CyberCRM
Final Report

August 6, 2024

Matthew Atanas

Patrick Cannell

Noam Gariani

Christopher Muniz

Will Tatum

CyberCRM

Contents

Executive Summary

The main customer need was to be able to track student information using a more efficient method than spreadsheets. To solve this problem, the application allows for the customer to create, read, update, and delete students and their records as the client sees fit. This directly addresses their main concern of mass data management and is created to allow the client to efficiently scale the size of the data and integrate it with the existing method. To accomplish the goal of integration with the current process, the project implements an export and import for CSV files. Along with this, in order to further address the client's needs, data visualization capabilities were added for the various fields of interest.

The stakeholders in this project are the client John Romero along with other program directors under the Texas A&M Cyber Security Center. These will be the main users of the application with potentially having student workers performing administrative tasks. One of the client's concerns was privacy and this was addressed by making the super user unable to access the students view. This prevents anyone with super user access from seeing private notes or other information. Along with that, role based access control prevents student workers from messing with any important information and keeps program directors from accessing global information such as the audit log.

Links

Combined Video: https://youtu.be/Zw9q37Ps10M

Presentation Video: https://youtu.be/-cK5os5YDec

Demo Video: https://youtu.be/0OegA46IqjY

-

Website: https://cybercrm-7ccb791b98d3.herokuapp.com/

Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2705347

GitHub: https://github.com/tamu-edu-students/CyberCRM

Google Drive: https://drive.google.com/drive/u/2/folders/0APLVAcl1WKtoUk9PVA

Code Climate: https://codeclimate.com/github/tamu-edu-students/CyberCRM

Sprint Information

Total

Getting a working website on Heroku that had proper user authentication and RBAC controls that was connected to the postgres database. The website was able to show student data in a table where the data was displayed in a way that could be filtered and sorted along with having full CRUD capabilities. The application also many other abilities:

- Add notes (full crud)
- Audit logs
- Add custom options and attributes
- Change views as needed
- Custom home page with data analytics that can be dynamically generated

Story Points:

Matthew Atanas	10
Patrick Cannell	18
Noam Gariani	38
Christopher Muniz	12
Will Tatum	33

Sprint 1

Get a website deployed on Heroku. Have the website require authentication to see data, and complete a basic layout of the website. Connect the postgresql database and have data be displayed.

Story Points:

Matthew Atanas	0
Patrick Cannell	2
Noam Gariani	3
Christopher Muniz	2
Will Tatum	3

Sprint 2

Have the ability to display data from the database as well as creating new changes and updating as needed. With the data that is displayed it should be able to be sorted and

filtered through. The pages of the website will also be better designed and more pages will be added as needed.

Story Points:

Matthew Atanas	1
Patrick Cannell	3
Noam Gariani	12
Christopher Muniz	2
Will Tatum	6

Sprint 3

This sprint was building out functionality along with polishing issues from prior sprints.

- add the ability to create notes for students and restrict notes access to only that program director that created it.
- For super users managing the site, functionality to create, update, read, and delete users will be added.
- For the different programs in the cybersecurity center, there will be tables connected with the students table that display information on scholarships and cybercenter programs.
- Information will be segmented where program directors only see their programs and student workers have minimal access to information about other students.
- Additionally, on the navigation bar you will see your image icon, name, email, and other information deemed relevant in an appropriate spot.
- Tables will now have the deactivate and activate feature rather than the delete button. Along with that, having the entire row be the show button rather than a dedicated column for it. There will also be input validation on user input.
- Lastly, there will be cleanup like fixing RBAC to where only authenticated users can see the associated pages and forms will have the correct input fields.

Story Points:

Matthew Atanas	4
Patrick Cannell	5
Noam Gariani	11
Christopher Muniz	3
Will Tatum	10

Sprint 4

This sprint we further implemented and fix things that have not been completed in previous sprints:

- Fixing and building upon the custom attributes
- Adding full CRUD to notes and being able to filter through them
- Improve access control for the different roles
- The ability to deactivate instead of delete
- Adding more tables and connecting it to the student table

In addition to these tasks we finished the website styling to make it a finished product with some more functionality and features. This includes having built out the home page content, statistics for students and programs, and the ability to change views given a higher role. Lastly, for security purposes we included logging capabilities for every action performed on the website.

