The Development of Taskmaster

Team Yar
Mackenzie, Eric, Ifeanyichukwu
University of Regina
ENSE 374
Tim Maciag
December 5, 2021

Table of Contents

List of Figures

Executive Summary

Our team's final project is a scheduling tool that is aimed at students and instructors. The goal of our application is that it allows instructors to see the workload of students, and for students to be able to see their upcoming schedule. Our final product is the first minimum viable product of our application, which we named Taskmaster.

This report discusses our initial project documentation, some of the feedback we received and how we used it to better our project, and the development process, which includes our approach and the final result. We also discuss what makes our final product a minimum viable product (MVP), and we answer the reflection questions more thoroughly than we did during our project presentation.

All three team members were full-stack developers, and we each had designated pages that we worked on from view to functionality. Our goal was to complete our first MVP, and we did so successfully. It contains many key user interactions between both the instructor-type users and the student-type users. While this MVP does not currently fulfill its final goal of alerting instructors that their students have multiple assignments due on the same day, we are happy with this version of it. The purpose of an MVP is to build a starting point that can be expanded upon later, and we feel like our MVP does that. Therefore, we have created a successful MVP of our Taskmaster.

1.0 Introduction

The purpose of this report is to give a detailed account of the documentation and production of an application that we, Team Yar, chose to name Taskmaster. We believe that both students and instructors want to see the best work from the student in all projects and assignments. The goal of Taskmaster is to be an application to allow an instructor of a particular course to view when students have conflicting or "clashing" assignment due dates with assignments of another course. With this information, instructors can communicate with one another to better structure the assignment due date schedules for the courses, so that the student can have ample time to complete all assignments to the best of the student's abilities. This would fix the common student problem of needing to split time between assignments, which could lead to the student sacrificing sleep to finish the clashing assignments on time, or the student sacrificing one assignment for the sake of another that has more weight in terms of grade. Our team's set goal for the project was to finish our first minimum viable product (MVP), which consisted of creating a strong base for the application so it could be easily iterable and reliable. Setting this seemingly simple goal was the best choice as it allowed us to learn and develop our skills when it came to full-stack development, and the outcome of this was understanding how the model, view, and controller interacted with one another in this specific project, as well as methods of how to interact and manipulate the database. We believe that with these two outcomes, iterating the application further and getting future MVPs done would be quick and simple to implement as all the hard questions about how to interact with the database or how to make persistent session variables are answered, allowing our team or future developers to use the created code as a reference manual on how to extend the application to have more user experiences. It is our opinion that though the goal of Taskmaster is not in line with the current implementation of the application, the fact that our team was able to reach our goal for the project to complete the first MVP, and with more time, are confident in our abilities to implement future MVPs within a two-week timeframe. This all culminates in the achievement of the main goal of the project as it relates to the course itself, which was to gain industry-quality experience with a medium-scale project. This report will help explain why our team came to these conclusions.

2.0 Initial Project Documentation

The first action of the project (beyond the initial project idea brainstorming, which was not officially documented), is to create a business case document. This document describes the background of the project, business needs, and cost-benefit analysis of different ways to complete or not complete the project. Unfortunately, this document did not provide much use to us. This document would be more useful in a real-world application, such as trying to convince managers of a company to divert time and company resources towards a particular project, and while it would prove more useful out in the industry, it is still good to experience filling out this document.

The next document was the project charter. For the project charter, we detailed the goals of our project, and we redescribed the "why" of our project. Furthermore, we gave ourselves objectives for the project: create the application, clearly display the schedules for both user groups, and give notifications to instructors of clashing assignments. Of these three objectives, only the last was not fully realized in the current MVP. What proved useful in the creation of our project was the overall project milestones, which we used to give internal deadlines for the

milestones of the project. This way, we could gauge our progress and see if we needed to speed up if need be. As the project continued, these milestones were informally changed to be more appropriate, especially in terms of the database, since we did not know much about how to implement one when we first made our milestone goals. Eventually, the dates of these milestones were edited in the document to show what we changed them to during the development process.

In the beginning, the roles and responsibilities document and the RACI chart did not see much use since we did not know how we were going to split up the responsibilities during development. When the time came to start coding for the application, once we decided how to split up the work, it was easier to distinguish who was responsible for what, and these two documents were updated accordingly. A good example of when the rules of RACI (Responsible, Accountable, Consulted, Informed) were followed was when Eric needed to change how the user schema was set up to give better functionality to his code. He consulted and informed Mackenzie, who was Responsible and Accountable for this part of the code, and also needed to use the user schema as a part of her code, so she was aware of the change.

