Food Watch

Pool 2 Development Process Evidence

July 7, 2015

IPT Associates, LLC
700 Technology Park Drive Suite 204 • Billerica, MA 01821
A Veteran Owned Small Business (VOSB)
www.iptassociates.com

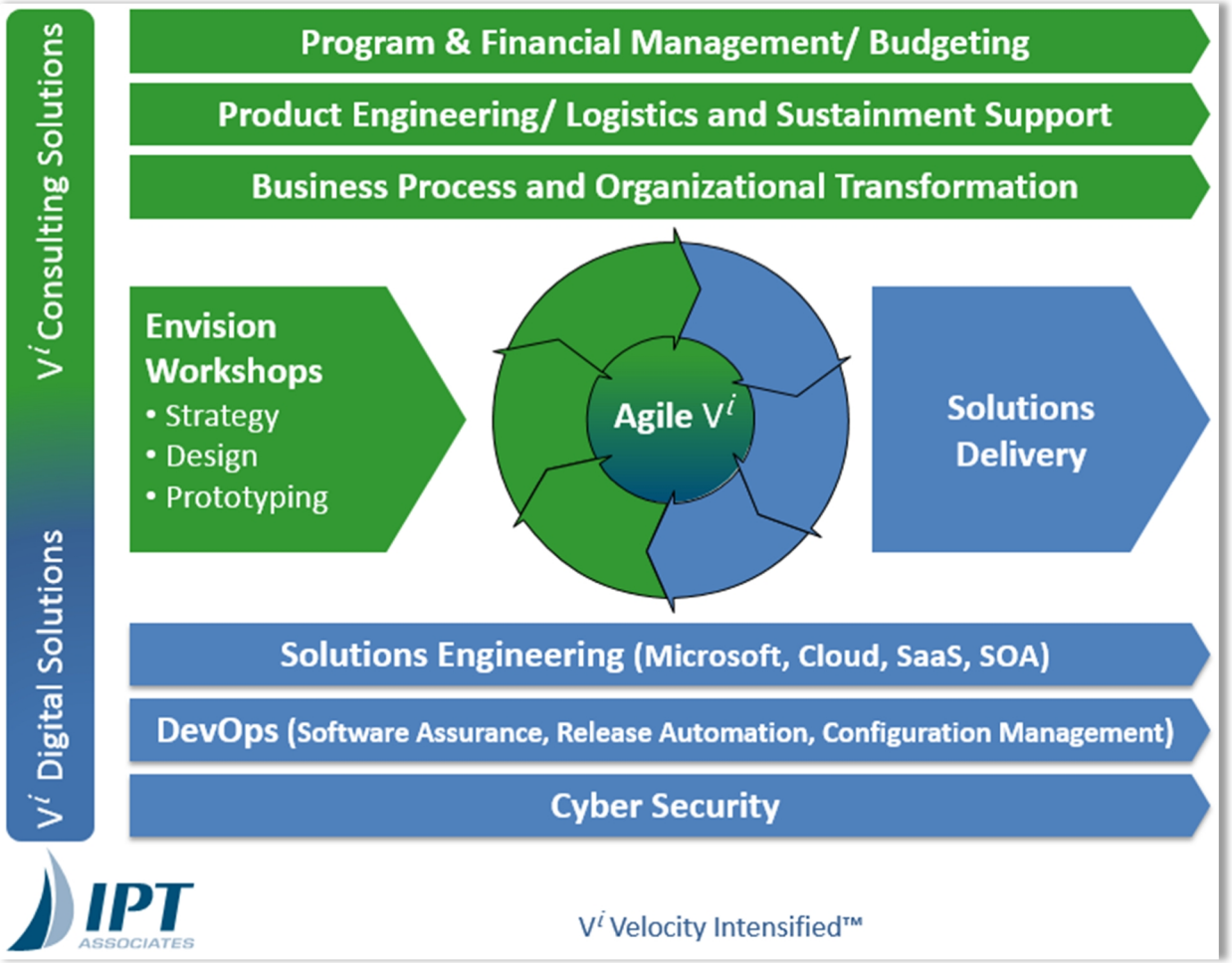
Table of Contents

1. Food Watch Process Evidence (Pool 2)

About this Document

This document has been prepared to describe the processes IPT Associates, LLC (IPT) used for the design and development of the Food Watch 1.0 prototype. As required in the General Services Administration (GSA) Agile Delivery Services BPA proposal (RFQ #4QTFHS150004), the Attachment E: Evidence Criteria document was provided for Pool 2 and submitted via the GSA eBuy portal. This document aligns the IPT Agile *Vi* process against said criteria to satisfy the evidence requirement for the proposal.

Agile Vi is IPT's process for accomplishing Agile/DevOps development. IPT has created Agile *Vi* so we can bring a repeatable process to our customer's that has proved results. IPT firmly believes that creating quality software is a collaborative team effort and the customer is the MVP.



The Food Watch prototype is an example of IPT's full process compressed into mini-sprints to meet the proposal deadline. In a typical development process, these sprints are usually two to three weeks, for the purpose of this prototype we went with three two-day Sprints to show the entirety of the process.

Sprint 1: Envisioning – This sprint is where we meet with the customer, review requirements and existing design assets, prepare our RDD (Requirements and Design Document) to ensure we understand the

customer's intentions and review with the customer for acceptance. **Sprint Goal:** Customer vision delivered to IPT.

Sprint 2: Development – Based on the RDD, one or more sprints are scheduled within a release window. A typical release usually contains two to four development sprints, with each sprint having at least one customer review session at the end, but typically we involve the customer as often as we can and find appropriate. **Sprint Goal:** Develop the use cases into functional software.

Sprint 3: Delivery Preparation – The final sprint before a release ensures that we have a quality feature set for the customer, and that any critical feedback the customer desires is incorporated into the release product. (If customer input is not feasible to implement, some feedback tasks move to the next release iteration.) This sprint is also where release notes/install guides/packages for release are finalized if required. **Sprint Goal:** Deliver a working prototype that satisfies the customer vision.

The information that follows in this document contains descriptions, screenshots, and photographic evidence that line up the activities that occurred within our three sprints, to the activities that require evidence to comply with the RFP as it pertains to Pool 2.

2. Required Evidence

a) Assigned one leader, gave that person authority and responsibility, and held that person accountable for the quality of the prototype submitted

Prior to Sprint 1, typical customer outreach occurs and a basic desire for a product is expressed. During IPT's Sprint 1: Envisioning phase, the **Need Statement** that drives overall design and development process is garnered. The team leader is ultimately responsible for making this happen and was assigned by the corporate leadership team to the project.

Josh was selected to be the Food Watch leader and **Technical Architect.** Josh has 15 years of experience leading and architecting technology driven projects for the Air Force and commercial customers throughout all phases of the software development lifecycle. Josh's experience allows him to inherently understand how to assemble a lean, agile team that can embrace Agile V[i] processes and deliver the Food Watch solution. The IPT team held Josh accountable for the Food Watch prototype delivery, quality, and customer satisfaction.

From this point forward, all tasks and activities (including the assignment of the project lead) were captured in our project / configuration management tools to help break down the project's work elements. For Food Watch, IPT used Microsoft's Visual Studio Online. IPT's backlog board within Visual Studio can be seen in *Figure 2* below.