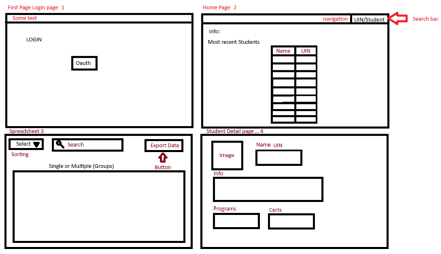
Story Points:

Matthew Atanas	5
Patrick Cannell	8
Noam Gariani	12
Christopher Muniz	5
Will Tatum	14

Design Diagrams (UI and ERD)

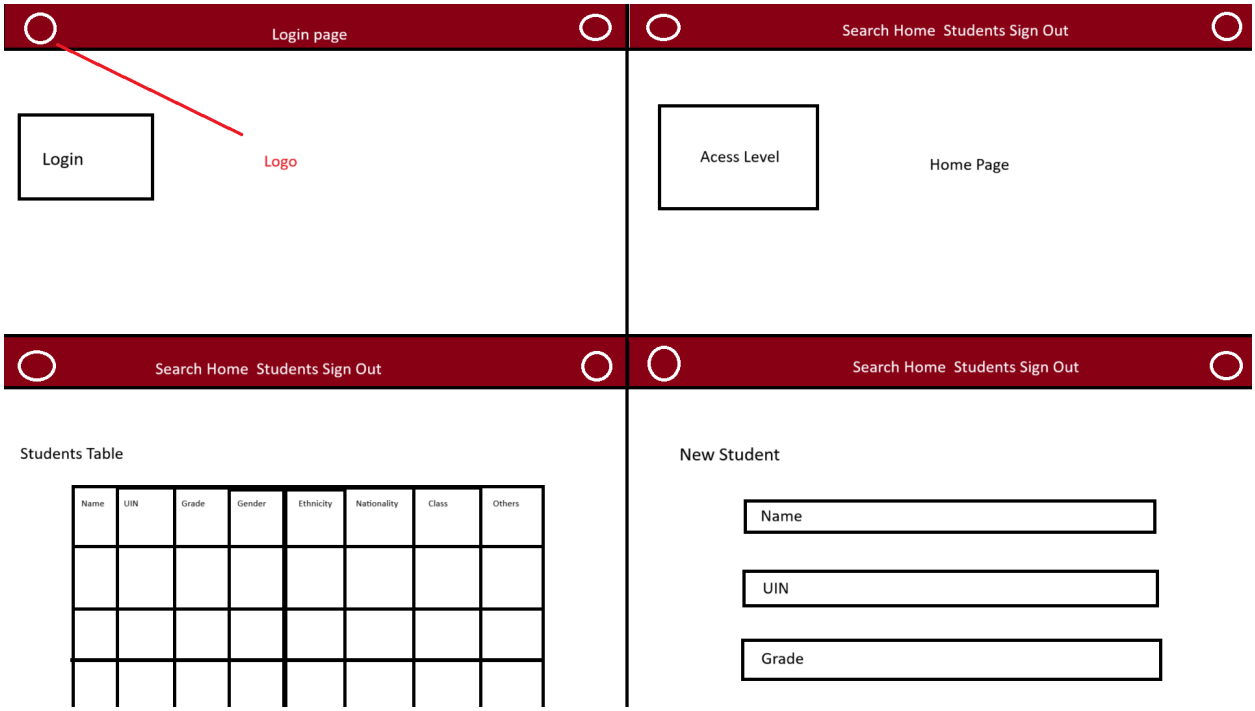
UI Design

Sprint 1



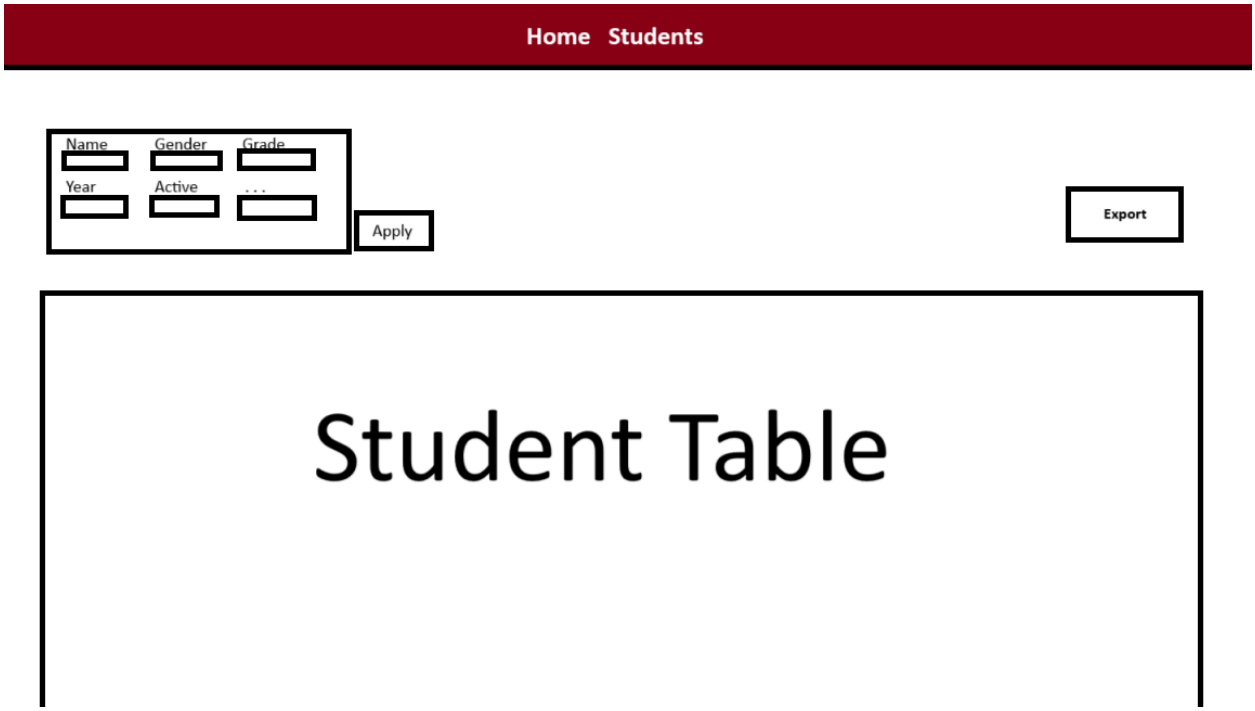