The most important initial document was the project scope statement. With this document, we got on the same page of what the application deliverables should be so that we could complete the first MVP on time. This prevented a lot of scope creep. Doing this task before the implementation of the project proved useful to us as we never second-guessed each task of the project after this point. We knew exactly what needed to be done and how it contributed to the overall completion of the project at large.

The last two initialization documents were the stakeholder register and the accompanying stakeholder engagement plan. Like with the business case, because of the nature of this course project, these documents were not used to their full extent, but these documents would be invaluable out in the industry. It would be a great resource for the requirements digging phase since talking to the stakeholders and users of the application can give insights into the exact business needs of the application that users may want. Additionally, these stakeholders would make a good selection of testers for our application. It is important to note that Tim Maciag is present as one of the project stakeholders but is absent from the stakeholder engagement plan document. We did this intentionally as it is stated in the course to never reveal the engagement plan to the stakeholders because doing so could have negative effects on how the team interacts with the stakeholder. These documents were very useful in helping the team to visualize the linear progression of one task of the project to the next and how these tasks all contributed to the overall result of the project.

3.0 Feedback and how we used it

The first piece of feedback given to us by our peers involved the vlogs that we posted on YouTube to discuss our project. The criticism we got was that we should have used a PowerPoint presentation instead of just talking throughout the session. We considered this piece of feedback for our final presentation as it would make it more interesting to use this form of multimedia to improve the focus of our audience. The feedback we received to use PowerPoint was taken into consideration because it was noted that the audio can be unclear at times, and it is better for the understanding of the viewers. This piece of feedback was mostly for our first vlog, as in our second and third vlogs we showed each piece of documentation on the screen as we discussed it, so some context was supplied to the viewers that way. We chose to use this piece of criteria for our final presentation so that our classmates could better understand what we were

discussing. Furthermore, a PowerPoint presentation can be shared for someone who may have missed our presentation, where they can view our vlog at their own convenient time.

The second piece of feedback involved the application of a Readme file in our GitHub. We had not yet done so, but we dived into completing it before we handed in our project, and included instructions on how to install it. The Readme file is important because it is the first file that the user can read, especially on GitHub, since the markdown file is displayed on the main page. Our Readme file contains a brief introduction to our project, along with the installation instructions.

Another piece of feedback we got was to split up the RACI chart and make sure that everyone has a proper title on the roles and responsibilities document. When we first made the documents, we were unsure how to split up the work, so we did not know who would be front-end, back-end, or full-stack developers. Once we figured this out, we were indeed able to update these documents with our exact roles and responsibilities, and we made an update on the RACI chart by including our responsibilities, accountabilities, who gets consulted and who gets informed.

4.0 Development

4.1 Development Exploration and Process

When the documentation for our project was first created, we were hesitant to commit to roles such as front-end developers and back-end developers because we still had not learned how to implement a database yet. We decided that once the time came, we would decide how to split up the pages and who was going to do what. We had 7 planned pages in total for our first MVP. The signup/login page, the instructor view of the calendar page, the student view of the calendar page, the add assignment page, the edit assignment page, the add course page, and the join course page. We decided to split it up as follows: Ifeanyichukwu did the signup and login page, Eric did both calendar pages because there would be many similarities and places to reuse code, and Mackenzie did the last four pages. Since the last four pages all consisted of just a few input fields and a submit button, it was decided that whoever got the last four pages would also combine the CSS files and make the colours, headings, and fonts consistent between pages, so Mackenzie also did that. When the time came to add functionality, we opted to continue to be in charge of our separate pages for simplicity. Thus, we were all full-stack developers.

The first task we completed was to outline our pages using HTML and CSS. Once this was done and the CSS was consistent between pages, we each converted our pages to the .ejs format so we could use it easily using the tools we learned in the lab. Next, we worked on our app.js, which would run with nodemon. We all worked on the same app.js file. Eric had the idea that we should make a test file called debug.js. Debug.js contained code so when it was run, it stored test users, courses, and assignments to use for testing. This was incredibly useful for Eric’s calendar views and Mackenzie’s four pages. Eric was able to work on being able to display values from the database without having to wait on Mackenzie’s pages to be functional first.

