.2.2 Sample Screens / Story Board 14

6.2.3 Open Issues / Questions Plate Management 21

6.3 Experiment Management Requirements 22

6.3.1 Overview 22

6.3.2 Sample Screens / Story Board 23

6.3.3 Open Issues / Questions Experiment Management 24

6.4 Results Analysis Requirements 25

6.4.1 Overview 25

6.4.2 Sample Screens / Story Boards 25

6.4.3 Open Issues / Questions Results Analysis 26

6.5 Special Feature Proposals 27

6.5.1 Overview 27

6.5.2 Interactive Visualization 27

6.5.3 Audit Trail 29

6.5.4 Four Eyes Protocol 30

6.5.5 Flexible Plate Template Editor 31

6.5.6 Plate Sharing 31

6.5.7 Support Machine / Equipment File parsers 31

6.5.8 Open Issues / Questions Special Features 31

7. High Level Design 32

7.1 Architecture 32

7.1.1 Introduction 32

7.1.2 Model 33

7.1.3 Controller 33

7.1.4 View 33

7.1.5 Components 33

7.1.6 Database 34

8. Project Planning and Estimating 35

8.1 Iterations 35

8.1.1 Overview 35

8.1.2 Iteration Calendar 35

8.2 Stories and Estimates 36

8.3 Main deliverables 36

9. Project Risks 37

9.1 Risks Identified 37

10. Appendix Story Backlog March 2nd 38



# Document Administration

## Document Change History

| Date | Author | Version | Description of Change |
| --- | --- | --- | --- |
| 23rd February 2015 | Alan Orcharton | 1.0 | Initial Document |
| 24th February 2015 | Sean Sinnott | 1.1 | Added information to section 5.4 (results import) – and some questions related to the selection of points for linear regressions. |
| 25th February 2015 | Alan Orcharton | 1.2 | Merge comments from Mark and Alex |
| 1st March 2015 | Alex Zaman | 2.2 | Updated Plate Editor Text and mockups. |
| 1st March 2015 | Alan Orcharton | 2.3 | Updated Project Overview sections. |
| 3rd March 2015 | Alan Orcharton | 2.4 | Updated Plate Editor Text and mockups. |
| 3rd March 2015 | Alan Orcharton | 2.5 | Updated Experiment Mgr and Architecture. Also added stories and estimates. |

## Document Content Owners

| Role | Name |
| --- | --- |
| Author | Alan Orcharton |
| Contributor | Alexander Zaman |
| Contributor | Tim Stefanski |
| Contributor | Mark Ford |
| Contributor | Sean Sinnott |

## Document Reviewers

| Name | Role |
| --- | --- |
| Peter Henstock | Customer |
| Robert Zupko | Consultant |
| Alexander Zaman | We99 Front End Developer |
| Tim Stefanski | We99 Front End Developer |
| Sean Sinnott | We99 Front End Developer |
| Mark Ford | We99 Back End Developer |
| Alan Orcharton | We99 Back End Developer |

## Document Approvers

| Name | Role | Signature / Electronic approval | Sign-off Date |
| --- | --- | --- | --- |
| Peter Henstock | Customer |  |  |
|  |  |  |  |
|  |  |  |  |

# Project Overview

## Project Vision

Our overall vision is to create a centralized web application that will provide tools to assist scientists in managing experiment plate sets and with the analysis of experimental results.

The team focus is providing outstanding tools to assist in analyzing the dose response characteristics potential drugs.

We will provide beautiful interactive visualization tools that will assist the scientist in:

* Creating and managing experimental plate sets for dose response experiments
* Performing quality control checks on plate results and plate controls
* Performing Interactive analysis of dose response results.

The team will store the experimental results and the analysis of the results so that the historic experimental analysis can be reviewed or reproduced at any time.

Scientists will be able to save the results for further analysis or publish their analysis, making it available to all other users.

## Current Process Problem being solved

New drug development is a time consuming and expensive process. In the early stages of drug discovery millions of compounds are screened and tested on the road to finding a few candidates for further exploratory development.

High throughput screening is a method scientists’ use in the early stages of drug discovery to test millions of potential drugs. The method allows scientist to quickly identify candidates. While the process is highly automated, a disparate selection software of software tools are often used to manage plate sets and analyze results. Commercial software to manage the process is often extremely expensive

The We99 team is proposing releasing an integrated suite of open source web based tools. The team believes that the tools will enable scientists to be more organized and efficient when screening and analyzing potential drugs. Time savings will allow scientist to more effectively eliminate compounds earlier in the discovery cycle potentially saving companies millions of dollars.

# Assumptions and Constraints

## Assumptions

## Constraints

* Software Solution must be free. All source code for the solution is to be made available to the open source community. Any part of the system may be openly used or further developed.

# Scope

## In Scope

## Out of Scope

# Proposed Functionality

## Introduction

The purpose of this document is to assist the We99 project team and the customer gain a common understanding of the high level functionality of the Biomedical Plate, Assay and Result Management system. The requirements have been documented in storyboard format. A short description of the functionality is included with some mockup screen shots. This format is intended to provide a good overview of the system from a users point of view.

## Overview

We propose a web-based software solution to facilitate the creation of Biochemical plate specifications, the management of experiments (assays), and the analysis of results.

We have broken the requirements into sections to facilitate the review and analysis of the functional requirements as follows:

**Security**

Requirements associated with logging into the system and defining user roles within the system.

**Plate Management**

The definition, creation, and storage of biochemical plates and plate sets to be used in experiments

**Experiment Management**

The definition, creation, and storage of experiments conducted.

**Results Analysis**

Functionality associated with loading, analyzing, reviewing, and storing experiment results.

**Special Features**

Proposals for advanced features to enhance the product

# Business (Functional) Requirements

## Security Functional Requirements

### Security Overview

We propose that all users of the system must have

* An email address – for use as a unique identifier and during the login process.
* An account – to logon to the system
* A User role – Administrator or Scientist

Nearly all of the operations in the system require a specific permission granted to the user in order to access the feature. These permissions are grouped into Roles and a User is assigned a single Role. This approach allows for easy expansion of the system to include user defined roles or customization of the existing roles.

A user with the Administrator role has access to create and remove user accounts.

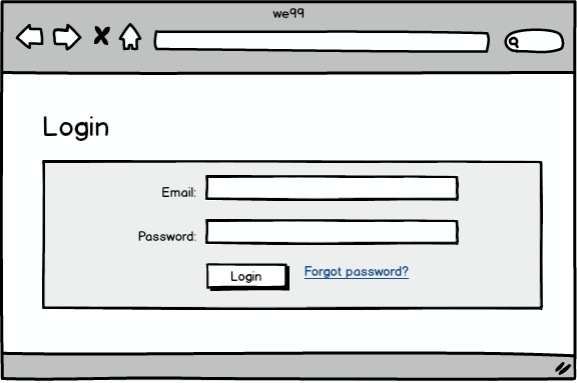
We propose requiring the following information for a user account:

