Final Report

Bencoolen Software - CSCE 431 Singapore

Summary

As implemented, Bytes meets the customer needs we mention in our additional project planning. Food Insecurity is an often overlooked problem on college campuses, and the infrastructure already exists to allow for peer-to-peer sharing of university meal plans. Bytes takes advantage of this infrastructure and creates a platform where students can share university meal credits.

We've implemented functionalities for both integration with existing school infrastructure, as well as independent sharing. School integration is simple, and provides an interface for meal credit donation and reception that hooks up directly to a third-party school meal plan api. The independent functionalities lets students post their availability, and allows recipients to accept that availability and allow them to meet up at a predetermined location and time to receive a free swipe into a dining hall.

User Stories

Sprint 1

Landing Page (Tyler) ~2hrs

- Create Route for landing page
- Create a Controller (?) for landing page
- Create HTML/CSS for the landing page
    - Navigation to login page, etc
- Protected route for only people who are not logged in

Login Page (Bryan) ~3hrs

- Create a route for the login page
- Create a controller for the login page
- Create HTML/CSS for the login page
    - Denote donor account of recipient account
- Allow users to log in with OAuth
    - Strictly only using their school email address

Logout Button (Bryan) ~1hr if coordinating with loginpage

- Create a button for the logout feature

- Create a controller for the logout button
- Create HTML/CSS for logout button
- Should route to home page of site with no user logged in

Student Account Creation (Kaijie) ~3hrs

- Upon successful login with OAuth:
    - Search database for them (uin)
        - If they don't exist, make them an account
        - If they do exist, fetch their information
    - Access the number of meal credits student has
    - Redirect to a profile page (protected route)
    - Populate information

See Excess Credit Amount (Simone) ~2hrs

- Upon profile being populated
    - Create a field in profile page
    - Populate that field with the number of excess credits the student has
        - Show expiration date (?)
        - Donor: meal swipes left
        - Recipient: meal swipes received & unused
- HTML/CSS to show this (perhaps another page so, route, controller etc)

See Credit Donations (Nick) ~2hrs

- Create / Maintain a list of ts in profile data
    - Every time a user uses these credits, update this persons “donations”
- Create route for donation history page
- Create controller for donation history page
- HTML/CSS for page
- Populate donation history with this list

Transfer Credits (Keegan) ~5hrs

- Create a route for credit transfer
- Create controller & model functions for logic
- Create HTML/CSS for page where users can define information about the credits they want to give
- Once submitted, do updates on their number of credits
    - (per customer, once credits are donated that’s final)
- Define the logic that goes into pooling credits together.
    - Remember we’re tracking credits in a history, so we need to keep track of this, even if they’re pooled! (this is probably going to be hard, and maybe not the best way either)
    - Maybe a CreditDonation object? Id, uin, date_given. All up to you we just need to track. In receive credits we can do logic that splits up the poo

See Available Credits (Kaijie) ~2hrs

- Create view for displaying total credits available and/or max that the user can withdraw
- Create controller for updating the number
- Create HTML/CSS for the display
- Should be updated as users donate credits and as users withdraw/receive them.

Receive Credits (Kelvin) ~5hrs

- Create route for credit receiving
- Create controller & model functions for logic
- Create HTML/CSS for page where users can define information about the credits they want to receive
- Once submitted, do logic that gets from the credit pool database, deletes, and adds to recipient account
- *Work closely with the transfer credits person and get on the same page with how things should work*

Sprint 2

View updated excess credit amount (Keegan) ~2hrs

- Implement the necessary API calls to fetch and update the excess credit amount. ~.5hr
- Update the frontend components to display the latest excess credit amount. ~.5hr
- Implement tests for api logic ~1hr

Notify user when credit pool is low on request (Tyler F) ~3hrs

- Choose value to set threshold of min credits needed in pool ~.5hr
- Implement notification system that alerts user when pool is low ~1hr
- Display notification on UI ~1hr
- Develop tests ~1hr