Sprint 2

This is a more streamlined version of our original design, and is the basis for how we wanted the website to look after this sprint.



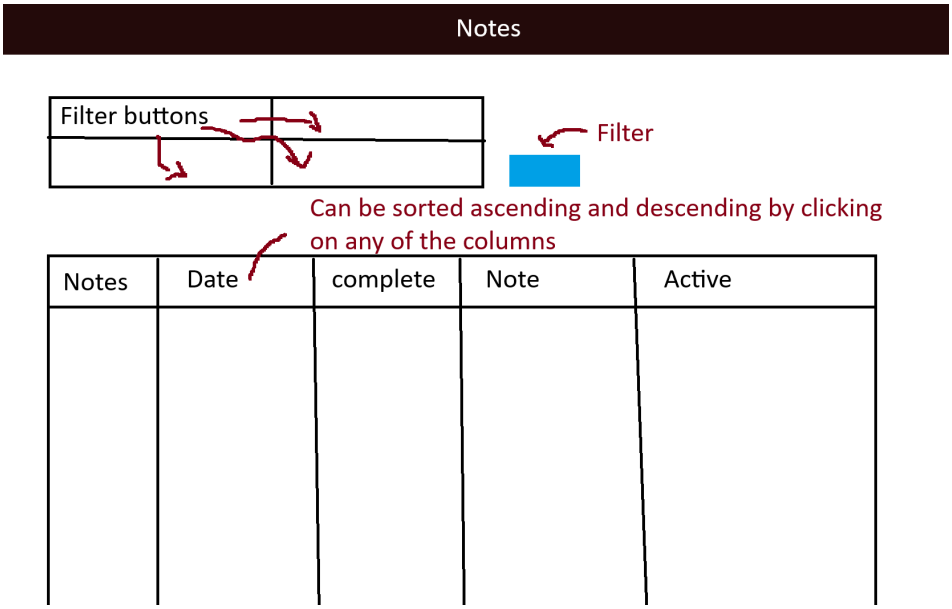
Sprint 3

This is a rough sketch of how the filters would look on the website.



