Automatic Reassessment and Relearning System

Transition to Vue in Assistments 2.0

A Major Qualifying Project Report

Submitted to the Faculty of

Worcester Polytechnic Institute

In Partial Fulfillment of the Requirements for the

Degree of Bachelor of Science

By

Peter Jankowski

With Assistance From Shikha Pandey and Dev Patel

Advisors:

Neil Heffernan, PhD, Professor of Computer Science

TABLE OF CONTENTS

ABSTRACT

Recently, the need for online learning platforms has greatly increased in the face of the COVID-19 crisis. To address this matter, online learning platforms have been created to assist students in the learning process. ASSISTments is a free tool for assigning math content online. Students receive immediate feedback while teachers receive data insights to drive meaningful instruction (ASSISTments.org, 2020). This project involved the design and development of a new version of ASSISTments' Automatic Reassessment and Relearning System (ARRS) that would serve to improve the UI experience and assist both students and teachers in the online learning process. This project launches the transition of the front-end of ARRS from Java Server Pages (JSP) to Vue.js, a progressive javascript framework used to build user interfaces. Vue.js was utilized for its modern layout and to provide users with easy access to the functionality and features of ASSISTments. Feedback from the ASSISTments Front End Team aided in the design of nine UI Mockups of new ARRS web pages. Four of these pages were developed and implemented in the ARRS ASSISTments 2.0 system. Ultimately, through the addition of Vue.js, this project aims to drive improvement of ARRS and open the path to prospective future changes in ASSISTments.

ACKNOWLEDGMENTS

The team would first like to acknowledge Professor Neil Heffernan, Jason King, and Jack Gonsalves for their wonderful support and guidance throughout this project. The team would also like to thank Ryan Emberling for being a great source of ASSISTments front-end code knowledge and navigation with Vue; his constant willingness to guide and support project efforts was thoroughly appreciated. In addition, the team would like to recognize Cristina Heffernan for aiding in the development of this Major Qualifying Project and providing invaluable insight into the minds of current ASSISTments users. Finally, the team would like to thank Sidharath Chhatani for his help in designing the most recent set of ARRS mockups.

1. INTRODUCTION

According to the Organization of Economic Cooperation and Development, “The COVID-19 crisis has forced education systems worldwide to find alternatives to face-to-face instruction. As a result, online teaching and learning have been used by teachers and students on an unprecedented scale” (OECD, 2020). Furthermore, new data from the RAND Corporation shows that many teachers are still struggling with online-learning, as teachers in all-remote environments have reported higher student absenteeism and less student work completion than teachers in face-to-face classrooms. These online teachers also said that they needed more support and guidance in planning instruction than their colleagues who were teaching in-person (Schwartz, 2020).

To address this matter, online learning websites have been created to assist students in the learning process, but teachers have routinely been struggling to find a teaching format that fits their needs, as well as the needs of their students. There are many options in the market that schools and teachers use but these options have their own limitations. Fully online programs are known to widen achievement gaps and often are unaffordable as well as students in online education, particularly disadvantaged students, underperform and on average experience poor outcomes. Furthermore, online education has failed to improve affordability and frequently costs more than in-person alternatives (Fain, 2019).

ASSISTments is a viable solution to this problem as it is an online tool designed to give teachers access to a wide variety of content as well as give students real-time feedback about different assignments. ASSISTments is able to enhance math curricula with actionable data for teachers and immediate feedback for students. With reports designed to give teachers overviews of each class, assignment, and student, teachers are able to tailor their lesson plans to better help

their students. Importantly, ASSISTments has been demonstrated to demonstrably aid student learning, as an experiment conducted by SRI International to test ASSISTments' efficacy showed 75% more learning in classrooms using ASSISTments when compared to the learning that normally happens in a year. The experiment randomized schools to either use ASSISTments or continue their regular methods. There were three findings: first, teachers who used ASSISTments reliably changed their classroom practices; second, students reliably scored higher on a standardized test; third, ASSISTments began to close the achievement gap: students below the median on the prior year test learned an equivalent of two years of math in a single year (ASSISTments.org, 2020).

As ASSISTments continues to grow, so does the need for version upgrades, and as the company made the decision to transition from version 1.0 to 2.0 of ASSISTments, decisions were made to redesign the entire front-end as a whole, as well as the Automatic Relearning and Reassessment System (ARRS). The company began the redesign of ASSISTments in search of a front-end web development framework with a redefining component architecture and ecosystem. Subsequently, Vue.js was chosen as the framework of choice. ASSISTments began to transition to using Vue.js in the Summer of 2020. Inevitably, ARRS had to follow as well, tasking the team with converting all pages from Java Server Pages (JSP) to Vue.js to provide ARRS with a modern layout and better structure and support that JSP failed to offer.

The goal of this project is to contribute to the transition of the front-end of the Automatic Relearning and Reassessment System within ASSISTments 2.0 from JSP to Vue.js. The project aims to redesign the previous ARRS implementation by applying and combining key aspects from the existing 1.0 and 2.0 system, as well as keeping in mind how to implement efficient and

intuitive user navigation, attractive visuals, and consistency in the web page design to meet the current needs of teachers and students using ASSISTments.

After a thorough analysis of the preliminary code for ARRS in ASSISTments 1.0, existing features of ARRS in ASSISTments 2.0, initial discussions and user testing, the requirements and needs for the revitalized version of the ARRS web pages were established. The final outcome of this project was nine UI mockups of seven distinct teacher and student version ARRS web pages, as well as the development and implementation of four of those core pages within the ARRS ASSISTments 2.0 system. Through the creation of the new pages and features, the project hopes to aid in the development of the ARRS 2.0 system and build upon the current ASSISTments paradigm.

The rest of this paper is organized as follows. Chapter 2 provides a detailed background on the history and development of ASSISTments and ARRS. This section also introduces Vue.js as a new framework being implemented and goes into the depth about the current design of how the Student and Teacher pages are established in the ARRS system. Chapter 3 outlines the User Testing Process and Overall Needs for teacher and student users visiting the ARRS pages. This section then goes into detail outlining the premise of each individual page, and showcasing the original nine UI mockups that were created as part of this project. This section also contains mockups that were made more recently in order to bring the design of ARRS more in line with the rest of ASSISTments 3.0 and prepare for a closed beta of ASSISTments 3.0. Chapter 4 highlights the development process of the four pages that were implemented before the decision to redesign these pages was made. Chapter 5 concludes the project as a whole and contains recommendations on how this project should be picked up in the future tying in personal

experiences and technical and conceptual changes that would make the future of the project successful.

2. BACKGROUND

This section serves to describe the overall purpose of the ASSISTments for teachers and students. The motivation behind the ARRS component of ASSISTments, the usage of Vue, and the prior implementations of ARRS are described.

ASSISTments

ASSISTments is a widely used web based learning tool that provides teachers with the ability to assign homework problems to their students and monitor their class's progress. In 2013, roughly 50,000 new student accounts were generated online through ASSISTments, which led to an average of 4,000 daily users in 2014 (Heffernan, N. & Heffernan, C., 2014). The site enables student users to receive real-time feedback pertaining to their responses and overall progress while providing teacher administrators the ability to manage questions, analyze reports to identify areas of success and failure, and personalize lesson plans. Ultimately, instructors are able to leverage the tool to fit the individual needs of their students and students are able to receive assignments with meaningful feedback. The ASSISTments tool is updated frequently by its users feedback, both student and teacher, and by researchers in order to unveil its full potential and ensure that it seamlessly and efficiently assists in the exchange of instruction and knowledge between teacher and student.

Automatic Reassessment and Relearning

A key aspect of the ASSISTments website that streamlines its efficiency is ARRS, which is an acronym for Automatic Assessment and Relearning System. ARRS is designed to fully reinforce and cement the skills and knowledge that students set out to learn. This is accomplished

through the “automatic reassessment” of topics that a student has previously studied and mastered, thus pushing students to retain the knowledge that they have learned. If the student is unable to receive satisfactory marks on the reassessment, the student “relearns” the topic and performs additional problem sets until they are able to receive satisfactory marks, thus indicating remastery. The main goal of ARRS is to ensure that students retain the mastery of skills they have gained from completing assignments that are referred to as skill builders. The system will periodically test students with similar problems they have completed to ensure they still know how to complete that problem. This system iterates until it is determined that the student has developed comprehensive mastery of the skill or topic.

Figure 1 demonstrates the originally intended default flow of an ARRS assignment, with reassessments given asynchronously for different students. After a student has successfully completed a skill builder assigned by their teacher, the system will automatically generate a reassessment assignment with one problem from the bank of problems from the skill builder. Then, those problems will be assigned to the student one week later. The student will then start on reassessment level 1. Once the reassessment assignment becomes available to the student they have to answer one question correctly and move to the next reassessment level and another reassessment assignment will be generated and assigned for 2 weeks later. However, if the student answers the question incorrectly, the original skill builder will have to be completed again, followed by the previous reassessment level they just finished. This process will continue until the student has successfully completed four reassessment levels at which point the student will have completely mastered the skill.

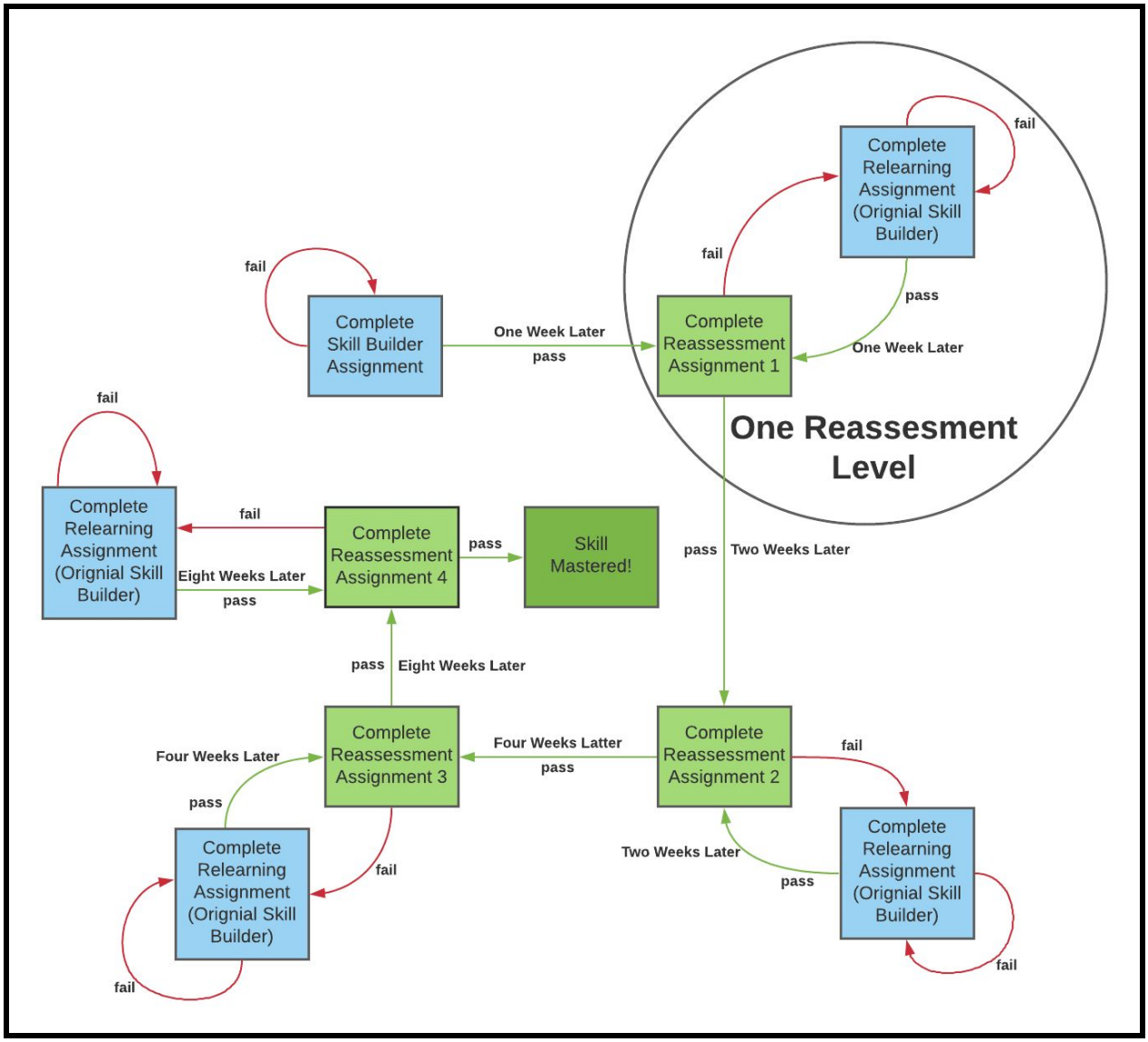

Figure 1: A graphical explanation of how asynchronous ARRS works. (Used with permission from Jack Gonzales)

In the latest redesign of ARRS for ASSISTments 3.0, the default flow of an ARRS assignment has been changed to ensure that students are kept aligned in terms of their progress. This is called synchronous ARRS, and the intention of synchronous ARRS is to both simplify the information presented to teachers instead of forcing them to keep track of where each student is

in the assignment, as well as simplify the initial implementation of ARRS for a beta release. With this redesign, an alternative term used to describe the entirety of an ARRS assignment as an ARRS Lifecycle was proposed for internal ASSISTments usage. With synchronous ARRS, students would no longer be stuck retaking reassessments on ARRS assignments, with the potential to fail endlessly and never finish an assignment. Instead, each reassessment was to be given one attempt and one due date, after which the student would be moved to the next level of the assignment, and the need for intervention to help the students that had fallen behind would be highlighted to teachers. This redesign is shown in Figure 2.

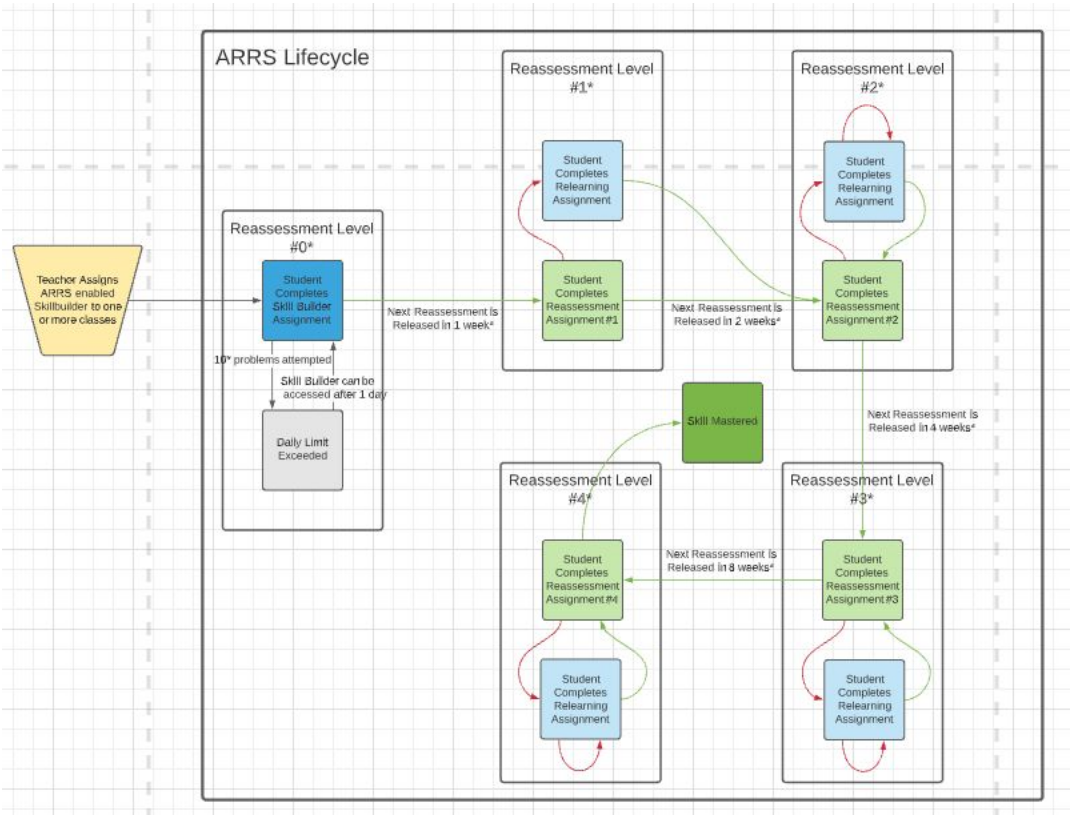

Figure 2: A graphical explanation of how synchronous ARRS works. (Used with permission from Jason King)

The concept of dividing and staggering reassessments and potential relearnings has been proven to be beneficial because students are able "...to store information in memory in a manner that makes the information more resistant to forgetting than non-spaced repetitions" (Thalheimer, 2006). This method of spacing out also provides students with the capacity to perform significantly better on standardized tests because they are able to access a deeper library of knowledge than students who learn individual topics and then move on without revisiting previous topics. This was proven through a study by Soffer-Goldstein in 2014 analyzing 97 eighth grade students to determine the effectiveness of ARRS. Results from the study depict that students who conducted staggered reassessments and applicable relearnings tested significantly higher on assessments administered at the end of the study session than those who did not. Ultimately, the results from this study illustrate that "ARRS is an effective method for increasing student retention of critical knowledge and skills and in maintaining high levels of proficiency" (Soffer-Goldstein et al., 2014).

Within ASSISTments, teachers can also view and analyze reports on assignments that leverage ARRS. Views within the reports section provide information on individual student performance, completion and timeliness of assignments, and overall comprehension of curriculum. The reports section of ASSISTments details the aforementioned information in a clear and effective way, which in turn enables teachers to easily address individual student needs and develop tailored lesson plans to cover areas of struggle while automatically reassessing mastered topics.