Hide receive option from donors (Keegan) ~1hr

- Check if user is a donor ~.33hr
- Hide receive option if user is donor ~.33hr
- Write cucumber test ~.33hr

Hide donate option from recipients (Keegan) ~1hr

- Check if user is a recipient ~.33hr
- Hide donate option if user is a recipient ~.33hr
- Write cucumber test ~.33hr

Update recipient credits with API (Kelvin) ~2hrs

- Implement the necessary API calls to and update the recipients credit ~1hr
- Update the frontend components to display the current credit amount ~.5hr
- Update the frontend components to display the current credit amount ~.5hr

Visual Styling for Login and Landing page (Bryan) ~2hrs

- Style landing page using bootstrap ~1hrs
- Style login page using bootstrap ~1hrs

Visual styling for Receive and Donate Page (Bryan) ~1.5hrs

- Style transfer page ~.5hr
- Style receive page ~.5hr
- Style transaction detail page ~.5hr

Prerequisite for creating recipient account (Kaijie C) ~2.5hrs

- Establish criteria for creating a recipient account ~.5hr
- Check if user fits criteria when creating recipient account ~1hr
- Flash message if user does not meet criteria ~.5hr
- Implement required tests ~.5hr

Sprint 3

Visual Styling for the profile page ~ 1hr (Kelvin)

- Add CSS styles for home page ~1hr

Responsive Interface throughout the Website ~5hr (Simone)

- Make the design style consistent, kind of like a brand.
    - Develop a brand for Bytes ~2hr
- Overhaul existing CSS to flexbox to make sizing responsive & add branding style ~ 3hr
    - Landing Page ~0.5hr
    - Profile/Home page (work with above person) ~0.5hr
    - Donation page ~0.5hr
    - Transfer page ~0.5hr
    - Transaction View Page ~0.5hr
    - Credit Pool Page (work with below person) ~0.5hr

Credit Pool View ~6.5hr (Kaijie)

- Add the capability for a credit pool to track number of donated and received credits ~1.5hr
- Create page for a credit pool

    - It will show school name (credit pool name I guess) ~0.5hr
    - It will show total number of donated and received credits from a credit pool ~0.5hr
- Show daily metrics for donations and requests ~2hr
    - Could do this several ways, each transaction has a datetime on it, could query for that
- Make a graph of usage ~2hr

Button to switch user type ~ 2hrs (Kelvin)

- Create visual responsive button that indicates what type the user is switching to
- Add checks to make sure that the user can switch (i.e. a user with too much credits can't switch to a recipient)
- Create confirmation prompt to switching user type
- Create route to update user type

Metrics on home page ~ 4.5hrs (Keegan)

- Brainstorm a set of metrics that would be interesting to a new user ~0.5hr
- Gather data for that statistic ~2hr
- Develop an array on the home page that will randomly pick an element and show the user that statistic. (i.e. "10,100,234 meals provided!" Or "3414 students making a difference in their community!") ~2hr

White-label user interface ~2hr (Nick)

- Allow user to input their school based on their .edu email address ~0.5hr
- Update school logo according to student's school ~1.5hr

Sprint 4

Post in-person donation availability (Bryan) ~4hrs

- Create a table that handles in-person donations posting ~0.5hr
- Create a route for making a new in-person donation posting ~1hr
- Create a form page for posting in-person donation availability ~1.5hrs
    - User should be able to set time, date, location
    - Date can be a set date or a recurring weekday
- Create a button to go to form page from user's in-person donation listing page ~0.25hr
- Implement required tests ~0.75hr

Available in-person donation listing (Keegan) ~2hrs

- Create a page listing available in-person donations ~1.25hr
- Create a button in the navigation bar to reach this page ~0.25hr
- Implement required tests ~0.5hr

