Pivotal Tracker Kanban Prototype

COLORADO SCHOOL OF MINES 2017 FIELD SESSION

Ann Gustafson | Emily Dederick | Christopher Bonin | Gerald Ung

CLIENT

Morgan Whitney

Table of Contents

1. INTRODUCTION

1.1 Client Description

Pivotal is a software and services company with a focus on increasing the effectiveness of other companies through agile development practices. Pivotal Tracker is a project management application designed for engineers. It allows for increased collaboration between coworkers and the productivity of the team by having a log of all tasks, called stories, that the team must complete. Prior to beginning any story, the team allocates points to each story, declaring how difficult each seems. If any story is pointed as a five or higher, utilizing Tracker's Fibonacci pointing system, the team discusses how to break the story down even more to make it more convenient and efficient for the team. Using the point system and a velocity tracker, Pivotal Tracker can show how long the project is expected to take and can show which tickets can be completed within a week. If the team's velocity increases or decreases, Pivotal Tracker reflects these changes accordingly.

As we can see in Figure 1, a Kanban board allows for a visual representation of the task at hand and allows other team members to see what work needs to be done before other tasks can be completed. Pivotal would like to incorporate a Kanban Prototype into their Pivotal Tracker system to allow teams to visually see the work being done, ensuring that work flow is occurring smoothly, and that the teams are continuously improving as time goes on.

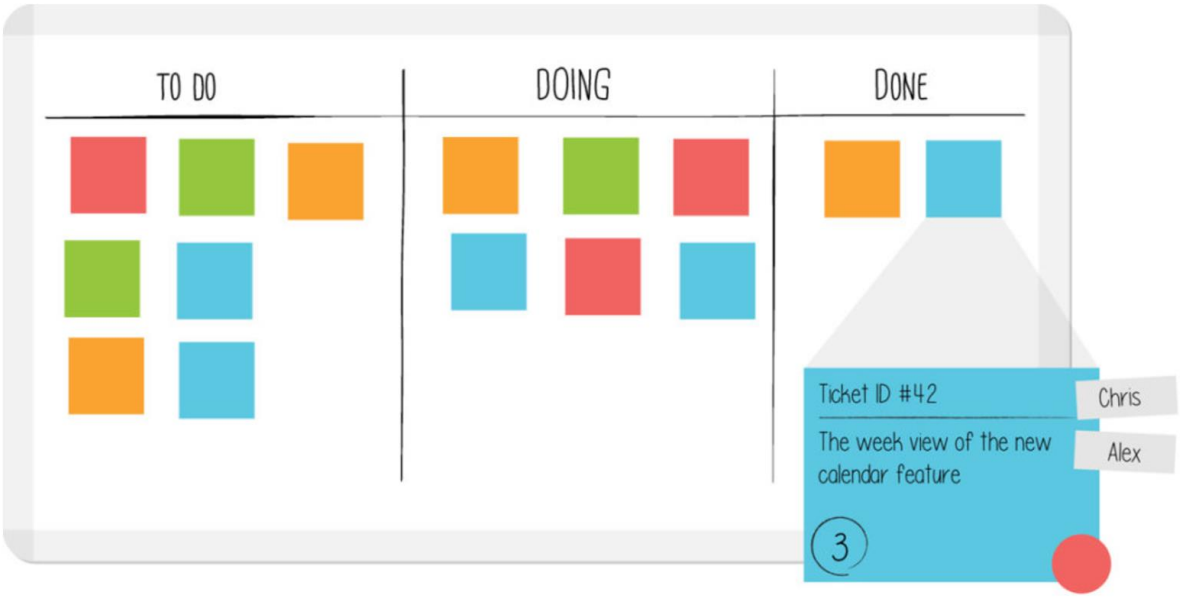

Figure 1: A Kanban board is typically performed on a whiteboard, where it is easy for team members to move a card from one column to another. Cards are usually color-coordinated showing the importance of each story and contains a description of what needs to be done.

1.2 Product Vision

Pivotal has asked us to create a Kanban Prototype for Pivotal Tracker that is easier for non-engineers to use, as seen in Figure 2. Ideally, this single-page web application would be used by Pivotal's

employees looking for a less cluttered and more simple version of Pivotal Tracker with the requirement that the Kanban Prototype be as simple and user-friendly as possible.