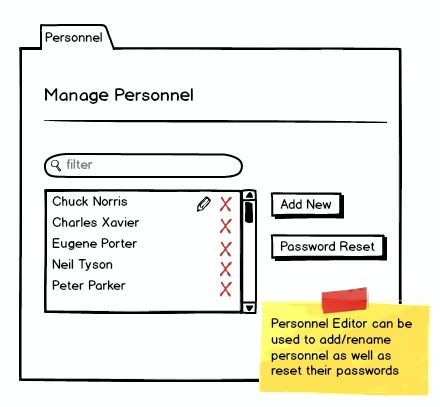
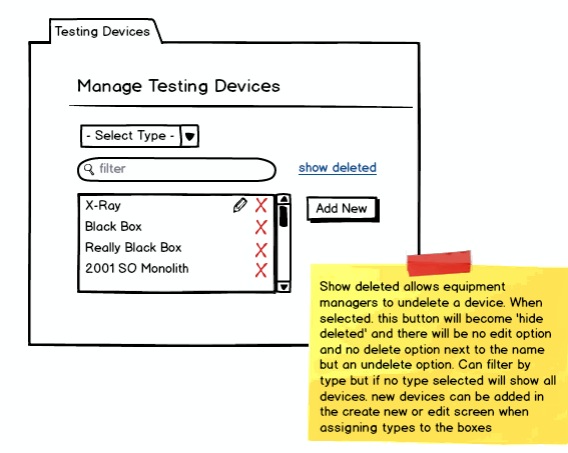
* **Email address** – used at the username. Email addresses help ensure unique usernames. The email address may by used in future for sending notification to the user.
* **First name / Last name** – used to identify the user in a more friendly form.
* **Password** – a user supplied password for authentication.

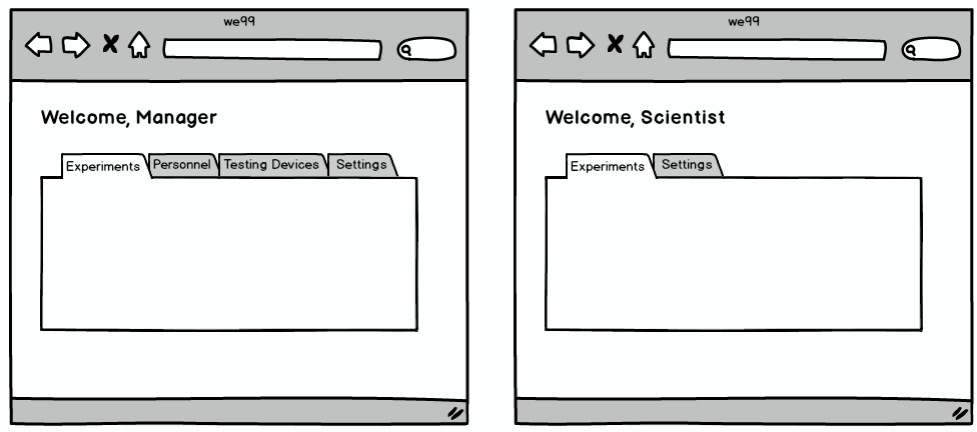
Users with an Administrator role have access to create and remove user accounts and to reset user passwords.

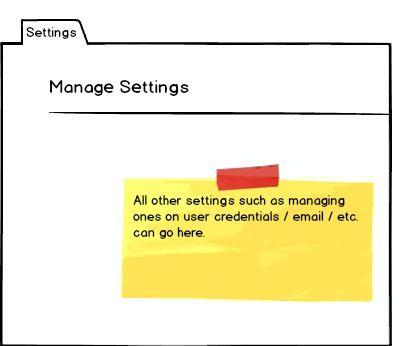
Administrative users also have the ability to add data on test equipment used in experiments. This information can be used to help identify defective equipment and functionality can be expanded to incorporate file parsers to assist the user in loading results produced from a machine that may be in a non-standard format (see special features).

### Sample Screens / Story Board









### Open Issues / Questions Security Section

| # | Description | NOtes |
| --- | --- | --- |
| 1 | We can possible use Testing devices to manage import export formats of files associated with the device |  |
| 3 | Organization Units – Is there a need to allow organizational groupings to manage accounts in large corporate environments | Is a nice idea for large organizations |

## 

## Plate Management Requirements

### Plate Management Overview

Plate Management covers creating a specification of the contents of microtitre plate sets to be used in experiments.

The plate management service offers a central place for the user to manage templates and create plate sets for their experiment. The experiment manager offers an overview of existing plate sets and their status. It also offers quick access to create new plate sets.

**Plate Templates**

Plate sets are created using plate templates. Plate templates contain labels that act as place holders for how compounds and controls should be arranged on plates. The process of creating plate templates is covered in detail in the storyboards below.

Plate templates are stored in the system and can be searched based on several criteria.

* Number of positive
* Number of negative controls

**Plate Details**

Plates are identified with a plateID (a.k.a barcode). Plates are assumed to be rectangular and can be created in any rectangular dimension. The dimension is specified using the number of rows and the number of columns. The product of rows x columns defines the number of wells that a plate possesses. Wells themselves are identified by the plate to which they belong and their row and column coordinates.

Plates are made ready for experiment by filling its wells with a specific concentration of a compound.

The concentration of a compound is specified via a number and a unit of measure (e.g., 20 uM).

**Creating New Plate Sets**

New plates can be created using the ‘Add New’ button from the experiment manager. This leads to a plate set creation wizard. The user uses this wizard to provide relevant information such as:

* The name of the plate set
* Plate type (make / model)
* # of plates and their IDs
* Dosages per plate

**Plate Import / Export**

Plate set setup information can be saved and reused using the import and export feature. Imports and exports are stored in clear text json format. This allows clients to export the data into their own processing systems or even edit the plates manually if the changes are very minor.

The export feature just requires a plate set to be selected and will save the file.

The Import feature receives a json export and asks the user to reassign some of the unique fields (plate set name and plate Ids). It will populate the database based on the associated json.

**Updating Plate Set Status**

Plate Sets can be updated from the main plate manager screen using the links next to the plate in the ‘Action’ column.

When a plate created, it reaches a preparing state. ‘Mark prepared’ will mark the plate as prepared and assign the marker as the preparer.

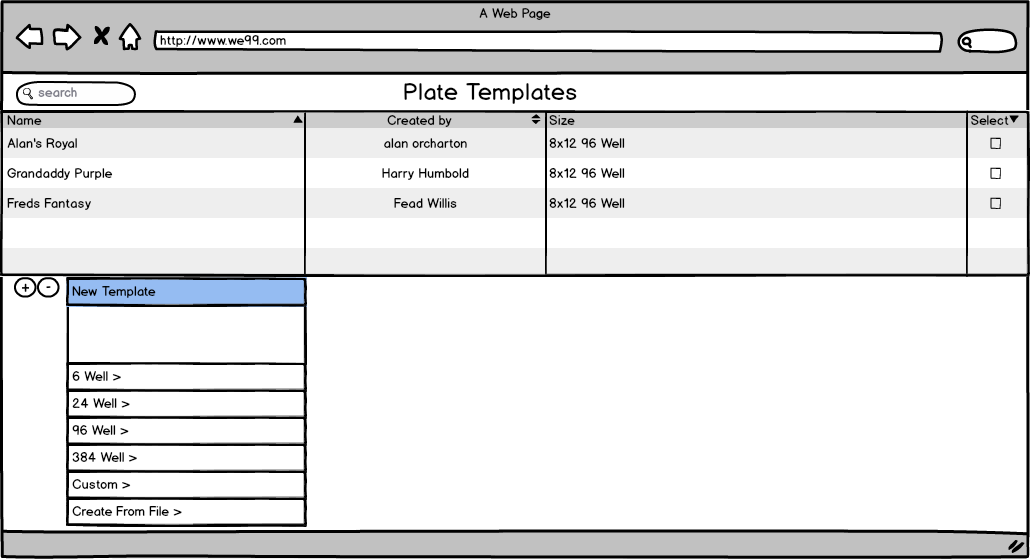
When the plate is prepared, the ‘send to device’ button will be available. Upon clicking it the state will change to ‘in testing’ and the marker will be identified as the tester.

