Create Order with OnPageLoads & OnClick events in your Agile Project  
This is a theory on how to logically break down a large or small Web Application to fit within your Agile Project and guarantee you are building a Web Application with a solid foundation.

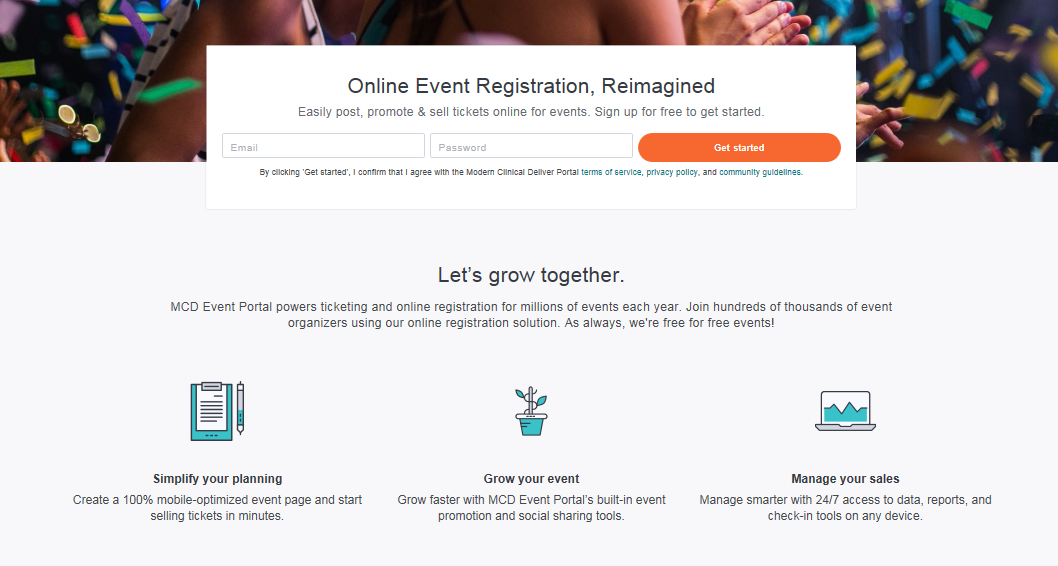
Written by Carl Boudreau, Software Architect for Modern Clinical Delivery  
carl\_boudreau@optum.com

*If you haven’t read my article “****Start with a Wireframe! Agile is not an excuse to start coding without a plan****” I would recommend you read it and save you and your team a lot of frustrations while developing your application.*

The challenge is how we provide continuous delivery with iterative Customer Validation and Feedback that the Team and Project is on track. Oftentimes, we build something exactly as per requirements, but it turned out that's not what business stakeholders wanted at all. Instead of finding that out six months after the project started, Scrum helps you avoid that by providing something tangible to show users by the end of the first sprint. Then you get feedback and continue to iterate until you build something the customer is happy with and accepts.

Consider a small web application use case that is an event registration that collects attendee information for a series of events. First we need to know who the attendee is. What is the attendee first name last name, email, phone numbers? Next which event(s) is the attendee interested in? Will the attendee bring guests? Does the attendee or their guest have any special needs? Are there seats open for the Attendee and their guest? If we have collected the Attendee’s information, and there are open seats then we need to register the Attendee and change the availability of seats open after this Attendee registration.

It would be a mistake to create a User Story that would save the Attendee registration data for an event if there were not enough seats available. This mistake is often made when deciding how to divide a project into digestible Program Increments (PI). Business wants to see “Core Functionality” so we often see the first story that read like “As an Event Coordinator I need to see how many people are registered for my event.”