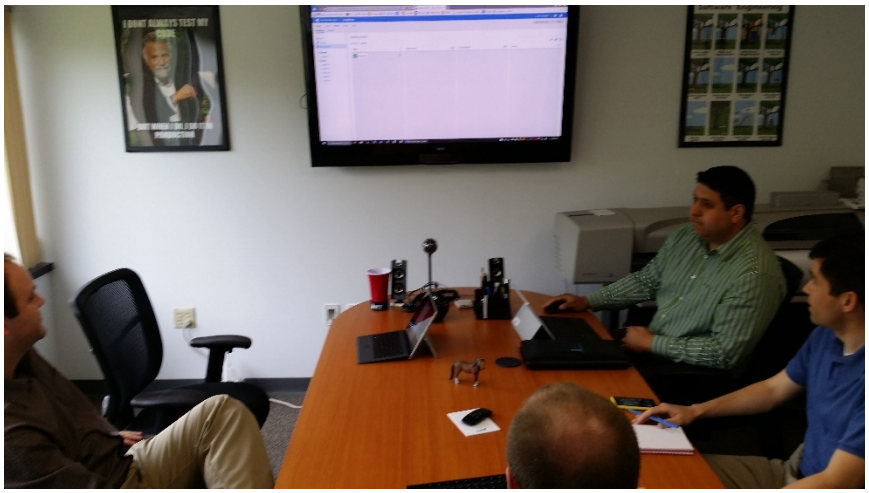


***Figure 1*. IPT's Technical Architect leads Food Watch team at the kickoff meeting.**

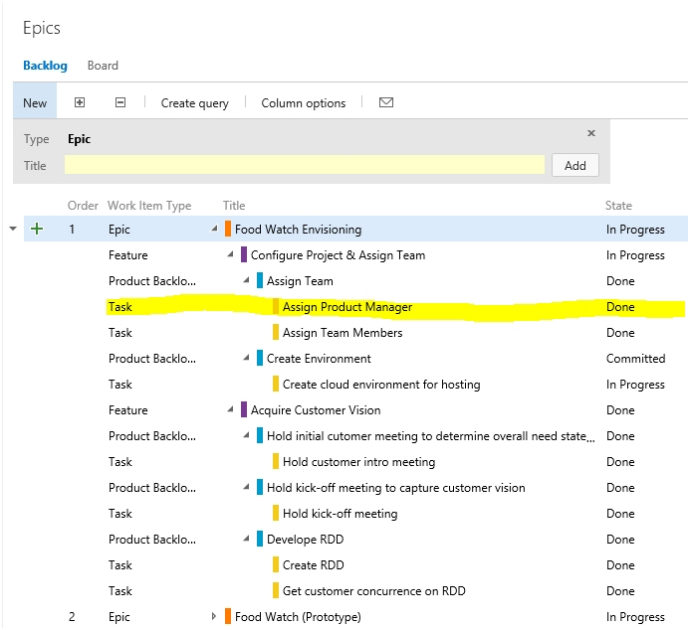


***Figure 2*. Backlog board for IPT's Envisioning Sprint.**

b) Assembled a multidisciplinary and collaborative team that includes a minimum of two of the labor categories limited to the Development Pool labor categories as quoted in Attachment C.

The Technical Architect worked with the corporate leadership team and department leads to assemble a project team and assign positions based on the scope of the project. The Technical Architect's in-depth understanding of IPT's Agile *V*^{i} processes allowed him to structure an estimated level of effort (LOE) required to rapidly deliver a quality and customer-focused Minimum Viable Product (MVP). To effectively deliver an Agile *V*^{i} solution to the customer, IPT engaged a lean but seasoned staff bolstered with the necessary Agile and DevOps experience designing and developing digital services, designing web applications, using automated testing frameworks, etc.

Figure 3. **IPT's Food Watch kickoff meeting and team assembly.**

The work elements previously broken down were then assigned to each team member based on their experience and expertise as they apply to Pool 2 activities.

Technical Architect: Josh
Front End Web Designer / Developers: Dan
Front End Web Designer / Developers: Dave
DevOps Engineer: Justin

Team members were then added to the Food Watch Visual Studio Online team site so they could be assigned tasks for sprints, as seen in *Figure 4* below.

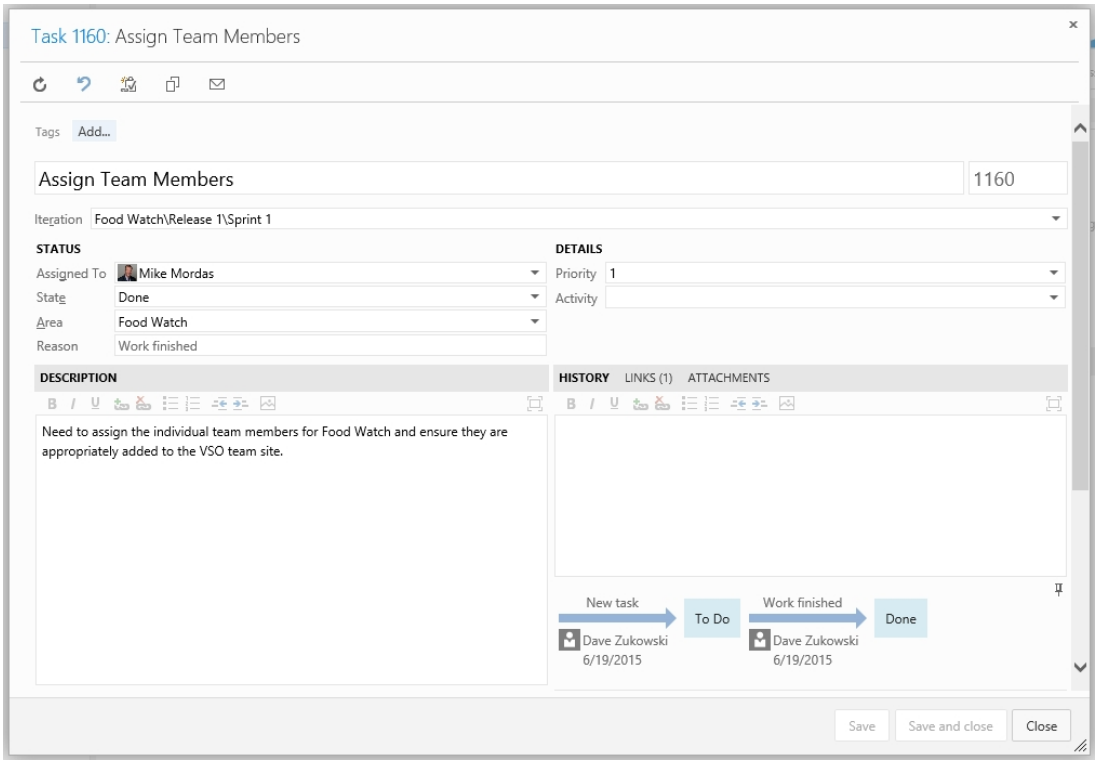


Figure 4. **Detailed task info screen for IPT's team assignments.**

c) Used at least five modern and open-source technologies, regardless of architectural layer (frontend, backend, etc.)

IPT leveraged ten (10) modern open javascript frameworks to accomplish Food Watch. All frameworks used, as depicted in *Figure 5* below, are open-source technologies licensed by MIT.

