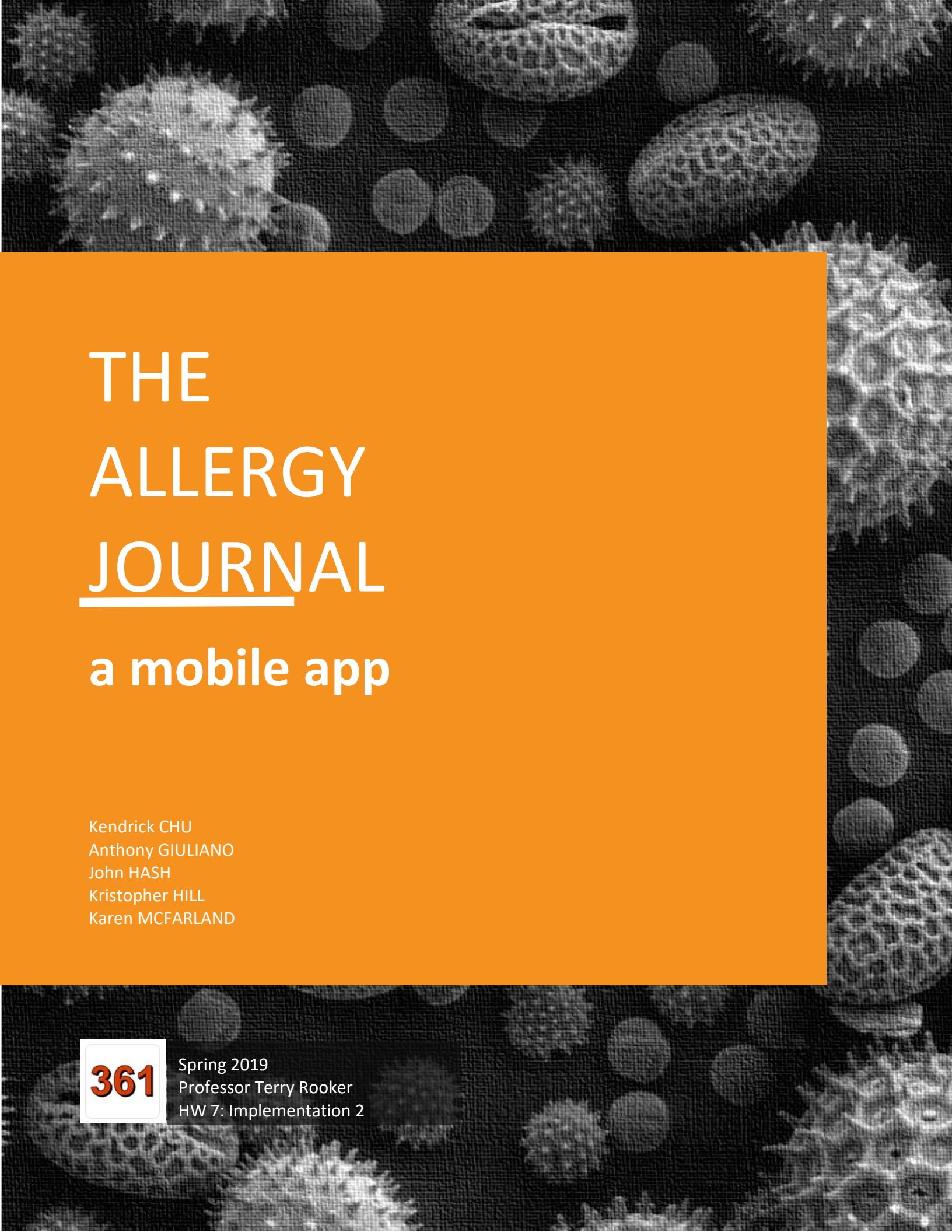


# THE ALLERGY JOURNAL

## a mobile app

Kendrick CHU
Anthony GIULIANO
John HASH
Kristopher HILL
Karen MCFARLAND

361

Spring 2019
Professor Terry Rooker
HW 7: Implementation 2

## URL To Software Prototype

The application's homepage is found at the following url:
http://web.engr.oregonstate.edu/~hashj/appHome.html

The website application works on all major browsers (Chrome, Firefox, Safari, and Microsoft Edge). The page cannot be properly viewed in Internet Explorer. In order to view any of the pages that have links on the homepage, users should be either on the Oregon State University campus or connected via VPN. The homepage will load over any network.

