Project Null
Group 114 - 4
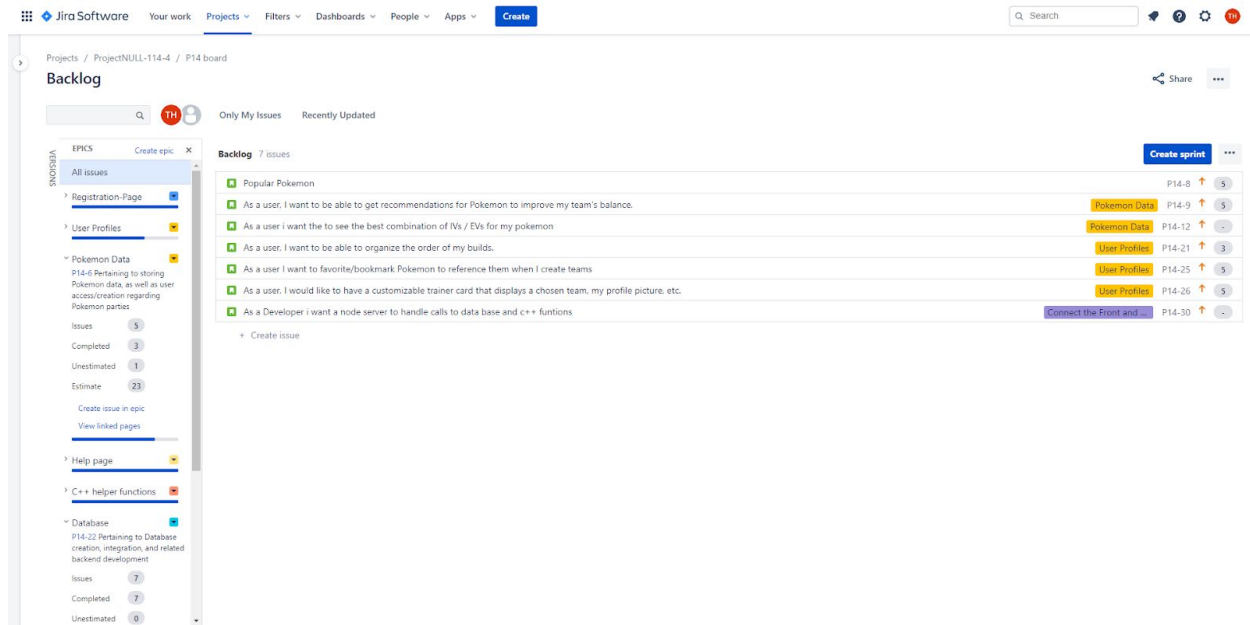
Ben Deguire
Trustin Pham
Christian Polanco
Kaleb Park
Taryn Hovenkamp
Jamie Joung

Project Description (200 words): Project Null is a Pokemon Team Builder created for the purpose of organizing teams for both casual and competitive purposes. Pokemon is a video game that many players enjoy playing at a competitive level, as it requires lots of strategy and planning. Every Pokemon has unique statistics called EVs, which are important to players to create the strongest teams. The Pokemon Team Builder allows users to customize these EV values as they see fit, using our web application to organize their existing teams, or possible teams for the future. As a user, you can create an account where you can build your teams and save them in order to access them later, as well as customize their features, such as whether they are shiny (color variation) or giving them nicknames to reflect a real team or for fun. After saving a team, you can log in and access them once more via the profile page. The user profile page allows the user to preview all of their teams at once, checking the stats you set before, adding a new team, or deleting a team you previously created. Newer players can learn more about the game by accessing our information pages, as well as reference other sources.

Project Tracker: *Jira*
https://csci-3308-fa20-114-4.atlassian.net/secure/RapidBoard.jspa?rapidView=1&projectKey=P14&view=planning.nodetail&selectedIssue=P14-2&quickFilter=2&epics=visible&issueLimit=100&atlOrigin=eyJpIjoiNGJiZjY1NmFkMDEyNGE3YjlkZDUzMmRjNTcyMmY0NGUiLCJwIjoiaiJ9
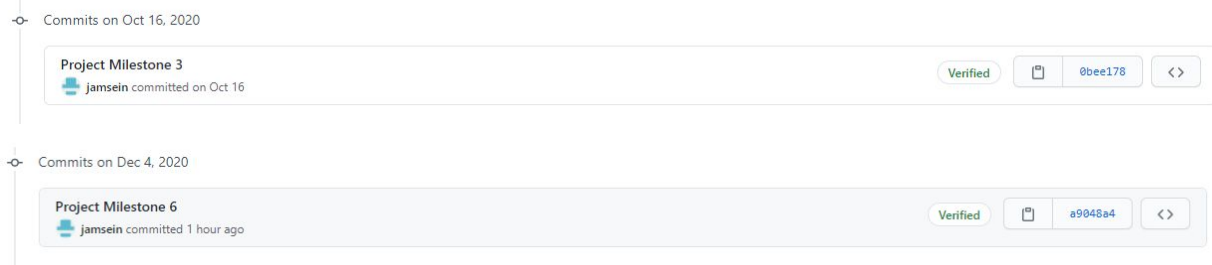   ❖ Agile Project Management Tool