Recipient accept in-person donation (Simone) ~1.5hrs

- Create a route to accept an in-person donation ~0.5hr
- Create recurring button on in-person donation listing listings to accept an in-person donation posting ~0.25hr
    - Update database so recipient is associated with the accepted in-person donation
- Make accepted in-person donations not visible from the listings page ~0.5hr
- Implement required tests ~0.25hr

User's accepted in-person donations view (Nick) ~2hrs

- Create a page that shows all accepted in-person donations a user is a part of ~1.5hr
    - This page should work for donors and recipients and query accordingly
- Implement required tests ~0.5hr

Donor edit in-person donation posting (Tyler) ~2hrs

- Create button to edit screen for posting that hasn't been accepted by a recipient ~0.25hr
- Create a screen for editing existing unaccepted posting ~1.5hr
    - User should be able to edit time, date, and location or delete the posting
- Implement relevant tests ~0.5hr

Cancel in-person donation posting (Simone) ~2hr

- Create a route for canceling an in-person donation ~ 1hr
- Create a button so that users can cancel an accepted in-person donation posting ~.5hr
    - Other side of donation should be notified of cancelation
- Confirmation popup when user attempts to cancel ~.25hr
- Implement required tests ~.25hr

Confirming in-person donation (Keegan) ~1hr

- Create a button that allows recipients to confirm an in-person donation occurred ~.75hr
    - Confirm donation date was before the current date
- Implement relevant tests ~.25hr

Sprint 5

Polish Styling for all pages (Nick) ~3hrs

- Style profile page ~0.6hr

- Style transfer page ~0.6hr
- Style transactions page ~0.6hr
- Style meetings page ~0.6hr
- Style admin pages ~0.6hr

Admin User (Simone) ~2hrs

- Add an administrator option to the user model table ~0.5 hr
- Enable checking user table for admin permissions ~1 hr
    - Allow application to show admin-specific info to admin only
- Implement relevant tests ~0.5 hr

Admin Page (Tyler) ~2hrs

- Create an admin landing page ~1 hr
    - Should be mostly empty, features will be implemented in Admin Functionality
- Create a button on the main page linking to admin page ~0.5 hr
    - Only display button if user is an admin
- Implement relevant tests ~0.5 hr

Admin Functionalities (Kelvin) ~4hrs

- From the admin landing page, allow admin to view the school's credit pool ~1 hr
- Add capability for admin to manually add credits to the school's credit pool ~1 hr
- Add capability for admin to manually remove credits to the school's credit pool ~1.5 hr
- Implement relevant tests ~0.5 hr

Allow users to send feedback in the app (Kaijie) ~4hrs

- Create a controller to handle form submission and rendering ~0.5 hr
- Create routes for form actions ~0.5 hr
- Create a view for the form with text fields (name, email, message, etc) ~1 hr
- Use ActionMailer to facilitate sending of the form via email ~1 hr
- Add button to link to form at bottom of profile page ~0.5 hr
- Implement relevant tests ~0.5 hr

Merge School & CreditPools (Simone) ~2hrs

- Merge the School and Credit Pool tables into a single Credit Pool table ~2 hr

Legacy Software

Bytes was not a legacy project and did not start from a previous project's code, instead we started from scratch.

Team

Bryan

Bryan was the Product Owner in Sprint 3, the Scrum Master in Sprint 5, and was a developer for all other sprints.

Kaijie

Kaijie was the Scrum Master in Sprint 4 and was a developer for all other sprints.

Keegan

Keegan was the Scrum Master in Sprint 1 and the Product Owner in Sprint 5. No changes happened during those sprints, but in all of the others, he was a developer.

Kelvin

Kelvin was the Product Owner in Sprint 4 and a developer for all other sprints.

Nick

Nick was the Scrum Master in Sprint 2 and a developer for all other sprints.

Simone

Simone was the Product Owner in Sprint 2 and a developer for all other sprints.