The application's source can be found in the zip file uploaded along with this PDF.
Filename: CS 361 Group 24 Code Submission.zip

Note that the source code is spread out into multiple server files. With as many user stories as we were trying to implement, it would have been logistically impossible for us to all coordinate on and use a single node. Further, it would have meant that one of us would had to have given their OSU credentials to the group for everyone to work on the same node. We decided it was better to link all the parts of the application together through the main webpage. The entire application does use a single database, however

# User Stories Information:

## Pair Programming

The following table shows the pairs of teammates who worked on a given user story's tasks together.

| User Story | Kendrick CHU | Anthony GIULIANO | John HASH | Kristopher HILL | Karen MCFARLAND |
|---|---|---|---|---|---|
| 0.1 Environment Setup - Webpages | X | | X | | X |
| 1. User Story: Add Product | | X | | X | |
| 2. User Story: Record Product Use | | X | X | | |
| 3. User Story: Record Reaction | X | | X | | |
| 4. User Story: View Product Use Summary | X | | | | X |
| 5. User Story: View Reaction Summary | X | | | X | |
| 6. User Story: View Product Inventory | | X | X | | |

## Relevant Unit Tests

The following table shows the relevant unit tests that were conducted.

| User Story | Relevant Unit Tests |
|---|---|
| 0.1 Environment Setup - Webpages | Navigation: Homepage links were added to all the pages. Each of these were tested. We tested that the pages will load in all commonly used browsers. Usability: We tested to make sure fonts were readable and links were clearly marked. |

| User Story | Relevant Unit Tests |
|---|---|
| 1. User Story: Add Product | Select Menu: Tested for confirmation that the list of products in the Products table were selected from the DB and inserted into the menu.<br>Submit Button for Add New Product: Tested to confirm that page is refreshed after clicking on the button (select menu is set back to default, all other forms are cleared).<br>Barcode Field: Tested for min and max lengths, to confirm that it would only accept strings of length 13.<br>Submit Button for Request a New Product: Tested to confirm that (a)the page refreshes and the fields are cleared, and (b) clicking this button prompts an alert confirmation that their request was submitted.<br>Database: Tested to confirm that product ids of products in the Products table that are added by a customer are added to a bridge table along with the user's id. Also tested to confirm that products submitted for admin add are added to UnderReviewProducts table. |
| 2. User Story: Record Product Use | Inputs: Date field was tested to make sure calendar dropdown worked and that the field data type was correct. Freeform user inputs were tested to ensure that they allowed users to enter data.<br>Value conversion: Checkboxes do not natively use "true" and "false." We tested that checkboxes were actually "checked." We tested that checked and unchecked boxes returned "1" and "0" values to match the database.<br>Selectable body regions: We clicked on and off the body images regions to ensure that the proper form was displayed.<br>Product dropdown: We tested to ensure that the proper data were displayed in the dropdown. |
| 3. User Story: Record Reaction | Inputs: Date field was tested to make sure calendar dropdown worked and that the field data type was correct. We tested that the freeform user inputs allowed user to enter data.<br>Value conversion: We tested that checkboxes were actually "checked." We tested that our client function properly converted checked boxes and returned "1" and "0" values to match the database.<br>Selectable body regions: We clicked on and off the body images regions to ensure that the proper form was displayed. We clicked the edges of the body outline to ensure that the image map closely aligned with the outline. |
| 4. User Story: View product Summary | Database: Checked to see if proper data was displayed from appropriate tables. Tested if changes in backend data changed data in webpage. Tested if product use submission changed data in webpage. |
| 5. User Story: View reaction Summary | Database: Checked to see if proper data was displayed from appropriate tables. Tested if changes in backend data changed data in webpage. Tested if record reaction submission changed data in webpage. |
| 6. User Story: View Product Inventory | SQL query: We tested the query alone to make sure the correct product barcode and name was returned.<br>Webpage view: We tested that the proper row data appeared in the webpage table. We tested that the CSS properly styled the tables. |

## Problems encountered

The following table shows the problems that were encountered.