Github Repository: https://github.com/CSCI-3308-CU-Boulder/114-4

**Deployed Web App:** http://projectnull.herokuapp.com/
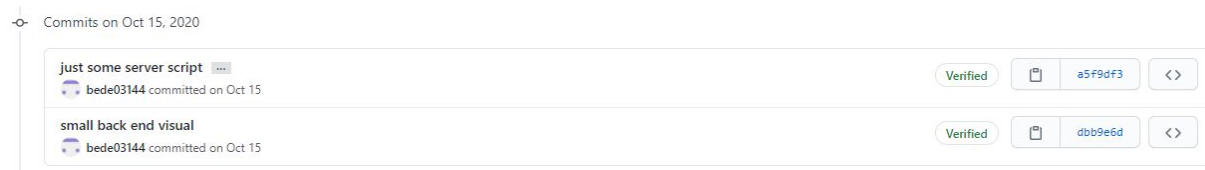
Contributions:

Note: Since we utilized pair-programming, Jamie and Taryn's commits reflect the work conducted by both members

Jamie: Mostly worked with Taryn to establish the backend throughout the entire project. We created the postgreSQL databases that held the necessary data for the Pokemon, as well as databases regarding the users as we implemented a user authentication system as well. We also initially connected it into Heroku before we covered it in the lab to establish the application onto the server rather than localhost. Using Visual Studio Code (and its LiveShare feature to directly collaborate), we wrote queries and edited them as needed by teammates, wrote code for queries, used pgAdmin to manage and view data, and more. Regarding users, we did a significant amount of research to figure out a method of user authentication, settling on using Nodejs and Express to create the registration and login

portion. In addition, we worked to create the table of Pokemon teams that would be stored for each user, and fixed bugs to ensure teams were saved to the specific user and not other users. Instead of using testing software, we manually tested cases to fix bugs and ensure our work would be fit to be implemented with the frontend. Also helped create the initial layouts/design of the website.
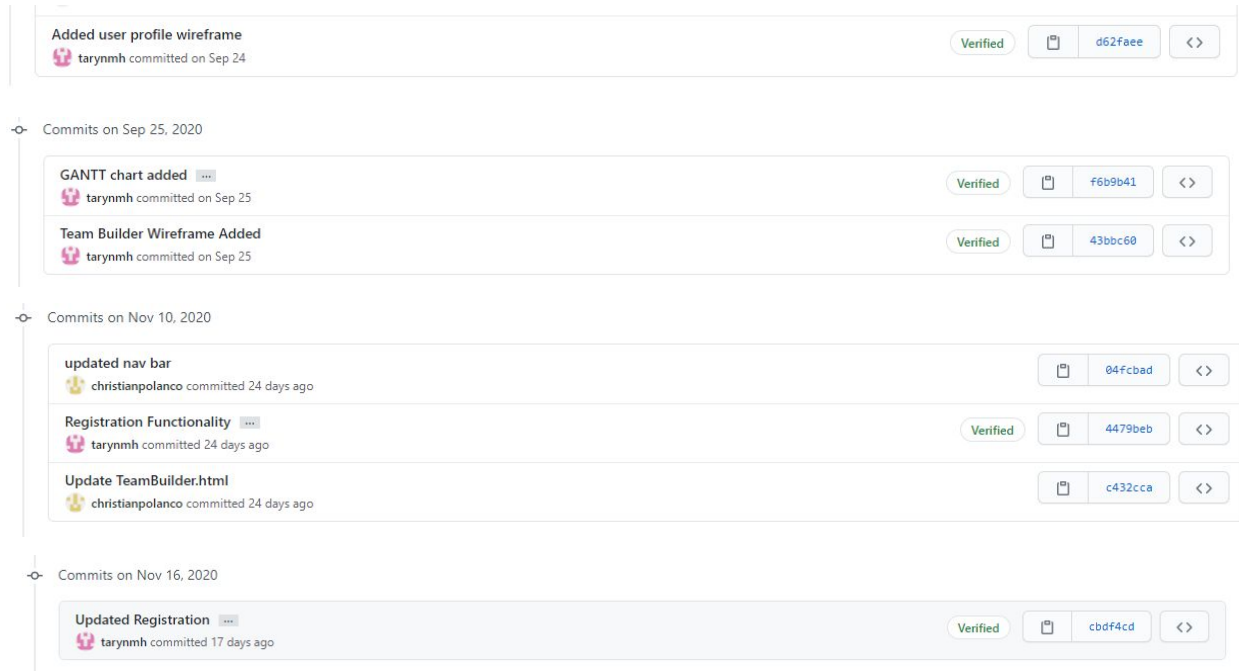
Taryn: Worked with Jamie on the database creation and management, as well as a few other backend portions. We utilized Visual Studio LiveShare to write queries, pgAdmin for the actual database management, and established a connection to Heroku using their servers. Our work included creating and inserting 800+ Pokemon into a table, creating a users table, and creating a table to hold each user's teams. Together with Jamie, we also established the user authentication system, which included a login and register portion. We implemented the relevant users table in the database, as well as the Node.js portion of the login/registration. Within the Node.js, we utilized node express as a framework while also drawing from our knowledge we obtained in the relevant labs. As for testing, we frequently found bugs when we were using localhost to view the site (and once we had integrated HTML/EJS, Node.js, and postgreSQL together).