Excel Online

Food Watch

	A	B
1	**Library**	**License**
2	Durandal	MIT
3	jQuery 1.10.2	MIT
4	require.js	MIT or New BSD
5	Bootstrap	MIT
6	Modernizr	MIT
7	Font-Awesome	MIT
8	Knockout.js	MIT
9	moment	MIT
10	.NET	Apache 2.0

Figure 5. **Modern open source technology and license listing.**

d) Deployed the prototype on an Infrastructure as a Service (IaaS) or Platform as a Service (PaaS) provider, and indicated which provider they used

IPT used the Microsoft Azure Web App Service for hosting Food Watch. Azure Web App Services are components of the Microsoft Azure Cloud's App Service PaaS offering. These services provide an

enterprise ready managed platform for rapid development, deployment, monitoring, scalability, and extensibility.

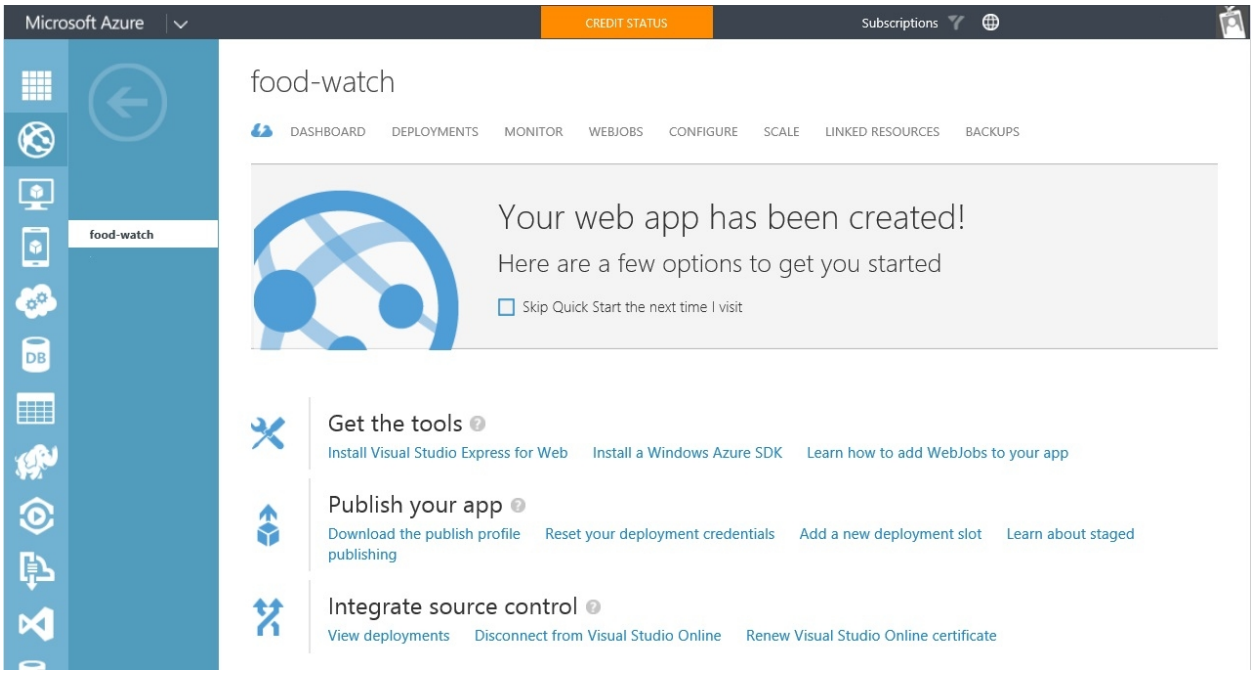


***Figure 6*. IPT's MS Azure deployment.**

e) Wrote unit tests for their code

Unit tests were written and executed as part of our continuous integration activities. As a general rule, unit tests are written to cover algorithms and business rules. IPT applies this logic to our strategy when it comes to unit testing. Rather than achieving 100% coverage, the tests will aim to target the highest cost-to-benefit ratio. To reduce cost, algorithms and business rules must be written with few dependencies where possible.

Overcomplicated code shall be refactored. Trivial code will not be unit tested. Server side code will be written in Visual Studio unit testing framework. Client side unit testing (testing of HTML and JavaScript) will follow Durandal's guidelines:

http://durandaljs.com/documentation/Testing-With-PhantomJS-And-Jasmine.html
http://jasmine.github.io/
http://phantomjs.org/

Integration tests may be written to prove the whole system components works together. The tests will be configured to run automatically upon committing to source control. Failing tests will prevent automatic deployment.

IPT applied a portion of this unit testing strategy to Food Watch as our team wrote a unit test for Food Watch using the Visual Studio testing framework (see *Figure 30*). However, due to the schedule limitations, prototype requirements of Food Watch, and our desire to keep the Food Watch application simple, the IPT team did not feel there was a strong need to write unit tests targeting the client side code.

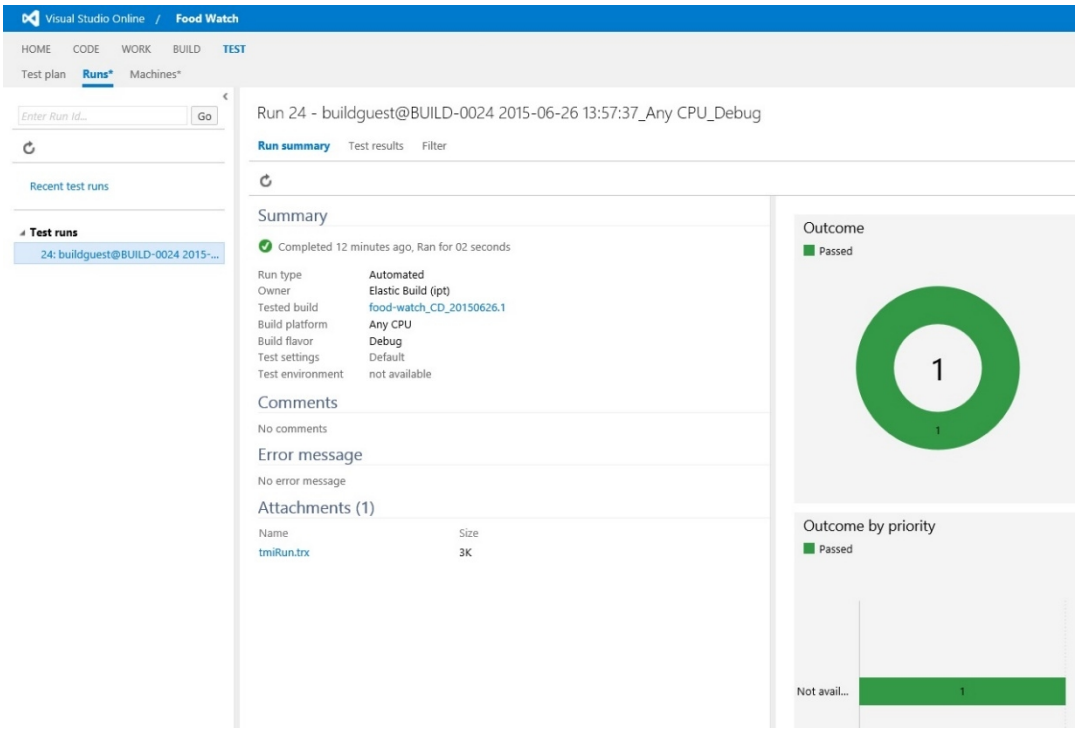


Figure 7. **Screenshot of unit test execution using Visual Studio's testing framework.**

f) Set up or used a continuous integration system to automate the running of tests and continuously deployed their code to their IaaS or PaaS provider

IPT used a combination of Microsoft Visual Studio online and Azure web apps to create a PaaS based continuous integration environment.