| User Story | Problems Encountered |
|---|---|
| 0.1 Environment Setup - Webpages | No problems were encountered. |
| 1. User Story: Add Product | Barcode Field: standard barcodes on most products are 13 digits long. We were using a number type input field which does not allow max or min lengths to be set. We chose to change the input type to text so that we could set the min and max lengths to 13 [*change based on grader feedback*] Database: we realized after the first sprint that to implement this story the way it was written in the user story, we needed to add a bridge table in the database to connect users to the products they use |
| 2. User Story: Record Product Use | Database Interaction: We ran into a problem with the feature where we automatically enter the date if the user fails to enter one. We had an issue where the database expected 'null' but a blank input from an HTML input tag returns a empty string. We had to fix that query to accept an empty string. |
| 3. User Story: Record Reaction | Database Interaction: We ran into the same problem described above where we automatically enter the date if the user fails to enter one. We had an issue where the database expected 'null' but a blank input from an HTML input tag returns an empty string. We had to fix that query to accept an empty string. We had an issue where the instanceID was not updating. The table that was accepting user reaction data must have unique rows. If a user selected the same body parts, and the reaction instanceID did not update, it was rejected. We fixed this. |
| 4. User Story: View product Summary | We decided to add the affected body parts to this report. That required us to collapse body parts that showed up in separate rows into a list for the reports. |
| 5. User Story: View reaction Summary | We decided to add the affected body parts to this report. That required us to collapse body parts that showed up in separate rows into a list for the reports. |
| 6. User Story: View Product Inventory | No problems were encountered. |

## Actual Time Required for Each Task

The following table shows the time require for each task in a given user story.

| User Story | Time Spent on Tasks |
|---|---|
| 0.1 Environment Setup - Webpages | Total: 1 hour |
| 1. User Story: Add Product | HTML: 1 hour; Server-side JS: 1.5 hours; SQL queries: 1 hour; Total: 3.5 hours |
| 2. User Story: Record Product Use | Modify SQL queries: 0.5 hrs; Client-side JavaScript: 3 hrs; Server-side routes, database connection, JavaScript: 3 hrs<br>Total: 7.5 hrs |
| 3. User Story: Record Reaction | Modify SQL queries: 0.5 hrs; Client-side JavaScript: 3 hrs; Server-side routes, database connection, JavaScript: 3 hrs<br>Total: 7.5 hrs |
| 4. User Story: View product Summary | HTML, CSS: 0.5 hours; Server-side JavaScript: 1.5 hours; SQL queries: 1 hour; Total: 3 hours |
| 5. User Story: View reaction Summary | HTML: 0.5 hour; Server-side JS: 1 hour; SQL queries: 1 hour; Total: 2.5 hours |
| 6. User Story: View Product Inventory | HTML, Handlebars, Express Server-side Javascript: 1 hr; SQL query: 0.5 hrs<br>Total: 1.5 hrs |

## Current Status

The following table shows the current status of each user story in the sprint.

| **User Story** | **Current Completion Status** |
|---|---|
| 0.1 Environment Setup - Webpages | Implemented and fully tested. |
| 1. User Story: Add Product | Implemented and fully tested. |
| 2. User Story: Record Product Use | Implemented and partially tested. There are so many possible combinations of products, descriptions, and body parts that every combination cannot be tested. |

| User Story | Current Completion Status |
|---|---|
| 3. User Story: Record Reaction | Implemented and partially tested. There are so many possible combinations of descriptions, and body parts that every combination cannot be tested. We carried out tests on many combinations, but we could not exhaustively test. |
| 4. User Story: View product Summary | Implemented and fully tested. |
| 5. User Story: View reaction Summary | Implemented and fully tested. |
| 6. User Story: View Product Inventory | Implemented and fully tested. |

## Remaining Work to Be Completed

The following table shows the remaining work to be completed for each user story in the sprint.

| User Story | Current Completion Status |
|---|---|
| 0.1 Environment Setup - Webpages | Complete. |
| 1. User Story: Add Product | Complete. |
| 2. User Story: Record Product Use | Complete. Minor bugs: 1) The product use instance number isn't displayed properly at the top of the HTML page. 2) The form is not robust to multiple submits. For example, if user enters a date and product, but forgets to enter a description before hitting submit, it cannot be added. 3) More on screen instructions should be given to the user to ensure that data is entered properly. More instructions could also disambiguate "location" and whether that means geospatially or bodily. However, the overall functionality required by the user story is complete. These bugs will be fixed in a future sprint. |
| 3. User Story: Record Reaction | Complete. Minor bugs: 1) The reaction instance number isn't displayed properly. 2) The form is not robust to multiple submits. For example, if user enters a date, but forgets to enter a description before hitting submit, it cannot be added. 3) More on screen instructions should be added. However, the overall functionality required by the user story is complete. These bugs will be fixed in a future sprint. |

| User Story | Current Completion Status |
|---|---|
| 4. User Story: View product Summary | Complete. Minor bugs: 1) It doesn't specify the left or right side of the body in report. However, the overall functionality required by the user story is complete. This bug will be fixed in a future sprint. |
| 5. User Story: View reaction Summary | Complete. Minor bugs: 1) It doesn't specify the left or right side of the body in report. However, the overall functionality required by the user story is complete. This bug will be fixed in a future sprint. |
| 6. User Story: View Product Inventory | Complete. |

# HW 5 Diagrams

## Usefulness of Spikes and Diagrams

### User Story 1: Add Product

The UML diagram for this user story was helpful, because it helped us identify what we did not do during the last sprint, and how to fix our code to fully implement the user story the way it was written.