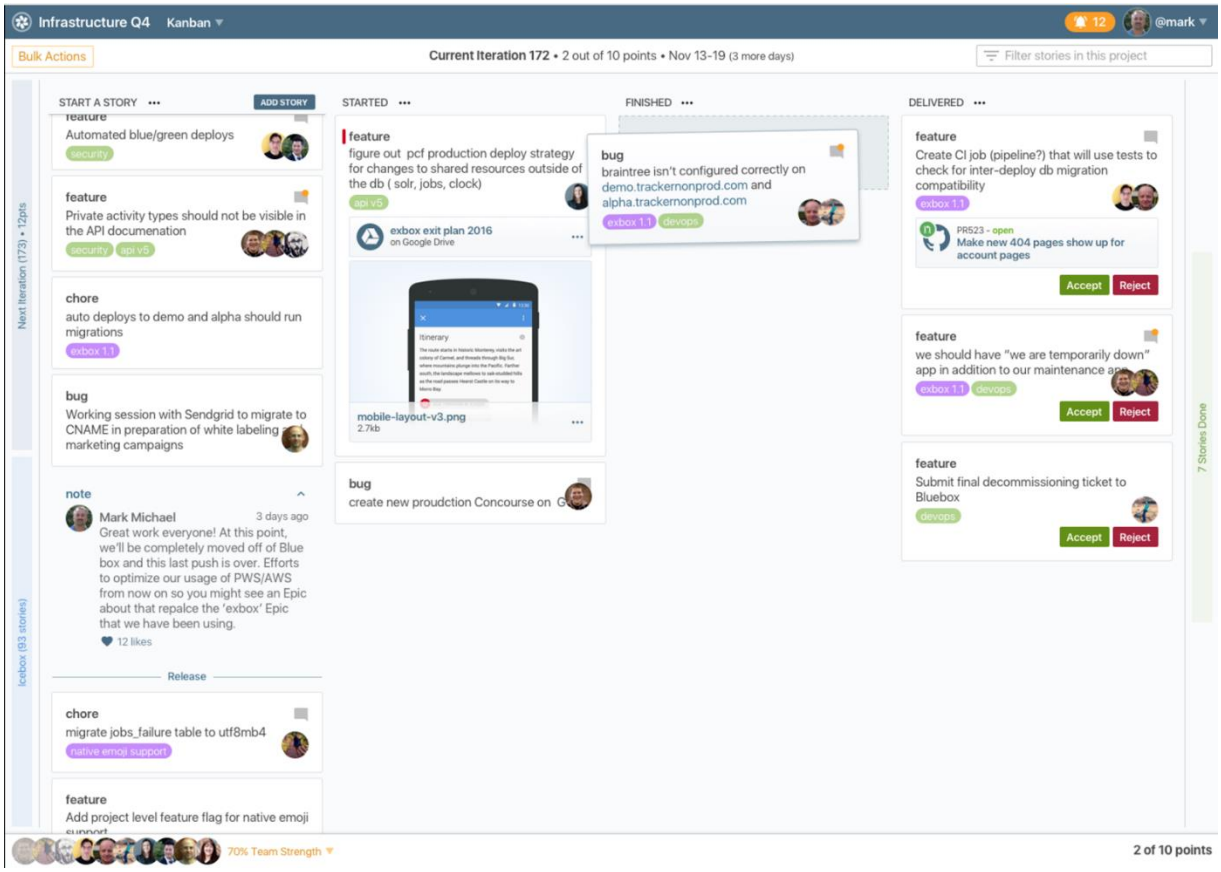

Figure 2: This is a visual representation and design scheme of what the client had in mind at the beginning of the project. This view is much more user-friendly compared to the traditional Pivotal Tracker view.

All the states used in Tracker are translated into the Kanban Prototype by changing the states to *Ready*, *In-Progress*, *Finished*, *Delivered*, and *Done*. Located in the "*Ready*" column is the *unstarted* and *rejected* stories. At any time, users can pick a story from the "*Ready*" column and move it to the "*In-Progress*" column by clicking a start button. Unlike traditional Kanban boards were users must wait for others before they can progress, this Kanban Prototype allows multiple users to work in conjunction with one another with real-time updating. This allows users to make changes in either the Kanban Prototype of Pivotal Tracker and have the update appear on both sites instantaneously. Once a story is completed, users can then move it to the "*Finished*" column where they can evaluate their progress before moving the card to the next column.