Tyler

Tyler was the Product owner in Sprint 1 and the Scrum Master in Sprint 3. In all others, he was a developer.

Sprints

Sprint 1

The goal of Sprint 1 was to have a deployable proof of concept with basic features like profile creation, credit requesting, and credit transfer all with a CI/CD pipeline in Github. This work would provide a solid foundation for us to build on top of going forward. We would divide up user stories into smaller subtasks, prioritize tasks that are foundational, and assign developers to each story to be worked on.

We completed our sprint goal, we have a proof of concept deployed that lets users create an account with OAuth, request credits, and transfer credits– all with a ci/cd pipeline in github.

Tyler Fredericksen	1 points	5.6%
Keegan Reynolds	3 points	16.6%
Kelvin Zheng	3 points	16.6%
Simone Kang	2 points	11.1%
Bryan Yan	3 + 2 points	27.7%
Nick Ludwig	2 points	11.1%
Kaijie Chen	2 points	11.1%

User Stories for Sprint 1

Landing Page (1pt)

As **a *visitor of the website***,
in order to ***learn about the website and sign-in***,
I want ***a landing page that gives me information about the website, vision of the project, and how to join***

See Excess Credit Amount (2pts)

As a ***student with excess meal credits***,
in order to ***quickly see how many swipes I have to donate***,
I want a ***visible counter of credits I will have in excess***.

See Credit Donations (2pts)

As a ***donor student with excess credits,***
in order to ***know if my credits have been received and used,***
I want a ***page to look at data about my credit donations.***

Login page (3pts)

As a ***student***,
in order to ***get access to my account***,
I want ***the ability to sign into Bytes using my verified school email address***

Logout page/button (2pts)

As either **a donor or recipient user**,
So that I can **protect my information and prevent misuse of my credits**,
I want to be able to **log out of the service**.

Create Student Account (2pts)

As a ***student user***,
In order to ***access the features of the website***,
I want ***a way to create an account, and connect it with my meal plan***.

Transfer Credits (3pts)

As a ***student with excess credits***,
So that I can ***feed hungry students***,
I want to ***transfer some of my credits to another student***.

See Available Credits (2pts)

*As **a recipient of meal credits**,*
*So that I **know how many credits are available to me**,*
*I want to see **how many credits I can receive and/or the total number of credits available.***

Receive Credits (2pts)

As a **student in need of credits,**
So **that I can get aid in getting food**
I want **a way to request credits**

Sprint 2

The primary goal for sprint 2 was to work on the external API that mimics that TAMU meal service and integrate it with our MVP. We planned to enable the updating of credits within our meal plan system when a donation is made or received through the external API. We wanted certain options to be hidden on the landing page depending on the role of the account logged in. We also planned on enhancing the visual styling and user interface of the web application using the Bootstrap framework. There were a total of 12 points this sprint.

We achieved all of our goals this sprint. We have an external API service that is deployed and mimics the current TAMU meal service. This API is connected with the bytes app and when transactions are made in Bytes, the change is reflected in the API. There are now different things that are hidden based on the role of the user. Our website also looks more polished due to more styling done during the sprint.

Tyler Fredericksen	2 points	14.3%
Keegan Reynolds	3 points	21.1%
Kelvin Zheng	1 point	7.1%
Simone Kang	2 points	0.0%

Bryan Yan	4 points	28.6%
Nick Ludwig	0 points	0.0%
Kaijie Chen	2 points	14.3%

User Stories

View updated excess credit amount (1pt)

As a **student with excess credits**,
I want to **accurately see how many credits I have**,
so that I can **know how many credits I want to donate**.

Notify user when credit pool is low on request (2pts)

As a **food insecure student**,
I want to be **notified when the pool has insufficient credits for my request**,
so that I **know why my request isn't being fulfilled**.

Hide receive option from donors (1pt)

As a **student with excess credits**,
I want the **option to receive credits to be hidden from my view**,
so that I **cannot misuse the website's functionality**.