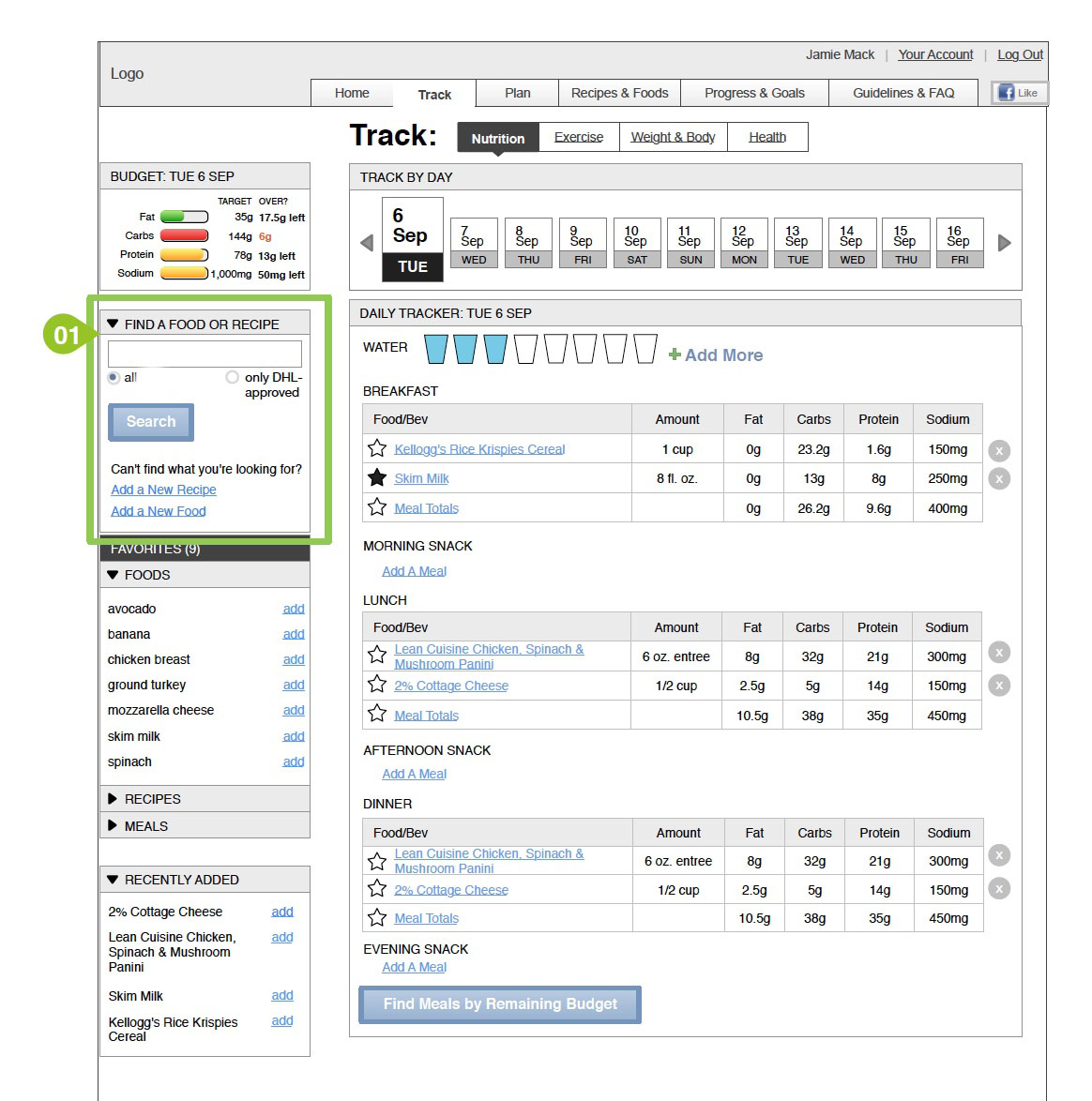
The problem with this being the first story, is the Preliminary Work that needs to be performed before this story can be demoed for acceptance. For example; where is the story that describes a Screen that the Attendee can select an Event(s), and how did the Event(s) get listed on the page to be selected? So maybe the first story should have been “I need a Web Site to log into.”   


Assuming you read “**Start with a Wireframe! Agile is not an excuse to start coding without a plan**”, or onboard with creating a wireframe to describe your Web Application. Determining where the first User Story and the following User Stories should start and end can be determined by considering the life cycle of a page, or for more complicated pages; blocks of functionality within a page.

Let’s look at the above MCD Event Portal login page. This page has an OnPageLoad and 4 OnClick events. The OnPageLoad event happens when the User enters a URL into a browser and loads the login page. It has 4 OnClick events; the “Get Started Button”, “Simplify your planning” image/button, “Grow your event” image/button and the “Manage your sales” image/button.

This page indicates that you should have 5 stories ordered by priority of the Business/Product Owner to complete this page. When you start writing a story ask yourself “Do I need any data prior to showing this page?” If you answer yes then you need a story prior to writing this story to collect the data needed. For this login example above you don’t.

Now let’s look at a more complicated page. The page shown below is from a nutrition management web site I build. This page demonstrates that a large amount of unique data based on a UserId is needed for the one OnPageLoad event.



Because this page displays data specific to a user there are over 70 data elements needed before this page could be rendered and shown to the user. There are over 16 unique OnClick events that bring the user to another page. Therefore to build this page we may need 17 or more stories. Unlike the Login Page each of these 17+ stories requires data like the UserId before loading the next page. This is a good indicator that your User Story is going to be broken into a number of tasks such as “Define Data Contract for Track Page”, “Create Unit Test for Track Page”, “Create API for Track Page”, and “Create UI Test Script for Track Page” for example.

By following this theory you will provide order for the natural progression of User Stories. Therefore demonstrating functionality and gaining more velocity to get your stories accepted.