When the story card has been moved to the "*Delivered*" column, project leaders can choose to either accept or reject the story. If accepted, the story card is moved into the "*Done*" column; however, if the story is rejected, it gets a label saying that it has been rejected and is moved into the "*Ready*" column to be reviewed.

Users of the Kanban Prototype can create custom columns by providing a column name, state, label, and position using the custom column form. These columns can be set to have a maximum limit of how many stories can be shown, allowing the screen to be less cluttered and distracting for users. Users can change the position of the custom columns in the Kanban Prototype as well as edit the column information to the user's preference. Stories can move freely amongst the static and custom columns, changing to the proper states to appear in the given columns.

The client also wanted the Kanban Prototype to talk with Pivotal Tracker. This direct communication back and forth between the two would allow for real-time updating. When a change was in one, it was reflected in the other.

Ideally, the story cards would be clickable and once clicked on, would provide further information about the story, such as a brief description as well as seeing comments made by other users. Out client deemed these extra story features as optional and would have been included in the Kanban Prototype once the required portion of the project was functioning and accepted.

2. REQUIREMENTS

2.1 Functional Requirements

Our goal for this project, as stated by the client, is to build a functional single web page application. The client has requested that the following be present in the application:

1. Be able to login with Google's API.
2. Be able to input the API key from Pivotal Tracker to see a list of projects linked to the user's account.
3. Be able to create custom columns.
4. Be able to edit custom columns.
5. Be able to delete custom columns.
6. Be able to move user story cards from one column to another.
7. Be able to see real-time updates on both Pivotal Tracker and the Kanban Prototype.
8. Be able to set column limits on custom columns.
9. Be able to move user story cards to and from custom columns.
10. Be able to click the user story card to see more information related to the story (optional requirement).
11. Be able to add labels and place flags on user story cards (optional requirement).

2.2 Non-Functional Requirements

We were not required to use any specific languages for this project, but the client did suggest we use certain tools such as:

1. ReactJS for front-end development since Pivotal uses ReactJS for their front-end development.
2. MongoDB as the database as it is provided by Pivotal Cloud Foundry.
3. Ruby on Rails for back-end development which is supported by Pivotal Cloud Foundry.
4. Google API login to keep track of user information and for added security.

3. SYSTEM ARCHITECTURE

In developing this single web page application, we learned that the application would need to be able to deploy to Pivotal Web Services (PWS) and more specifically their Cloud Foundry. Based on the PWS website, we discovered that it was best if we used Ruby on Rails for our backend, and used MongoDB for our database as we could connect the two using gems. For the front end of the application, we used ReactJS to display information and functionality to the user.

To ensure the client that we were making significant progress, we would upload and deploy a working version of the Kanban Prototype to PWS (http://tracker-kanban.cfapps.io). In addition to uploading a working version to PWS, we would get feedback from the client allowing us to make any corrections necessary.

For real-time updating to occur, the Kanban Prototype needs to continuously communicate with Pivotal Tracker, allowing changes made in either application be reflected in the other. To do this, we made the client side do as little work as possible and forced the database as well as the Rails application to do most of the work. In order for this relationship to work, the Rails application would talk to both the Tracker API and MongoDB. Information from the Tracker API would be sent to the client after it had been filtered via the Rails application. This filtered data would also be sent to the database, allowing for faster rendering times.

ReactJS can utilize the data in conjunction with JavaScript to create a visual representation for the user. For this to happen, ReactJS uses JavaScript components to create DOM objects, which are displayed when the web page is loaded. One of the qualities ReactJS is that it will render only the components that change. This is beneficial to us as the states of the cards will change at any time and instead of re-rendering the entire web page, it would be best to just render the cards that have changed states.

Prior to the implementation of ReactJS, users would have to forcibly refresh the webpage to see if any changes happened; however, with ReactJS, we could allow the web page application to refresh automatically when a change is made. The idea to allow the page to refresh automatically comes from the fact that Pivotal Tracker updates automatically when a change is make, alerting all other members of the project.