VUE

Vue is, in short, a modern open source front end Javascript framework for building user interfaces. Vue is a reactive framework, which means that it can dynamically update components as the data in use is updated. According to the documentation for Vue, “A Vue app attaches itself to a single DOM element then fully controls it.” (You, 2020). Once a new Vue instance has been bound to the HTML, the data used within the DOM element is handled by the Vue instance. Vue itself has a number of directives that can be used to perform specific manipulation of the data. Vue is also a component based framework, with the intention of allowing users to build applications out of small, reusable components.

There are a number of reasons why ASSISTments Foundation has begun transitioning to use Vue as a frontend framework. To start with, most modern web applications are built using modern Javascript tooling, and in this regard, ASSISTments is currently in the process of catching up. ASSISTments 2.0 currently uses Java Server Pages, (JSP), for a frontend, which is less popular and less supported than Javascript alternatives. Also, using a separate frontend application with Vue from the backend of the application follows the REST (representational state transfer) style of software architecture. This is intended to help the frontend to be performant and scalable, and also makes updating parts of the frontend less disruptive to the backend, as they are seperated. (Fielding and Taylor, 2000).

ASSISTments more specifically chose to use Vue instead of alternative frameworks because it was the most natural fit for the application. Vue has a solid ecosystem of libraries, tooling and support. This has provided Vue with a strong level of popularity among frontend Javascript frameworks, which in turn has encouraged the creation of additional component

libraries such as Vuetify, which are immensely helpful resources to use. Additionally, Vue allows for the interface of each page to be handled at a granular level in a straightforward manner. Robust tooling goes hand in hand with maintainability as well.

For this project in particular, because ASSISTments is moving towards using Vue as a frontend framework, and since ARRS has not been released in ASSISTments 2.0, the decision was made to use Vue rather than writing or editing pages in JSP that would be overwritten later. More about how this decision to move to Vue affected the project can be found in the "Discussions" section.

Current Implementation:

The Spring 2020 *Automatic Reassessment & Relearning System in ASSISTments 2.0* MQP Project featured the overarching goal of providing ASSISTments 2.0 with a minimum viable product of ARRS functionality. The codebase was not directly used due to the upcoming transition to Vue, but the team will be able to borrow from its design to determine which pages are needed and what the requirements for each page shall be. As a result the first step, outlined in the coming sections, was to perform an evaluation of the ARRS implementation in both ASSISTments 1.0 and 2.0 so that the extent of the project needs can be ascertained.

Student Pages

On the student side of ASSISTments, students are granted the option to view a Student Landing page and an Assignment Report page. The Student Landing page's main function is to display assignment information, including past, current, and upcoming assignments; reassessments and relearnings; and allow for the completion of a selected assignment. Additional information such as assignment type, assignment status, and due date are also provided for

clarity. The Student Landing page may be accessed via Google Classroom through the selection of an ARRS assignment. Figure 3 depicts the current design established by the *Automatic Reassessment & Relearning System in ASSISTments 2.0* MQP. The Assignment report page allows a student to view their progress and results for an ARRS assignment and reassessments for that assignment.

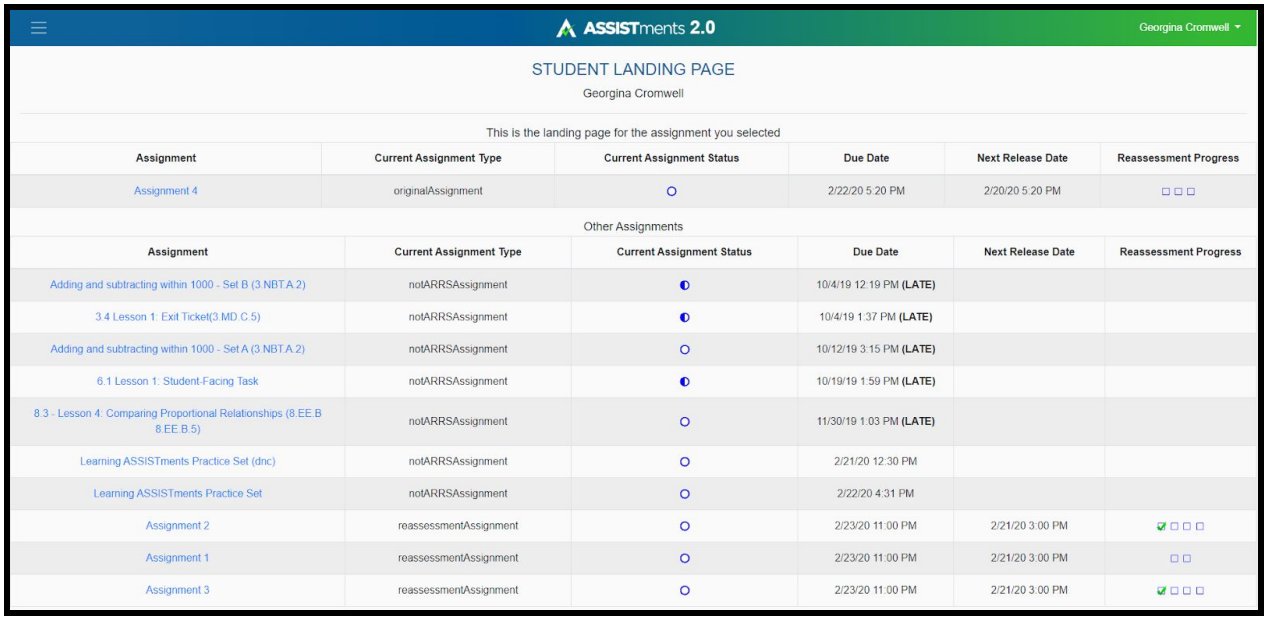

Figure 3: An example of the student landing page. (Used with permission from Jason King)

Teacher Pages

On the teacher side of ASSISTments, users are granted the option to access Landing, Settings, Assign, Class, Students, and Report pages. The Landing page allows a teacher to view their various classes, and clicking on a specific class opens the Report page. This Report page lists enrolled students and statistics pertaining to their individual and collective assignment progress. A detailed, individual Student Report page can also be accessed by selecting a particular student, depicting late assignments and Reassessment progress. The Settings page

permits a teacher to update the ARRS release schedule for specific classes from the default values. If the settings are modified, changes are implemented across all applicable assignments assigned after the settings are saved. Available edits include reassessment schedules and number of instances. An example of the aforementioned pages implemented in the current design created by the *Automatic Reassessment & Relearning System in ASSISTments 2.0* MQP are available in Figures 4-6.

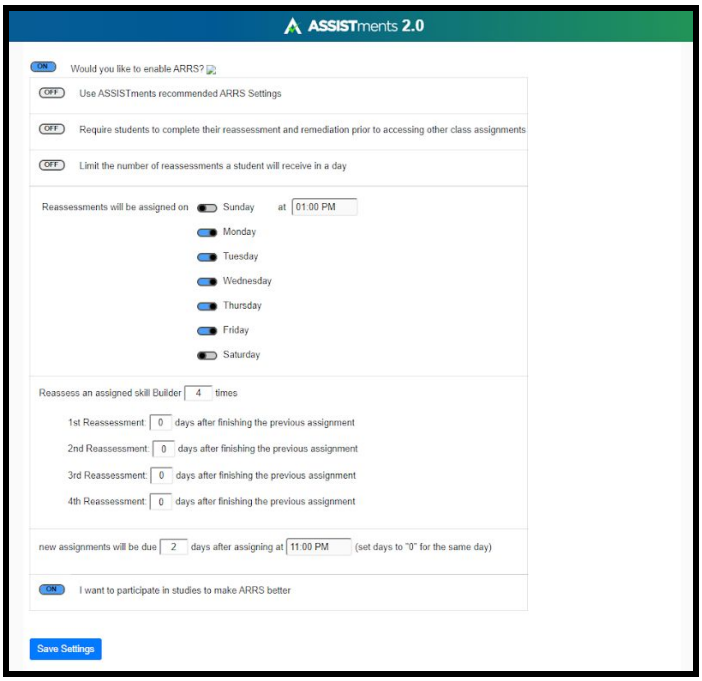

Figure 4: An example of the settings page. (Used with permission from Jason King)

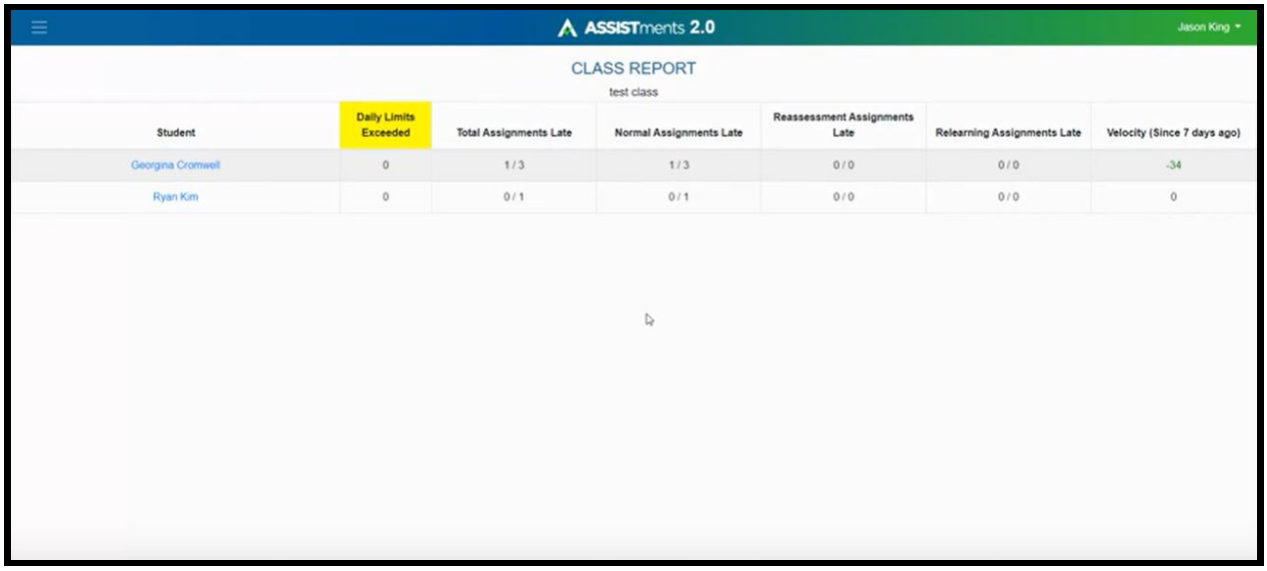

Figure 5: An example of the class page. (Used with permission from Jason King)

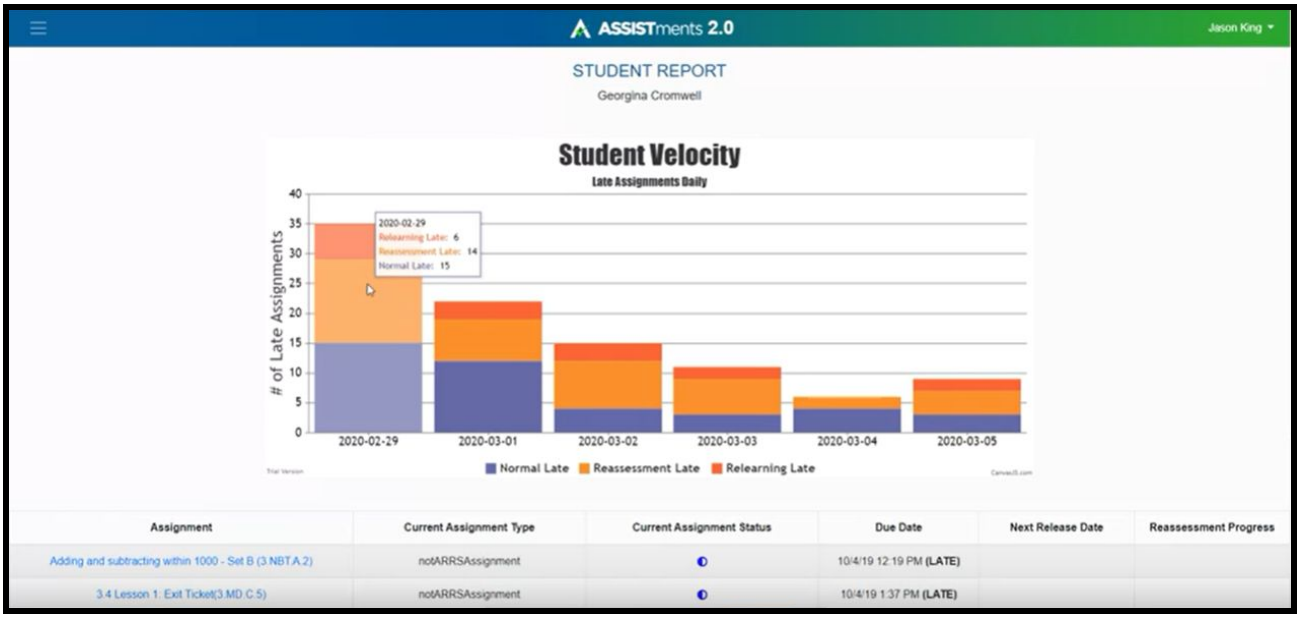

Figure 6: An example of the student report page. (Used with permission from Jason King)

3. DESIGN

This section outlines the process behind the first set of ARRS UI Mockups design and implementations in Vue for ASSISTments 2.0. In order to inform design and development, user needs were first identified and analyzed. This process is captured in the User Testing and Needs section. Next, nine UI mockups were drafted to meet the needs, which are outlined in the Conception section. Critique and feedback of the mockups that was received from the Front-End Team is discussed in the Feedback section. Finally, the last part of this Design section discusses new mockups that were made after these mockups were partially implemented.

User Evaluation

While the team was unable to perform user testing with completed pages in Vue due to time constraints, user testing was previously performed with some of the mockups for selected Adaptive Homework pages with volunteer ASSISTments teacher users. Since this feedback was performed before the project was consolidated, the mockups were not yet made for the ARRS pages and the feedback that was received did not directly pertain to ARRS. However, the interviews were able to lend other forms of valuable insight to the team. For example, it was garnered that teachers tend to strongly dislike navigation through Google Classroom to access different assignments. Moreover, it was learned that access to an immediate visual representation of key information found on a page, such as a student velocity graph, is appealing to teachers. Refer to Appendix B for the script utilized when conducting the user interviews.

Needs

Since the core functionality of ARRS had already been established through the previous project, the requirements of this project were not concerned with reestablishing the purpose of ARRS. Instead, in the interest of a fresh start in terms of implementation, the team was tasked with presenting ARRS in a way that would improve upon the prior representations' shortcomings. With that in mind, the overarching aims of the page designs were to render the homework assignment process easy and efficient for teachers and develop a clear solution for student users to simplify the process of completing reassessments. Feedback received from user evaluations on Adaptive Homework also emphasized the need to minimize the usage of sites such as Google Classroom and Canvas between pages and assignments, as well as provide clear visual representations of what information was most important for each page.

Conception

The next step was to create mockups of what each page was to look like using Figma, a collaborative interface design tool that features a vector graphics editor and is primarily web-based. This was done to ensure both the familiarity with the pages that were to be implemented and to provide an opportunity to propose different ideas on how the pages should be structured. In total, nine mockups were made for seven distinct pages, each of which was reviewed by members of the ASSISTments team and iteratively adjusted. The following sets of mockups were all made before a decision was made to redesign ARRS and bring it more in line with the rest of ASSISTments 3.0, which is further discussed later in this paper.

Student Pages

The following sections discuss mockups made specifically for student users and also includes examples of the mockups.

Student Landing Page

The student landing page was a significant focus of the team's time and implementation because it allowed access to the other ARRS specific student pages, and also featured new ideas in design. The student landing page's mockup went through three design implementations. The original student landing page design mockup was highly similar to the student landing page implemented by the previous project, but with some minor changes centered around how the page's information was to be formatted in a table, as seen in Figure 7.

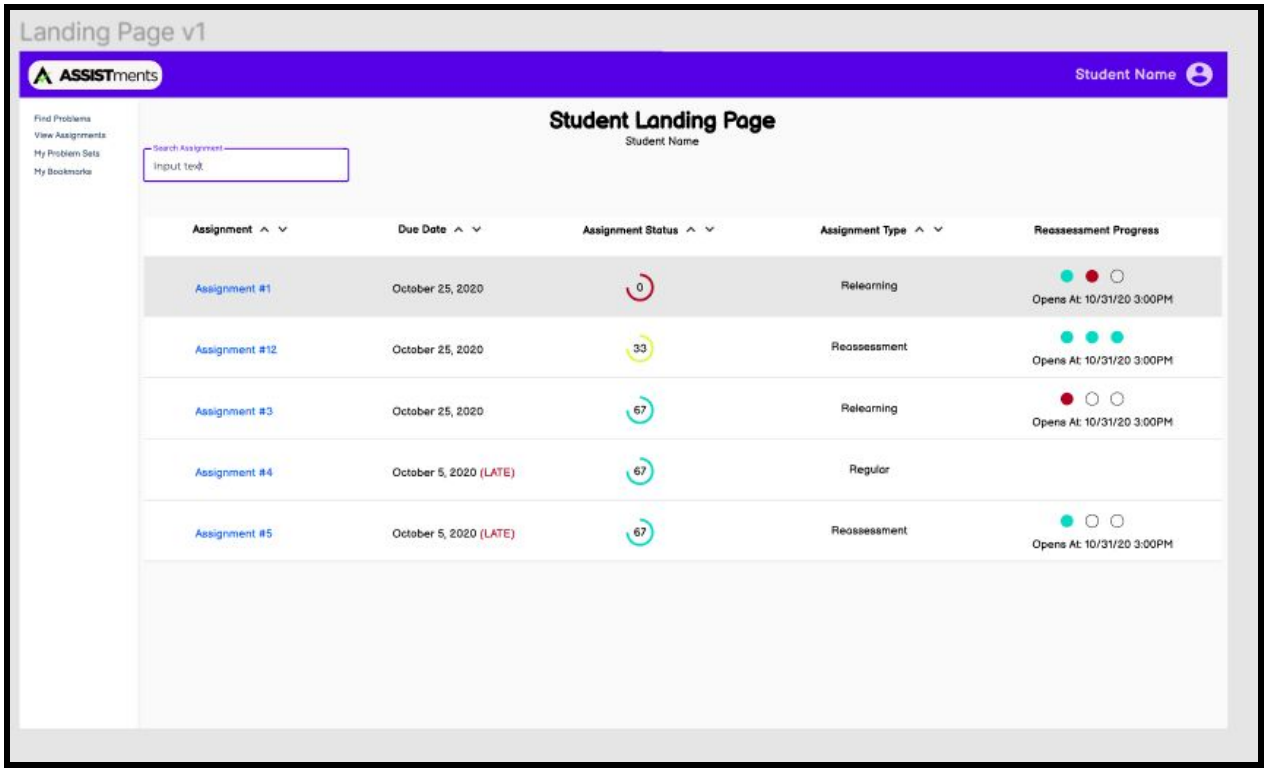

Figure 7: The original mockup for the new student landing page.