For these few weeks working on app.js, we communicated the status of our pages. Mackenzie and Eric especially had to communicate a lot, because Eric’s code would be retrieving the data that Mackenzie’s code stored, so they communicated on how to best save the data for the calendar page to be easily displayed. Ifeanyichukwu’s login and signup page did not

depend on Eric's pages or Mackenzie's pages, but both Eric and Mackenzie needed session variables to access and change user data. Other than redirecting to other pages and accessing session variables, the code between pages did not interact with each other very much, which worked well for us. The rest of the pages displayed and changed information depending on the database.

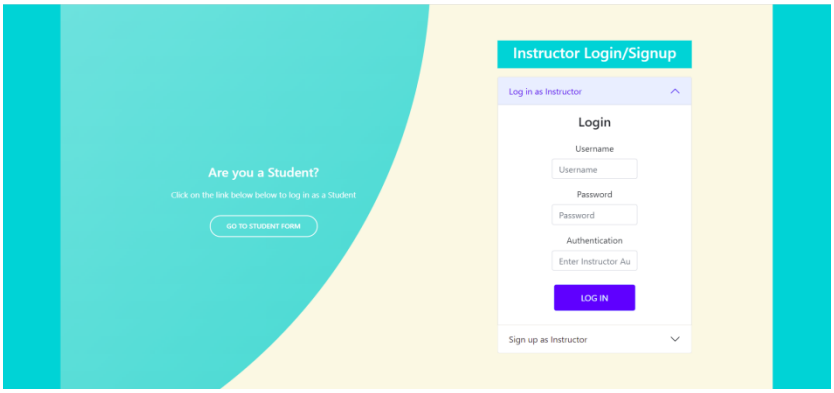
4.2 Outcome of MVP: User interactions

Our team was able to complete our first planned MVP, which was our goal. Our MVP has two types of users: the instructor type and the user type. There are many key user interactions for users of both types, and they will be discussed next.

4.2.1 Instructor User Interactions

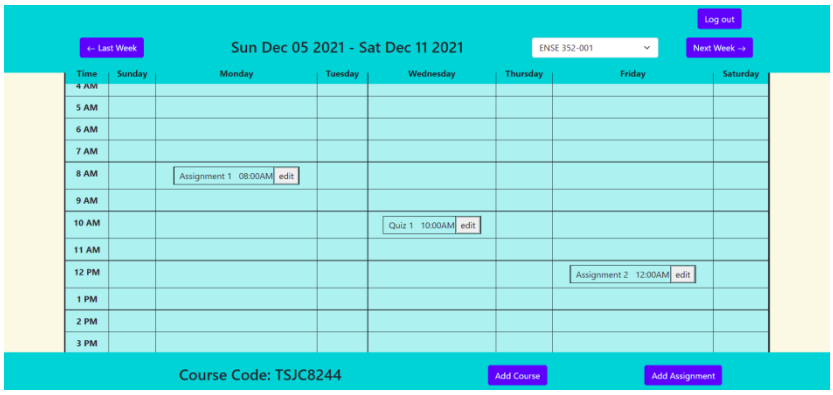
There are a few key interactions for a user of the instructor type. An instructor can sign up and log in with their account. To sign up and log in as an instructor, an authentication code is needed. The authentication code for both signup and login is "grades." If the signup fails, the login/signup page is reloaded. Figure 1 below shows the instructor login form. The accordion can be clicked to view the signup form, which is the same in appearance, with one small change of a confirm password field.

Figure 1: Instructor Login



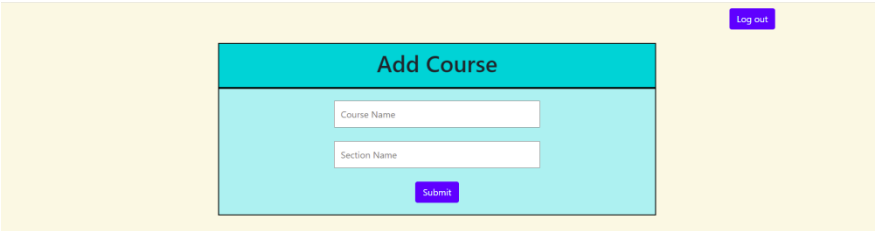
Once the instructor is signed in, the landing page is the instructor calendar view. This can be seen in Figure 2 below. If the instructor has just created a brand new account, the page will be empty as they have no courses or assignments yet. If the instructor is logging in to an existing account, the calendar will be populated with their courses and assignments already. If there is at least one course, the course that appears at the top of the dropdown will automatically be chosen as the course being viewed by the instructor. If there are multiple courses, they can be selected from the dropdown and the calendar will update accordingly. The course code at the bottom of the screen will change depending on what class is being viewed at the time. If there is at least one course, the "Add Assignment" button is available.