When the plate is ‘in testing’ the upload analysis will become available. This link raises a popup allowing the user to upload the analysis provided by the machine. Upon upload, the state will change to ‘tested’.

In the tested state, users will be able to reach the analysis interface using the ‘view analysis’ link.

### Sample Screens / Story Board

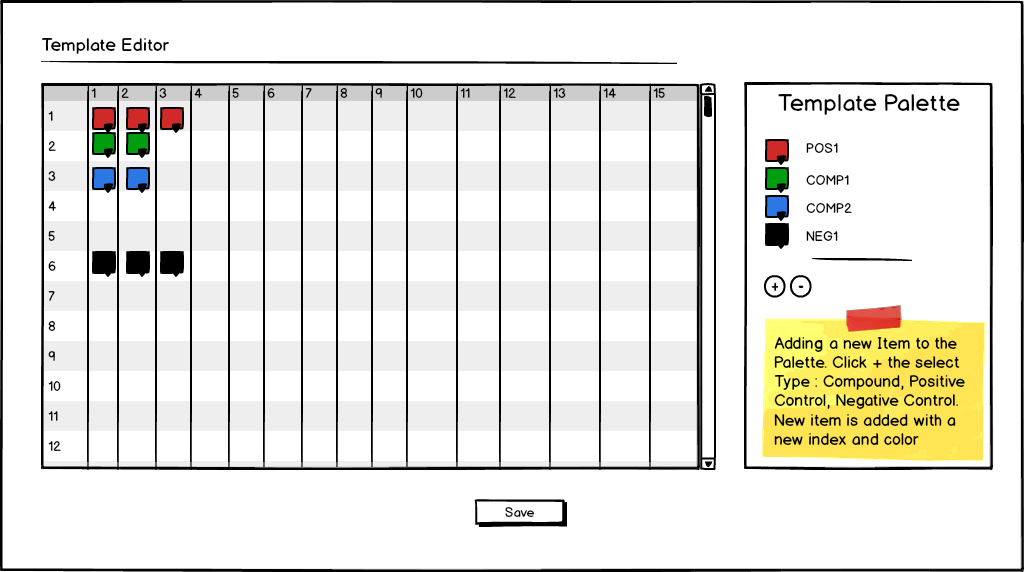
**Plate Templates**



Plates are created from plate Templates.

* Scientists can create new templates using a template editor (next diagram).
* Scientists can also create new custom (rectangular) plates with the plate editor.
* A more common approach may be to upload existing CSV templates into the system.

**Plate Template Editor**

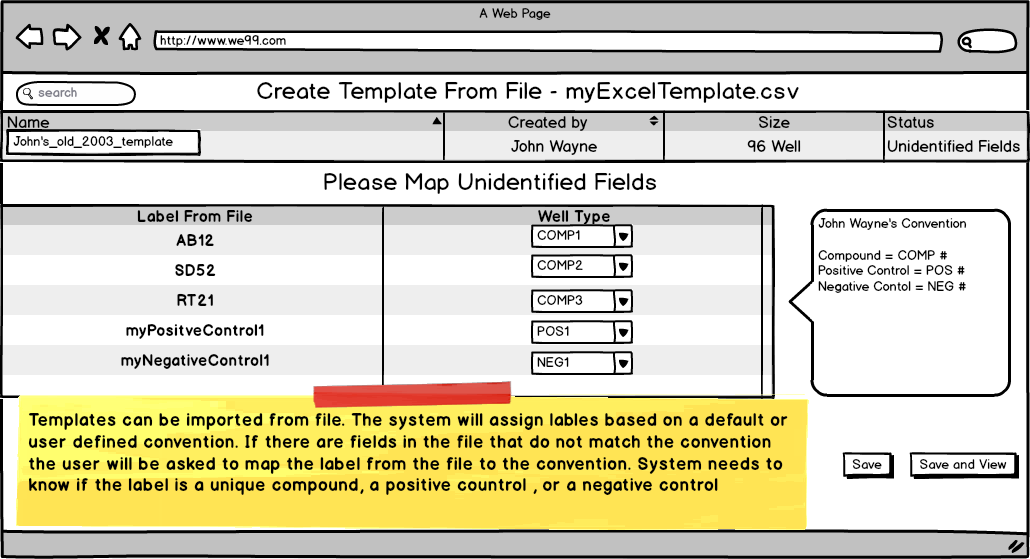


The template editor allows scientist to create new plate templates. To each template the scientist adds various compound labels, and label for positive and negative controls.

When adding a new compounds, positive controls, or negative controls the scientist selects the “+” button in the Template palette. The scientist needs to select if s/he is adding a compound, negative, or positive control. Based on the selection a new item is added to the palette. Items get a new label and color following a convention e.g.

* Compound: COMP{number},
* Negative Control: NEG{number},
* Positive Control: POS{number}.

The Scientist can select the color from the palette and add the item to a particular well on the template.



Another option for creating template will be to create a template from a CSV file. Scientists may have a number of templates for previous experiments in excel format that can be exported to a CSV file.

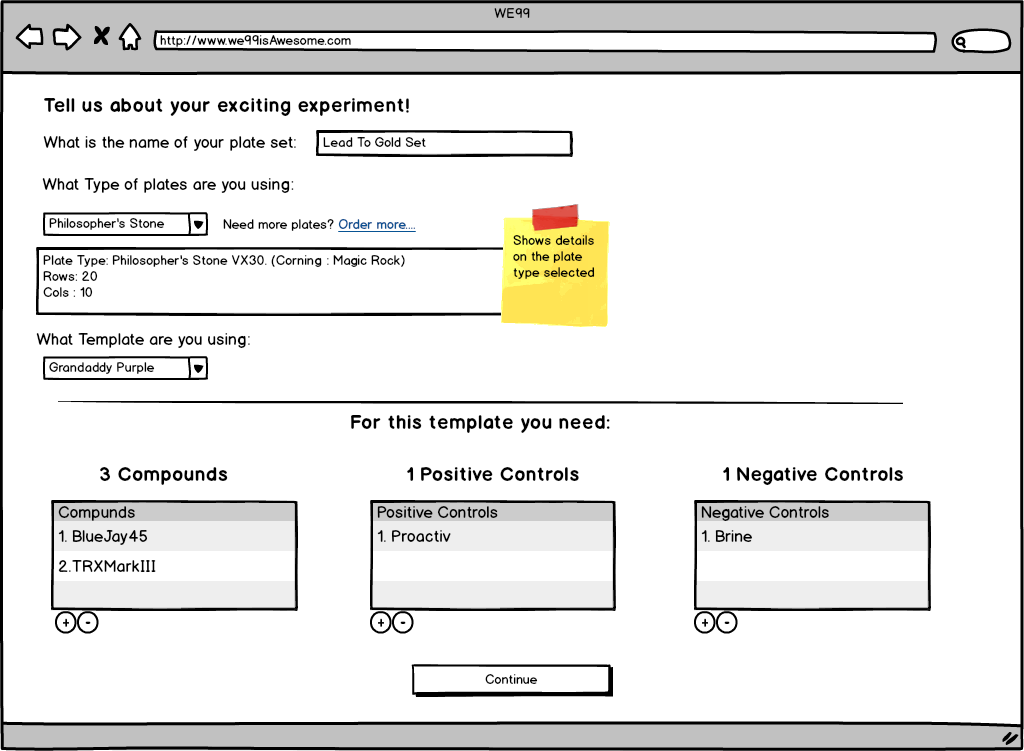
We99 propose importing the CSV file to create templates using a labeling convention. Templates need to identify unique compounds, positive controls, and negative controls. A sample convention could be as follows:

* Compounds: COMP{Number}
* Positive Controls: POS{Number}
* Negative Controls: NEG{Number}

If the CSV file contains labels following the convention the import will be extremely simple. All the Scientist would have to do is provide a name for the imported template and save it to the database.

