**Local Foods Connection**

Carter Markegard, James Msekela, Liam Li, Nate Williams, Ricardo Xi, and Shuyi Qi

***Overview***

Our project is to develop a website for Local Foods Connection, a non-profit organization in Grinnell that, in the words of their mission statement, “partners with local farmers to provide fresh environmentally sustainable food and nutrition education to Grinnell residents in need.” Our website extends this mission by giving residents a tool for searching for locally-sourced food that they can access directly from the source and getting information about local-food centered community events.

The site is currently near-deployment. It contains a home page with a food provider search-bar, info cards, and a map, an events page which lists local recurring and one-time events, a sign-up page for food providers who want to be listed on the site, and an admin portal for both food providers and the page admin to edit site information.

***Major Parts of the Project + Gems***

Home page: Display all farms’ information in a collection of farm cards as well as pins on an embedded google map. Farms are searchable (including all fields) via the search bar.

Individual farm pages: Shows information (name, picture, description, address, phone number, etc.) of a farm.

Events page: Displays events (both recurring and one-time) in a grid of cards. Events can be added to this page via the Events tab in the admin portal.

Sign up page: Allows farmers to sign up for a farm on the website. Their farms will not be displayed until reviewed by the admin.

Sign in page: Allows registered farm owners to sign in and edit their own farm’s information; also allows admin to sign in, approve farms, edit farm information, delete farms, edit events, etc.

How these fit together: A Grinnell resident can browse a list or a map of local farms on the home page. If they are looking for a particular farm, or even a particular produce item or any other piece of farm information, they can search for it see a list of farms that include that search term in any of their fields. They can then click on the farm’s card to see the farm’s details. If they are interested in local food-related events, they can go to the ‘Events’ page. For a farm owner, if not a registered user, they can sign up by going to the ‘Sign Up’ page and filling out the farm form. Once registered, farm owners can edit farm information by signing in via the ‘Sign In’ Page. Laurel can approve and edit farm information by signing in via the same page and accessing the sign-in portal.

Gems: The main gems we used are ActiveAdmin & Devise for user system, Pundit for authorization, and Rspec & Capybara for testing. This semester for testing, we added Selenium-webdriver and headless-chrome for running tests in a chrome browser. We can use the debugger to pause the test (by inserting the line ‘debugger’ into the code) and inspect databases in the same way as we use rails console or JavaScript consoles in a Chrome browser. (Note: You might run into problems when using headless-chrome; if the error message says something like “expected chrome version xxx”, make sure to check your chrome browser version and update if necessary). More gems and documentation can be found in the Gemfile.

***Workflow***

● What worked

* Consistently meeting as a group every Friday.
* Conducting a compulsory weekly briefing on Friday’s with our project mentor Alex.
* Consistently assigning work in pairs.
* Open communication via slack channel between all group members and project mentor.
* We mostly assigned work based off of who wanted to do what. Because of this most of the pairs had no problem working on the same assignment/task for the next sprint as well.
* Before project presentations and demos each group member would work on preparing and speaking on whatever it is they worked on during the past sprint.

● What didn’t

* Roles remained fixed despite talk of rotating assigned roles (roles such as communicator) after fall break or around half way through the semester.
* Only met with the community partner twice throughout the semester. Ideally we would’ve loved to meet with Laurel more, however, she was extremely busy and was often unavailable.
* It was frequently the case that certain pairs (we tried to pair program consistently) often reprised their roles working on the same assignments. We might have wanted to assign different tasks and thus increase variability in the work each pair was doing every week.

***Behavior-Driven Development***

When we first got the code, we explored the happy paths for user, farm and admin to see if any features were broken. Since our project was mostly feature-complete, we created user stories based on our observation of what worked and what needed to be fixed, which we presented during our demo for Sprint 1. We then combined the user stories with the Remaining Work part in the last team’s final report to plan for what features to complete, add and modify. The user stories were also confirmed and updated with our community partner during meetings. For example, the Admin group showed Laurel the current features of the admin portal and found out that the admin would want to use the admin portal to edit and delete farms while any other farm is not allowed to do so. We thus implemented these two features according to the updated user stories. The other two groups also implemented features according to this pattern. In addition, our tests in the spec/ directory were written based on the desired behavior of the website. The features specs include different scenarios and use phrases like “it should…” to describe each spec. Overall, our development process reasonably follows the principles of BDD.