One idea that the team explored was consolidating all of the information about each ARRS assignment with the rest of the assignment, so that all reassessments and relearnings in the same sequence could be accessed together. With this in mind, the team made a second version of the mockup, as seen in Figure 8, with assignments grouped between ARRS and nonARRS and cards containing the progress information for each assignment. If the assignment was an ARRS one, its reassessments and learnings were also displayed.

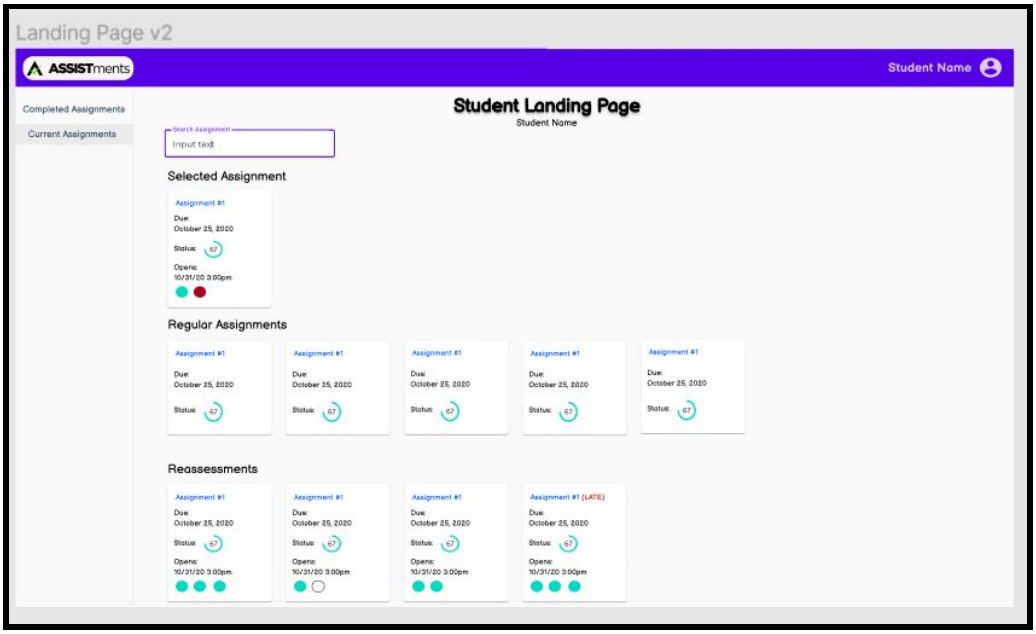

Figure 8: The second Figma Mockup for the student landing page.

The aim of the final design was to present this information in a clean, immediate manner that would not be confusing to users. To that end, the main design changes for this page were centered around one page, to provide more immediate information to the user about their upcoming assignments. For this, many new components were designed. These components were a timeline of upcoming assignments and a characteristic listing of ARRS assignments and all of

their different parts, with the original assignment, reassessments, and relearnings compiled together on one assignment card, shown by a progress indicator component as pictured in Figure 9. The progress indicator component was conceptualized as a miniaturized version of the student's overall upcoming assignment timeline, but specifically for one assignment and its information. This progress indicator would use different colors and symbols to differentiate between completion statuses and assignment types, respectively. The final mockup for the student landing page is displayed in Figure 10.

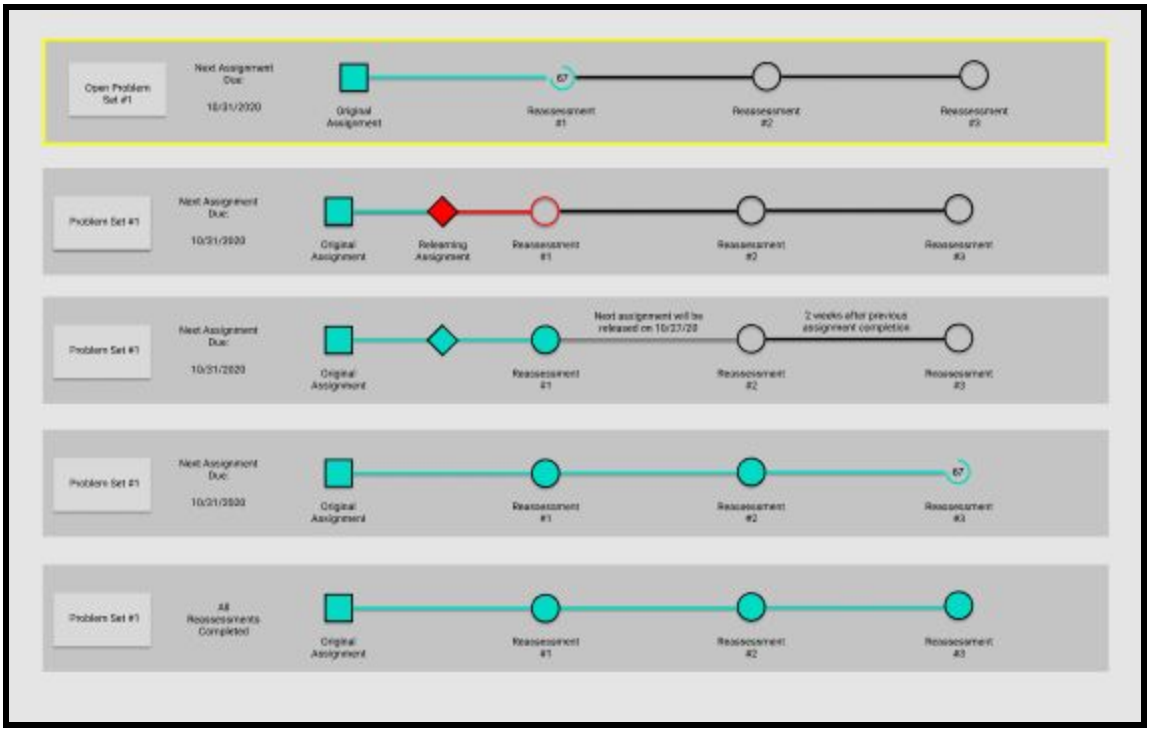

Figure 9: Initial mockups for the progress indicator component for ARRS assignments.

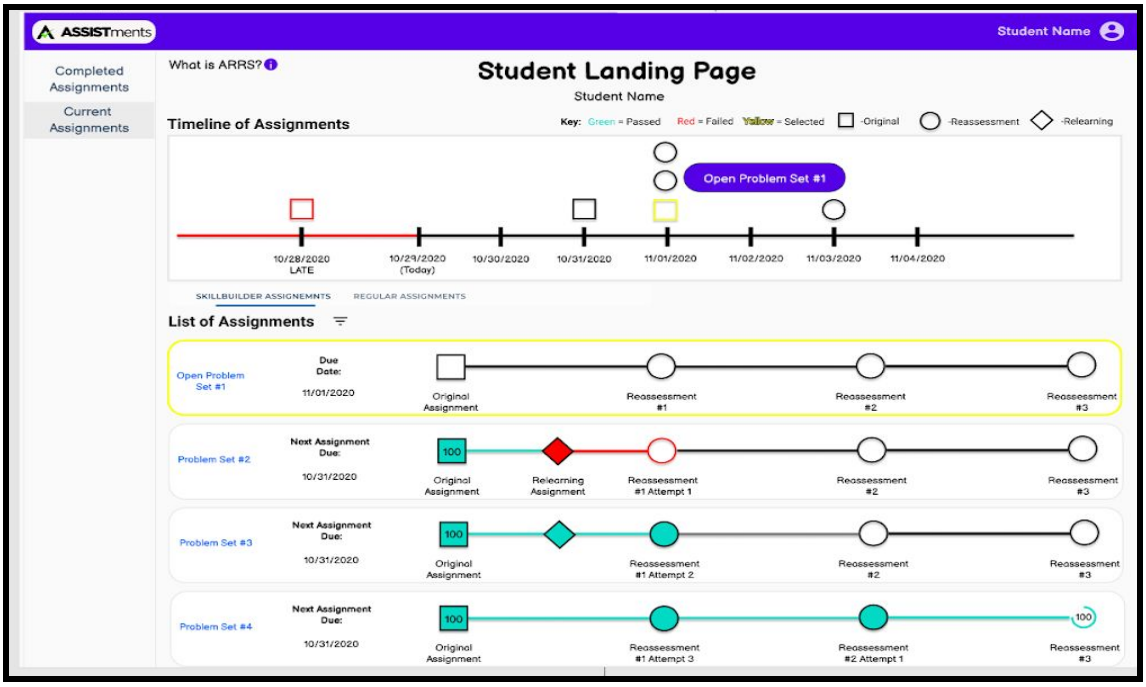

Figure 10: The final figma mockup for the student's landing page for the original implementation of these pages in this project.

Student Assignment Report Page

During typical use, a student may desire to explore the results of an ARRS assignment in more detail. As such, the team felt that a specific report page for an ARRS assignment would be necessary to display and handle information that regular assignments lack. To that end, the team made a mockup of what a student's assignment report page would look like for an ARRS assignment as seen in Figure 11. While the original design was highly similar to what a nonARRS assignment's report page would look like, a progress indicator component was included for the ARRS assignment, which would allow users to toggle presented information at the bottom of the page by selecting either the initial assignment, a reassessment, or a relearning

in the chosen assignment's sequence. The final mockup of the student assignment report page is captured in Figure 12.

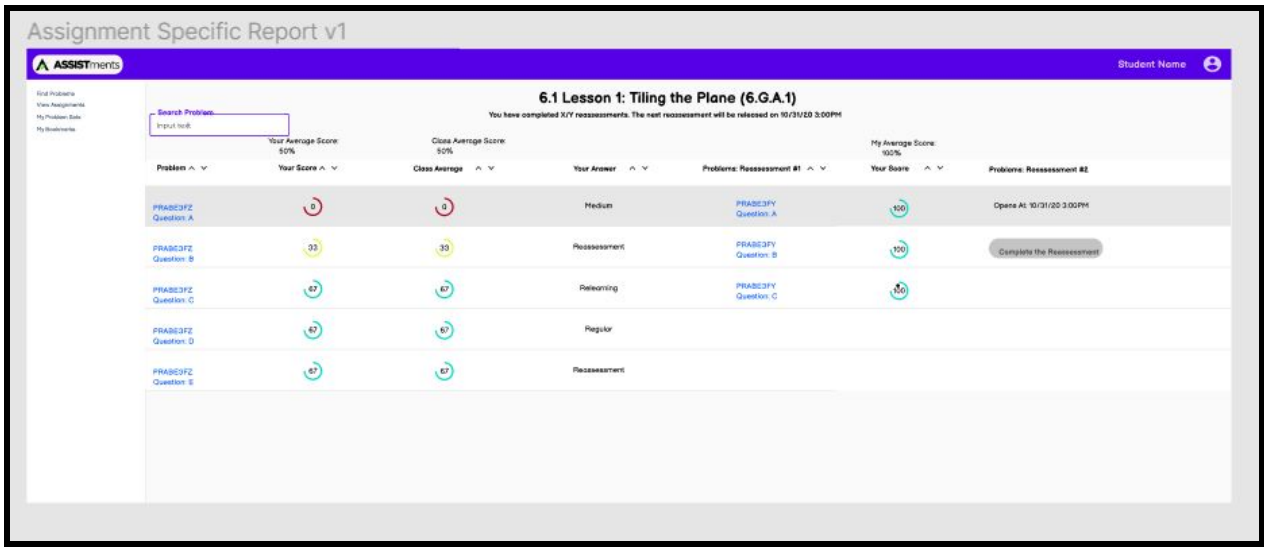

Figure 11: The initial mockup for the student's assignment report.

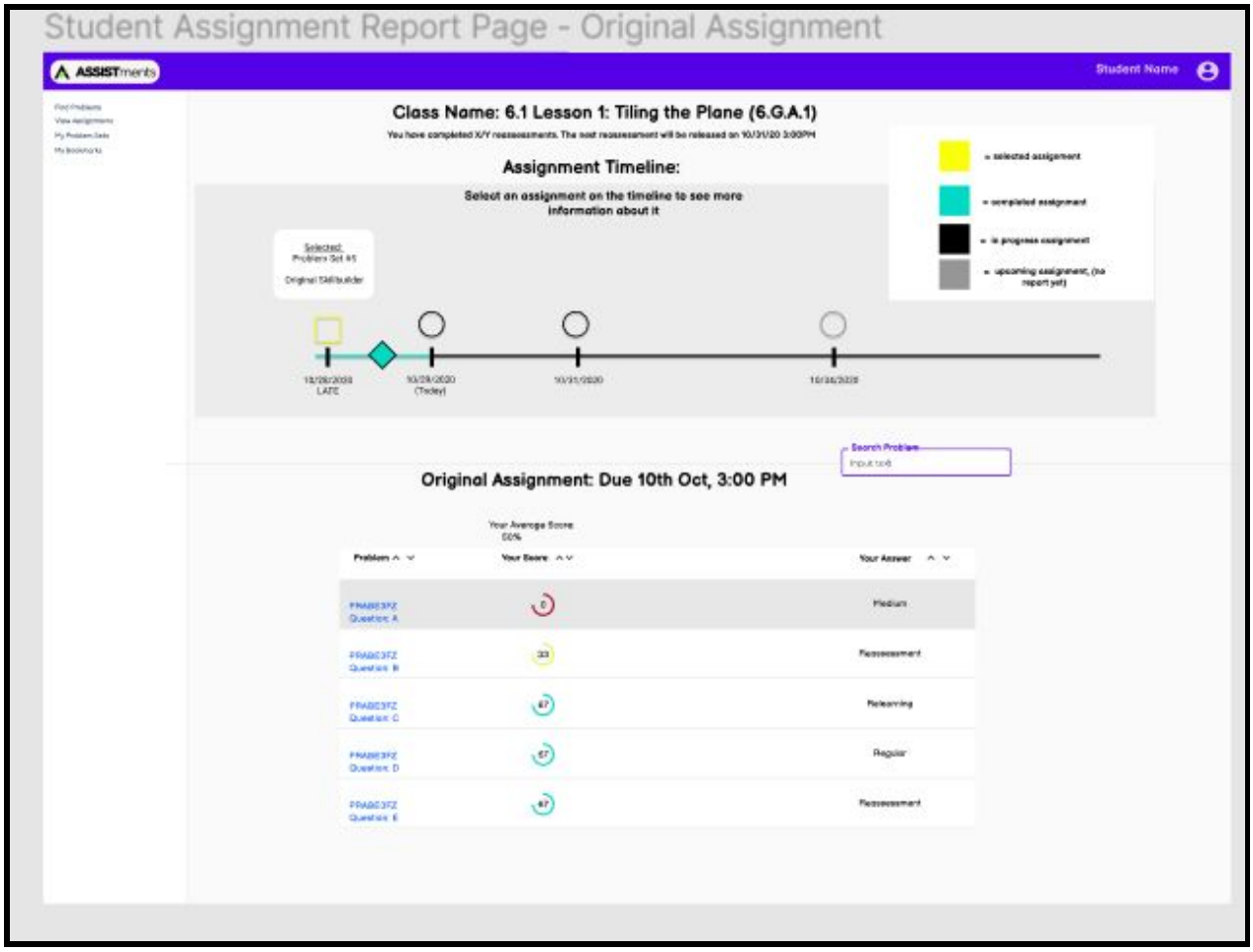

Figure 12: The final mockup for the student's assignment report in the original implementation of these pages in this project, with the original assignment selected.

Teacher Pages

The following sections discuss mockups made specifically for teacher users and include images of the designed mockups.

ARRS Settings Page

This page allows a teacher to modify their ARRS related settings, including ARRS toggling, reassessment assignment, and similar. The team felt that the content of the previous

project's settings page was sufficient, but it was still imperative to agree upon and plan the page's design prior to implementation. To that end, the team only made one significant iteration of an ARRS Settings Page, which took advantage of a similar layout to the original project page. The final mockup is captured in Figure 13.