4. TECHNICAL DESIGN

The design of the Kanban Prototype can be broken down into three components: Pivotal Tracker integration, the Pivotal Tracker API, and the web application. The user first interacts with the Rails application which, at sign-in, speaks with Google's API. Once a user is logged into the Kanban Prototype, the Mongo database is used to store the user's session information. Every time a user logs into the Kanban Prototype, their username, typically their email address, and their Pivotal Tracker token is stored in the database. If the user fails to provide their token are enters it incorrectly, they are promptly directed to enter their token. The Rails application uses the token value to retrieve and present a list of projects that the user is assigned to. Once a token is saved to the user profile, they can click on any of their assigned projects and see it in the Kanban Prototype view. The inner workings of this process are displayed below in Figure 3.

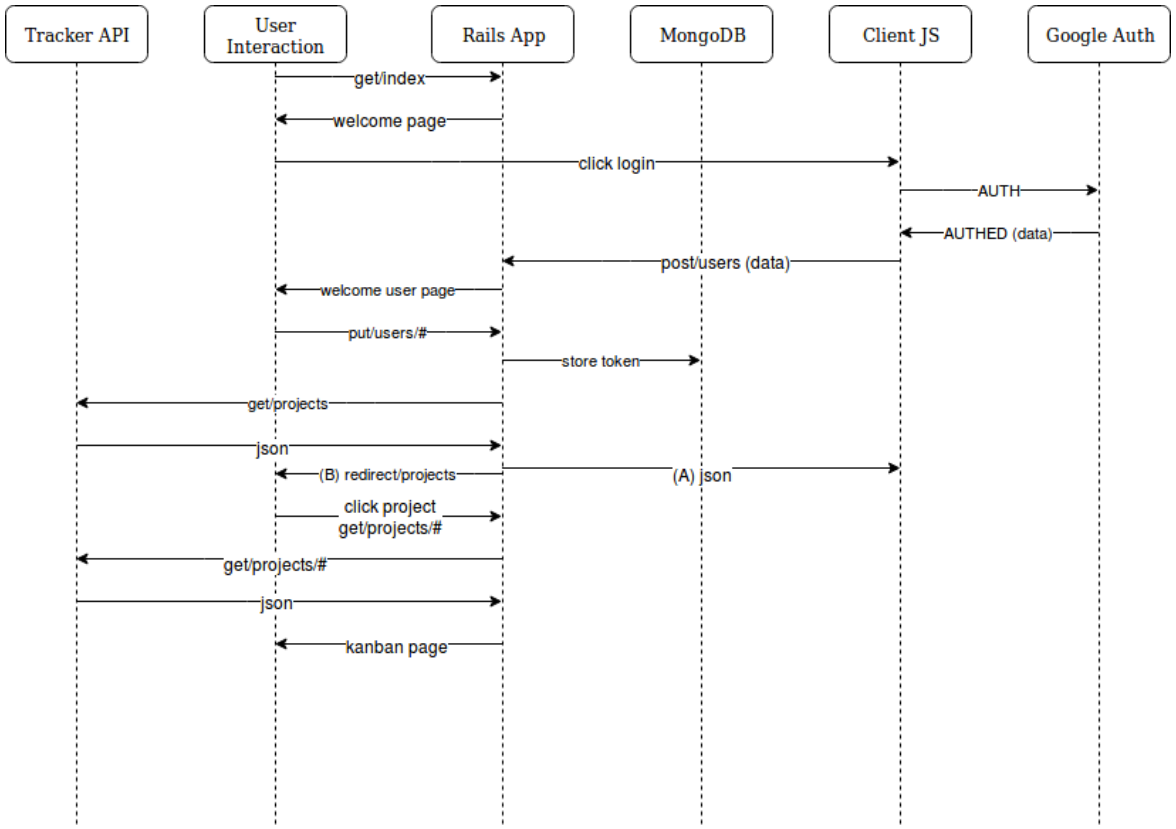

Figure 3: This is a diagram illustrating the flow of communication between the elements involved in the application. This was provided by the client to explain his thoughts on how the application should be set up.

The web application is statically generated according to Rails, and as stated earlier, any updates to the page is controlled by ReactJS. The incorporation of ReactJS allows us to render objects that have been changed, allowing the user to see these changes without having to reload the entire page. The Rails application uses Ajax calls to the database to see the changes that are important to the user. Figure 4 shows how the components should communicate with one another.