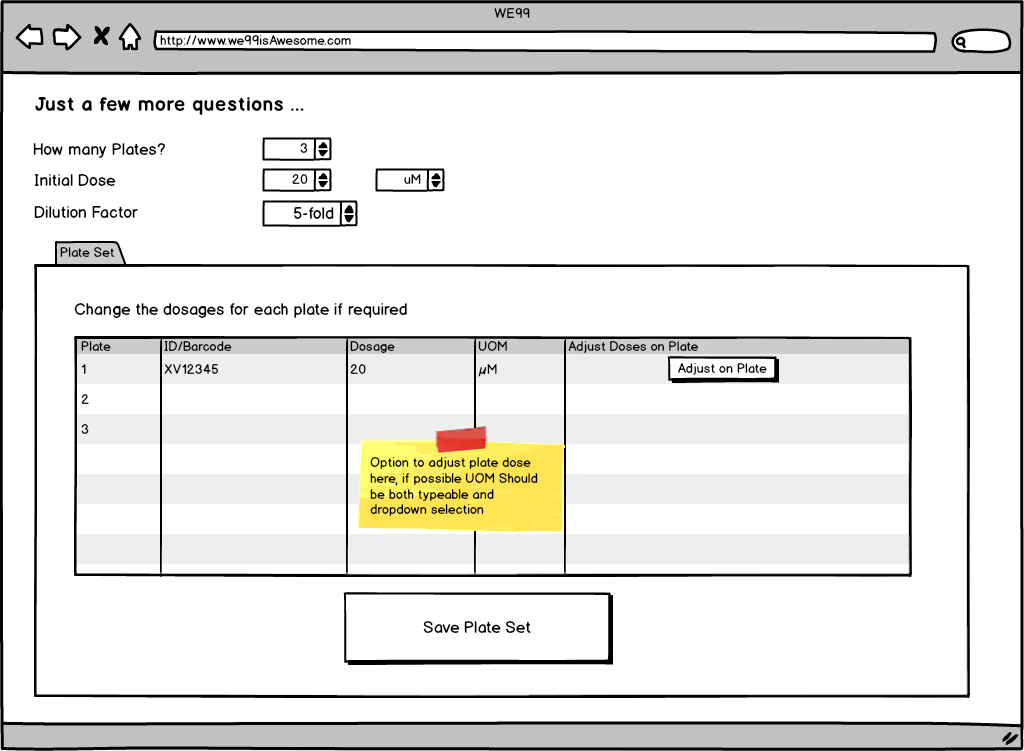
If the CSV file contains labels that do not follow the convention, then the Scientist will be prompted to map the unidentified fields to labels that follow the convention.

This concept could be extended to allow Scientists to have their own labeling convention for plate templates allowing them to import existing CSV templates easily.



The scientist then creates plates sets for the experiment. During this stage the Scientist needs to map the real compounds and controls that are to be on the plate to compound and control labels on the template.

* The scientist provides a name for the plate set.
* Then selects the type of plate to be used (see ordering diagram). The plate type contains details regarding the actual physical plate to be used e.g. manufacturer, material etc.
* The scientist then selects the template to be used for the plate set.
* The system will determine number of compounds, positive controls and negative controls that need to be specified for the plate.
* The scientist adds the compounds and controls using a dropdown or picker from available compounds and controls stored in the database.
* Once complete the scientist selects continue



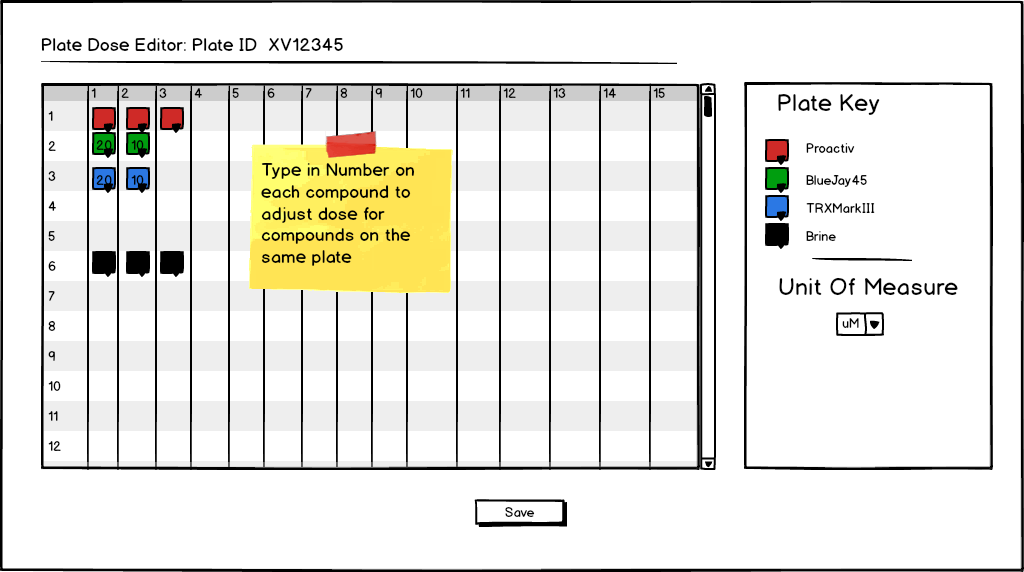
The next step is to apply the dosage of the compounds for the plate. There are two ways to create dose response plates in the system.

1. Multiple plates each plate with diluted compounds

* The scientist selects the number of plates.
* Then selects the initial concentration.
* Then specifies a dilution factor.
* The system generates plates with different concentrations of compounds on each plate
* The system will generate a unique ID for the plates, the scientist may override this to match any barcode ID that the physical plate may possess.

1. Single plate with multiple concentrations of compounds.

* The scientist can choose an initial does and no dilution factor
* The scientist can then select “Adjust on Plate” to access the Plate Dose Editor



* Using the plate dose editor the scientist can alter the concentration of each compound by changing the dosage value on each compound.