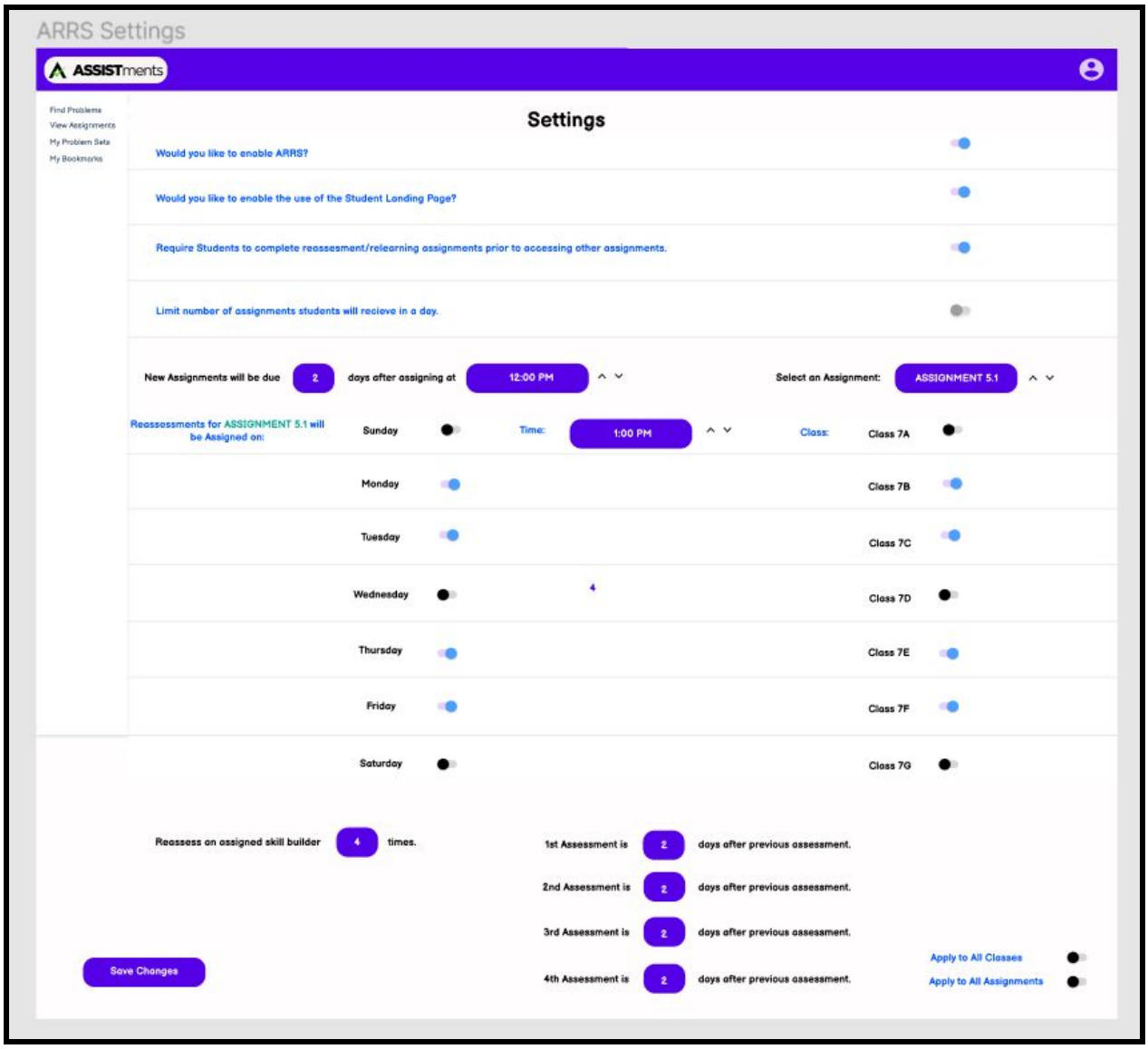

Figure 13: The final figma mockup of the teacher's ARRS settings page in the original implementation of these pages in this project.

Teacher Class Report Page

The broadest of the teacher pages for displaying information on ARRS assignments, the teacher class specific page allows a teacher to view the overall status of one of their classes. The intention of this page would be to allow teachers to view this information speedily. For that, the team decided that providing teachers with an immediate visual indicator for how their class was faring would be best suited to this particular page. To fulfill this goal, the team decided to add a version of a student velocity graph to the main view of the class report page with the goal of showing the relative progress of all students. An initial mockup of the teacher class report page and the student velocity graph can be seen in Figure 14 and 15, respectively. The team also felt that, since the page was so saturated with information, it would be applicable to implement tabs to toggle between information focused on students and information focused on assignments. An example of the tabs implemented can be found in Figure 16 and 17.

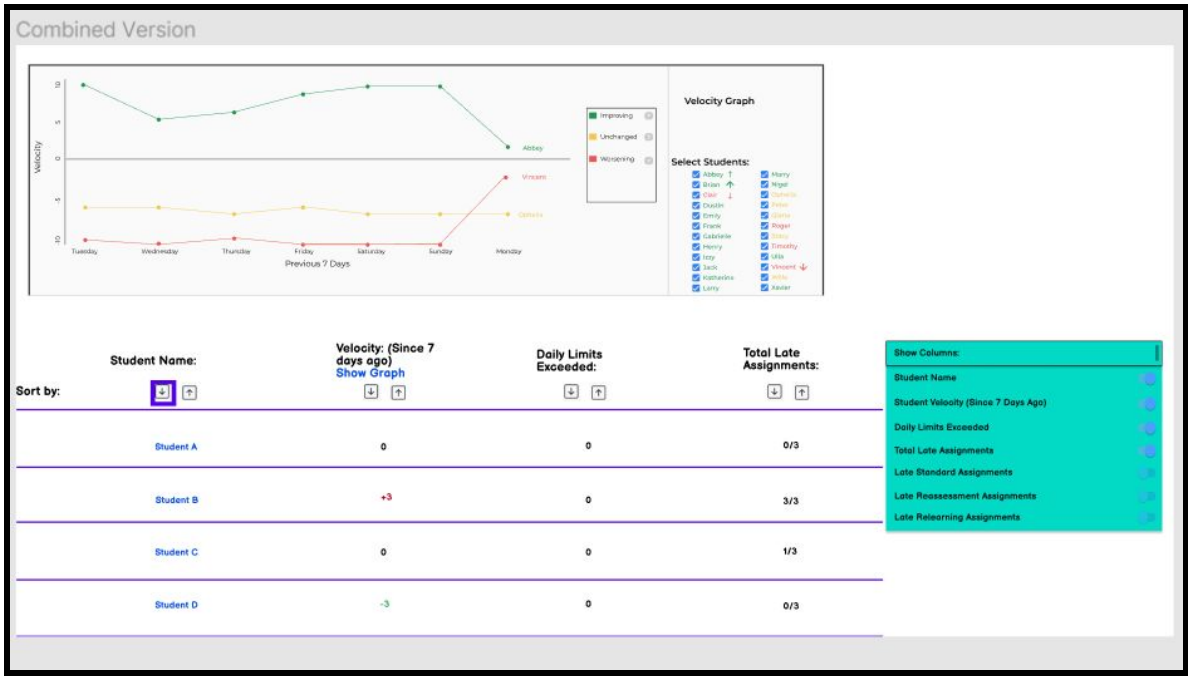

Figure 14: An initial mockup of the teacher's class report page.

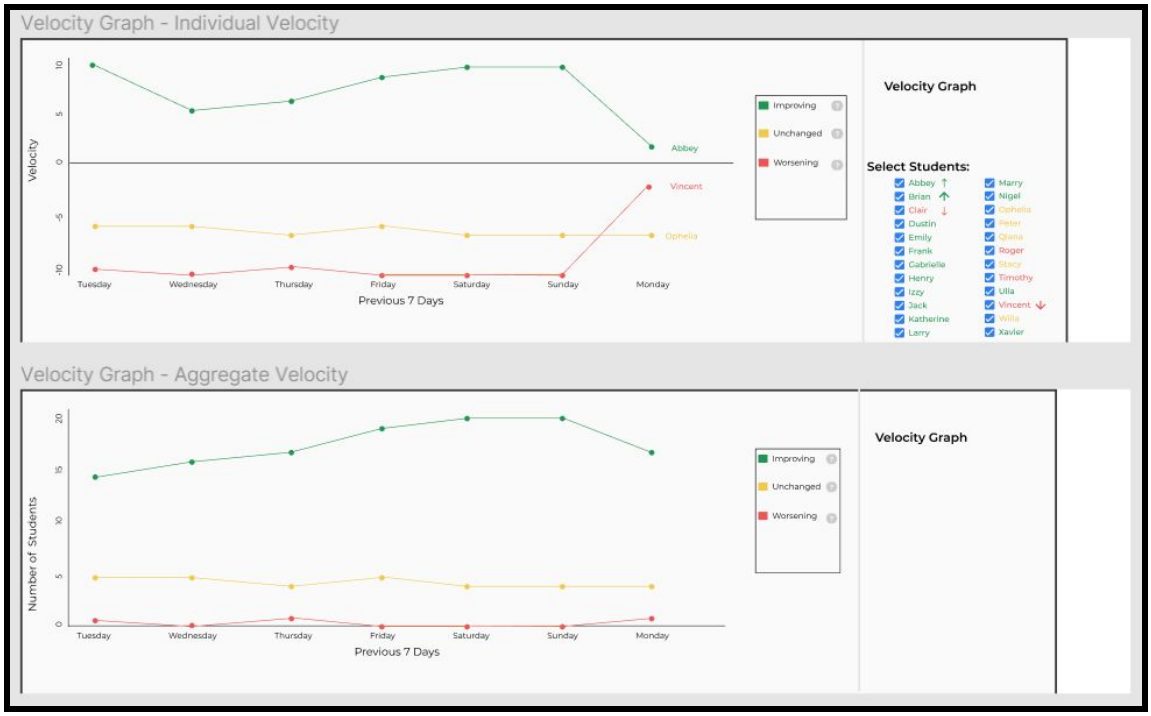

Figure 15: Mockups of the student velocity graph.

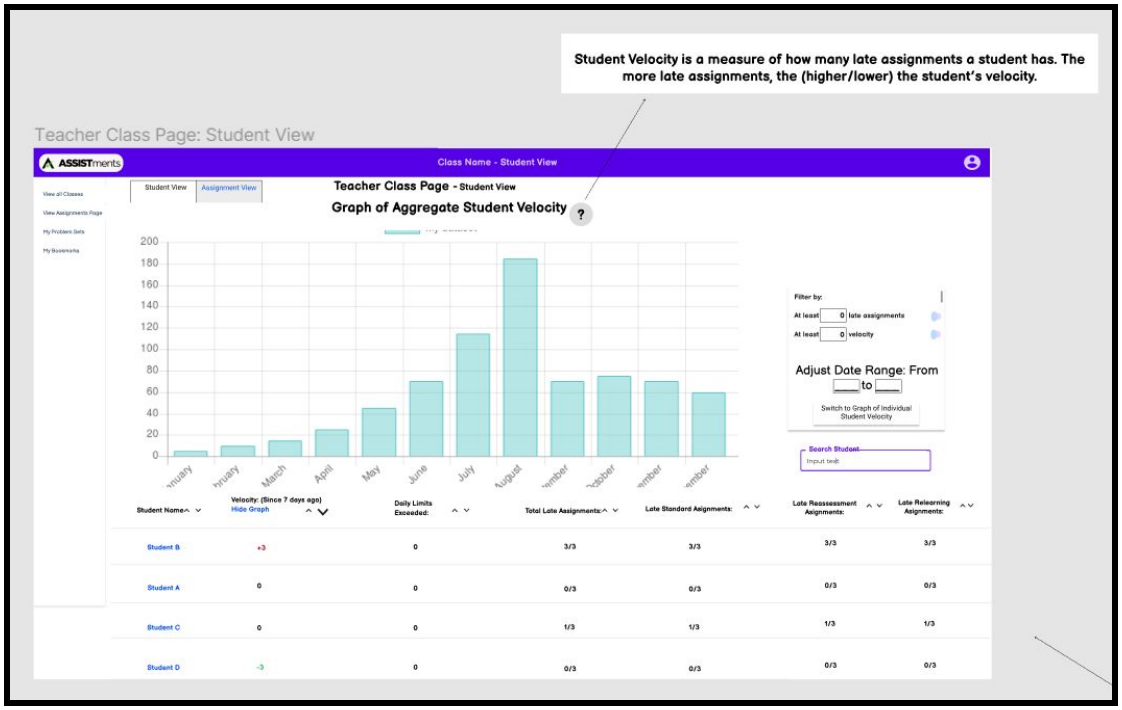

Figure 16: The final mockup of the teacher's class report page in the original implementation of these pages in this project, on a tab focused on students.

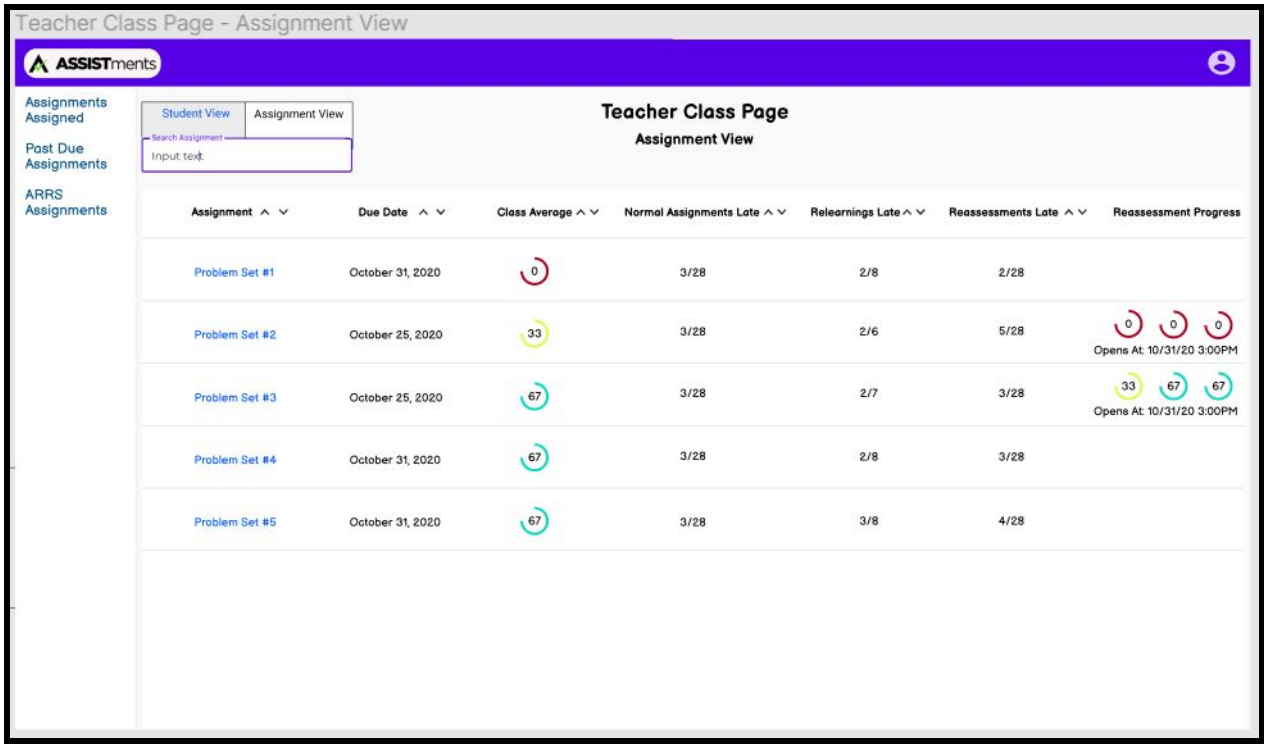

Figure 17: The final mockup of the teacher's class report page in the original implementation of these pages in this project, on a tab focused on assignments.

Teacher Assignment Report Page

This page is intended to show a class's progress on one particular ARRS assignment to the teacher. With this, the team felt that a progress indicator of the assignment could be leverage as the main focus of the page. The team intended to allow teachers to view each part of the ARRS assignment, in detail, without showing all of the information related to the assignment at once. To accomplish this, the progress indicator could be implemented to allow teachers to adjust which part of the assignment they are viewing by selecting either the initial assignment or a reassessment, which would alter the bottom of the page to display information on that particular aspect of the assignment. The team also came to the agreement that, since assignment relearning would only be applicable if students incorrectly answered their reassessment, it would be most

practical to omit relearnings from the page in the event that certain students would not have one to show. The final mockup for the teacher assignment report page can be visualized in Figure 18.

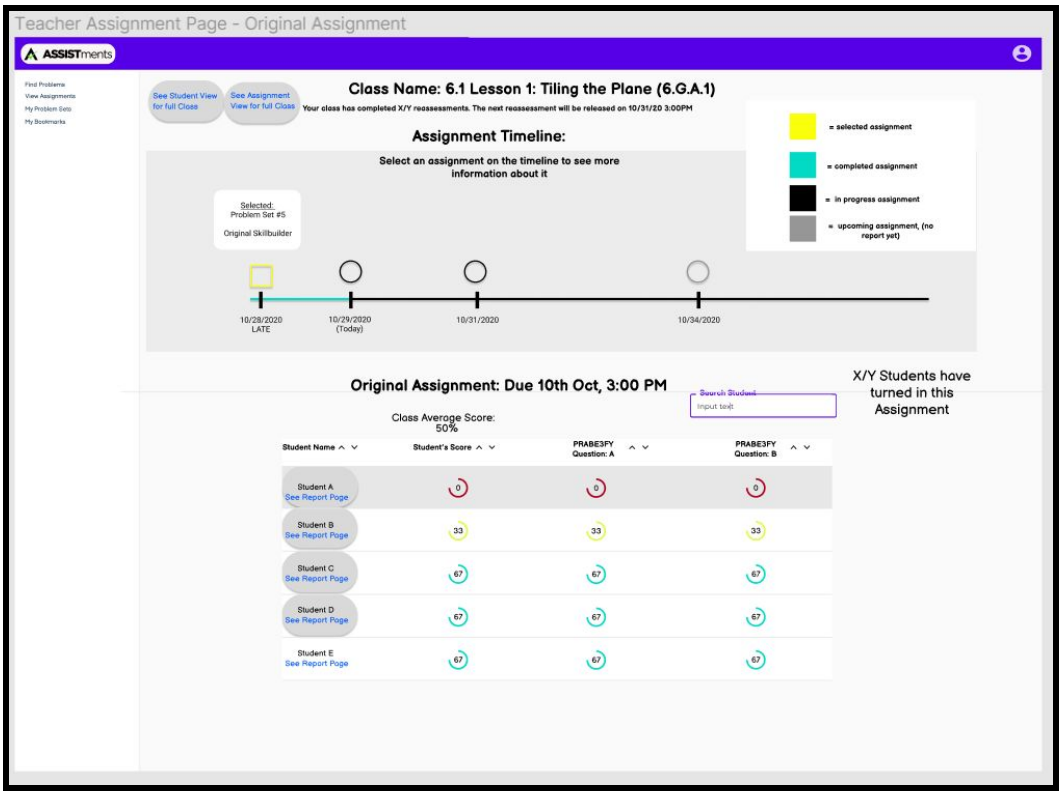

Figure 18: The final mockup for the teacher's assignment report page in the original implementation of these pages in this project, with the original assignment selected.