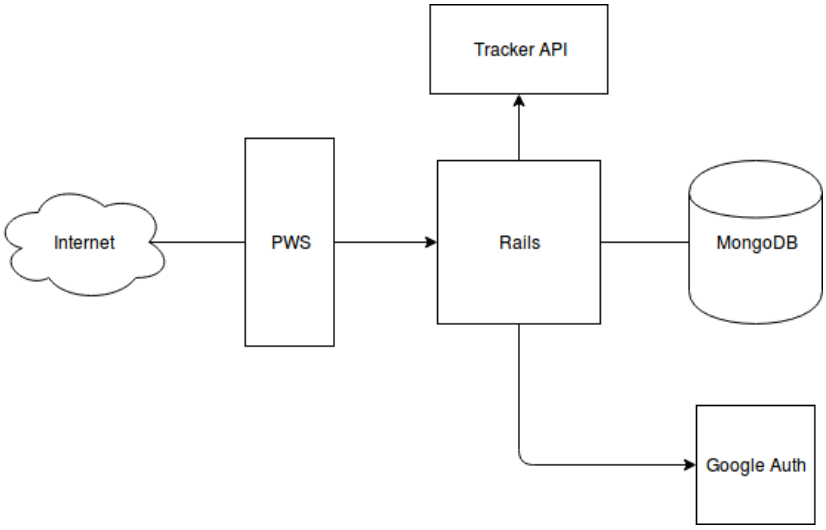

Figure 4: This is a visual representation of how the Rails application communicates with all the components

Technical Problems:

- Syncing the database.
- Hiding the sign-in and sign-out buttons
- Retrieving the Pivotal Tracker token
- Parsing the API token json data to show all projects.

5. DESIGN DECISIONS

5.1 Languages

We used Ruby on Rails for our backend to make our system easy for Pivotal to merge into its system later should they choose to do so. Using Ruby on Rails over other backend languages like Python or JavaScript was suggested to us by the client, as they have created similar projects to ours using Ruby on Rails with little difficulty.

The frontend of our Kanban Prototype was written in ReactJS as it provides an excellent interface for single-page applications. The client wanted the application to be very smooth in use to make the website feel like a live application instead of a static website. Pivotal also has its whole application written in ReactJS, making our Kanban Prototype easy to incorporate into their website if so desired.

5.2 Database

We chose to use MongoDB as our database after also looking at SQL. We chose to use a documented-oriented database rather than a relational database because we needed a database that could handle repeated data, in which a relational database would not handle. The MongoDB sandbox is also free to use and is supported by Pivotal Cloud Foundry, helping us choose our database.

6. RESULTS

The goal of this project was to create a single web page application in the style of a Kanban Board to increase productivity while using Pivotal Tracker. Our application has been tested and works well in the current versions of Google Chrome, Firefox, and Safari and works will on both desktops and mobile devices. We did not have time to implement a clickable card to show more story information or allow users to configure multiple flags on the cards. After voicing our concerns about the amount of time left on the project, our client made the executive decision to make the deployment of flags and card information optional.

We also could not implement drag and drop into our project because of compatibility issues associated with the gem used to mount ReactJS to Ruby on Rails. We attempted to implement drag and drop through the react-rails gem and through HTML5 but neither worked with our existing code. Rather than rewriting our project in a new language to use drag and drop, we opted to add buttons on the cards that allowed them to move left or right. There were other features we would have loved to incorporate into our project, such as editing story cards and adding comments; however, we did not have time to include them. Overall, we have a working product that can easily be integrated into Pivotal Tracker's current site if desired to give users an efficient yet less complex project-planning experience.

We learned through this experience the importance of planning for the entire project in the beginning stages of a project. Had we done some research on how to mount our front-end and back-end languages together early in the planning stage, we would have known which gems allow for relatively simple drag and drop integration or decided that drag and drop would be too time consuming, giving us time instead to try to implement other features. We discovered within ourselves that we could learn new languages in a short time frame, especially with a fast-paced project such as this one. No team member had been exposed to either ReactJS or Ruby on Rails but we were able to learn just enough to get through the project by utilizing online tutorials.