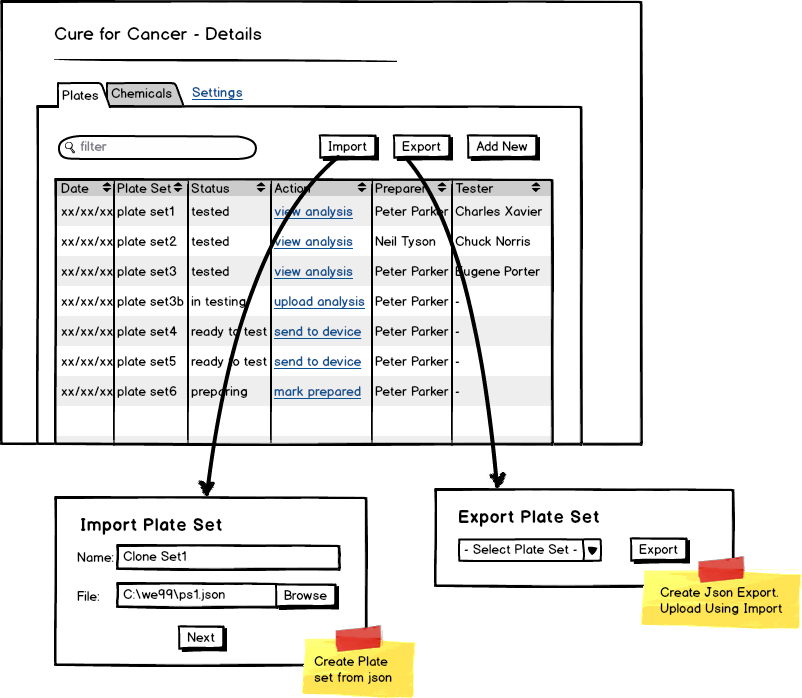
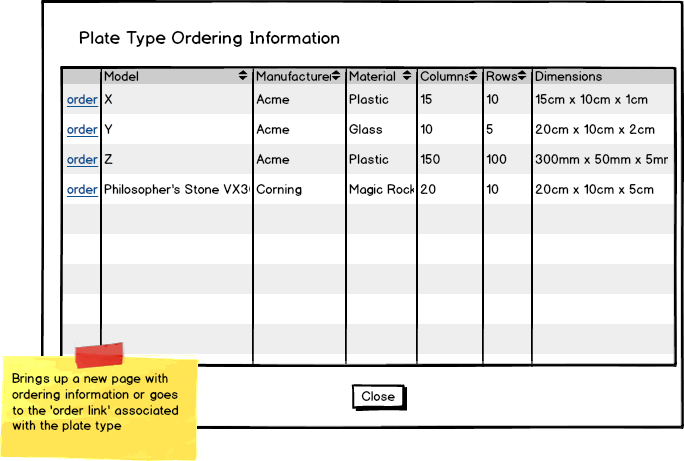


Plate Sets can be exported from the experiment manager into various formats.

This will allow the scientist to create the physical plate to the specification created in the system.



Information regarding the Physical plate inventory on hand is also stored within the system. Scientists may order more plates through external links.

### Open Issues / Questions Plate Management

| # | Description | NOtes |
| --- | --- | --- |
| 1 | Ability to create plates of any dimension. does not account for non-rectangular (e.g., round) plates or 3d plates. | Robert confirmed, this is NOT required |
| 2 | Exporting plate specs. Is there a standard format? Do we need to handle with different parsers tied to equipment type? |  |
| 3 | We should consider having add plate set wizard to produce a json document server side. then the import wizard and the add new plate wizard would both use the same call to populate the database |  |
| 4 | Concept of “Direction” for doses. Is this something we need to record for templates? |  |

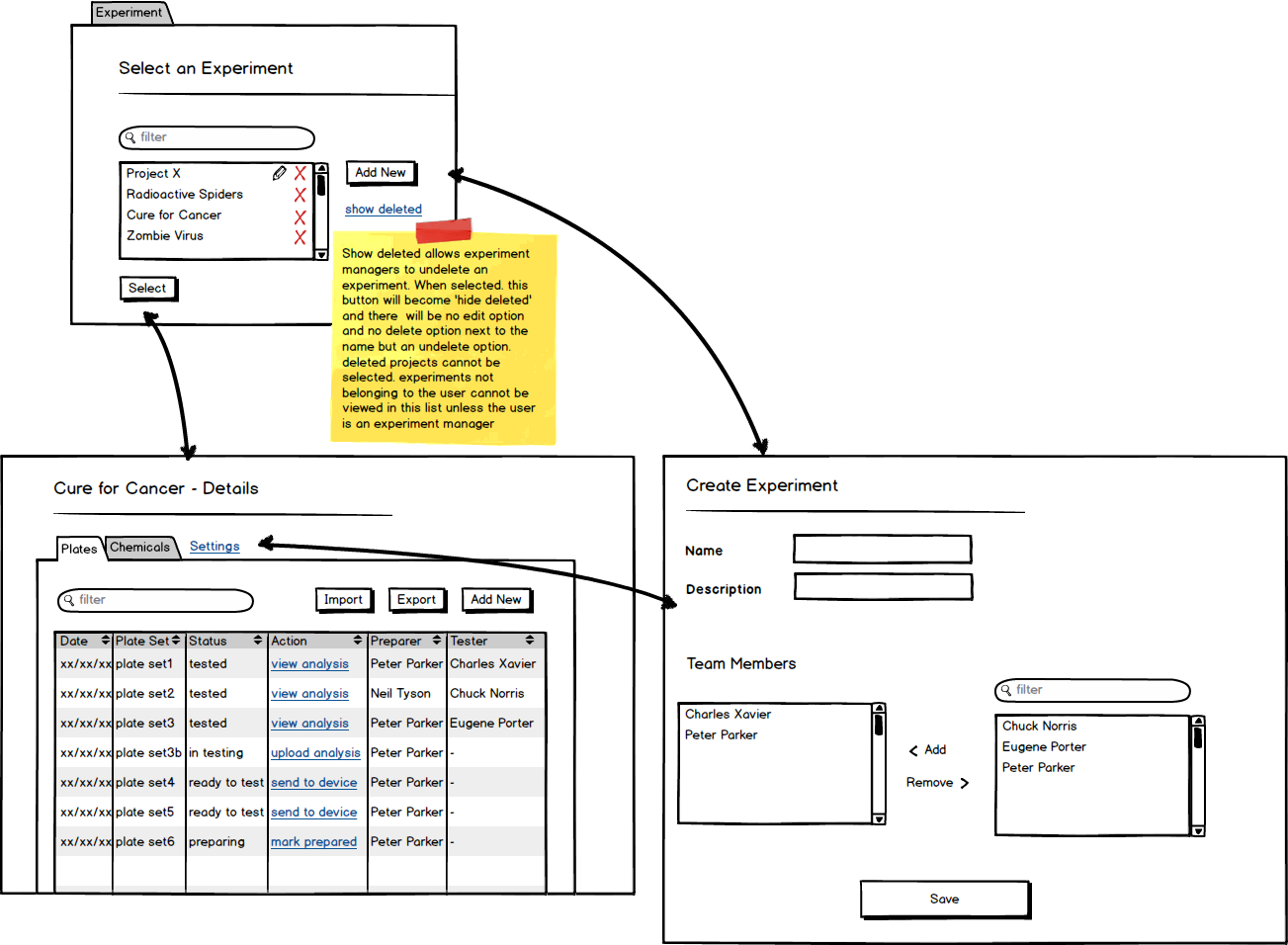
## Experiment Management Requirements

### Overview

Experiments can be created and deleted by an administrative user. An experiment is identified with a unique name. A description of the experiment can also be included. An administrative user can also create teams of scientists for an experiment.