Figure 2: Instructor Calendar Page



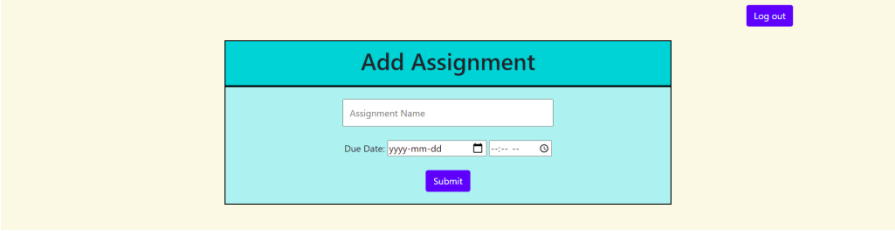
The instructor can press the "Add Course" button to add a course. This takes the user to a new page, where they can add a new course. The instructor can add in the course details, such as the name (which is required) and the section (not required). Upon submitting, the instructor is taken back to the calendar page. The Add Course page can be seen below in Figure 3.

Figure 3: Add Course Page



The instructor can press the "Add Assignment" button to add an assignment to a course. The assignment will be added to the course that is being viewed on the calendar at the time that the button is pressed. To add an assignment, the instructor must fill out the assignment name, due date, and due time (all required). This can be seen in Figure 4.

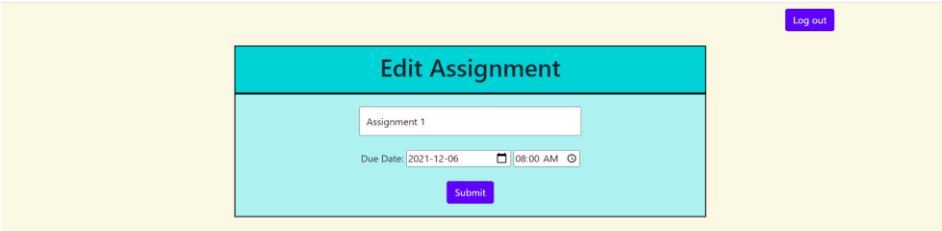
Figure 4: Add Assignment Page



Upon hitting submit, the instructor is taken back to the calendar page, where the assignment will be displayed. If needed, the instructor can use the "Last Week" and "Next Week" buttons to navigate the calendar and view the assignments associated with a specific course.

Each assignment on the calendar page has its own “Edit” button. Pressing this button takes the instructor to a new page, the “Edit Assignment” Page. The current assignment details are displayed in the input fields, and changing them before pressing submit saves the updates to the database. This is shown in Figure 5 below. Pressing submit takes the instructor back to the calendar page, where the changes to the assignment are reflected on the calendar.

Figure 5: Edit Assignment Page

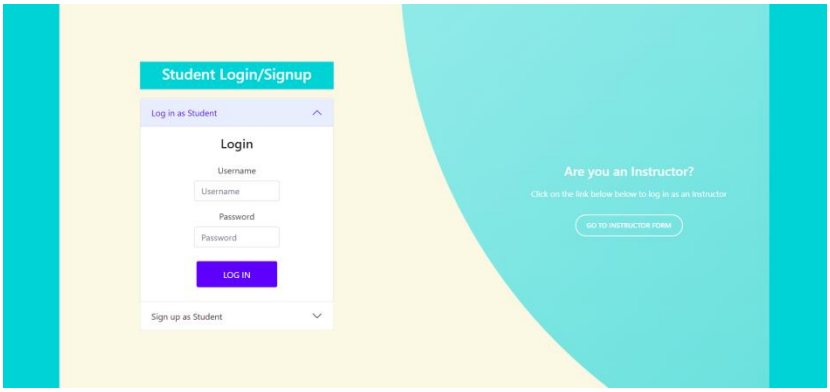


Instructors can add courses, add assignments, and edit assignments at any time. The instructor can log out using the “Log out” button at the top right, and as long as the database is not reset, they can log back in and view all their courses and assignments again.