Hide donate option from recipients (1pt)

As a **food insecure student**,
I want the **option to donate credits to be hidden from my view**,
because I **won't be able to donate due to my lack of credits**.

Update recipient credits with API (1pt)

As a **food insecure student**,
I want my **credits to be promptly updated when I receive a donation**,
so that I can **use the credits for my meals right away**.

Visual Styling for Login and Landing page (2pts)

As **a user**,
I want a **nice interface when logging in,**
so that I can **have a more visually appealing experience**.

Visual styling for Receive and Donate Page (2pts)

As **a user**,
I want a **nice interface when viewing my profile,**

so that I can **have a more visually appealing experience**.

Prerequisite for creating recipient account (2pts)

As a **food insecure student,**
I want to **meet the criteria for signing up to be a recipient**,
So that I can make sure that **I am not misusing the website**.

Sprint 3

Our goal for Sprint 3 was to completely **finish our MVP for Bytes**. We wanted to have a complete application that looks and feels professional and real, including flexibility across a variety of users and devices. On top of this base, we would be able to start adding extra features in the future. There were a total 13 points this sprint.

Our sprint goal was achieved, and we have a fully completed MVP for the original vision of Bytes. We have an application that allows users to login with their student emails, create donor/recipient accounts, and make donor/recipient actions according to their meal swipe count that is integrated with our external API. This app is also styled consistently and has the functionality to integrate with multiple schools. There are also bonus statistics and metrics screens scattered throughout. This sprint we completed all our stories, putting the finishing touches for this first iteration of Bytes.

Tyler Fredericksen	0 points	0.0%
Keegan Reynolds	2 points	15.4%
Kelvin Zheng	3 point	23.1%
Simone Kang	3 points	23.1%
Bryan Yan	0 points	0.0%
Nick Ludwig	2 points	15.4%
Kaijie Chen	3 points	23.1%

User Stories for Sprint 3

Improved Styling for Profile Page (1pt)

As a **student user**,
I want **improved styling for the profile page,**
So I have **a more visually appealing experience**

Responsive Interface (3pt)

As a **user of a non-traditional screen size**,
I want **responsive elements**,
So that I can **have a better experience when accessing the website on my screen**.

Credit Pool View (3pt)

As a **student using Bytes,**
I want to **be able to see a page detailing statistics about my schools Credit Pool,**
So I can **have a better understanding of the impact students like me are having.**

Button to switch user type (2pt)

As a **student with changing needs,**
I want **a button that allows me to switch user type,**
So that **I can switch between receiving and donating credits based on my circumstances.**

Metrics on home page (2pt)

As a **student about to sign up for Bytes**
I want to **see encouraging statistics on the home page**
So that **I am more likely to see my potential impact and sign up for Bytes**

White-label the user interface (2pt)

As a **student using Bytes,**
I want to **be able to see my school's logo even if I'm not from Texas A&M,**
So that **I can have a better, more personal, user experience.**

Sprint 4

Our sprint goal for Sprint 4 was to implement the ability for donors and recipients to schedule meetups so that donors can do in-person meal swipe donations that aren't contingent on their school integrating Bytes. There were a total of 15 points this sprint.

Our sprint goal was achieved. With our current product at the end of the sprint, the donors are now able to create a donation posting specific to their request, like time and location. On the other hand, recipients can view available donation postings and accept donations. After accepting donations, both the recipients and donors can easily view their postings on a quick-to-easy access point. During this sprint, we have completed all of our stories and successfully implemented the feature that was requested by our sponsor.