Teacher Student Report Page

This page is the teacher's analogue of the student landing page, in that it would show the ARRS information pertinent to a student in their class. Similar to the student landing page, this page was primarily intended to show information specific to that student's completed and upcoming ARRS assignments. The main difference intended for this page is that, instead of using the page's center space to depict a timeline of the student's upcoming assignments, a student specific velocity graph would be displayed to provide immediate indication of the

student's current status to the teacher. The initial and final mockup of the teacher's student report page can be found in Figure 19 and 20, respectively.

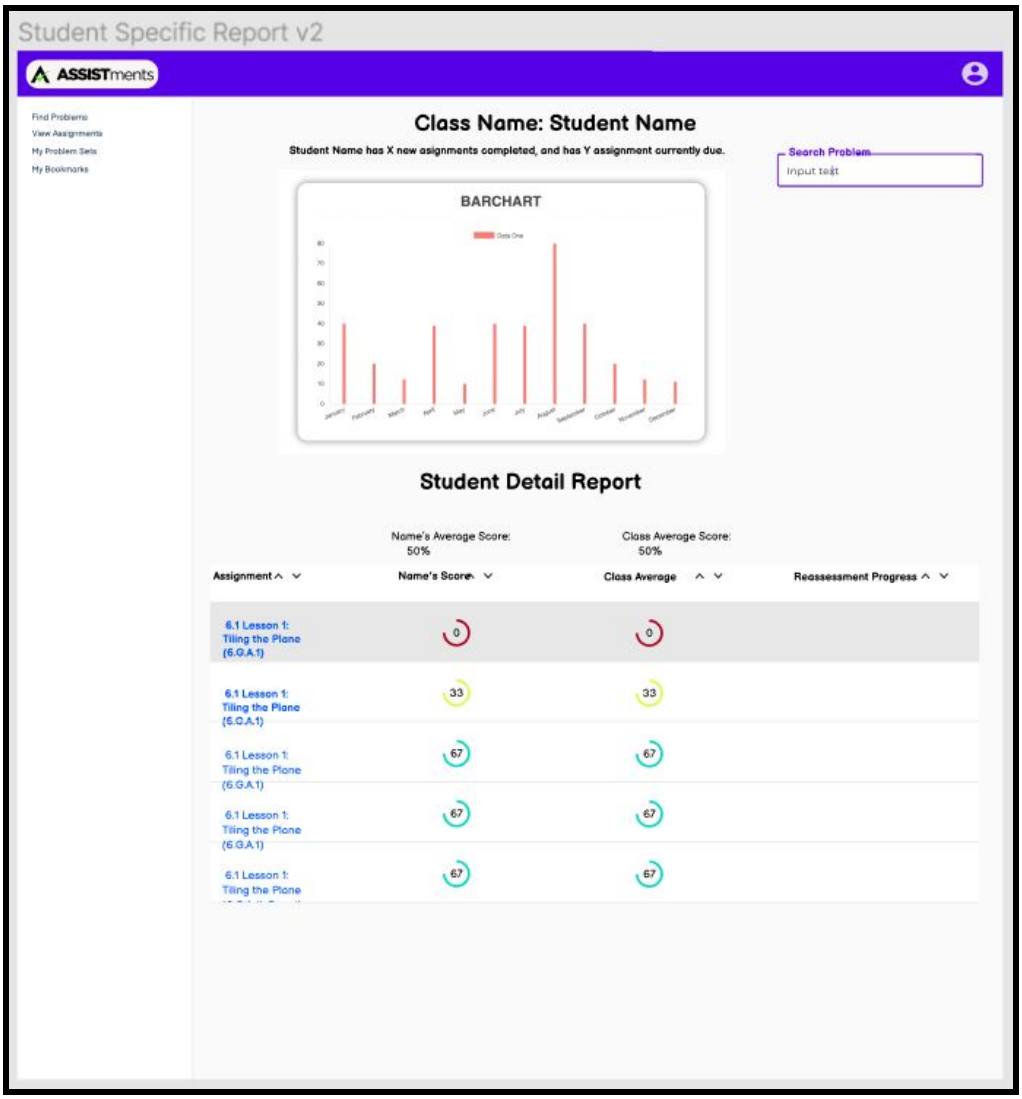

Figure 19: The initial mockup of the teacher's student report page.

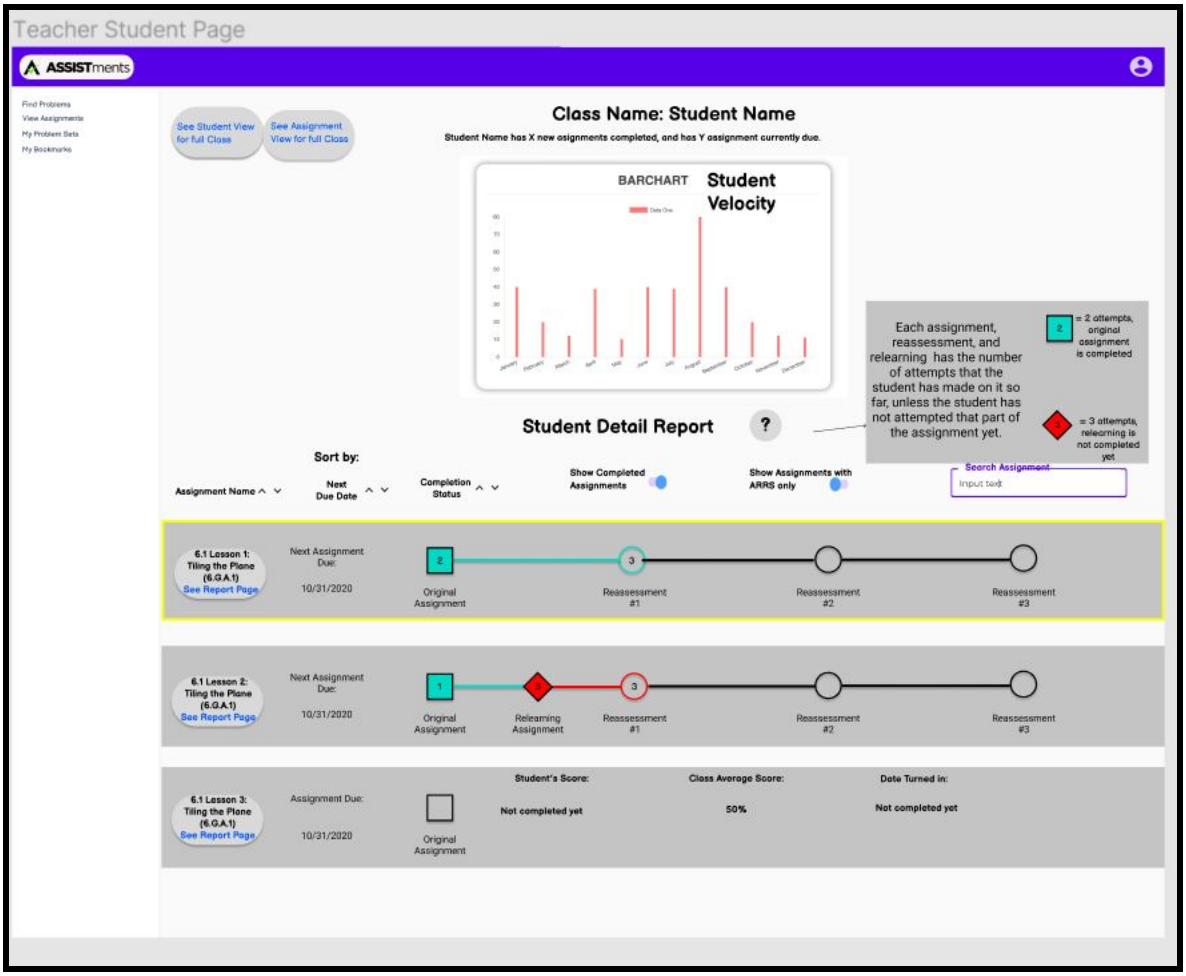

Figure 20: The final mockup of the teacher's student report page in the original implementation of these pages in this project.

Teacher Student and Assignment Report Page

This page is the most granular of the teacher's ARRS specific pages, in that it shows the details of an assignment for one specific student, similar to the student's assignment specific page. Due to its similarity to the student's assignment specific page, the team felt that the design between both pages could largely be kept consistent. As a result, the mockup differences between them were minimal and the final mockup is pictured in Figure 21.

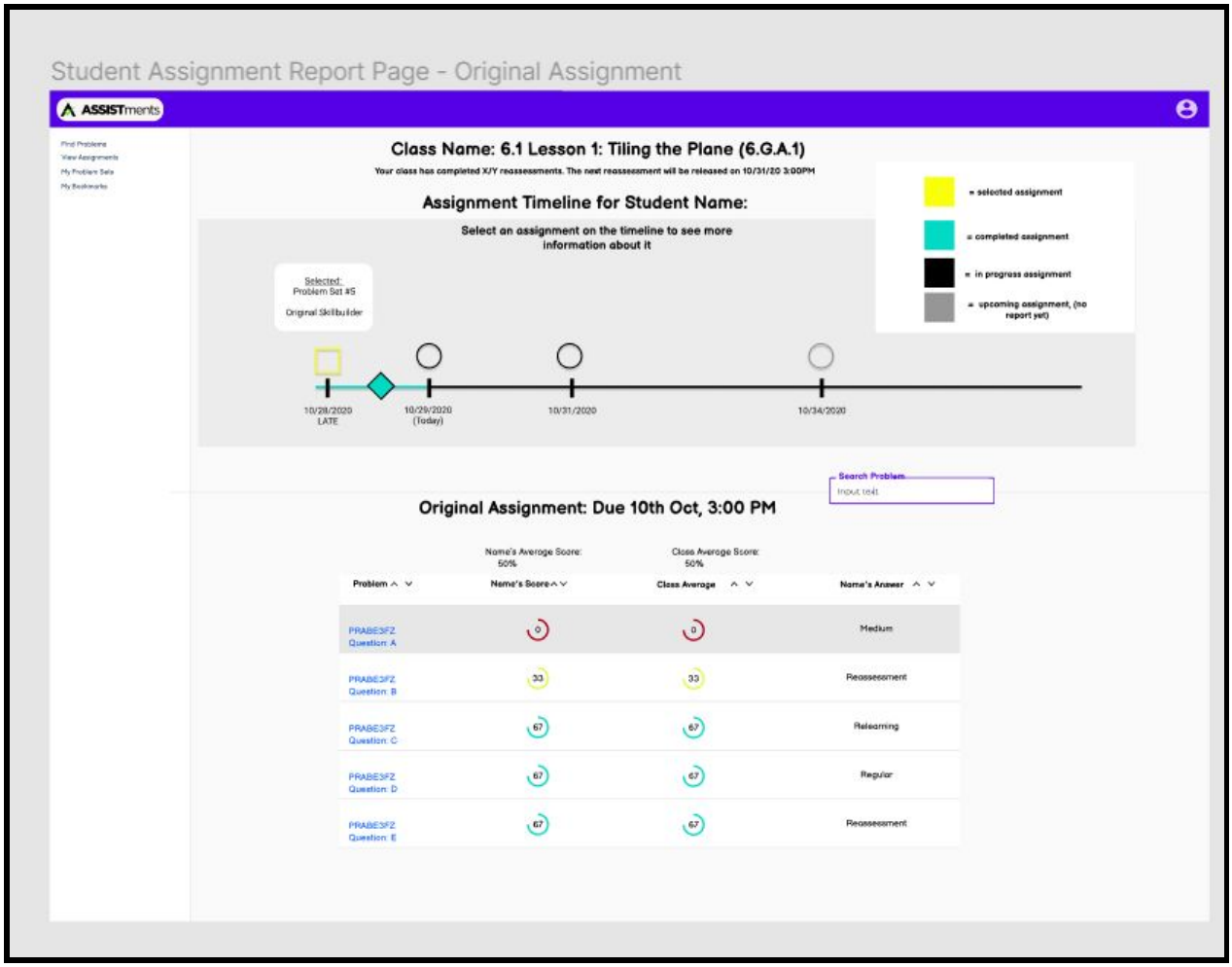

Figure 21: The final mockup of the teacher's student and assignment report page in the original implementation of these pages in this project, with the initial ARRS assignment selected.

Feedback

Throughout the mockup process, the team regularly shared mockup progress to members of the ASSISTments team for feedback on appearance and content. One of the first pieces of feedback that the team received was to construct a consolidated, graphical view of how all of the pages would interconnect so that it could be fully understood how each page would be established in ARRS. The full consolidated mockup is presented in Figure 22. This also helped to visualize typical user progression between pages and key information that they would seek to ascertain from each specific page.

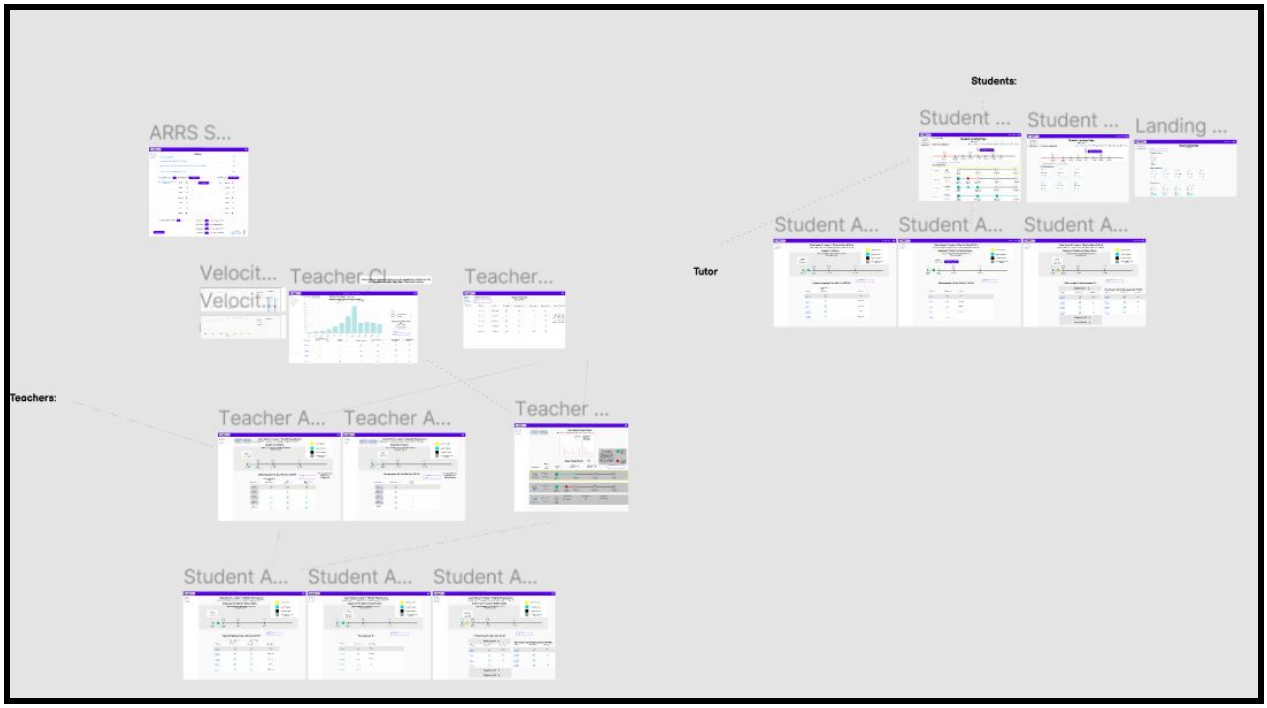

Figure 22: The full consolidated mockup view of each page, including connections between pages and different views for each page.

As mentioned earlier, when describing each of the individual pages in the mockups, the most significant addition was providing a unique visualization of information, varied from typical tables. The addition of page components such as progress indicators for ARRS assignments and a student velocity graph for teachers was motivated by feedback stating that the pages should clearly and immediately present their key information to users. This also served as a reminder to better highlight the initially selected assignment on the student landing page, even though information about the user's other assignments would also need to be displayed. Furthermore, feedback was received to consider allocating information on the pages into smaller, easy-to-digest subcategories. This ultimately led to the usage of tabs on some pages and explicit distinction between ARRS and nonARRS assignments on other pages.