7. APPENDICES

7.1 Additional Figures

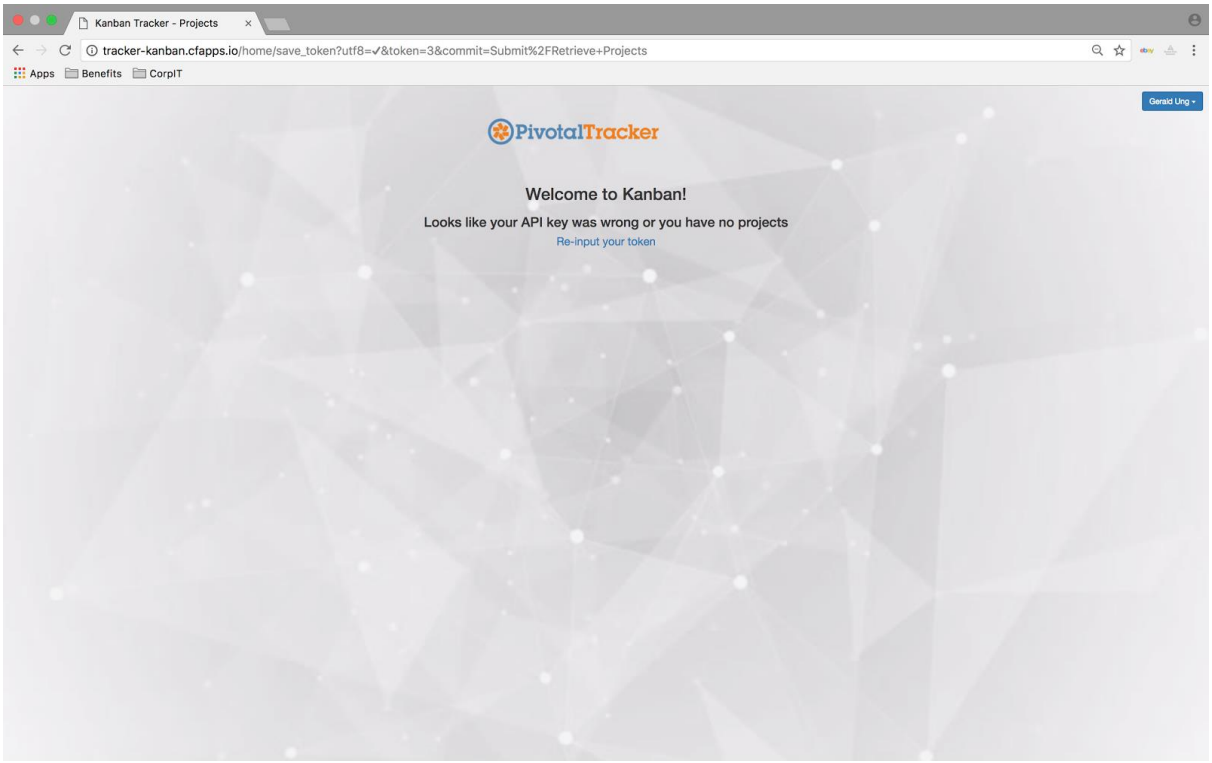

Figure 5: If the user enters a wrong token value, clears their token value, or is not assigned to any projects, they are redirected to this splash screen.