Tyler Fredericksen	2 points	13.3%
Keegan Reynolds	3 points	20%

Kelvin Zheng	0 point	0%
Simone Kang	3 points	20%
Bryan Yan	4 points	26.7%
Nick Ludwig	3 points	20%
Kaijie Chen	0 points	0%

User Stories for Sprint 4

Post in-person donation availability (4pt)

As a **student with excess credits**,
I want to **post the locations and times that I'm able to do in-person donations**,
So that I can **better facilitate in person donations that work with my schedule.**

Available in-person donation listing (2pt)

As a **food insecure student,**
I want **a way to see and accept in-person donation postings,**
So that I can **request an in-person donation that best fits my schedule.**

Recipient accept in-person donation (2pt)

As a **food insecure student,**
I want to **accept an in-person donation posting,**
To **notify the donor that I wish to receive a meal-swipe at the posted time and location.**

User's accepted in-person donations view (3pt)

As a **student user,**
I want **a way to quickly see all accepted in-person donation postings I'm a part of,**
So I can **quickly discern when to expect someone to meet up with me for a donation.**

Donor edit in-person donation posting (2pt)

As a **student with excess credits,**
I want **the ability to edit and delete in-person donation postings that are not accepted,**
So I can **easily make small changes to postings and keep them up to date.**

Cancel in-person donation posting (1pt)

As a **student with excess credits,**
I want **the ability to cancel an in-person donation postings,**
So I can **notify recipients if I'm no longer able to make it.**

Confirming in-person donation (1pt)

As a **food insecure student,**
I want to **be able to confirm a transaction happened,**
In order to **give credit to the donor who donated a meal-swipe to me.**

Sprint 5

Our goal for Sprint 5 was to polish the user experience by creating an administrator interface for the Bytes app to allow admin control of the system, allow an avenue for user feedback, and improve the overall visual styling for increased cohesiveness and presentability. There was a total of 12 points during this sprint

Our sprint goal was achieved. The product now has the ability for a user to be an "administrator". This lets them view the overall health of their school's credit pool, and manually force changes to the amount of credits in the pool. Additionally, users of Bytes are now able to provide feedback to the devs via a form on the app.

Tyler Fredericksen	2 points	16.67%
Keegan Reynolds	0 points	0%
Kelvin Zheng	3 points	25%
Simone Kang	2 points	16.67%
Bryan Yan	0 points	0%
Nick Ludwig	2 points	16.67%
Kaijie Chen	3 points	25%

User Stories for Sprint 5

Polished Styling for all pages (2pt)

As a **student user**,
I want **improved visual styling for all of the pages**
So I have **a more visually appealing experience**

Admin User (2pt)

As someone **in power on my campus**
I want to be able to be an "administrator" user
So I can **access higher privileges**

Admin Page (2pt)

As someone **in power on my campus**
I want to be able to **view the admin dashboard**
So I can make any **necessary changes**

Admin Functionalities (3pt)

As someone **in power on my campus**
I want to be able to **view and edit my schools credit pool**
So I can **make necessary changes and view the health of my schools pool**

Allow users to send feedback in the app (3pt)

As a **user of Bytes**
I want to be able to **send feedback to the Bytes team**
So I can **voice any concerns I have about the app**

Merge School & CreditPools (chore)

Customer Meetings

We had customer meetings every 2 out of 3 weeks, one as a progress check and one as a retrospective/planning meeting.

Sprint 1 Meeting

We demoed to the customer on Thursday December 11th at 8:00am over Google Meet. The meeting was essentially a rundown of where we were at, how much we had done. We explained we were pretty much done with all of the features for the sprint, but needed to finish getting to 100% test coverage (this happened a day before the sprint ended). He said that was okay and we demoed the features of the MVP for him. He was satisfied with the MVP and offered some advice on where to go next, such as simplifying the receiving process but said we had to think about that more. We asked some questions about the mock university api we had spoken about the week before, and he offered some guidelines about how it needs to be able to fit with **any** university entity, not just TAMU.

Sprint 2 Meeting

We demoed to the customer on Tuesday, February 6th over Google Meet. We demoed the features of the mvp for him, and he was generally satisfied with the product. One main point of his feedback was that our design and styling needs to be congruous throughout the application, specifically our profile pages. His other point of feedback was that we need to make the application more flexible. He suggested an adapting layer to the api that should work with different schools and third parties. We then spoke about expectations for sprint 3.