Redesigned ARRS for 3.0 Demo

Following the partial implementations of the mockups in the next section of this paper, the scope of ARRS was replanned, along with a new redesign of the pages for a beta version of ARRS, to be part of ASSISTments 3.0. The reason for this requested change was to bring the design and progress of ARRS more in line with the rest of ASSISTments 3.0, as well as introduce changes into the design of ARRS. This was intended to both better fit the time constraints of the beta release, as well as more clearly define the features belonging to ARRS. With this in mind, the first steps of this redesign was to rewrite the user stories for ARRS and run them by the rest of the ASSISTments team to prioritize the importance of each feature of ARRS. One of these design changes was to use a synchronous version of ARRS in the beta release of ARRS, which is described further in the Background section of this paper, with Figure 2 showing the synchronous version of an ARRS assignment. As part of this redesign, an application outline for the pages of ARRS was created, as shown in Figures 23-24.

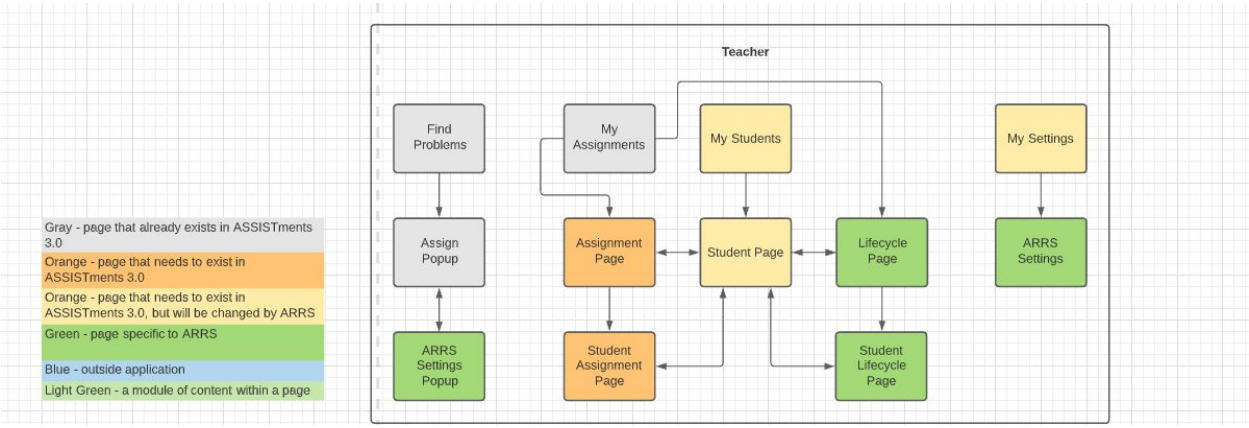

Figure 23: The new application outline for the pages of ARRS that are associated with teachers, (Used with permission from Jason King).

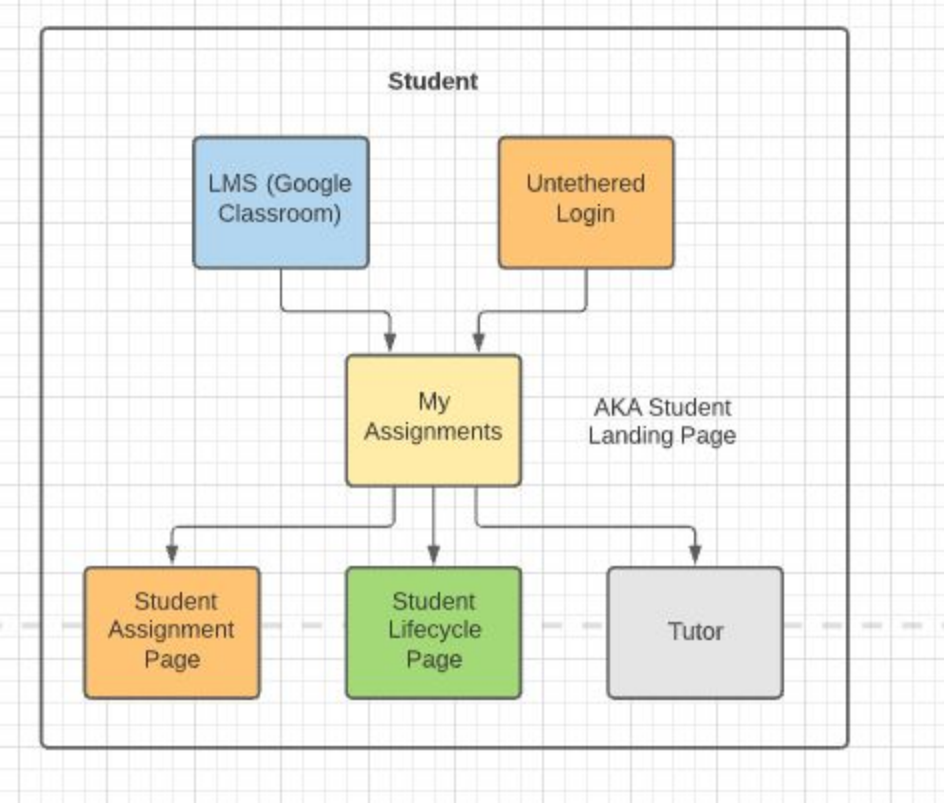

Figure 24: The new application outline for the pages of ARRS that are associated with students, (Used with permission from Jason King).

After the user stories were finalized, mockups were made for each of the pages that were assigned to be part of the first baseline implementation of ARRS for the beta release. The initial designs for many of these pages were made by Sidharath Chhatani, with refinement from the remaining members of the ARRS project team.

Student My Assignments Page

A new version of the assignments page in ASSISTments 3.0 was designed, with the intention of allowing students to view information about assignment with and without ARRS with the same formatting as had been previously designed as part of other ongoing design efforts. This would serve much of the same functionality of the previous designs for a student landing page, in that it would allow students to view upcoming ARRS assignments, reassessments,

relearnings, and serve as a point to access other ARRS related pages for students. The mockup for this page is shown in Figure 25.

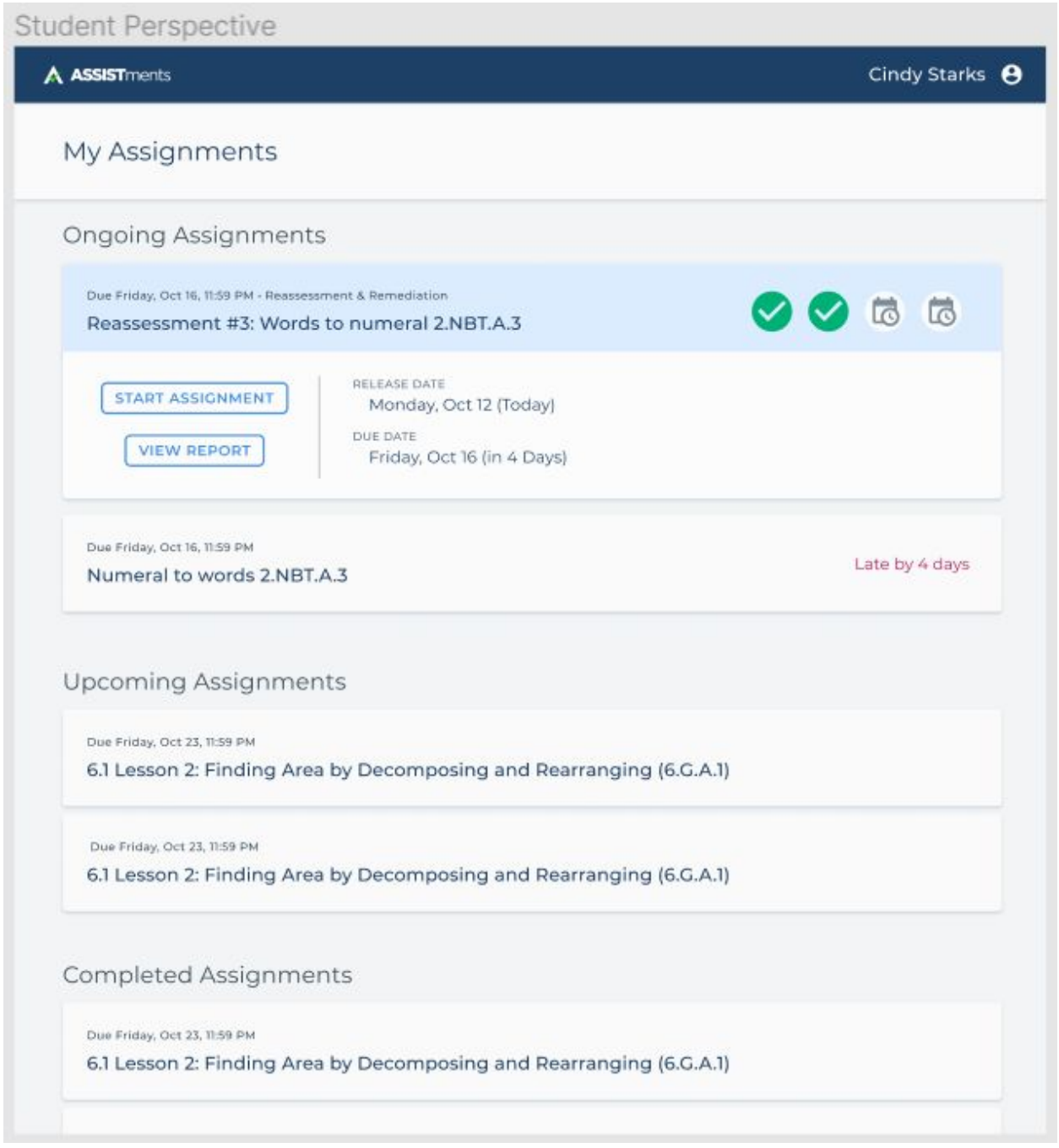

Figure 25: A mockup for a student's My Assignments page, (Used with permission from Jason King).

Other student pages were not mocked up as part of the latest set of mockups, since they were not seen as critical to a beta release of ARRS. The new review of the user stories of ARRS called into question whether students should be able to view specifically what past questions they have seen on ARRS assignments that are still ongoing, since studying questions immediately before taking a reassessment would seem to defeat the purpose of ARRS.

Teacher My Assignments Page

A similar mockup for allowing a teacher to view the assignments that they have released to their classes was designed, as seen in Figure 26. In particular, at the top of this page is a direct link to the reassessment and remediation report for ARRS assignments, which was included due to the importance of prominently displaying ARRS to teachers and to encourage usage of ARRS so that it can be evaluated.

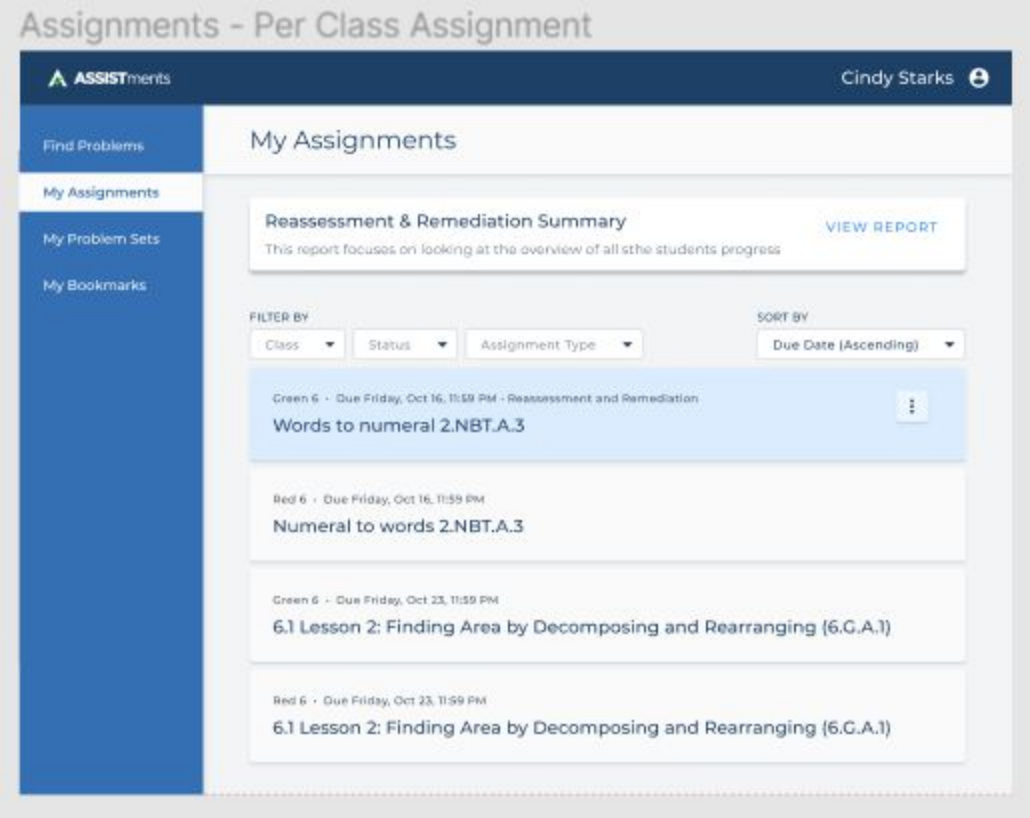

Figure 26: A mockup for a teacher's My Assignments page, (Used with permission from Jason King)

Teacher Report Pages

Two different report pages were mocked up for teachers, with the report for a single ARRS assignment shown in Figure 27. In this report, all parts of an ARRS assignment are accessible between the different options at the top of the page, with a table showing the results for the currently selected part of the ARRS assignment for all students shown at the bottom of the page. A new set of symbols representing the different stages of progress on an ARRS assignment were made for this page, as shown in Figure 28.

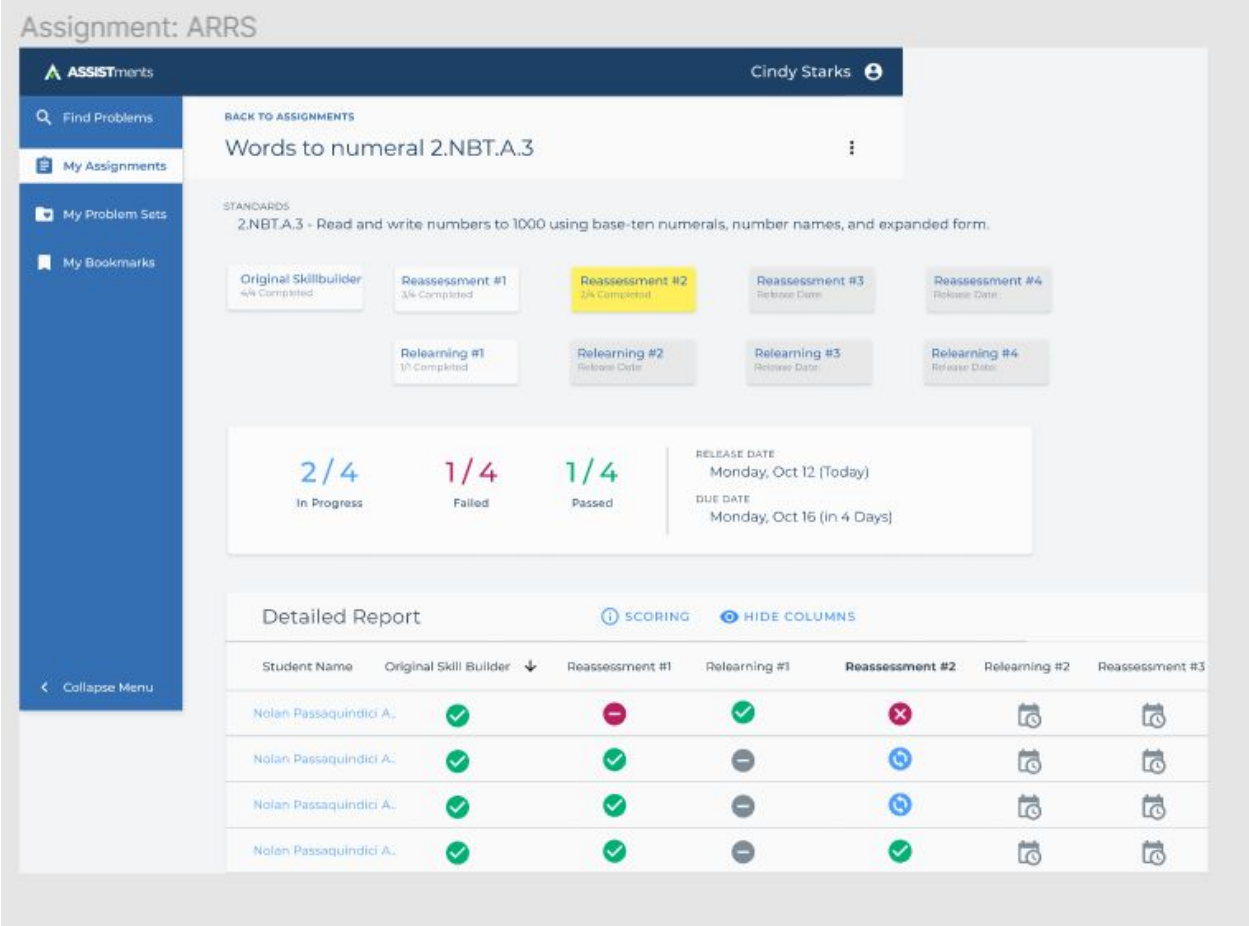

Figure 27: A mockup for the redesign of a teacher's report on a single ARRS assignment, (Used with permission from Jason King)

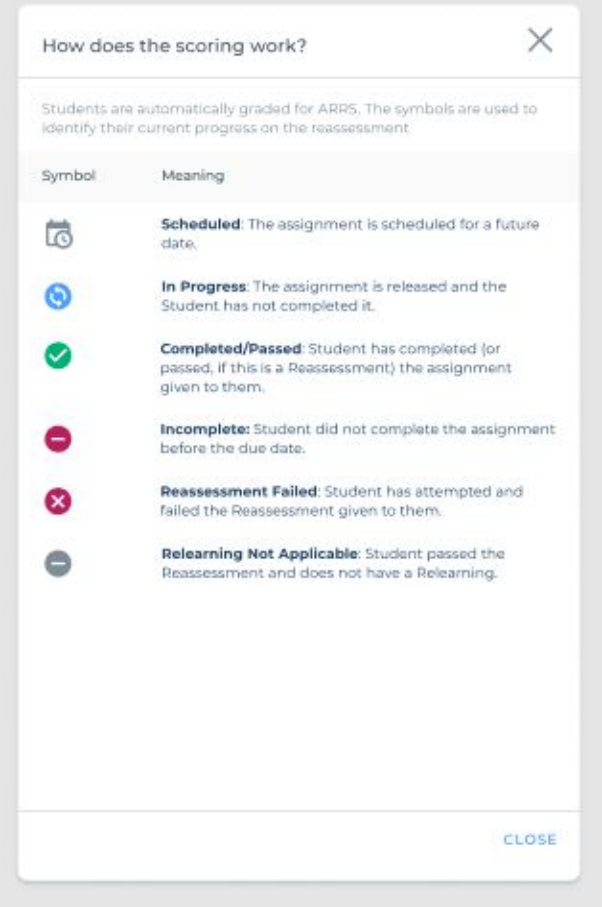

Figure 28: A mockup from the redesign of a teacher's report on a single ARRS assignment of symbols used to show student progression on the assignment, (Used with permission from Jason King)

Additionally, an overall reassessment and remediation summary report of student progress in the class on ARRS assignments was redesigned to better fit the new concept of synchronous ARRS assignments, since students could no longer retry relearnings and reassessments with ever extending due dates. This report was split into two tabs, as seen in Figures 29-30, to show the recent and all time progress of students in a class on ARRS assignments.