Our Food Watch deployment history can be seen in *Figure 8* below:

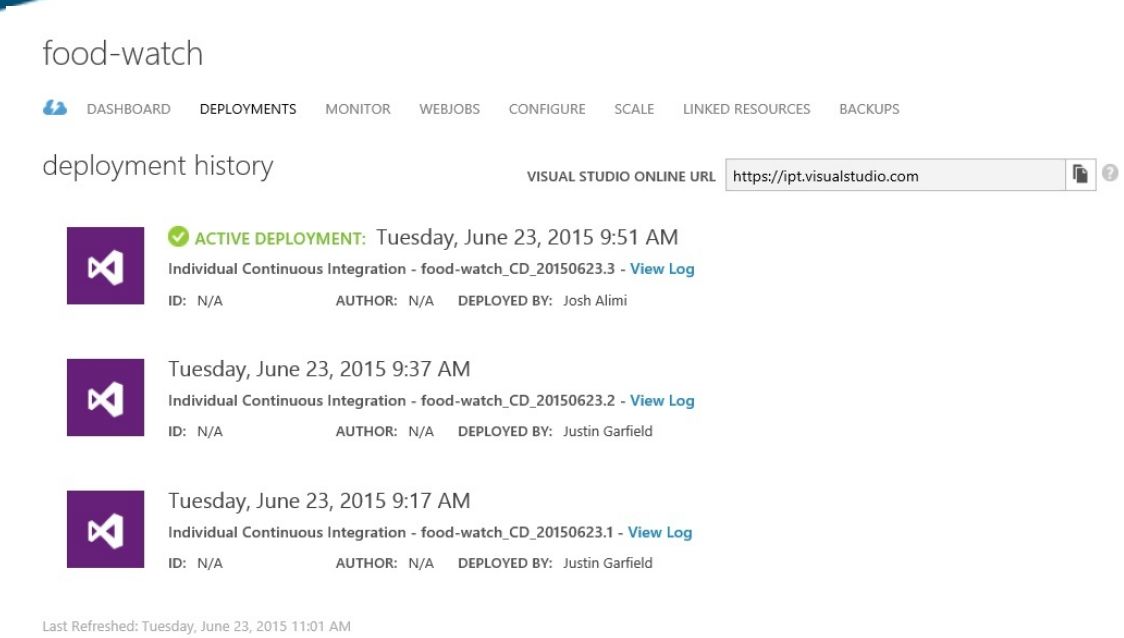


***Figure 8*. Food Watch's continuous integration build results.**

g) Set up or used configuration management

IPT set up and used Visual Studio Online with a GIT based repository for configuration management on Food Watch.

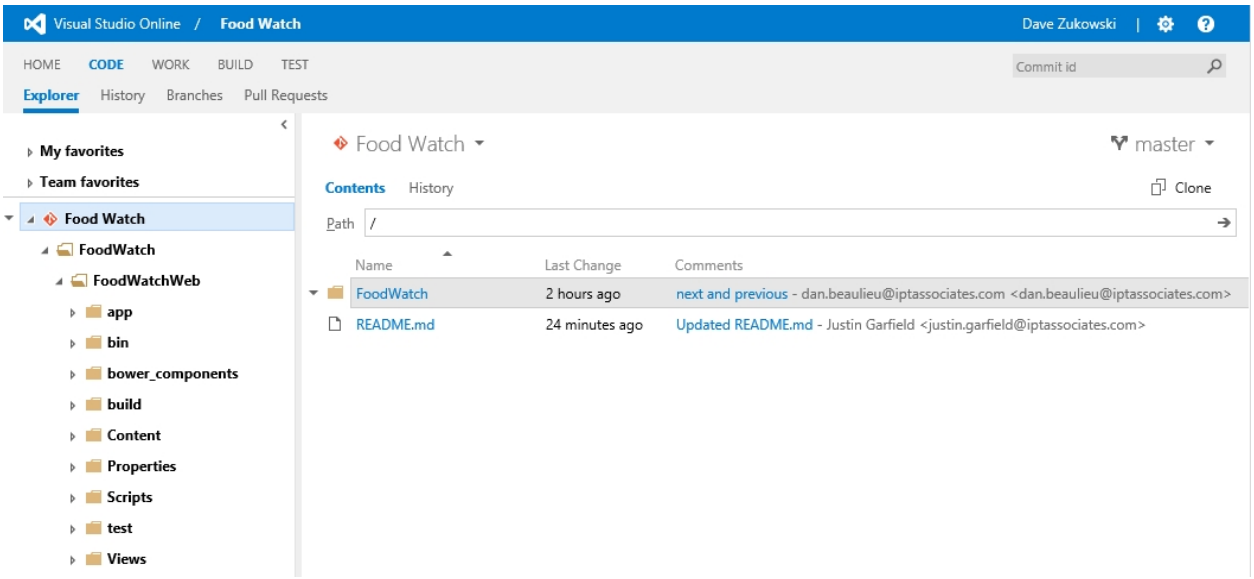


***Figure 9*. IPT's configuration management in VSO.**

h) Set up or used continuous monitoring

Food Watch's continuous monitoring was set-up and configured through the Azure management portal.