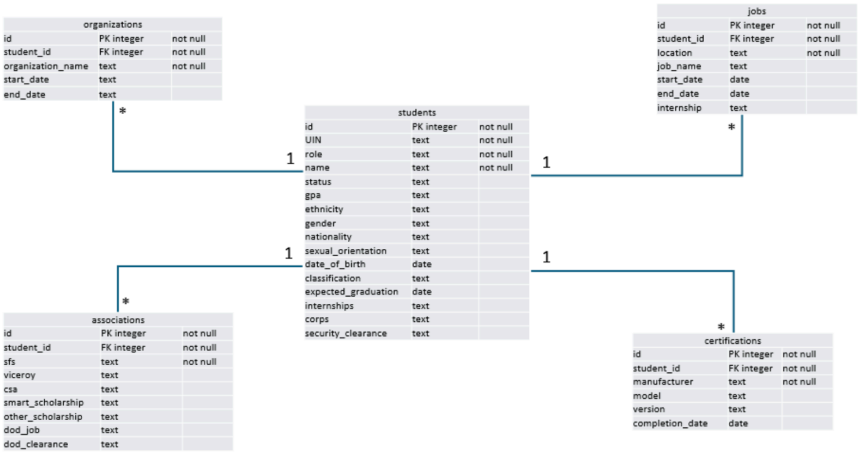
Sprint 4

This is a rough sketch of how the filters would look alongside the notes table.