4.2.2 Student User Interactions

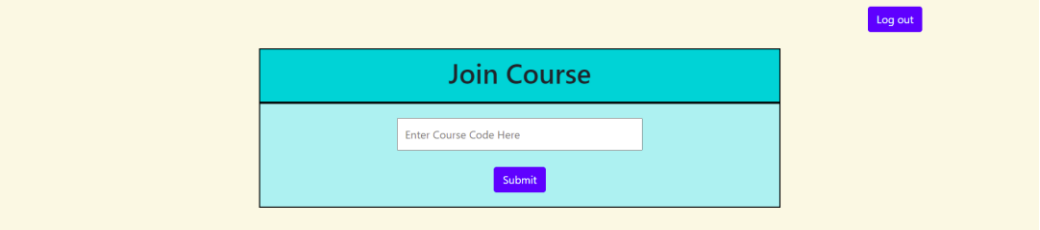
The student type of user has a few key interactions as well, but not as many as the instructor type. A student can sign up and log in with their account. All that is needed is a username and a password. This is shown in Figure 6 below. Again, the signup form is revealed by clicking on it, and the only difference is a field for password confirmation.

Figure 6: Student Login/Signup



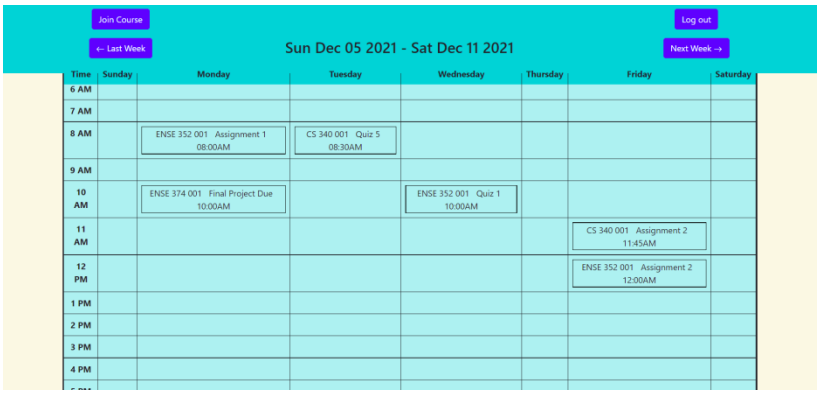
Once logged in, the landing page for a student is the student version of the calendar screen. From here, a student can view all assignments from courses that they are enrolled in. To join a course, the student can click the “Join Course” button at the top left. This will take them to a new page, the “Join Course” Page. This page is displayed in Figure 7. The student must enter the course code given to them by their instructor, and upon pressing submit, they are directed back to the calendar page.

Figure 7: Join Course Page



The student can now view all assignments from all courses they have joined and can use the "Last Week" and "Next Week" buttons to navigate their calendar. An example of a student enrolled in three different classes is shown in Figure 9.

Figure 8: Student Calendar Page



4.3 MVP Justification

Our project fulfills a true MVP. It is usable, so therefore it is "viable." We opted to include logging in and out as part of our MVP to make our application hold some use to both types of our users. It is also not our final envisioned product, which is what makes it "minimum." In fact, our MVP does not even fulfill the primary reason that we created this app. We wanted instructors to be able to view the percentage of students that have clashing assignments, either due on the same day or due within a certain period that can be selected by the instructor. We decided early on in development that that feature should be moved to our second MVP, and to just focus on the basic application first, since we needed a good basis for our application. Looking back, we are glad we made that decision because while we planned to implement that feature if we had time to get to it, we ended up not having time after all, and only completed one MVP. Another thing that makes our project an MVP is the fact that it would be easy to continue to develop if we chose to do so. MVPs are constantly being added to create the next MVP, and we built ours with the intent to move on to the next MVP if we got there.

5.0 Reflection

5.1 How did we feel about this project? Do we think we were successful?

We feel pretty good about our project, and we consider ourselves successful. We left ourselves in a good place so that if we were out in the industry and had to continue our project,

we could easily pick up from where we left off to implement features for the next MVP. We are proud that we got done, and that our code works as we envisioned. It is a good feeling to get your code to work, especially because our code interacts with one another's. It was cool to see our entire project working together between different pages that different people coded.