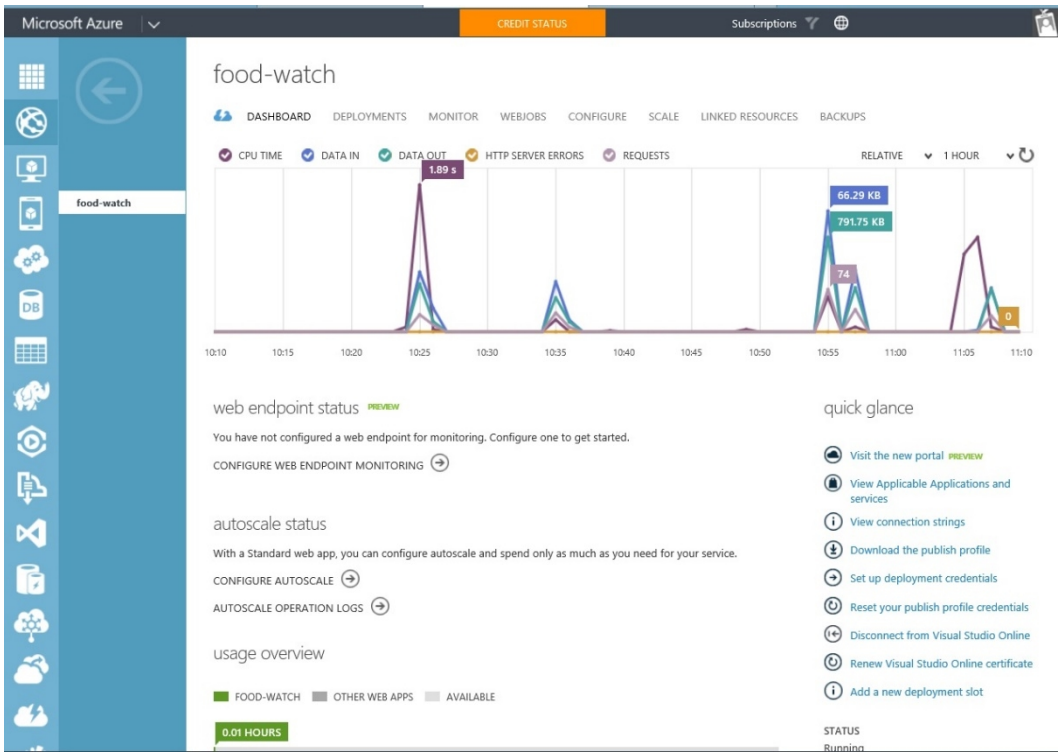


***Figure 10*. Screenshot of Food Watch's continuous monitoring configured via MS Azure management portal.**

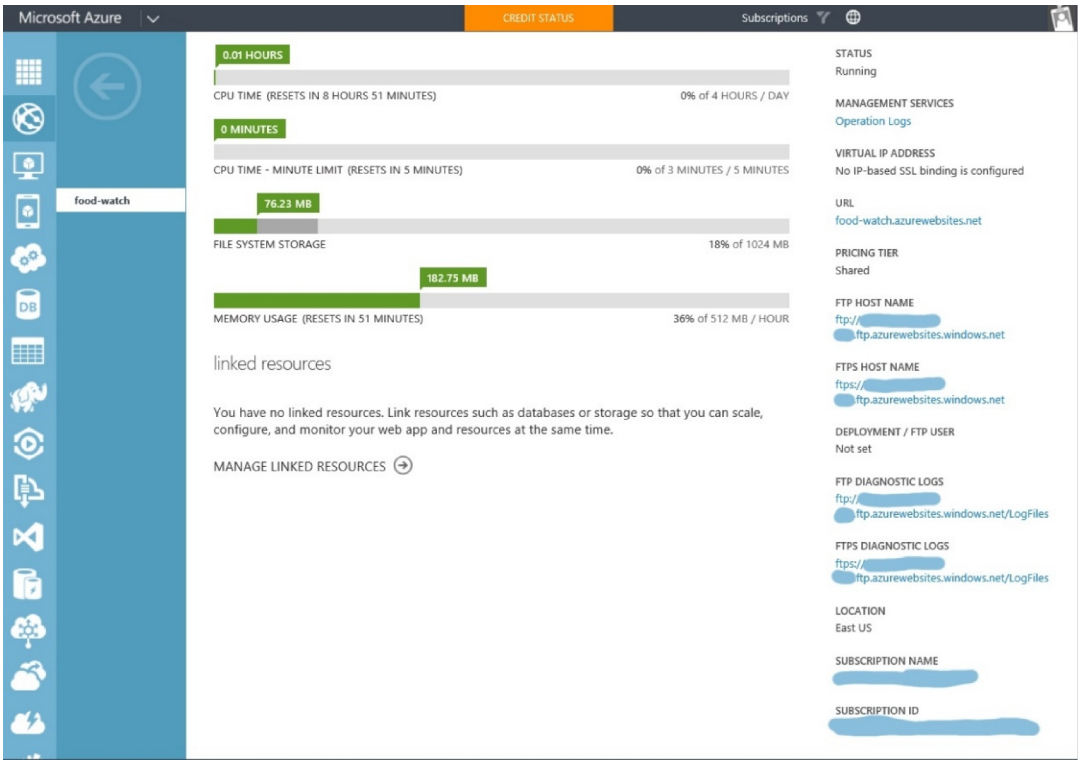


***Figure 11*. Screenshot #2 of Azure continuous monitoring.**

i) Deploy their software in a container (i.e., utilized operating-system-level virtualization)

IPT used the Microsoft Azure Web App Service for hosting Food Watch. Azure Web App Services are components of the Microsoft Azure Cloud's App Service PaaS offering. These services provide an enterprise ready managed platform for rapid development, deployment, monitoring, scalability, and extensibility.

Azure Web Apps provide an isolated secure environment for web application components to execute without direct access to the underlying operating system and establish a consistent operating environment to deploy solutions into.