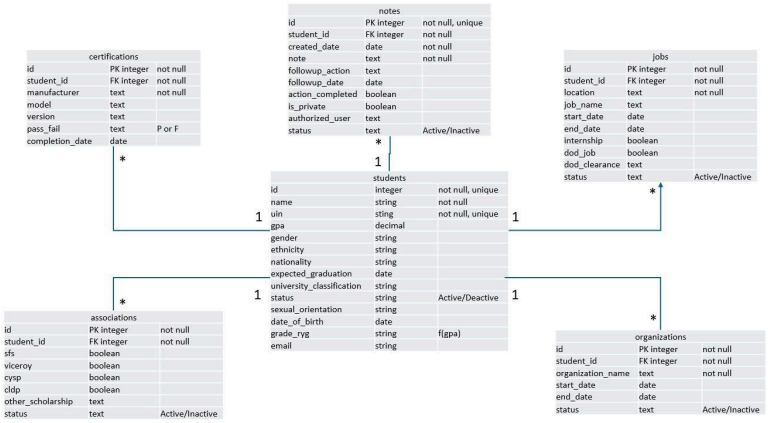


Entity-Relationship Diagram

Sprint 1



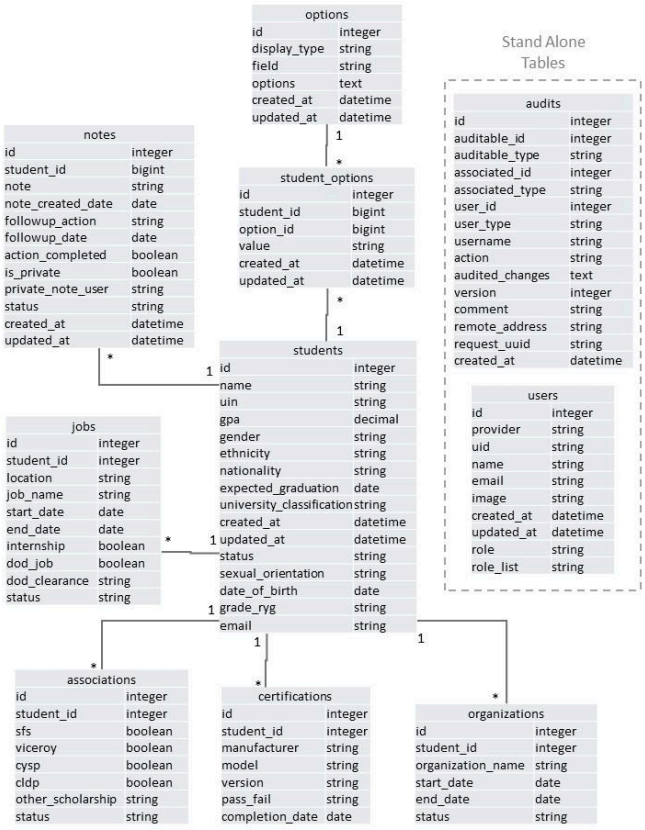
Sprint 2



Sprint 3

The biggest new addition is the notes table. A major piece of functionality is the ability to record and date meetings with students. Additionally, a status column will be added to each table except for certifications. The status column enables soft delete so that records can vanish but remain in the database. The students table is the only table that has been created, and the table is populated with realistic data.

Sprint 4



Project Management

Configuration Management Approach

We didn't use spikes because we didn't know it existed at the time. However in retrospect it might have been useful to research activities to better assess how long a given story will actually take.

For every story the individual assigned to that made an associated branch on GitHub and when it had some work done it was pushed to GitHub but a pull request (PR) was not made yet. Once work had been mostly completed the PR was made, PR link submitted to pivotal tracker, and the GitHub actions were run to check correctness of the PR before merging to main. If everything passes then there is a required review of the code before it could be merged. Once it is reviewed it can be merged and the branch deleted. We did have a "dev" branch that was supposed to be an intermediary between the main branch

that is always correctly deployed and the to be merged feature branch however we ended up not using it much throughout the project.

We had four major releases with tags that were submitted to Canvas based on each sprint. There were many more deployments near the beginning of the project to get Heroku working and then later it was almost every time that something new was added to the main branch.

Team Roles

The team roles rotated every week so everyone would have a chance to be in a leadership position. Below are the roles along with the sprints associated with who was that role in the given sprint.

Christopher Muniz

Sprint 2 - Product Owner

Sprint 4 - Scrum Master

Matthew Atanas

Sprint 1 - Scrum Master

Noam Gariani

Sprint 1 - Product Owner

Sprint 2 - Scrum Master

Patrick Cannell

Sprint 3 - Product Owner

Will Tatum

Sprint 3 - Scrum Master

Sprint 4 - Product Owner

Tools

Heroku

Most of the issues with Heroku was the initial setup of the website and database to be deployed. Once this was completed in the first sprint and many hours spent on figuring out how to work with Heroku it tended to operate fine. However there were some weird issues that occurred throughout the starting sprints like missing gemfile or procfile but it just wasn't recognized. And additionally the standard Ruby buildpack didn't work and a

different one had to be used to fix other issues. There were many minor issues like this that took a long time to figure out. Lastly, the database setup with Github and connecting it with environment variables was difficult and we were overage in cost. This was fixed in later sprints by decreasing the postgres database abilities in Heroku.

Initially automatic deployments were used until there were many failed deployments occurring on every push to main. From here it was changed to manual deployments. It was also realized that for every deployment the database had to be migrated again. And near the beginning there were times where it had to be reseeded and even completely reset. Because of this manual deployments that gave more control were better. In the last sprint where we had more stuff done automatic deployments were turned back on. For any deployment that went the migration in the database still had to be done. Sometimes rubocop would break a line of code that broke the deployment.

Gems

Here is a list of notable gems that we used past the generated ones:

- csv
  - To work with csv's
- brakeman
  - security auditing
- rubocop, rubocop-capybara, rubocop-factory_bot, rubocop-rails, rubocop-rspec, rubocop-rspec_rails
  - Code Quality
- simplecov, simplecov-json
  - Code coverage
- omniauth, omniauth-google-oauth2, omniauth-rails_csrf_protection, omniauth-test
  - Authentication
- chartkick
  - Making the statistics and charts
- tailwindcss
  - Making css easy
- pg
  - postgres

Of course there were many other gems under different environments as well like "test", "development", and "production". And some other gems that weren't covered but used for the project. Many gems were used to streamline development.