5.2 What did we like about our project? What did we dislike?

It was interesting to experience the development process in the field of engineering. It was fun implementing the tools that we learned in the lab, but it was difficult to learn new things alongside implementing them. We learned about passport just as we were starting to need it, as the last lab's due date overlapped with the beginning of the coding sprint during November. There were also some things that we had to research while developing them, such as how to use object ids between JavaScript and Mongoose since we dealt a lot with them.

5.3 What are we most proud of throughout the entire project experience?

We are proud that we got so much done in such a short amount of time. We were one of the groups that had some visuals and some basic skeleton code to show at the November 16th scrum, but we did not have any functionality at that point. We implemented all of the functionality within 2 weeks.

5.4 What did we learn about ourselves as we collaborated and worked on this project?

As a team, we learned that we communicated well both during Zoom meetings and over our Discord group chat, whether it was a meeting that took a few hours or a quick message to update each other on the status of our work. We held regular Zoom meetings every time that a project activity was getting closer to being due. We met for work sessions on Saturdays and filmed vlogs and finished up the work on Thursdays before the Friday due dates. When we did not have a meeting, we messaged frequently over Discord. Messages such as "I just pushed a new update, X should be working now," or "I was testing something and found a bug in this part of your code" were very common during the development stage.

Eric learned how to better communicate ideas and opinions in debates about the direction of the project. Eric and Mackenzie had a debate on how to best store the id data type in the database to retrieve it later. The two eventually figured out a system that made it easy for Eric to access data in the database and for Mackenzie to store data in the database.

Mackenzie learned that she can get a lot of work done in a different working environment. When she found it hard to focus in her bedroom, she switched to working in a different room and found it easier to focus and work on this project for longer periods to get her work done.

Ifeanyichukwu learned a lot about teamwork. He has been used to working alone, solving problems and doing tasks by himself, but after completing this project on our team, his perspective on group projects has changed.

5.5 How will we use this experience moving forwards?

We will all be using this experience moving forwards. We learned to manage our time well this semester, and two of us have co-op/internship jobs in the near future working in the industry where we will apply the things we have learned. We will also be using this experience for our engineering capstone project. We are glad we had this experience and we will continue to work on the skills we developed this semester.

6.0 Conclusion

Our team built a scheduling application that can be used by both students and instructors. It was designed to combat the struggle that students so often face with “clashing” assignments from different classes. All three of us on the team worked as full-stack developers.

The project is aimed at task management, for students to have ample time and opportunity to give in their finest work. We want our application to make it easier for students to keep track of assignments between classes so that they can submit their assignments and projects on time. We hope that our application also opens up communication between students and instructors about due dates and the workload of the students. Professors need to see their student’s best work, and students deserve a chance to hand it in.

The development of this project was a success for our team as the target goals were met. We took into account feedback from our peers to better our product. We managed to complete our first planned MVP, which was our goal and it was tested and approved to work as expected. The MVP application has two types of users, the instructor type and the user type. Each type of user can use our application as intended, which is why we wanted users to be able to log in and out.

The feedback we received was very helpful to our team in the completion of our project. The feedback received, from adopting a PowerPoint presentation to attaching a readme file and defining our roles on a RACI chart, was helpful given the impacts they had on our project. We made several decisions such as adjusting our presentation to accommodate the viewers, attaching a readme file to help clarify the coding structure and developing an Edits and Updates Log for an update on our GitHub.

The project has many fundamental user interactions for both types of users with a different interface for each. Among the significant strengths of the system is that the interface is friendly to use, allowing the users to have an easy time learning and using the system with much ease. It is also MVP viable, allowing the operations of login and out to differentiate the users and have their credentials to their platforms.

We count this project as a significant achievement toward realizing our career. It will be a great point of reference for future projects as it has shown us the potential of developing modern interactive systems. It helped us identify the issues arising with the design and implementation of the project, giving us a chance to work towards improving and perfecting future work. It also let us experience the development process the way that we might while working out in the industry.

We are happy to announce that we achieved our goal of finishing our first MVP, where we created the app that cleanly displays the schedules for both students and teachers. The outcome was a successful MVP of our Taskmaster application.