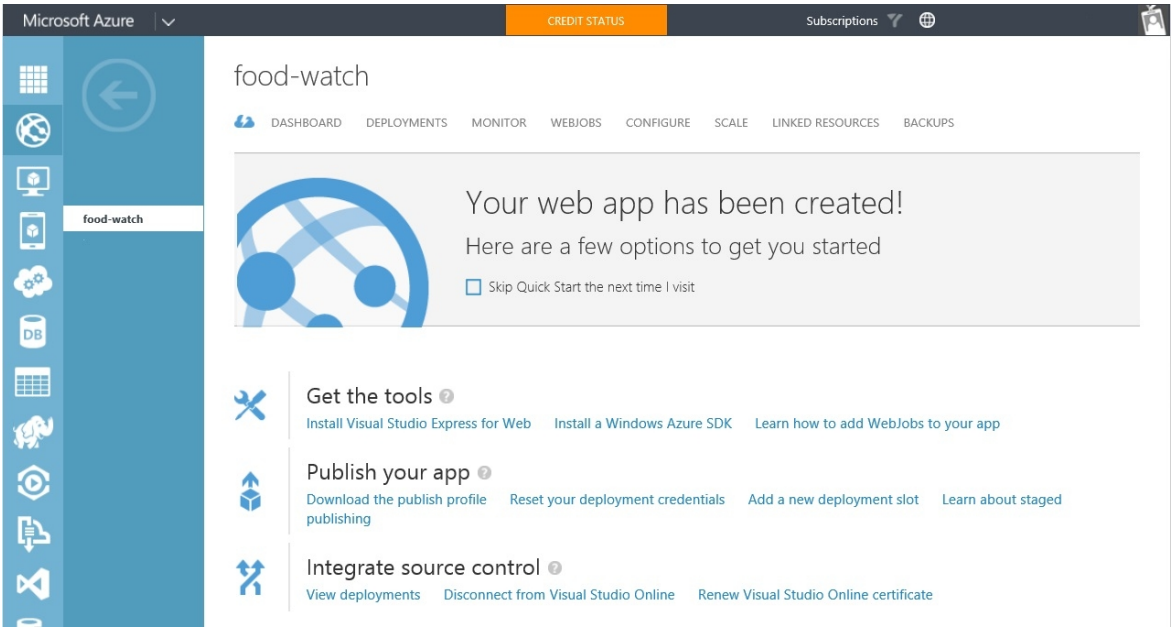
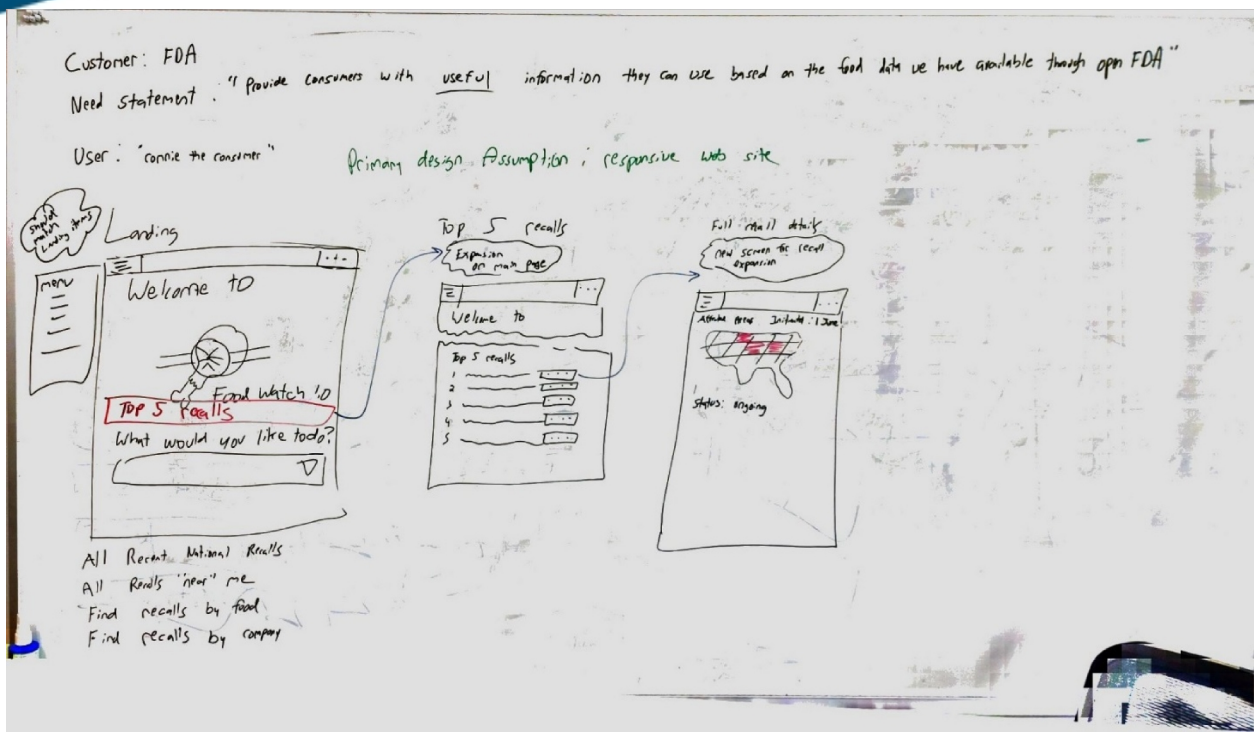
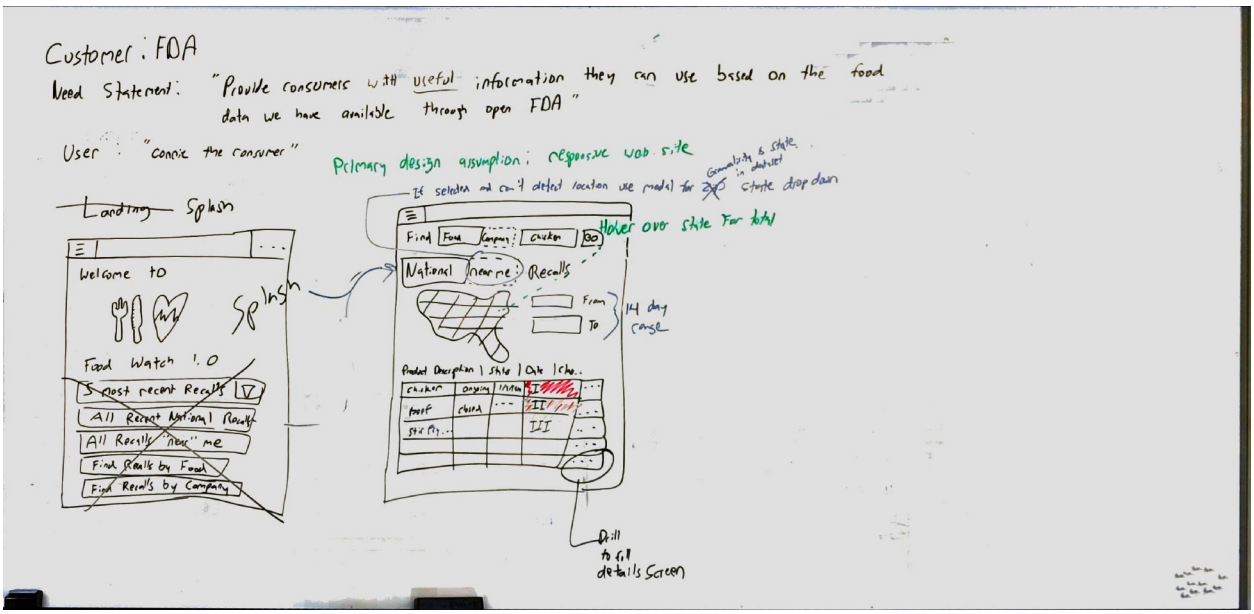


***Figure 12*. Azure Web app container overview.**

j) Used an iterative approach, where feedback informed subsequent work or versions of the prototype

In the development, the brainstorming and whiteboard sessions, team reviews, customer feedback, and usability tests drove the design through an iterative and collaborative process. The rough initial whiteboard drawings were iterated on by the team to establish the basic layout as well as look and feel of the product. Those were translated into the RDD as wireframes and further reviewed with the customer to ensure they met the need. From customer feedback the RDD was iteratively tweaked and updated until an agreed “baseline” design was reached and then the working prototype was developed. The working prototype was reviewed by the team and customer providing changes and tweaks to improve the overall interactive user experience.


Customer: FDA
Need statement: "I provide consumers with useful information they can use based on the food data we have available through open FDA"
User: "connie the consumer"
Primary design assumption: responsive web site
Landing
Welcome to
Food Watch 1.0
Top 5 Recalls
What would you like to do?
Top 5 recalls
Full recall details
All Recent National Recalls
All Recalls "near" me
Find recalls by food
Find recalls by company


Customer: FDA
Need Statement: "Provide consumers with useful information they can use based on the food data we have available through open FDA"
User: "connie the consumer"
Primary design assumption: responsive web site
Landing Splash
Welcome to
Food Watch 1.0
5 most recent Recalls
All Recent National Recalls
All Recalls "near" me
Find Recalls by Food
Find Recalls by Company
Find Food Company Location
National Near me Recalls
From
To
14 day range
Hover over state for total
Drill to full details screen