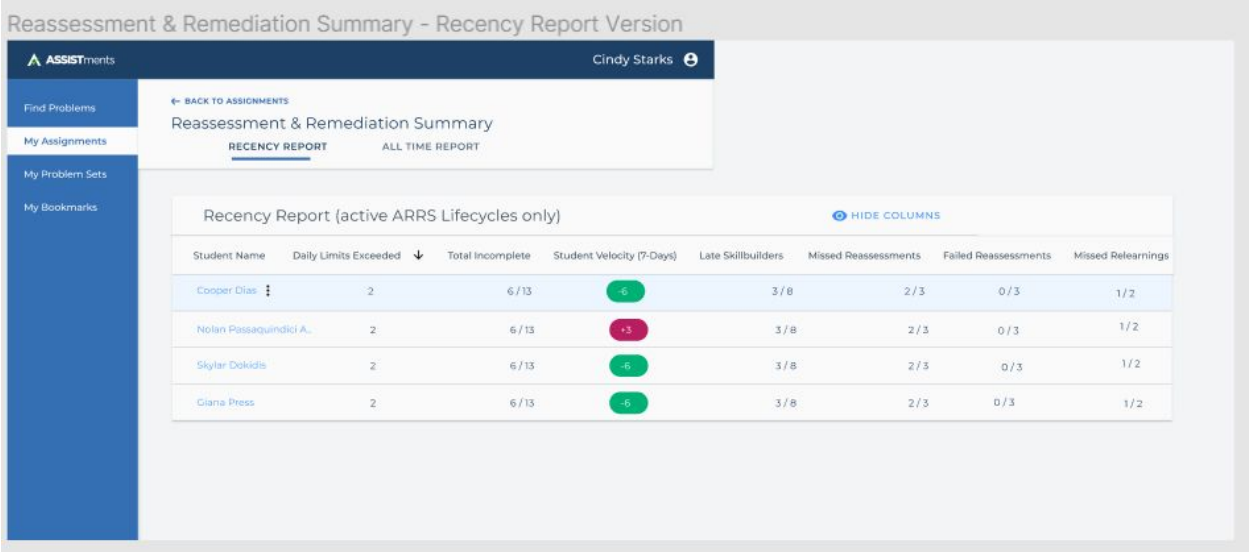

Figure 29: A mockup from the redesign of a teacher's reassessment and remediation summary report on a class's recent performance on ARRS assignments, (Used with permission from Jason King)

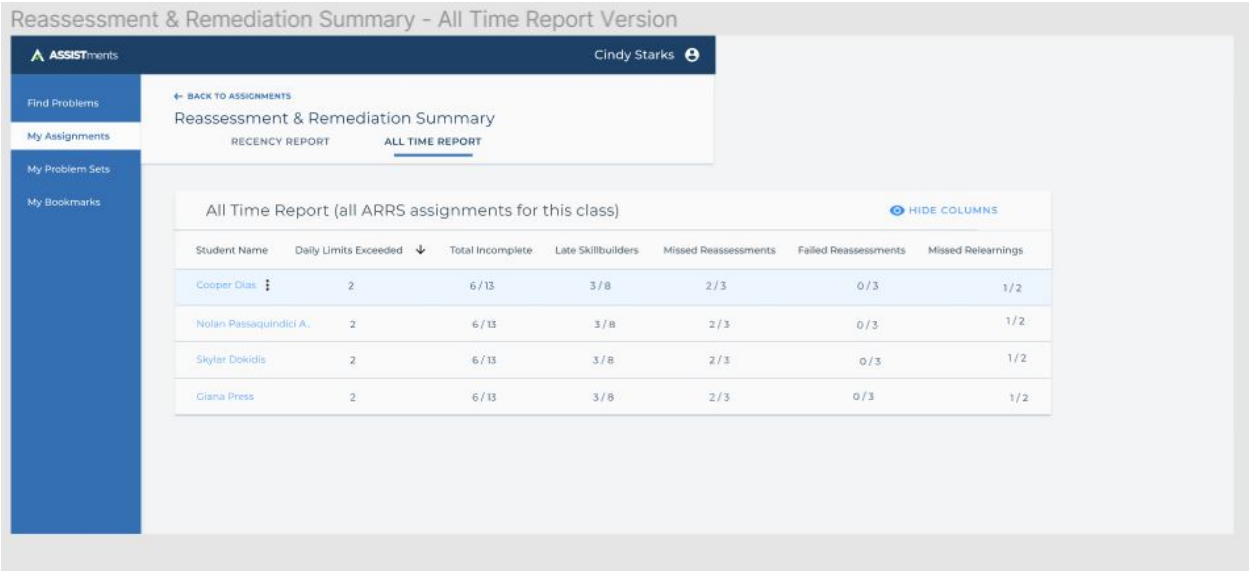

Figure 30: A mockup from the redesign of a teacher's reassessment and remediation summary report on a class's all time performance on ARRS assignments, (Used with permission from Jason King)

Backend Design

Once the user stories were finalized, the backend diagram of database tables necessary for ARRS was created, in order to make the process of hooking up the frontend pages of ARRS to the backend of ASSISTments go as smoothly as possible. Tables were outlined for storing information on all parts of an ARRS assignment, including the initial assignment, (also called the root assignment), reassessments, and relearnings. The goal of our table layouts was to minimize the amount of duplicated information between tables while still permitting students and teachers to access the information necessary to them on all ARRS assignments without requiring an excessively long query time. An initial diagram of the planned tables is shown in Figure 31, and a more finalized diagram is shown in Figure 32.

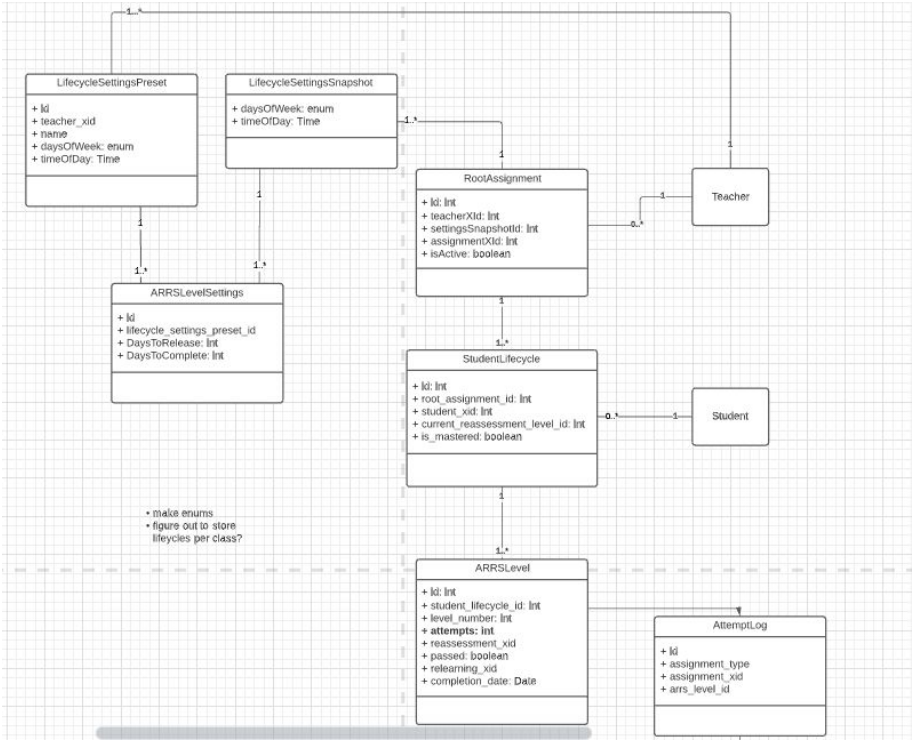

Figure 31: An initial diagram of backend database tables for the redesigned implementation of ARRS, (Used with permission from Jason King)

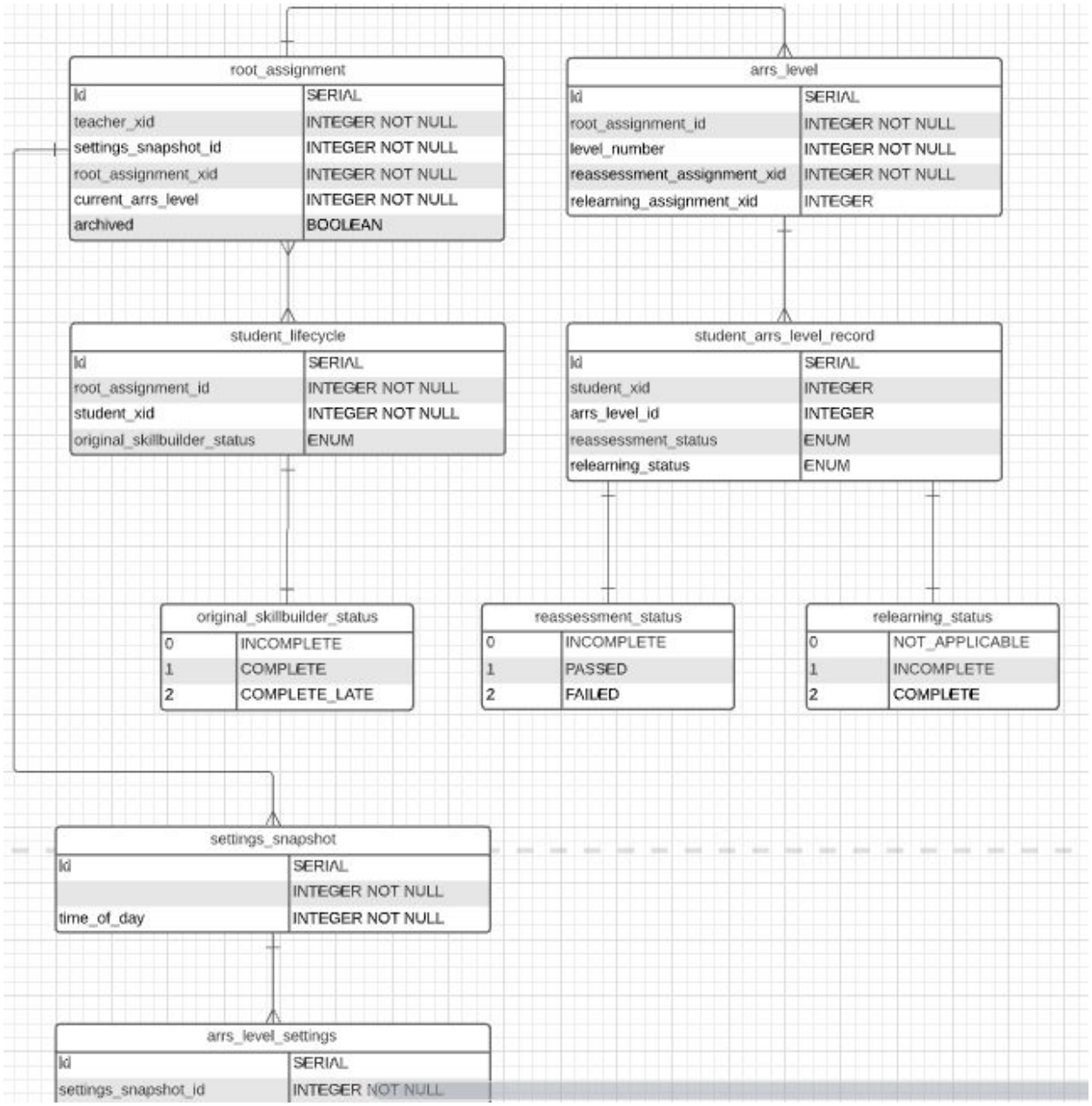

Figure 32: A more finalized diagram of backend database tables for the redesigned implementation of ARRS, (Used with permission from Jason King)

4. DEVELOPMENT

In this original attempt at implementation, the team prioritized the pages that were deemed to be critical ARRS groundwork to finish. It was decided that the most important pages to develop were the student landing page, the student assignment report page, the ARRS settings page, and the teacher's class report page. The team felt that in completing both pages for the student, as well as some of the broad pages for the teacher, the more granular pages for the teacher that reused the components could smoothly be completed later. With that in mind, a new branch was created of a repository which was being used to rope the functionality of TNG (the next generation of ASSISTments) to Vue. However, each of the following pages were implemented before the most recent redesign of ARRS was proposed, and as a result they are outdated compared to the newest designs for ARRS.

The four core pages were implemented without connection to the backend server. As a result, test assignment data had to be utilized instead of real data to simulate how each page would function. The team started by manually creating assignment records, which included reassessments and relearnings, by using a mirage js server as a placeholder for the actual server. In order to complete the class report page, simulated data for teachers and classes for the purpose of properly grouping student data was also made.

Development initialized with the student landing page. To begin, the group drafted a timeline of the student's assignments. This demanded a few design iterations to properly adjust the outward appearance of the timeline and to display reassessments and relearnings properly. The end result was a timeline that depicts a user's selected assignment, late assignments, and upcoming assignments while concealing any unreleased or completed assignments over the

course of the next two weeks. The progress indicator was implemented for specific assignments and the assignment specific cards were established as reusable Vue components so that potentially anyone could reuse them on other ARRS specific pages. To assist students and teachers with understanding what the timeline and progress indicators were showing, a modal help button component was erected for use on every page. This modal help button displays a guide to what the symbols, colors on the timeline, and the progress indicator represent. An example of the difference between the past student landing page and current version are outlined in Figure 33 and 34, respectively.

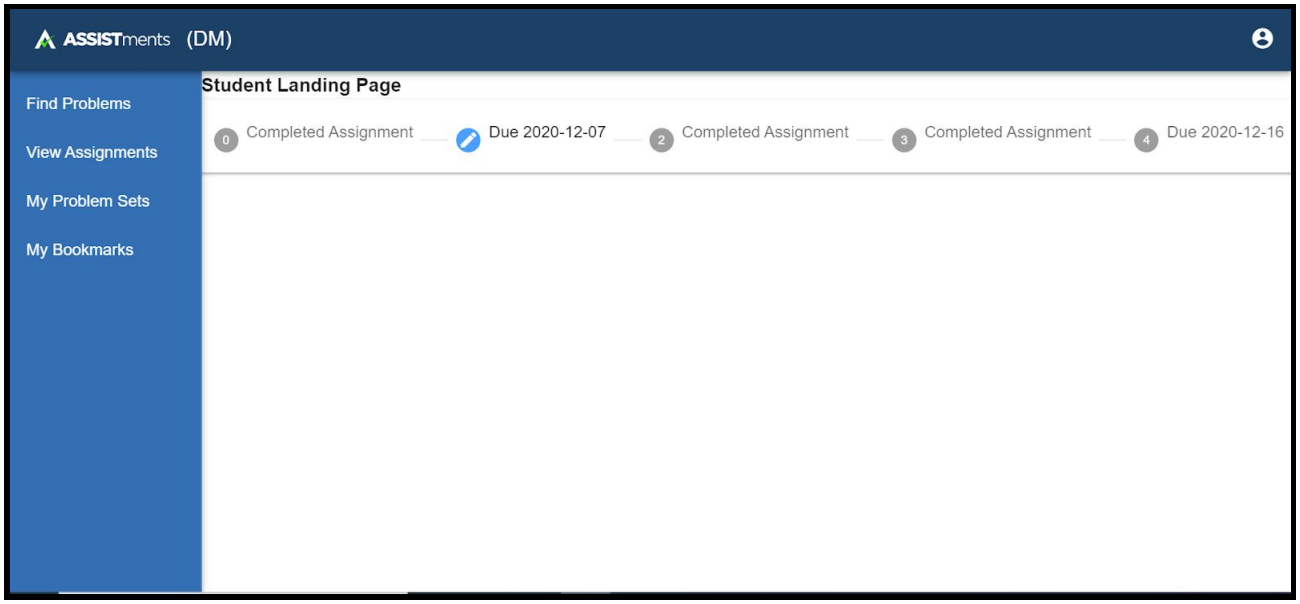

Figure 33: An earlier version of the student landing page in Vue.