***Test-Driven Development***

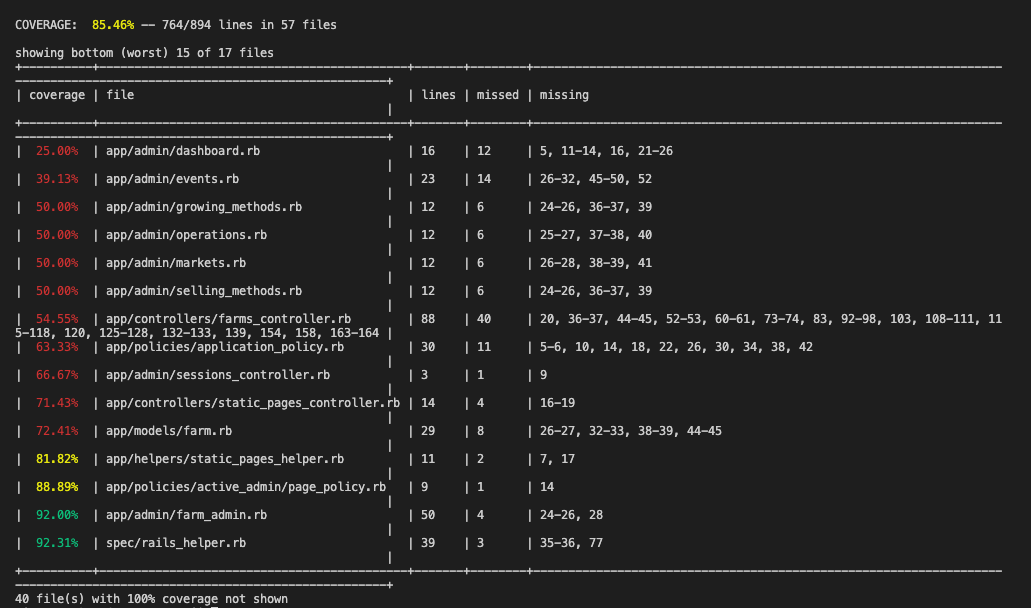
There were already tests for some features that were not correctly implemented. When we tried to fix errors in the code for the features, we used the tests to correct the code. This fits the “Add a test - Run all tests and see if the new test fails -write the code - run tests” cycle of Test-Driven development. We also went through the refactoring the code stage in the Test-Driven development cycle. It’s stated that “New code can be moved from where it was convenient for passing a test to where it more logically belongs.” When we merged any code from other branches to the master branch, we ran the tests and ensure that the code that has been merged is correct. Although we did not write tests strictly before we wrote the code to use tests as the software requirements, there were cases where the test group would communicate to the rest of the team about how a test should be written, which then raised a discussion on how a feature should be written, which led to a realization that two teams assumed that the same feature be developed in two different ways. Such discussions helped us clarify software requirements.

***Test Coverage***

In terms of feature coverage, the current specs cover all the functionalities of the website except for the map feature on the home page and its integration with the search feature. The completed specs are: ActiveAdmin controller spec, feature specs for checking whether header and footer elements are complete and whether each tab in the navigation bar links to the correct page, model specs that test for each of the six database, and the search feature on the home page. The map feature requires knowledge of testing for the Google Map API, which we did not get to explore yet because we spent a lot of effort debugging in order to have a JavaScript testing framework that is working. The future team will need to work on testing for the map by looking at how to test the Google Map API. When searching for a farm/any attributes of farms (like produce, address, phone number), only the farms that matched the search tokens should be shown on the map. So the integration of the map with the search function also needs to be tested by future teams.

The code coverage statistics are shown in the figure below, though the statistics do not accurately reflect our progress towards thorough testing. Most of the tests we completed this semester were for the search feature (written in JavaScript) and admin portal. The coverage statistics do not reflect coverage for JavaScript code but our tests covered all functions in the search.js file (we even fixed a small problem in one of the functions with the tests and testing tools). According to ActiveAdmin’s documentation, Dashboard.rb and other customized pages in the Admin portal don’t need to be tested because most of the functions are referenced to default functions in Activeadmin. Our tests already covered the customized functions in farm\_admin.rb by 92%.