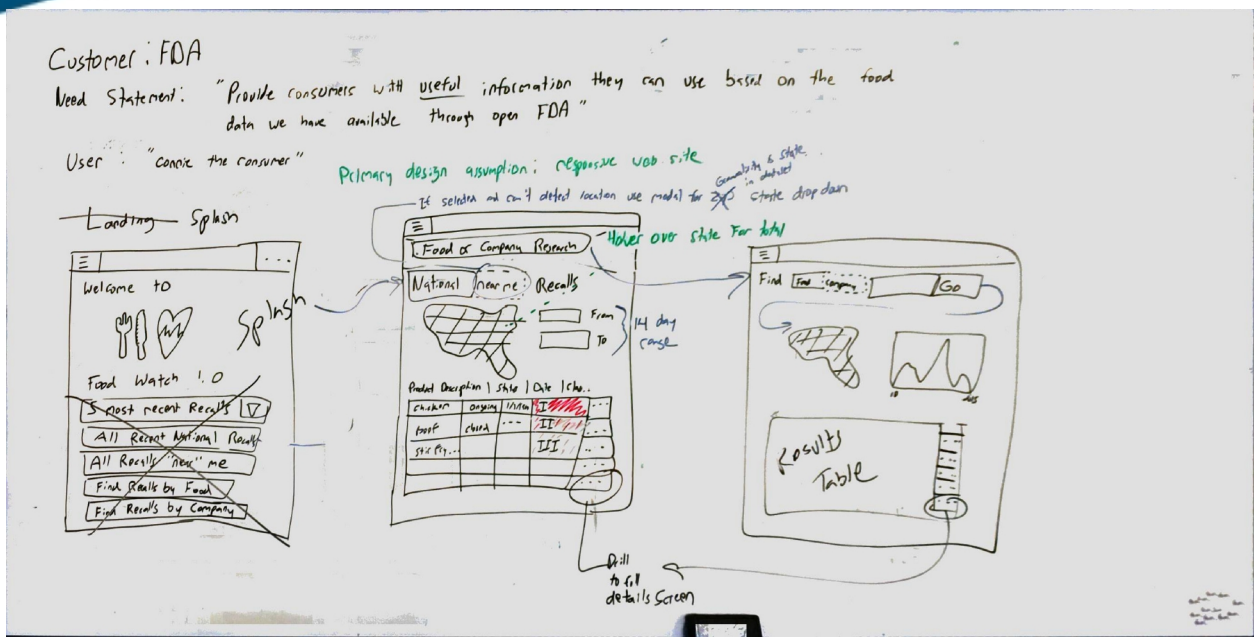
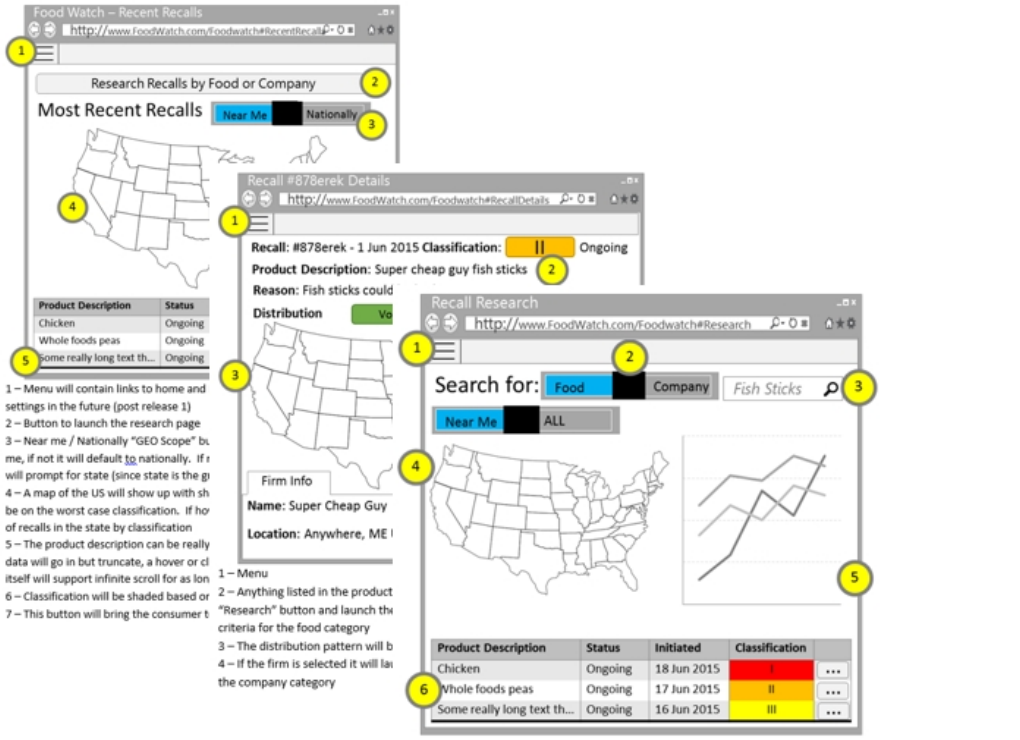


Figure 13. **IPT's brainstorming whiteboard design iterations.**



1 – Menu will contain links to home and settings in the future (post release 1)
2 – Button to launch the research page
3 – Near me / Nationally "GEO Scope" bu
me, if not it will default to nationally. If
will prompt for state (since state is the g
4 – A map of the US will show up with sh
be on the worst case classification. If ho
of recalls in the state by classification
5 – The product description can be really
data will go in but truncate, a hover or cl
itself will support infinite scroll for as lon
6 – Classification will be shaded based or
7 – This button will bring the consumer t

1 – Menu
2 – Anything listed in the product
"Research" button and launch the
criteria for the food category
3 – The distribution pattern will b
4 – If the firm is selected it will la
the company category

1 – Menu
2 – This is the research type toggle, either food or companies (firms) can be researched
3 – This is the search term, if "drilled" from another page this will be prepopulated and executed
4 – This shows an aggregate of the distribution pattern and otherwise behaves like the map on the landing page, and also allows the research to be restricted to recalls near me or "All"
5 – This is a histogram of recalls over time by classification
6 – This is the list of results based on the research and behaves exactly as the grid does on the landing page

Figure 14. **RDD Wireframes driven by the whiteboard sessions.**

Food Watch Team Sprint 2

June 22 - June
2 work

Backlog Board Capacity

Create query | Column options

	Title	State	Assigned To	Rema...
+	Create Environment	Committed	Mike Mordas	
	UC1 - Connie the Consumer wants to see all recent recalls	Committed	Dan Beaulieu	
	UC2 - Connie the Consumer wants to see all recalls based on f...	Committed	Dan Beaulieu	
+	Hold User Feedback Sessions	New	Mike Mordas	
	Hold Sprint 2 User Feedback Session	Done	Mike Mordas	

Figure 15. **Usability/Feedback Sessions as tasked in Visual Studio.**

Figure 16. **IPT's development prototype review sessions.**

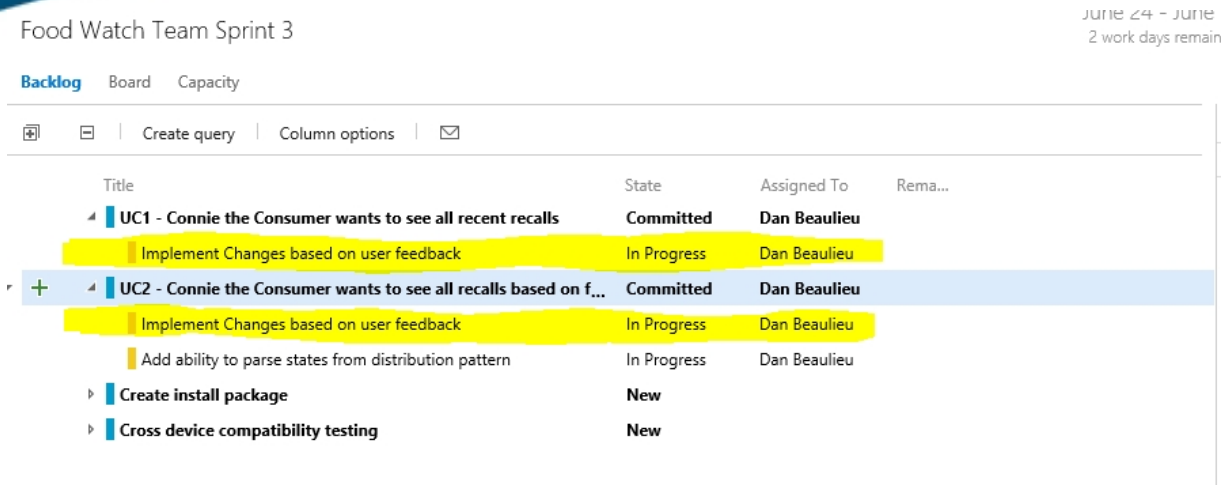


Figure 17. **New tasks for final prototype development sprint driven by Sprint 2 feedback results.**

In *Figure 18*, Connie reviewed the Food Watch development and suggested the "Browse Recent Results" be more visible than the "Learn More" button on the landing page. Connie also wanted a map of the United States with states highlighted so she could quickly see if her state was effected by a certain food product's recall. Connie also picked up that the recall firm was missing in the recall details, which was information she would have wanted. All of these usability test results were highlighted in Figure 18 as the front end designers/developers marked up their designs. The developers took Connie's feedback and incorporated the input into the subsequent iterations of Food Watch.