Experiments may use a single of a set of filled plates. The plates used in an experiment can be listed. A team member can add new plates to an experiment.

### Sample Screens / Story Board



### Open Issues / Questions Experiment Management

| # | Description | NOtes |
| --- | --- | --- |
| 1 | Adding Compounds. Is there a need to do this in a bulk import? |  |
| 2 | note adding a team member removes them from the list of available team members |  |
| 3 | Might need Need a separate role and management view for ‘mark as prepared’. | e.g., this is where a lab assistant could mark plates as prepared and the status is updated to the project. |

## Results Analysis Requirements

### Overview

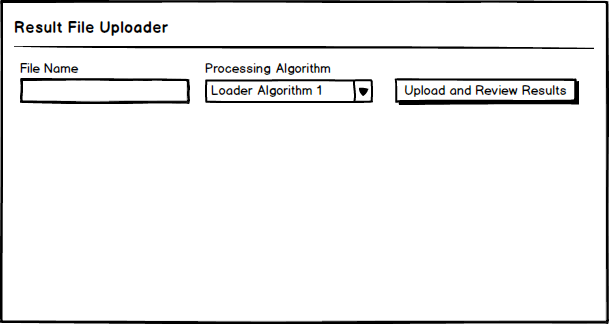
A plate will be analyzed by a machine - which will generate a structured text file as output. The user will upload the structured text file into the web application and select an appropriate parser to convert it into a result set. The results for the uploaded plates will be displayed to the user for validation in the form of a heat map.

While reviewing the results as a heat map the user will be able to read information about,

1. The results values.
2. The concentration of compounds in the wells.
3. The compounds in the wells.
4. What type of well’s are which (ex. EXP, POS, NEG).

### Sample Screens / Story Boards

Step 1 – Select a file and an algorithm to upload.



Step 2 – Review the results and save them if they look appropriate.

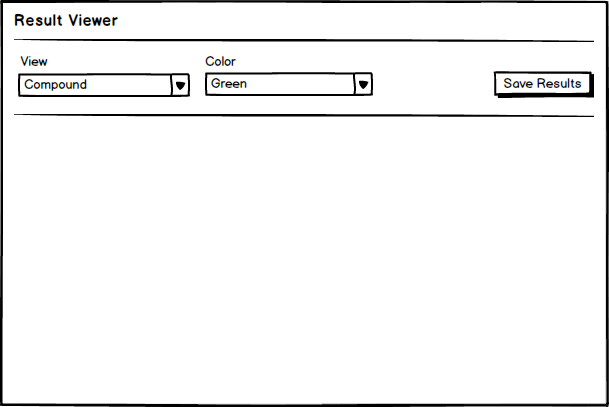


Plate 1

Plate 2

Plate 3

### Open Issues / Questions Results Analysis

| # | Description | NOtes |
| --- | --- | --- |
| 1 | When reviewing the result in the form of a heat map is there any workflow for processing the results beyond saving. For example -- if a plate looks like it has issues should the system allow them to flag the plate as problematic, etc? |  |
| 2 |  |  |
| 3 |  |  |

## Special Feature Proposals

### Overview

We have includes a few special features that we were considering implementing as part of the system.

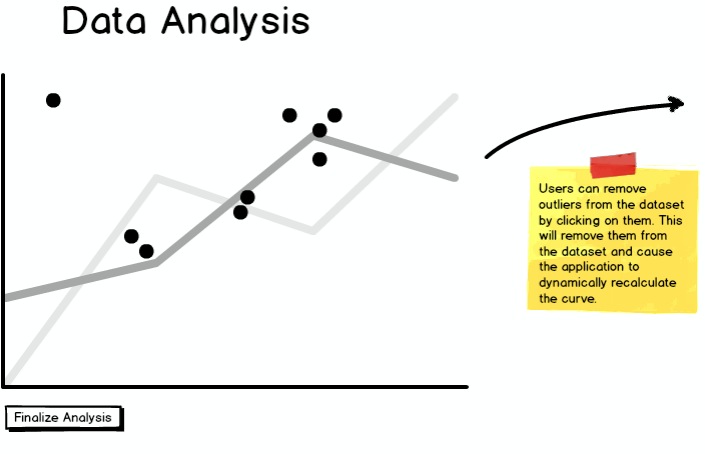
### Interactive Visualization

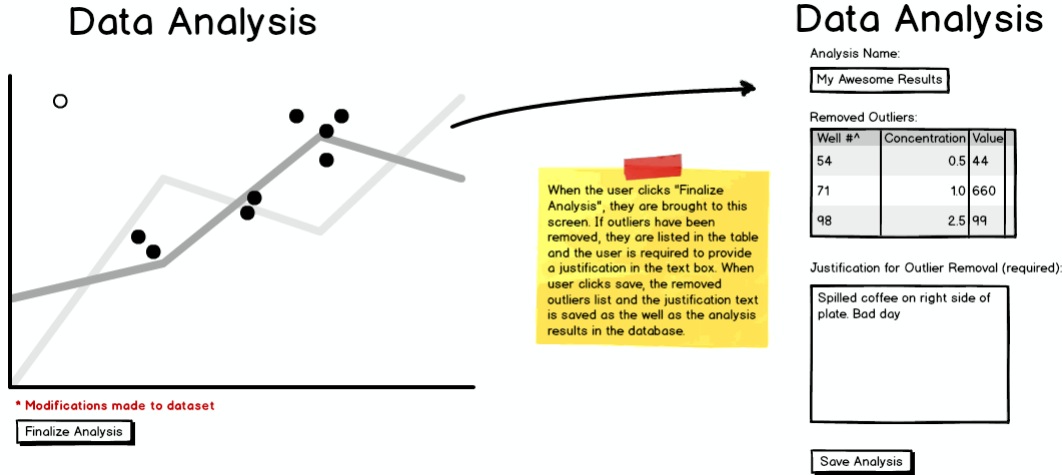
Allow the user to remove plot points from an Experiment's results. They will see the raw data from the processing in a scatter plot where we fit a curve to the data. The user can select points in the UI and remove them which will cause the graph to render the curve line again.

We will track which points were removed and enable users to provide comments as to why they were removed.

When removing points, you can offer a preview view. This would generate a new regression on the fly when certain points are selected to be removed. This can either be saved which would create the change and audit trail or the scientist could cancel changes.

#### Sample Screens / Story Boards



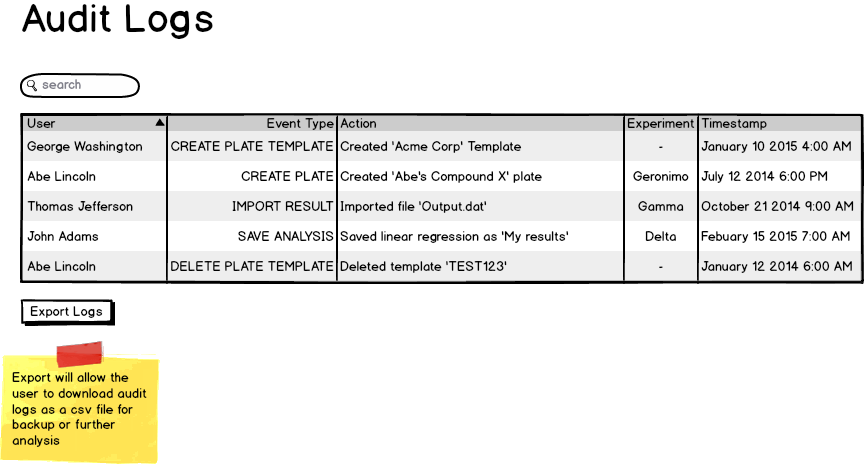


### Audit Trail

We could track whenever a Plate / Well / Dose / Experiment changes. This would allow users to know when something in the system has changed and may be necessary for internal reporting or perhaps regulatory agency requirements.