### User Story 2: Record Product Use

No new UML diagrams or spikes were performed during Sprint 2 for this User Story. Work on this story was a continuation of Sprint 1, and the previous diagrams and spikes were sufficient.

### User Story 3: Record Reaction

No new UML diagrams or spikes were performed during Sprint 2 for this User Story. Work on this story was a continuation of Sprint 1, and the previous diagrams and spikes were sufficient.

### User Story 4: View Product Use Summary

The diagram was useful in that it captured how "recent summary" was being defined. This could have been just as easily captured in the user story however. If it had been captured in the user story, then the diagram would not have been as useful and would also not have been needed as this was a simple story.

### User Story 5: View Reaction Summary

The diagram wasn't that useful because all the necessary information was captured in the user story. This was a fairly simple user story, and was not complicated to implement so a UML diagram was not necessarily needed to help structure the code.

### User Story 6: View Product Inventory

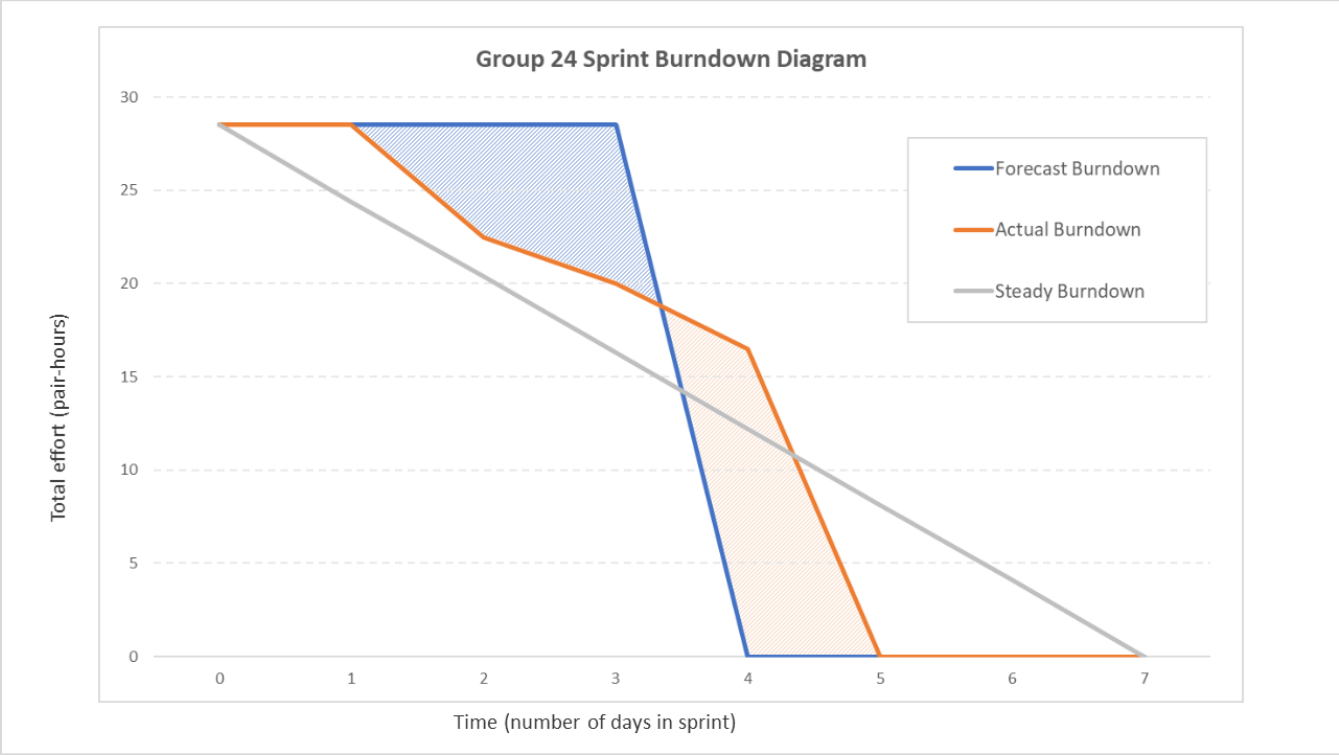
The simplicity of this story didn't necessitate creating a diagram or performing a spike.