Sprint 3 Meeting

We met with the customer on Tuesday, February 27th via Google Meets. In this meeting the customer shared his satisfaction with our current MVP and gave many suggestions for functionality that we can add in the next sprint(s). We are supposedly switching focus slightly, and moving towards a match-and-meet meal credit transfer system rather than going through schools themselves. This is for cases where the school we are trying to integrate with won't allow us to integrate within their meal systems. This functionality will be our main focus for the next sprint.

Sprint 4 Meeting

We met with the customer on Tuesday, March 26th via Google Meet. In this meeting, the customer was satisfied with our product with the newly implemented in-person donation feature. The main topic of discussion for the next sprint was improving the overall look and feel of the application. A new feature that our client expressed interest in was an admin role. The admin role will help manage the overall status of our application in case some sort of error occurs.

Sprint 5 Meeting

We met with our client on Tuesday, April 16 over Google Meet. In the meeting, we demonstrated our completed work with the customer, who was satisfied with the newly included features of the admin dashboard, capabilities, and the feedback form. We also discussed the plan for handing over the project to the next team that will continue to improve our app.

BDD/TDD Process

For this project we used Behavior Driven Development and Test Driven Development. We would write all of our tests before writing the code so we had a solid understanding of what the goal actually was before we started development. This was especially helpful with tools like cucumber, which allowed us to simulate what an entire workflow would look like before we actually had one made. Methodologies like this reduce the risk of getting stuck doing redundant or non-important work. Nobody wants to get through weeks of development for a sprint, only to learn that what they were developing wasn't actually what the client wanted or needed. TDD helps prevent this, and BDD as an extension keeps the end goal fresh in our minds with regular meetings with the customer to define specifications for what we should build through the lens of a customer and user.

Configuration Management

For this project, our configuration management was done through GitHub. We rely on the version control system provided by Git and the various features that it provides, such as branching and release management. In total, we had 57 feature branches, 415 commits, and 5 releases, one for each sprint. We also have implemented continuous integration and continuous

deployment with GitHub and Heroku, so that our latest changes on main will automatically deploy to the production environment on Heroku.

Heroku Production Problems

At times our code would work on our developer servers, but when we push to Heroku it would not work on deployment. One of the issues was that we have a local database for our developer environment, and an actual database on the production server. Additionally, these databases were different types, one being SQLite and the other being Postgres. Solving these issues came down to making sure all of the migrations we ran on local to make it work were also being run in production. Additionally we needed to make sure all the urls were correct in the production environment file.

Tools/Gems

We used several development tools & gems in this project. The largest was Github which let us collaborate with each other in an easy way. We also used CodeClimate & SimpleCov to generate code reports which gave us insights into the quality of our code and tests. These are important because it directly transfers to the quality of our final product.

In terms of gems, we used a lot of different ones. Capybara lets us write cucumber tests in an easy to understand way, using plain english to define functionality. Rubocop was another testing suite that offered code quality reports, and let us automatically fix some quality issues as well. Finally, OnmiAuth was an authentication gem that let us hook up Google Sign-in to Bytes in order to let users sign in with their school .edu email address to get to their school's pool.

Repo Contents

Everything that a person needs to deploy the project is in the GitHub repository except for credentials and an external API service that supports the project. They will have to follow the readme file on the repo to create their credentials. As for external API, we have created and linked an example repository below that will also need to be deployed for this project to be deployed and tested. Follow the readme in the example repository to make sure that it is set properly.

Important Links

Slack link

Github Repo

Tamu-Dining Github Repo

Pivotal Tracker

Heroku Site

Presentation Link

https://www.youtube.com/watch?v=qgmrS7lj04o&ab_channel=KeeganReynolds