These were used for the various features that we added. Many of the gems were useful for developing features very quickly because it was built in. Other gems were used for code quality and security auditing. This site:

https://gorails.com/tool_categories

Was a great resource for finding gems that would be useful and even thinking about new things to add within the website. Many things are prebuilt and don't need to be manually built because of the existing gems that are out there.

GitHub Actions

We used 4 GitHub actions.

1. **Test** - Includes: cucumber, rspec, simplecov
2. **Lint** - Includes: rubocop, brakeman
3. **CodeClimate**
4. **Pivotal Tracker**

First Github Action (Test):

This was used to run the rspec and cucumber tests that were made and used simplecov to integrate into the application. If any test was failing then the action would fail and an additional implementation that was added with simplecov was not allowing it to be below 90% which was a course requirement. Then by making the github action required it made it to where any pull request (pr) that was not passing a test for some reason or decreased the code coverage to below 90% could not be merged.

- **Benefits**
    - This was useful to make sure the new code did not break any existing code
    - Maintaining a high code coverage.
- **Problem**
    - This first test was sometimes annoying because it took so long to run as we made many more tests.

Second Github Action (Lint):

This action was the one that ran rubocop and security audit to make sure the project and code quality was good. If there were any warnings from rubocop that were not covered the action would fail. If there were any recent security vulnerabilities found by the brakeman gem then it would fail. This was a required github action so without it passing the pull request could not be merged.

- **Benefit**
    - This ensured high code quality with little to no security vulnerabilities. It also made sure that this part of the requirement for the project was satisfied.
- **Problem**
    - In one of the stories there was a security vulnerability in the implementation of someone's story. This stopped it from being merged however there wasn't enough time left in the sprint to fix the security vulnerability so it was marked as ignored so it could be merged to main. This is not an ideal solution but it was done.

Third Github Action (CodeClimate):

This GitHub action was fairly simple. It just pushed the test report to CodeClimate every time a merge to main was made. It then would show up on the CodeClimate page automatically.

- **Benefit**
    - We did not have to manually put it into CodeClimate and it would update automatically on every push to main.
- **Problem**
    - To get this GitHub action to work took a while to figure out because of simplecov. There were many issues but a notable one was that every reference of simplecov, cucumber, or rspec had to be at the very beginning of the file

Fourth Github Action (Pivotal):

This one always passed because the intention was to connect every commit to pivotal based on putting the id in the commit.

- **Benefit**
    - This would have been very convenient to connect every commit without doing it manually to the pivotal tracker. And it ran fairly quickly.
- **Problem**
    - This Github action never actually worked and we still had to put in the PR link manually to connect it to the pivotal story.

CodeClimate

We used CodeClimate to upload the test coverage and see any additional information that it had to show. Many smells and other warnings were fixed as needed. It was beneficial to easily display and show the coverage and some problems were getting the github action to work properly to submit the code to CodeClimate.

SimpleCov

We used this gem to get cucumber and rspec code coverage to upload and work. It was very useful and used with github actions. Some problems were that it was hard to get working sometimes because of some weird bugs on placement of simplecov statements.

Testing

For BDD we used cucumber and for TDD we used rspec. It worked very well and helped as needed however there was a learning curve which was the largest problem.

Repo Contents

The code itself is needed (cybercrm) to run and then there is the ".github" folder which contains all of the github actions. The gemfile contains all the needed gems and they tend to need to be updated with time.

Customer Meetings

Date: June 17, 2024

The meeting showed the current iteration of basic website deployment. There wasn't much to show in this meeting but he mentioned that there was no ability to search for students or see students in an effective way.

Date: June 26, 2024

The meeting lasted around 1 hour and he went over the design of the database. When he did the user study he mentioned that he would like to have more attributes and changed attributes for the student table. He mentioned that in detail in the meeting and notes were taken. He also mentioned how he wanted the search bar to work as it was in the home page but not functional. Additionally, he wanted to be able to sort through the students to be able to quickly see a student that he would like to pull up. Lastly, he mentioned that he would like to see notes for each student.