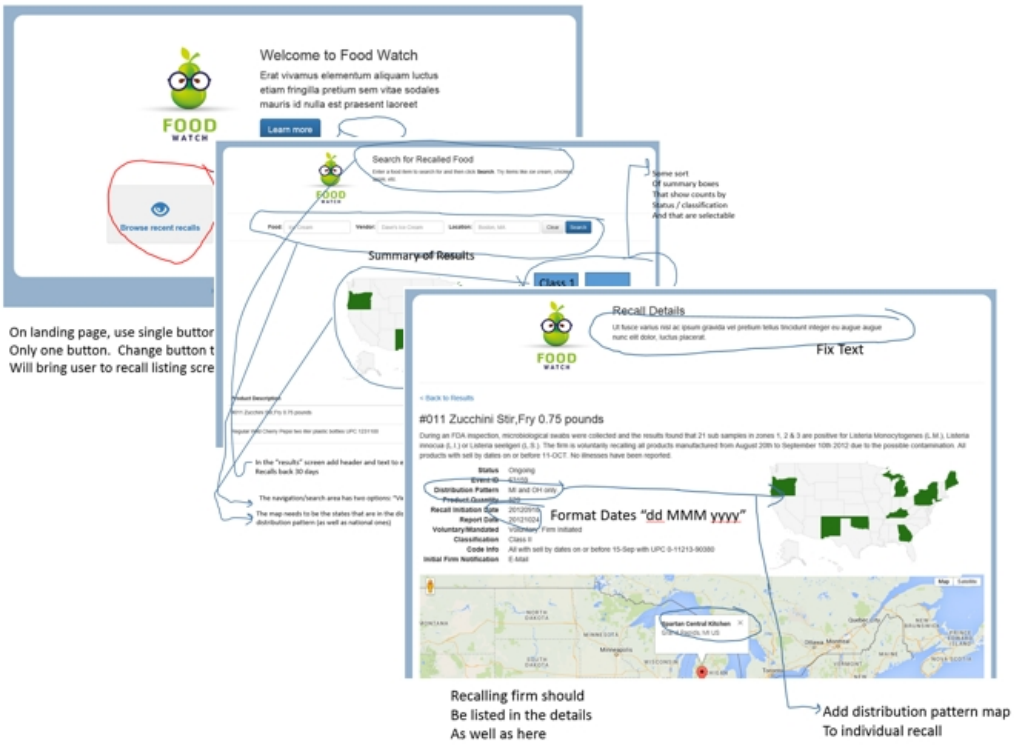


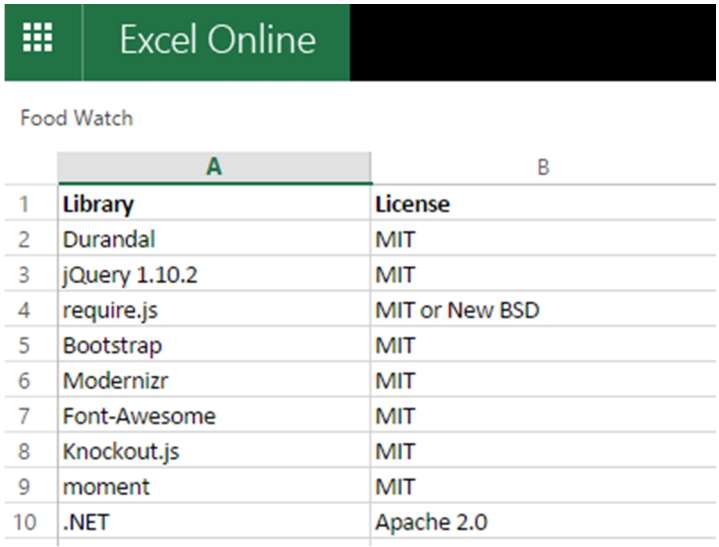
Figure 18. **Customer Review and Usability tests during development.**

k) Provided sufficient documentation to install and run their prototype on another machine

A Food Watch Installation Guide was provided in the documentation folder in our GitHub repository.

l) Prototype and underlying platforms used to create and run the prototype are openly licensed and free of charge

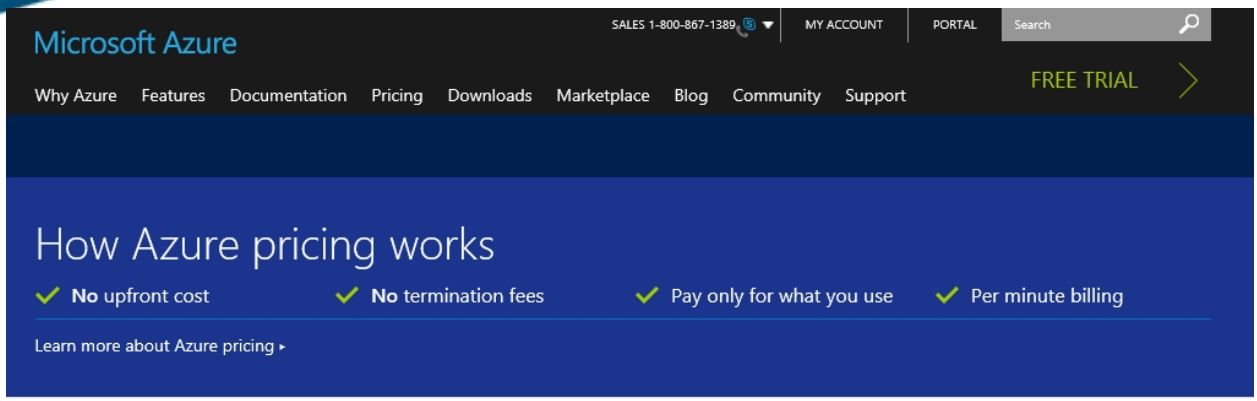
IPT leveraged ten (10) modern, openly licensed platforms to design Food Watch. All frameworks used, as depicted in *Figure 19* below, are open-source technologies licensed through the free MIT platform.



	A	B
1	**Library**	**License**
2	Durandal	MIT
3	jQuery 1.10.2	MIT
4	require.js	MIT or New BSD
5	Bootstrap	MIT
6	Modernizr	MIT
7	Font-Awesome	MIT
8	Knockout.js	MIT
9	moment	MIT
10	.NET	Apache 2.0

***Figure 19*. Platform license listing.**

IPT ran the prototype in the free version of Microsoft Azure, as *Figure 20* illustrates:



App Service brings together everything you need to create web and mobile apps for any platform and any device. The Free and Shared plans allow you to host your apps in a shared environment, while Basic, Standard, and Premium plans provide Virtual Machines dedicated to your plan. You can host multiple apps and domains in each instance you deploy within your plan.

	FREE Develop and test apps	SHARED Dev/test with higher limits	BASIC Go live with basic apps	STANDARD Go live with web, mobile, logic apps	PREMIUM Maximum scale and enterprise integration
Web, mobile, or API apps	10	100	Unlimited	Unlimited	Unlimited

***Figure 20*. Hosting container initial pricing.**