**\*\*Note:** there is one spec in farm\_admin\_spec.rb and another in home\_page\_feature.rb that are not working after our final merge during the last sprint and thus needs debugging (we highlighted them in the comments in the code).

****

***Remaining Work***

● Write a better centering/zooming algorithm for the map w/ javascript. The current one breaks down if there’s one food provider really far away from the others.

● Putting photos on website with via AWS or otherwise/

Currently, photos of farmers that users see when they open the individual farm pages are pulled from the original local foods connection page. Future work can be done on migrating this functionality to AWS. If done, users, when signing up, can upload a photo of their choice that would be shown on the website.

● Work with the community partner to finish the integration of their website and our website. Right now, our website has a very similar view and navigation bar as the community partner’s SquareSpace website. For the following work, people could ask community partner or have their account to add the navigation to our website. Our plan for integration was to have every tab of both sites present on both sites and the tab-links will either take you to the page on our site or send you to the page on the other site.

● Style the signup page/make it look cute.

● Fix the html/css issue where the farm cards on the main page are taller than need be

Especially when there is only one card.

● There’s a bug where if you click on a farm marker, you are given the information of the last farm card from the latest search.

● The map doesn’t center on grinnell, because it runs a blank search immediately. It should center on Grinnell initially (unless you want to change that).

● Finish testing: one test in the spec/controller/admin/farm\_controller\_spec.rb is not working; two tests for the search feature in home\_page\_feature\_spec.rb did not work after our last merge; no tests for the Google Map.

***Retrospective***

● Accomplishments

1. Working Search Bar
2. Integrated search bar with farm cards and map
3. Created map zooming and centering algorithms
4. Vastly increased test coverage
5. Added and refined several admin tools
6. Differentiated the user and admin accounts via ActiveAdmin
7. Added stable method for maintaining events
8. Improved events page styling
9. Completed tests for the models and static page features
10. Cleaned up legacy code
11. A working configuration for testing JavaScript code
12. Added gems that allow the testing process to be shown in a browser
13. Added tests for admin portal and search feature

● Tools and Learning Resources

1. Hartl tutorials
2. YOUTUBE!!!
3. Our mentor Alex
4. Stack Overflow
5. If a feature is broken, check old commits via github to narrow down what isn’t working.
6. If using a framework, check the github tickets for that framework

● Success Tips

1. Meeting with community partner more and early.

Meeting is much more efficient than writing emails. In our case, community partner is extremely busy and the period between sending her an email and having her feedback can be a week. We need to wait for the community partner’s input but without postponing our project progress. We would suggest group having the same issue to schedule with the community partner ahead of time. Do not worry what if there is no problems would occur then. You will always have questions for the community partner.

1. Class Mentor is very helpful

Our project mentor, Alex, has been very available and helpful for our project. We usually get many problems solved at weekly one hour meeting with Alex. Another reason that we have efficient conversations with Alex is that we use Slack channel a lot. Communication tools helped us a lot. Don’t be afraid of reaching out to him.

***Relevant Links***

Notes from our first meeting with Laurel: <https://docs.google.com/document/d/1rQrZViJXPX_B_4aztUMChBbfhEL2N4rWdqIDTnWQYjo/edit?usp=sharing>

Google Drive Folder:

<https://drive.google.com/drive/folders/1cGkrOL0-iwDZ1nvLQCNqfxCU0leLzBy7?usp=sharing>

Slack Channel:

<https://app.slack.com/client/T03CTDJ27/CN7AX8WKB>

Heroku Deployment:

<https://localfoodssearch.herokuapp.com/>

Github Repo:

<https://github.com/CSC322-Grinnell/LFC_Features>

***Sprint 5 Time Reporting***

Time spent in meeting with your community partner: 0 hours

Time spent in meeting with your alumni mentor: 1.5 hours per person

Time spent in working on any project deliverables: 31 hours

Time spent in developing technical skills: 10 hours