Date: July 10, 2024

The team met with John Romero for 75 minutes. Topics discussed were menu options, notes, Program Director vs Super User, custom fields, and soft delete. When editing or creating a student, the client wanted a drop menu to appear so that junk data is not added to the database. The notes section of the app will record commentary on students and dates for follow up actions. Regarding the Program Director role, the client required the only the Program Director role can edit and view students. The Super User role can update the user roles. Custom fields creation was mentioned so that future users can add or remove fields from the students table. Finally, Romero specified that that data should never be permanently deleted; the client wants soft delete functionality.

Date: July 31, 2024

The team met with John Romero for about 30 minutes. We went through the entirety of the website and he particularly liked the implementation of the audit logs and the ability to add custom options. Topics discussed were custom options, notes, changing roles without logging out, custom fields, student statistics, archives, filtering for the notes, and audit logs. He wanted the ability to change an option that was added in case it was misspelled and some other minor changes through the demo.

He mentioned that he wants continued work on this project so that the Cybersecurity Center could use this application. The survey will be sent out pending the completion of the final report.

-

All the stories for the given meeting could be seen under user stories with the corresponding sprint. Each demo was fully comprehensive and was able to show every story that was completed per given sprint.

User Stories

Description of all user stories (including revised/refactored stories in the case of legacy projects). For each story, explain how many points you gave it, explain the implementation status, including those that did not get implemented. Discuss changes to each story as they went.

Sprint 1

- As a program director, I need people to authenticate to their correct roles.
    - Assigned: Will Tatum
    - Completion: Done

- As a grader for this project, I need this to be a website hosted on Heroku.
    - Assigned: Noam Gariani
    - Completion: Done
- As a user, I want to see some basic page layouts.
    - Assigned: Christopher Muniz
    - Completion: Done
- As a program director I need access to personally identifiable information.
    - Assigned: Patrick Cannell
    - Completion: Done
- As a program director, I need to have a list of students sorted by name.
    - Assigned: Matthew Atanas
    - Completion: Removed due to being unable to complete it on time due to
    - complexities with Ruby setup. Will be put in the next sprint.

Sprint 2

- As a program director, I need to be able to delete a student.
    - Assigned: Noam Gariani
    - Completed: yes
- As a program director, I need to update student information as needed.
    - Assigned: Noam Gariani
    - Completed: yes
- As a program director, I need to be able to read all the information about a specified student.
    - Assigned: Noam Gariani
    - Completed: yes
- As a program director, I need to create a new student.
    - Assigned: Noam Gariani

  - Completed: yes

- As a program director, I need to be able to search through student data in a spreadsheet-like view to manage and easily see student data.
  - Assigned: Will Tatum
  - Completed: yes
- As a program director, I need to have information about students stored.
  - Assigned: Patrick Cannell, Noam Gariani
  - Completed: yes
- As a program director, I need to be able to sort through student data in a spreadsheet-like view to manage and easily see student data.
  - Assigned: Will Tatum
  - Completed: yes
- As a program director, I need the pages to be functional and designed well.
  - Assigned: Christopher Muniz, Noam Gariani
  - Completed: yes
- As a program director, I need to be able to read all the information about a specified user.
  - Assigned: Matthew Atanas
  - Completed: yes, although it changed to “As an unknown user, I need to be able to see my current status for permissions”

Sprint 3

- As a user, I want to see my information in an appropriate location.
  - Assigned: Matthew Atanas
  - Completed: July 9th
- As a program director, I need my inputs to be properly sanitized.
  - Assigned: Noam Gariani
  - Completed: yes, July 8th

- As a program director, I need to be able to click a row in the table to view it.
    - Assigned: Noam Gariani
    - Completed: yes, July 8th
- As a program director, I need to prevent bad actors from accessing data they should not.
    - Assigned: Noam Gariani
    - Completed: yes, July 8th
- As a program director, I need to be able to import CSV data to fill in student information quickly.
    - Assigned: Noam Gariani
    - Completed: yes, July 8th
- As a program director, I need to be able to create users.
    - Assigned: Will Tatum
    - Completed: yes, late July 12th
    - Late because focusing on OAUTH testing and GitHub Actions
- As a program director, I need to be able to read user data.
    - Assigned: Will Tatum
    - Completed: yes, late July 12th
    - Late because focusing on OAUTH testing
- As a program director, I need to be able to update user information.
    - Assigned: Will Tatum
    - Completed: yes, late July 12th
    - Late because focusing on OAUTH testing and GitHub Actions
- As a program director, I need to be able to delete users.
    - Assigned: Will Tatum
    - Completed: yes, late July 12th