## Sufficiency of HW 5 Diagrams

There are not any additional diagrams that we wish we had that were not mentioned in the last writeup. We continued to work on user stories from the last sprint, so there are no additional diagrams needed. One

exception is User Story 6, which is mostly a simple derivation of the product use summary from User Story 4. This was an add-on after we finished the promised stories and didn't need a spike or diagram.

## Sprint 2 Burndown diagram




The area shaded in blue shows that initially we were ahead of schedule, but as indicated by the area shaded in orange, we fell behind in our schedule. The steep drop shown in our diagram indicates that we had very tight deadlines since we had only a few working days in this sprint. All of our tasks fell due on one date. We successfully completed all of the tasks in our sprint.

## Refactoring

We did no refactoring during this sprint.

## Questions for Customer During the Sprint

Jonathan was pleased with the progress that we made on the app over the last two sprints. The priorities that laid out were followed, and we completed all the tasks promised by the end of Sprint 2. We walked Jonathan through all the webpages and their functionalities. He did not surprise us with any new requirement changes.

In preparation for Sprint 3, we conferred with Jonathan to set priorities. As in previous weeks, he continued to rank routine, non-app specific tasks at the bottom of the priority list and would like for us to continue with stories that showcase the product. Because we now have the ability to capture the most critical user product use and allergic reaction data, he wanted us to work on implementing the ability to actually associate product use with a specific reaction. However, we noticed from the shared homework documents this week that many groups began with a log in user story. Also, for our application in particular, different users are associated with different products, reaction instances, as well as other data. This functionality of allowing the application to switch between different users and display dynamic information for each user is currently absent because of our lack of a log in user story. He decided that maybe we should put that on our list.

## Integration Tests

### User Story 1: Add New Product

Dropdown Menu: Similar to the select menu in user story 2, the select menu in this user story, which is filled with products from the Products table in the database, needs to send a SQL query to the database to pull all of the products and dynamically add those products in <option> tags within the <select> tag hardcoded into the page. We tested that the SQL query for the menu was sent to the database, correctly returned all of the product names and their ids in the Products table, and then used those to dynamically build the product select menu for the user to see. We used the inspector in the browser and the user_products bridge menu to verify that the id values that were taken from the Products table were added to the value attribute of the <option> tags and that those id values matched the product names that were also added to the select menu. Finally, we tested that the submit button for this select menu sent the correct INSERT SQL query that added the product id to the user_product table.

### User Story 2: Record Product Use

Dropdown Menu: Creating the dropdown menu with the user's available products is one of the most complex parts of this user story. A hardcoded <select> tag in the webpage acts as a parent element for the <option> tags that are created dynamically with data (productName, id) from the products table in the database. We tested that the client side Javascript successfully sent a request to the server, that the server ran the correct SQL query, and the array of names and ids was properly returned to the calling function where the dropdown was created.

We tested that the option tags were assigned the correct value based on the id returned from the database associated with that product.
Product Use Form Submission: We tested that the submit buttons, which clicked, submitted to the correct Express routes on the server and ran the appropriate INSERT SQL queries. This was checked by logging and inspecting the query when it got to the server, looking in the appropriate database tables (for date, id, body parts, when, where, reaction description, etc), and by checking the reports webpage that displays the list of product use instances.

## User Story 3: Record Reaction

Reaction Form Submission: We tested that each of the two forms (one for submitting overall details, such as date, notes describing the place, and notes describing the reaction, and the other for specifying specific body parts) submitted to the proper route and ran the appropriate SQL INSERT queries by logging the query and inspecting the database tables and by checking the webpage that displays the list of reaction instances. Also, through verifying that the insertion worked by checking the reports webpage, we additionally tested the integration of all these parts plus the additional query to SELECT from the database and the code for displaying the report. This essentially tested the application from end to end.

## User Story 4: View Product Use Summary

This was ostensibly an easy user story to implement. Product summary report was tested to see if it had up-to-date data by documenting more product use instances through the application's product use form. After the data added by the form was confirmed to have been added by the database, the product use report was checked to see if it contained the new data. The data presented by the report was up-to-date and integrated with the application.