Ben: I did the majority of the server code and embedded js along with a few static html pages. I made our nav bar a partial to be used for new pages to be added. I did the server routing and request along with the handling of complex data between client and server and server to postgres. I made most of the queries, inputs,deletes and updates to our database. I did the js/query for our search bar which lets you search pokemon with up to three different search criteria. I did the js for populating cards of pokemon on the search page and also the js for adding them to the clients temporary team before it gets saved to the database. I made the delete function on the profile page. I made the save team function which creates and add your team to the database and also assigns you a number that will allow you to retrieve your teams from the database. I did the js for populating the profile page with your saved teams. I did the js for handling a user login and maintaining their specific session variables used to grab teams and store them in the right spot along with changing the UI to show that you are logged in. I did the js for selecting a chosen team to either be viewed or deleted on the profile page. I also created a lot of the structure diagrams earlier in the year.

Kaleb:

Wrote C++ library we didn't end up using. Wrote javascript functions for type comparisons to be integrated with node server and html. Broke my hand and never got to finish up connecting functions with html so we could show type comparisons.

Trustin: I worked with Christian to do the HTML for the website. I made templates to inject the node.js for the sign up page, the homepage, and helped Christian with the profile page.On the signup page the verification for the information was checked before we allowed the User to sign up for the page. The homepage acted as an information hub for the user to learn about pokemon. The profile page displayed the users teams and information about their pokemon. After that the code was passed onto other group members for them to work on the node.js

.

Christian: Worked with trustin to develop the HTML pages. Along with the Home page, the user page, sign in page and team builder page, we made templates for the Node implementations to be made more smoothly. The user page shows the amount of pokemon that the user has made so far and has the ability to delete or add a team. The team builder page allows the user to see the current team they are building and edit the stats of the pokemon through sliders. They can also search for a pokemon using the search bar to filter or search by name. The website as a whole is easily navigated with the navbar and redirecting link that are available throughout the website.

PUT IN GITHUB AFTER FINISH
 Include a README in your repository:
   - Describe repo organization/structure
   - Describe how to build/run/test code
   - If using a Continuous Integration system, provide a link to the CI status page

Repository Organization
   - **All Project Code:** Contains separated files that holds each portion of the code,
       - views: Stores all frontend user-interface elements of the website
           - Partials: contains ejs partial code for consistent elements of the website like the navbar.
       - Css: contains all css code
       - Js contains most of the js code that isn't embedded.
       - Pokemon_images: hold useful images that get displayed
       - server.js: server code that works with our pages and js.
       - login.js: helper for user authentication.
       - Package.json: hold all of our dependencies
   - **Milestone-Submissions:** Track records of our progress throughout the semester of Fall 2020
   - **C++ :** Holds the C++ code used initially to apply game logic of Pokemon
   - **Planning:** hold all our our graphical plans of architecture and html pages

- **Team Meeting Logs:** Mostly unused, as our main means of communication were through Discord.

How to Build/Run/Test Code:
Check out app on heroku: https://projectnull.herokuapp.com/
If you want to use/modify locally everything you need in the project code folder. You will need nodejs, we used version v14.15.0. Once downloaded it should contain all modules required but if there is a problem we have a package.json file with all dependancies in All project code folder. To try out simply 'cd all project code' in your terminal to get to the proper directory. then run 'node server.js' the server will be locally hosted to port 8080.

note: "All project code" contains all the used code in our build. anything outside the folder is for the class.