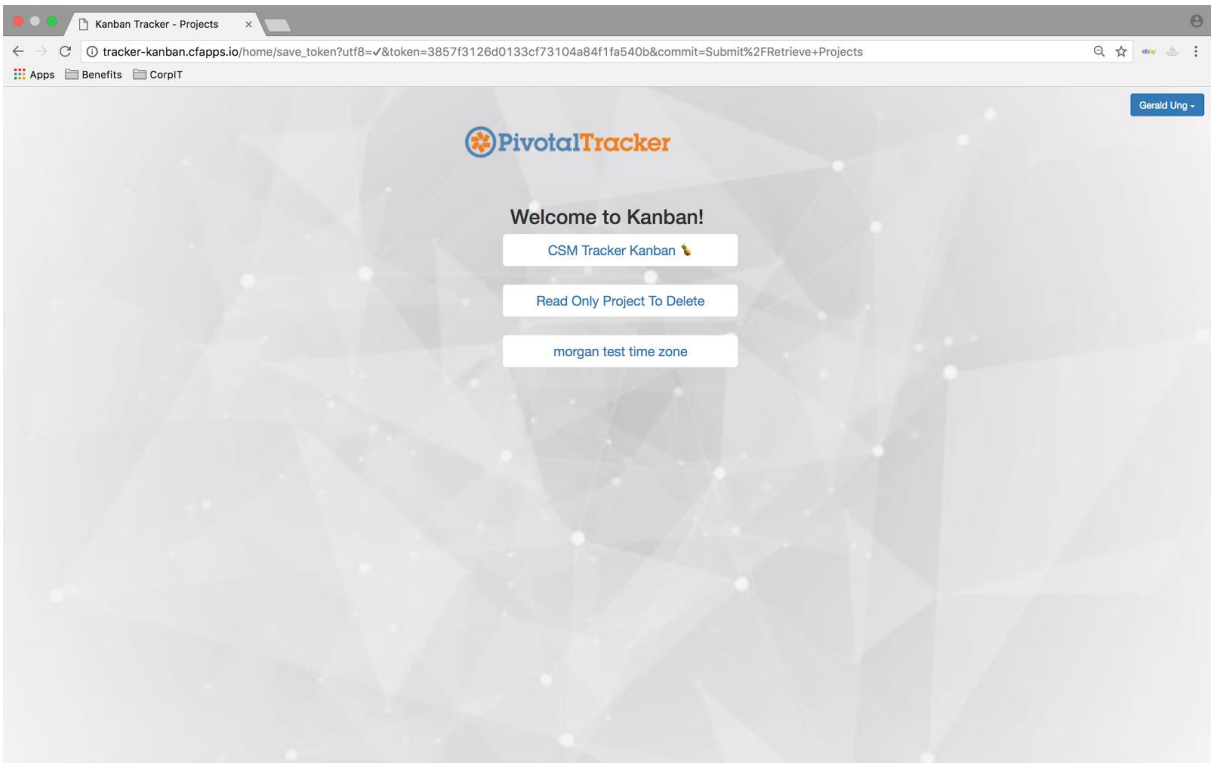

Figure 6: When an appropriate token is entered and the user is assigned to projects, they will be redirected to this screen which lists out all the projects. The user can click on any of the projects and see it in the Kanban view.