## User Story 5: View Reaction Summary

Reaction summary report was tested to see if it had up-to-date data by documenting more reaction instances through the application's reaction form. After adding the new reaction through the form and confirming with the database that the data had been added, the reaction summary report was checked to see if it contained the new data. The data presented by the report was up-to-date and integrated with the application.

## Sprint 3:

- User Story 1: As a registered user, I can log in and the application will authenticate my credentials. [5.5 hrs][Epic F] {Karen, Anthony} Due: Tuesday, June 11, 2019
    - Create login page [2.5 hrs]
    - Write SQL query that checks the user login credentials are correct [1hr]
    - Store the userID for use in other functionality [1 hr]
    - Incorporate URL into app's website [1hr]

- User Story 2 : As a registered user, I want the app to identify a likely allergen for me, in order to discover what I'm allergic to. [Part 1]
  [30 hrs] {Karen, John, Kendrick, Kris, Anthony} Due: Tuesday, June 11, 2019
    - Create algorithm to determine a "likely" allergen based on past allergic reactions and product use [30 hrs]
        - Formally define the problem as a machine learning (ML) problem [1hr]
        - Explore data [2hr]
        - Create and engineer features[1hr]
        - Create a training dataset, a test dataset and a validation dataset[1hr]
        - Create and train models (i.e. create algorithm)[8 hrs]
        - Evaluate trained models with validation data (i.e. validate algorithm)[2 hrs]
        - Model selection: Iterate until final model chosen[15 hrs]
        - Assess the final chosen model with test data (i.e. test algorithm)[1hr]

## Implementation Plan

| User Story | Task | Assigned To | Time Estimate | Due Date |
|---|---|---|---|---|
| 1. User Story: Login | Create login page | Karen; Anthony | 2.5 | 6/9/2019 |
| 1. User Story: Login | Write SQL query that checks the user login credentials are correct | Karen; Anthony | 1 | 6/10/2019 |
| 1. User Story: Login | Store the userID for use in other functionality | Karen; Anthony | 1 | 6/10/2019 |
| 1. User Story: Login | Incorporate URL into app's website | Karen; Anthony | 1 | 6/10/2019 |
| 2. User Story: Discover likely allergens | Formally define the problem as a machine learning (ML) problem | Karen; John; Kendrick; Kris; Anthony | 1 | 6/8/2019 |
| 2. User Story: Discover likely allergens | Explore data | Karen; John; Kendrick; Kris; Anthony | 2 | 6/9/2019 |
| 2. User Story: Discover likely allergens | Create and engineer features | Karen; John; Kendrick; Kris; Anthony | 1 | 6/10/2019 |
| 2. User Story: Discover likely allergens | Create a training dataset; a test dataset and a validation dataset | Karen; John; Kendrick; Kris; Anthony | 1 | 6/8/2019 |
| 2. User Story: Discover likely allergens | Create and train models (i.e. create algorithm) | Karen; John; Kendrick; Kris; Anthony | 8 | 6/10/2019 |
| 2. User Story: Discover likely allergens | Evaluate trained models with validation data (i.e. validate algorithm) | Karen; John; Kendrick; Kris; Anthony | 2 | 6/10/2019 |
| 2. User Story: Discover likely allergens | Model selection: Iterate until final model chosen | Karen; John; Kendrick; Kris; Anthony | 15 | 6/11/2019 |
| 2. User Story: Discover likely allergens | Assess the final chosen model with test data (i.e. test algorithm) | Karen; John; Kendrick; Kris; Anthony | 1 | 6/12/2019 |

## Customer Availability

Our customer was willing and able to meet with us on Wednesday, June 6, 2019. We reviewed the current progress and provided a demo of the work completed to date.

## Contributions of team members

| | Kendrick CHU | Anthony GIULIANO | John HASH | Kristopher HILL | Karen MCFARLAND |
|---|---|---|---|---|---|
| Pair programmed to implement tasks from Sprint 2 | X | X | X | X | X |
| Provided updates on the progress of Sprint 2 tasks | X | X | X | X | X |
| Attributed Sprint 3 user stories (due date, tasks, dependencies, story points) | | | | | X |
| Created Sprint 3 Implementation plan | X | X | X | X | X |
| Helped write the HW submission | X | | X | X | X |
| Burndown diagram | | | | | X |
| Stitch together document | | | | | X |
| Review final draft | | | X | X | X |
| Submit document | | | | | X |