Assuming that we have some type of relational database, we should hook into the changes at the transaction level so the audit trail is captured in the same atomic operation as the data is changed. Note: there's a good library from Hibernate for this called Envers.

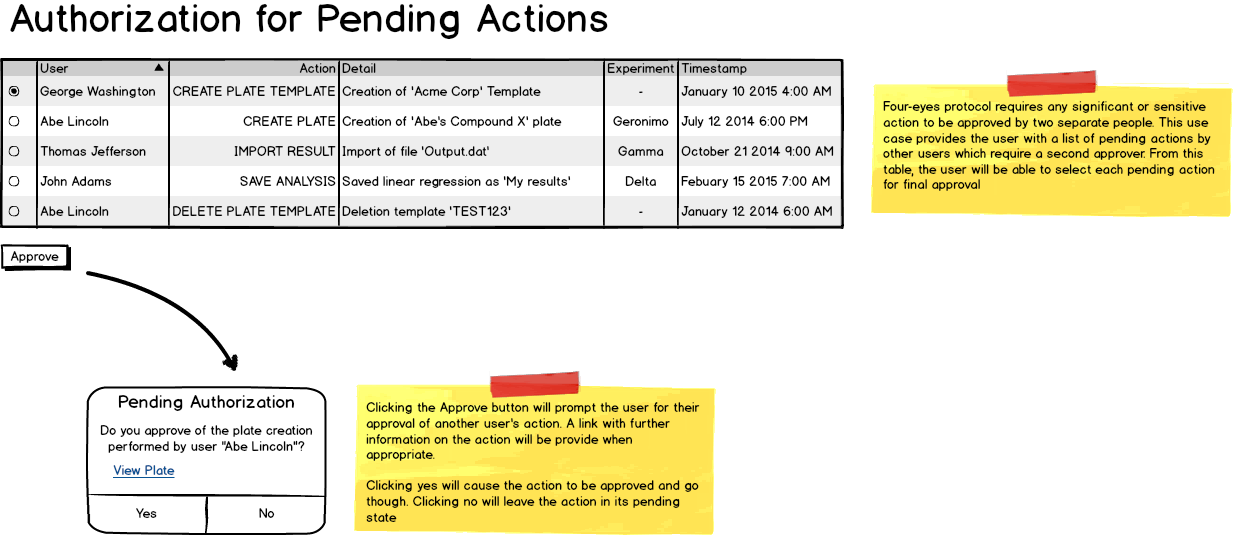
#### Sample Screens / Story Boards



### Four Eyes Protocol

Changes to experiment results or perhaps the plate configuration before an experiment should require a second person to review or approve the change. This would assume that we have an authentication model in place so we know which users own an Experiment and thus which users would be eligible to approve the change,

#### Sample Screens / Story Boards



### Flexible Plate Template Editor

Plate templates can be saved and loaded so that they can be shared. The save file is a clear format (e.g., csv) that could be edited directly with a text editor/excel. Alternatively the webapp interface can have a table that lets you design the plate.

### Plate Sharing

In some cases a plate may need to be shared across multiple Experiments. This may be a cost or resource issue. In order to support this, we need to associate wells to an experiment and not assume that a plate belongs to a single Experiment.

### Support Machine / Equipment File parsers

If we are storing data regarding the equipment used for an experiment we could possibly have the ability to customize output file formats for the type of equipment. This could also apply to parsing the results files from different machine types.

### Open Issues / Questions Special Features

| # | Description | NOtes |
| --- | --- | --- |
| 1 | For the data analysis portion of this they are represented as scatter plots. What should the x and the y axis of the scatter plot be? |  |
| 2 | For the data analysis portion how should the result be filtered. Would people want to look at a specific compound across an entire experiment? Only look at a single plate at a time? Review all plates in an experiment? |  |
| 3 |  |  |

# High Level Design

## Architecture

### Introduction

Our overall vision is to create a web application that can assist scientists with managing the data associated with their experiments. We have decided to develop our project as a Java web application. A web application will easily allow multiple users to access our program from whatever internet-capable device they have available, such a workstation or a mobile device. Our team has significant experience in the Java development environment, so choosing a Java-based framework was considered the most efficient choice.

### 

As a standard web app, we expect to see a Model, View, and Controller. This pattern is represented in the diagram where the model is the database that stores all of the data for the plates and the associated experiments. The controller part is the REST web services that provide all of the operations necessary to create and modify experiment data. Finally, the presentation layer is implemented in a HTML and Javascript and interacts with the controllers though standard AJAX/HTTP calls.

### Model

The data components in the application are modeled as plain Java Beans. The beans are named using domain in accordance with domain specific terms in order to better associate the data models with real world tasks needed to be done by the domain experts.

The domain model uses the standard JPA annotations in order to be able to create a declarative persistence layer for the application. While this persistence layer is primarily focused on a relational database implementation, it's not strictly required. It's conceivable for the customer to swap the persistence layer out at a future point in favor of another implementation without having to undertake a complete rewrite. This is the declarative nature of the persistence model. At best, it is loosely coupled to a relational database.

### Controller

The controller portion of the application is implemented via REST services. The service classes are implemented using the JAXRS standard specification. As such, they will run in any JAXRS compliant container. Furthermore, the services can be exercised by any REST client capable of producing the documented JSON and sending it to the documented URI's.

One added benefit of using the standard JAXRS annotations is that we can leverage the capabilities provided by one popular implementation (Apache CXF) in order to provide strongly typed client API's in order to assist with future integrations. Obviously future integrators are free to choose any library that suits their needs, but if they happen to choose Apache CXF or a similarly compatible library, then they can use our interfaces directly in order to generate dynamic proxies to invoke our services.

### View

The view component of the application is an application based on Angular JS. This framework is an open source HTML 5 framework from Google. The application will have knowledge of the set of permissions in the application and render the UI accordingly. It will exercise the service components and allow the user to interact with the plates and experiment results. All of the interactions from the client to the server will be accomplished through the standard REST service interfaces.

### Components

Both the front-end and back-end are design by contract. The REST services provide sufficient annotations and documentations to enable the implementations of a full application. The front-end will extract each of its service calls to a well defined service interface that could be swapped for a different implementation.

### Database

The default database vendor is the H2 project. This is a SQL 92 compliant database that has wide support across major libraries like Hibernate and the Java Persistence Architecture. The key thing to consider with a database vendor is whether you're locked into the database vendor through some vendor specific API. Specifically, is there some touch point with the vendor where the host application is interacting with the DB host in an interface that is not part of a standard interface like JDBC or JPA.