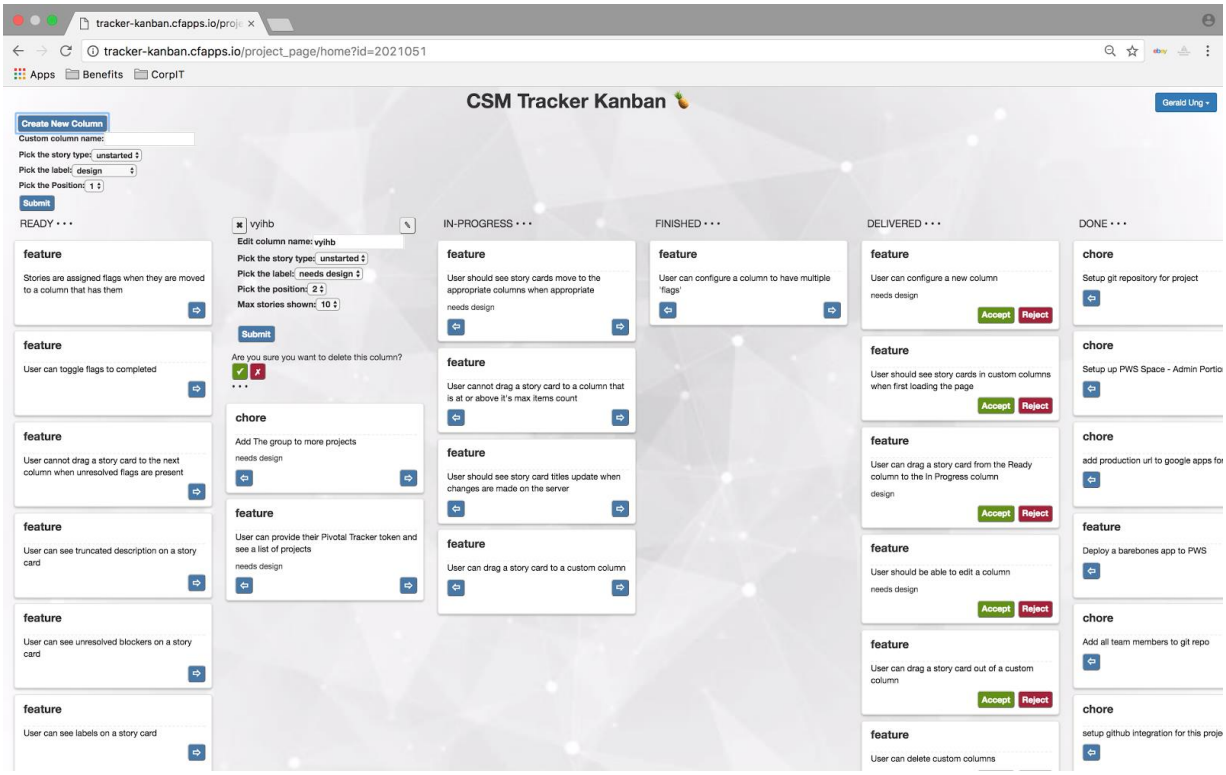

Figure 7: This is the Kanban view. In this view, we can see the forms for creating, editing, and deleting columns.

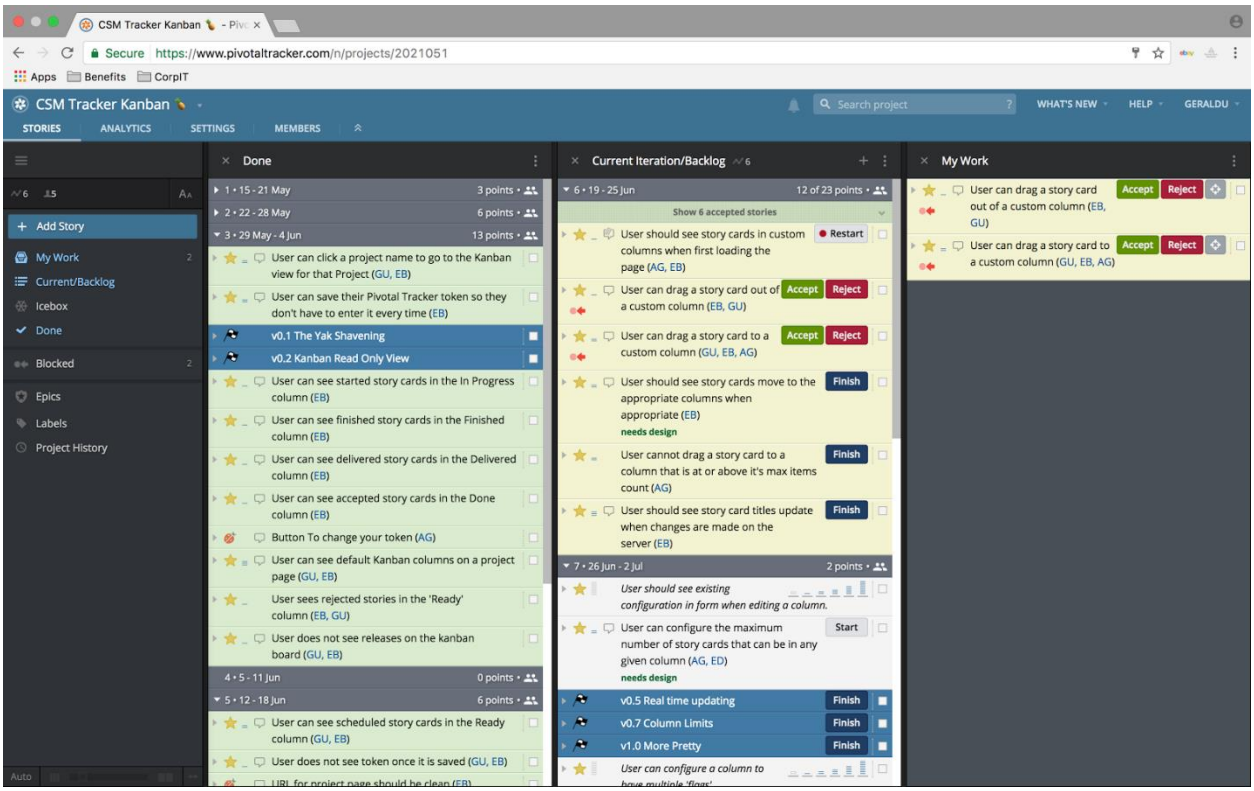

Figure 8: The traditional view when using Pivotal Tracker. Stories can be in any given state and users can have a hard time looking for specific stories.

8. REFERENCES

- *"What Is a Kanban Board?" LeanKit. N.p., n.d. Web. 19 June 2017.*