    - Late because focusing on OAUTH testing and GitHub Actions

- As a program director, I need to be able to add and delete custom attributes in the student table.
    - Assigned: Will Tatum
    - Completed: yes, late July 12th
    - Late because focusing on OAUTH testing and GitHub Actions
- As a program director, I need to have a well designed user CRUD control page.
    - Assigned: Noam Gariani
    - Completed: yes, July 12th
- As a program director, I need to have the colors of the grade column correspond to the text.
    - Assigned: Noam Gariani
    - Completed: yes, July 11th
- As a program director, I need information segmented based on role.
    - Assigned: Matthew Atanas
    - Completed: July 14th
    - Late because I focused on other assignments.
- As a program director, I need to implement the notes table
    - Assigned: Patrick Cannell
    - Completed: July 17th
    - Late because the story was much more complex than initially anticipated. The story entailed foreign keys, pulling from data from the database into the controller using special join commands, and creating an entirely new page that the user could navigate to from the show student page. Also, the notes table was seeded with realistic sample data.
- As a program director, I need to be able to have advanced filter functionality for the student table.
    - Assigned: Christopher Muniz

- Completed: July 17th
- Late because I couldn't fix merge conflicts so re-coded on a newer version of the project and had issues with cucumber.

Sprint 4

- As a user, I want to see my information in an appropriate location.
  - Assigned: Matthew Atanas
  - Completed: July 9th
- As a program director, I need my inputs to be properly sanitized.
  - Assigned: Noam Gariani
  - Completed: yes, July 8th
- As a program director, I need to be able to click a row in the table to view it.
  - Assigned: Noam Gariani
  - Completed: yes, July 8th
- As a program director, I need to prevent bad actors from accessing data they should not.
  - Assigned: Noam Gariani
  - Completed: yes, July 8th
- As a program director, I need to be able to import CSV data to fill in student information quickly.
  - Assigned: Noam Gariani
  - Completed: yes, July 8th
- As a program director, I need to be able to create users.
  - Assigned: Will Tatum
  - Completed: yes, late July 12th
  - Late because focusing on OAUTH testing and GitHub Actions
- As a program director, I need to be able to read user data.

  - Assigned: Will Tatum
  - Completed: yes, late July 12th
  - Late because focusing on OAUTH testing
- As a program director, I need to be able to update user information.
  - Assigned: Will Tatum
  - Completed: yes, late July 12th
  - Late because focusing on OAUTH testing and GitHub Actions
- As a program director, I need to be able to delete users.
  - Assigned: Will Tatum
  - Completed: yes, late July 12th
  - Late because focusing on OAUTH testing and GitHub Actions
- As a program director, I need to be able to add and delete custom attributes in the student table.
  - Assigned: Will Tatum
  - Completed: yes, late July 12th
  - Late because focusing on OAUTH testing and GitHub Actions
- As a program director, I need to have a well designed user CRUD control page.
  - Assigned: Noam Gariani
  - Completed: yes, July 12th
- As a program director, I need to have the colors of the grade column correspond to the text.
  - Assigned: Noam Gariani
  - Completed: yes, July 11th
- As a program director, I need information segmented based on role.
  - Assigned: Matthew Atanas
  - Completed: July 14th

    - Late because I focused on other assignments.

- As a program director, I need to implement the notes table
    - Assigned: Patrick Cannell
    - Completed: July 17th
    - Late because the story was much more complex than initially anticipated. The story entailed foreign keys, pulling from data from the database into the controller using special join commands, and creating an entirely new page that the user could navigate to from the show student page. Also, the notes table was seeded with realistic sample data.
    - Note that notes table full CRUD and private notes weren't in the final product because of merge conflict. It is still on a feature branch on GitHub.
- As a program director, I need to be able to have advanced filter functionality for the student table.
    - Assigned: Christopher Muniz
    - Completed: July 17th
    - Late because I couldn't fix merge conflicts so re-coded on a newer version of the project and had issues with cucumber.