Being vendor agnostic is a key design principle with WE99. We'll avoid any vendor specific queries or other enhancements. No vendor specific SQL, no stored procedures, and no assumptions about the platform executing our queries. This vendor agnostic layer is provided largely for free via the Java Persistence Architecture API. We can encode our DB access into our entities via annotations and an abstract query language that has late binding to a vendor specific query language at running.

The initial deployment of WE99 is designed to scale vertically as opposed to horizontally. This could change by rethinking the persistence layer to introduce an abstraction layer for JPA. This indirection would make it easier to swap out a traditional RDBMS with a NoSQL approach if needed. While there's still some work to do to support this, it's isolated to the service implementations and not part of the core interface with the presentation layer.

# Project Planning and Estimating

## Iterations

### Overview

We99 has planned the project based on the Agile Methodology. The project is broken down into Iterations. The We99 team has planned 10 iterations, each iteration being 1 week in duration. The iteration starts on the Thursday of each week and closes on the following Wednesday. Every iteration the project team will be delivering several units of functionality defined by stories.

### Iteration Calendar

The table below specifies the iteration calendar for the We99 Team

|  |  |  |  |
| --- | --- | --- | --- |
| **Iteration Number** | **Open Date** | **Close Date** | **Notes** |
| 1 | March 5th 2015 | March 11th 2015 | Milestone1  Presentation March 5th |
| 2 | March 12th 2015 | March 18th 2015 |  |
| 3 | March 19th 2015 | March 25th 2015 | Spring Break March 15th-22nd |
| 4 | March 26th 2015 | April 1st 2015 |  |
| 5 | April 2nd 2015 | April 8th 2015 | Milestone 2  Presentation April 9th |
| 6 | April 9th 2015 | April 15th 2015 |  |
| 7 | April 16th 2015 | April 22nd 2015 |  |
| 8 | April 23rd 2015 | April 29th 2015 |  |
| 9 | April 30th 2015 | May 6th 2015 | Milestone 3  Presentation May 7th |
| 10 | May 7th 2015 | May 13th 2015 | Final Report May 9  Presentation Faculty Committee May 14 |

## Stories and Estimates

The We99 are using Atlassian JIRA to manage and track project stories. <https://westeast99.atlassian.net/secure/Dashboard.jspa>.

As of march 2nd, The team have 30 stories in the backlog. Each story is sized with an attribute called story points. Each point represents a team man hours. The initial rough estimates predict that the team will require 360 points (man hours) to complete the project. Our initial estimates are rough, so doubling the estimate out estimated (man hours) for the project is 720 Hours. The team has a capacity of 750 hours for the project.

The We99 team predicts that they should be able to complete the scope of work for the project within the allotted timeframe.

At the close of every iteration the team will re-compute its “velocity”. That is the number of story points it is averaging per iteration. This is an iterative process and the team will adjust its estimates accordingly.

## Main deliverables

Milestone 2 – April 9th. Close of iteration 5

* Predicted Functionality completed
* Domain Model Completed
* Web Service Interfaces specified
* Web Services Implemented for Plate Editor
* Web Services Implemented for Plate Quality Control
* Web Services Implemented for
* Plate Template Editor for the creating of Dose Response Plates
* Front End Quality Control of Plates

Milestone 3 – May 7th. Close of iteration 9

# Project Risks

## Risks Identified

# Appendix Story Backlog March 2nd

|  |  |  |
| --- | --- | --- |
| **Issue Type** | **Key** | **Summary** |
| Story | [WE-17](https://westeast99.atlassian.net/browse/WE-17) | Update Report and Presentation with Architecture and Web Service Info |
| Story | [WE-4](https://westeast99.atlassian.net/browse/WE-4) | Import CSV Plate Maps |
| Story | [WE-19](https://westeast99.atlassian.net/browse/WE-19) | Update Report and Presentation with Front End Stack |
| Story | [WE-18](https://westeast99.atlassian.net/browse/WE-18) | Update Report and Presentation with Results Analysis Requirements |
| Story | [WE-20](https://westeast99.atlassian.net/browse/WE-20) | Update Report and Presentation with details of the Plate Map Editor |
| Story | [WE-3](https://westeast99.atlassian.net/browse/WE-3) | Basic LogIn Logout |
| Story | [WE-6](https://westeast99.atlassian.net/browse/WE-6) | Implement Standard Equations for Dose Response Curves |
| Story | [WE-7](https://westeast99.atlassian.net/browse/WE-7) | Front End - Support User-Specified fit Equations for Dose Response curves |
| Story | [WE-8](https://westeast99.atlassian.net/browse/WE-8) | BackEnd Services to support HeatMaps for plate results |
| Story | [WE-11](https://westeast99.atlassian.net/browse/WE-11) | Back End Services to Compute Z and Z' for QC'ing Plates |
| Story | [WE-12](https://westeast99.atlassian.net/browse/WE-12) | Plate Map Editor |
| Story | [WE-13](https://westeast99.atlassian.net/browse/WE-13) | Import and parse test results |
| Story | [WE-14](https://westeast99.atlassian.net/browse/WE-14) | Design Domain Model |
| Story | [WE-15](https://westeast99.atlassian.net/browse/WE-15) | Define Web Service Interfaces |
| Story | [WE-24](https://westeast99.atlassian.net/browse/WE-24) | BackEnd Results Parser - to parse and store result files |
| Story | [WE-25](https://westeast99.atlassian.net/browse/WE-25) | Front End Results File picker - select results file to parse |
| Story | [WE-26](https://westeast99.atlassian.net/browse/WE-26) | Front End - Display Single Heat Map for a Single plate |
| Story | [WE-27](https://westeast99.atlassian.net/browse/WE-27) | Front End - Display Multiple Heat Maps for Experiment |
| Story | [WE-28](https://westeast99.atlassian.net/browse/WE-28) | Front End - Display Single Dose Response Curve |
| Story | [WE-29](https://westeast99.atlassian.net/browse/WE-29) | Front End - Knock out Values on Dose Response Curve and Refit |
| Story | [WE-30](https://westeast99.atlassian.net/browse/WE-30) | Front End - Display plate statistics, and control well statistics across all the plates for all teams |
| Story | [WE-31](https://westeast99.atlassian.net/browse/WE-31) | Back End - Support Services for plate statistics, and control well statistics across all the plates for all teams |
| Story | [WE-32](https://westeast99.atlassian.net/browse/WE-32) | Front End - Support Combination views of Dose response curves |
| Story | [WE-33](https://westeast99.atlassian.net/browse/WE-33) | Front End - Specify values for fit parameters for all or a Selection of Dose response curves |
| Story | [WE-34](https://westeast99.atlassian.net/browse/WE-34) | Front End - Select %effect ranges for to plot non, partial and full responders Dose response curves |
| Story | [WE-35](https://westeast99.atlassian.net/browse/WE-35) | Back End - Store Raw Results file for reproduction of analysis |
| Story | [WE-36](https://westeast99.atlassian.net/browse/WE-36) | Back End - Define negative and positive controls so all normalized results are in terms of a percent effect |
| Story | [WE-37](https://westeast99.atlassian.net/browse/WE-37) | Back End - Store control data |
| Story | [WE-38](https://westeast99.atlassian.net/browse/WE-38) | Back End - Store SVG plots transforms and meta data for plots |
| Story | [WE-39](https://westeast99.atlassian.net/browse/WE-39) | Back End - Audit and store all Knock out values during dose response analysis |