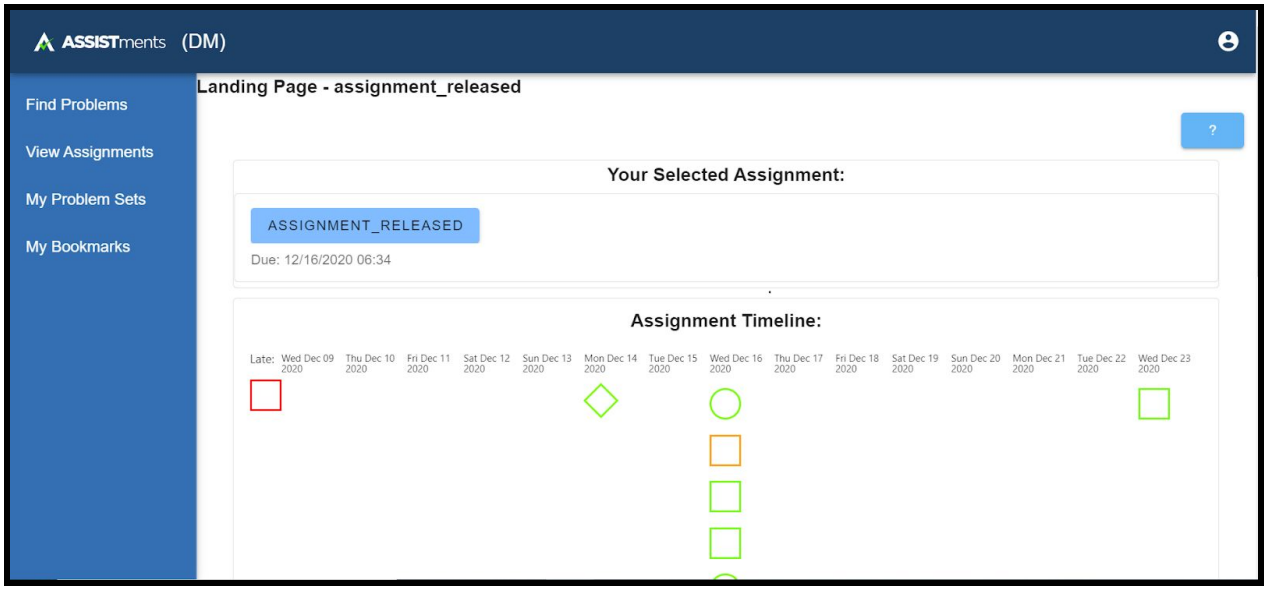

Figure 34: The current student landing page in Vue.

Next, the ARRS Settings Page was edited because it contains a great amount of information that ARRS would introduce. Moreover, the page is one of the first that a teacher will land upon after initializing ASSISTments, and as such it was also one of the top priority teacher pages. Secondly, this page manages the control for all the other pages in teachers view, as it enables and disables all ARRS settings, applies features to all classes and assignments, and decides when new assignments will be assigned and when skillbuilders will be reassessed. This page was reiterated thrice, as the first iteration was implemented through an external library called Bootstrap. After this, it was decided by the Front-end team that Vuetify should replace Bootstrap and so all components on the page were transformed to Vuetify components. Then, in the third iteration, increased options to allow teachers to digress from the default ASSISTments settings were implemented. Due to the fact that there was not a direct connection to the backend of ASSISTments, the main goal of this design was to ensure that the format and visuals for the page were set in stone. It was also a priority to make sure the settings could be updated and saved

locally for the purpose of seamlessly passing along the information to the database when connected to the backend. An early version of the settings page along with the finalized version are introduced in Figure 35 and Figure 36, respectively.

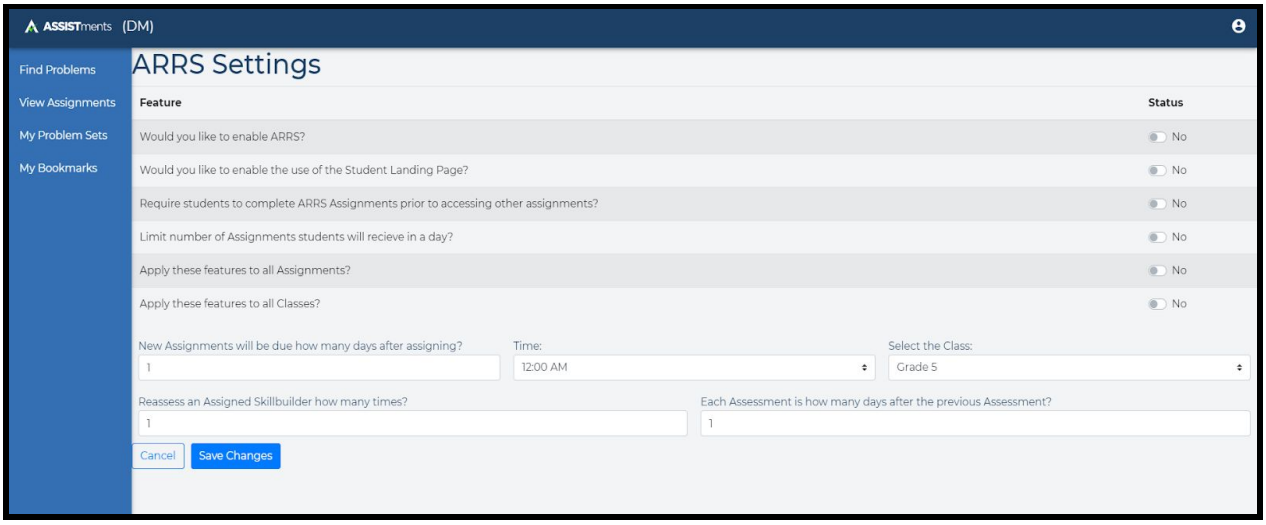

Figure 35: An earlier version of the ARRS settings page in Vue

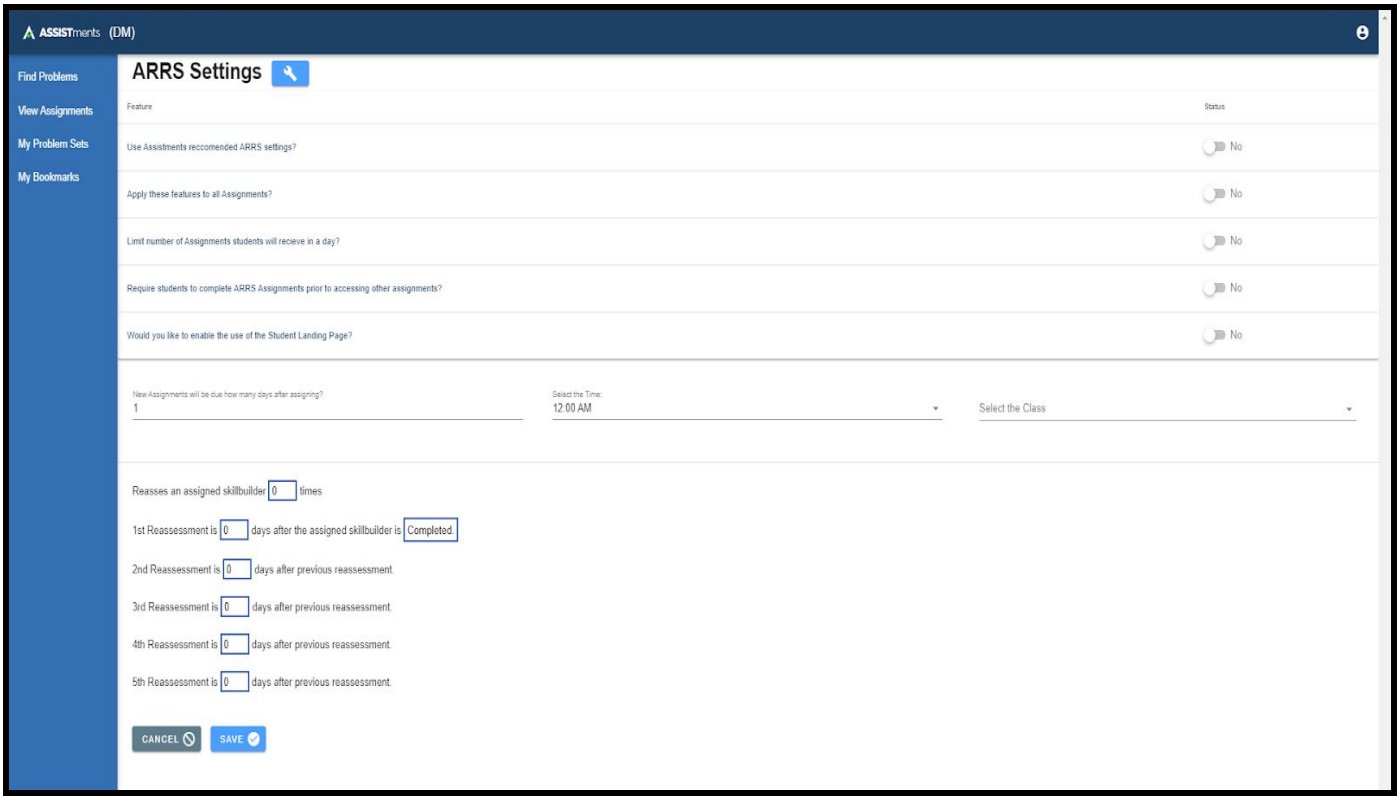

Figure 36: Final version of the ARRS settings page in Vue.

The team then switched focus to the Student Assignment Report page. For this page, the progress indicator component was modified to make each assignment selectable for use and the component was extended to pass the selected assignment's information to the page itself. On the mockups, it was originally decided that the bottom segment of this page would show information pertaining to the selected assignment in the timeline, but it was later decided that it would be prudent to coordinate the display of this information with other members of the ASSISTments team, since a component that displays an assignment's information would be in use for nonARRS assignments as well and may cause an overlap.

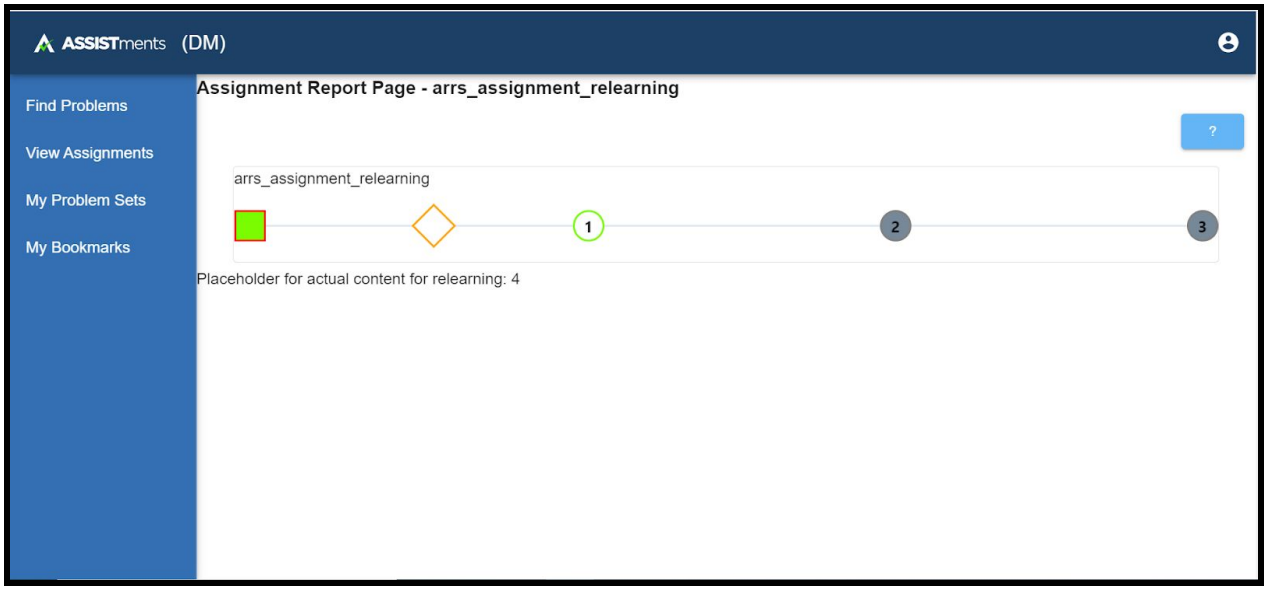

Figure 37: The student assignment report page in Vue.

Lastly, the group worked on the teacher class report page. Originally, the plan was to show all of the data for the students in a table, but then it was decided to focus on the ARRS side of the data to again reduce overlapping with other work. It was decided to show student name, late original assignments, late reassessments, late relearnings, and total late as fields on table. With this data, it could be easy to set up the computation behind the student velocity graph as well. For the table to be shown correctly, the mockup data had to be altered to connect classes,

teachers, and students. Once the actual data is used later on, the calls to the individual fields will have to be changed to access them. With the usage of Vue data table, individual columns can be filtered and allowed teachers to easily sort however they want. Once everything was set up, the logic behind determining late assignments had to be calculated and passed on to the table.

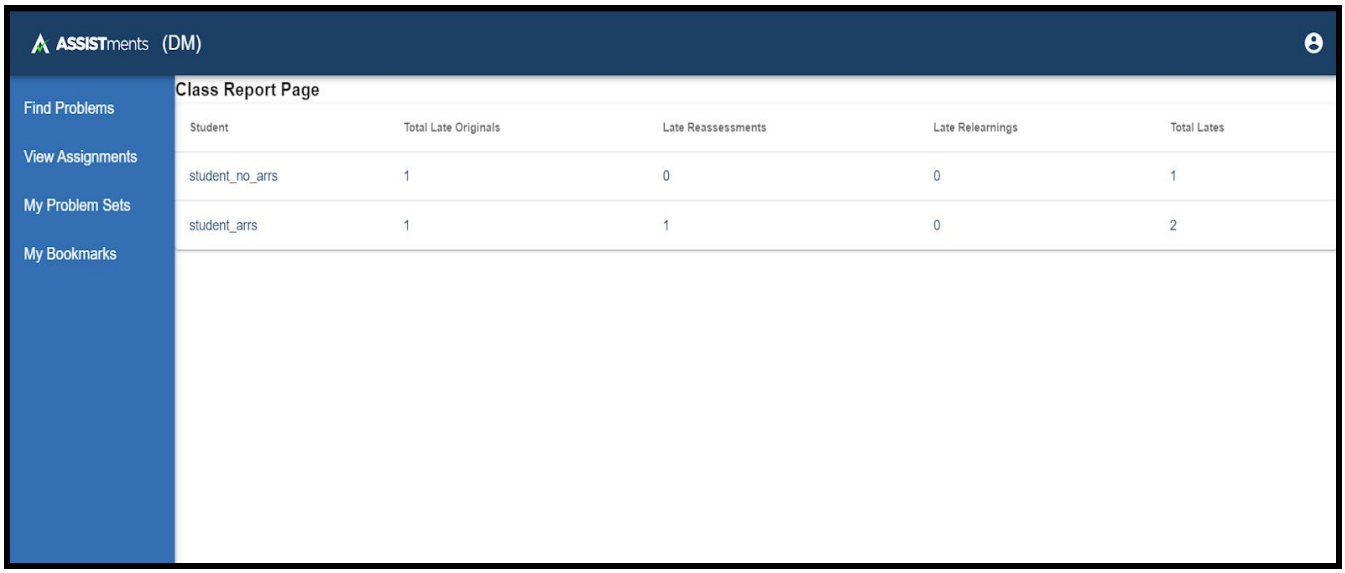

Figure 38: The class report page in Vue.

5. CONCLUSION AND RECOMMENDATIONS

This section describes the major conclusions of this proposal and recommendations regarding the future continuation of this project.

Conclusion

As ASSISTments continues to make the transition from JSP to Vue, ARRS follows. The goal of this project was to aid in the front end development of ARRS as it was introduced to Vue.js. After usertesting and the analysis of teacher and student user requirements, the team created UI mockups for seven distinct pages to be added to the new design of ARRS in Vue.js. Four of these pages were then developed and implemented as a start to this transition. As a result of the iterative design and development process, the team was successfully able to implement part of a viable product capable of addressing the varied needs of the teacher and student users that would utilize the new ARRS system, before the new redesign was proposed. Many of the suggestions that the team received were implemented, such as styling the pages uniformly, providing a key to understand ARRS, separating the different types of assignments under tabs, simplifying the timeline and ARRS rows, and easier navigation. With these implementations along with the newest planning behind ARRS the team hopes that these changes aid in the further development of the beta version of ARRS in ASSISTments 3.0. Any further changes, improvements, and ideas that were suggested during the design and implementation process were analyzed and allocated to future work.

General Recommendations

One of the most significant challenges that the team faced during the course of this project was a lack of definitive direction. The original intention of the project was to build upon the pages from the previous MQP's work so that the functionality of ARRS could finish being added to ASSISTments 2.0. However, the group was informed partway through setup that ASSISTments was transitioning to a new API version which consisted of Vue.js on the frontend. A decision was given to the team to either keep following the original instruction and implement pages that would be overwritten eventually, or learn to use Vue and start remaking the pages in Vue from scratch. The latter option was chosen since the team felt that writing code which would be overwritten shortly after the team left and graduated would not be productive or important to the development of ASSISTments. Furthermore, this led the team to not have a definite path of direction until Mid October, which was half way through the Fall semester. It is recommended that there should be a more thorough project pipeline, leading to less time being spent establishing what future project teams are to be doing, and more time actually performing development work. Lastly, there should be a more thorough communication line between the advisors as there were a lot of conflicting ideas the team had to process through, and there was limited access and meetings with the advisors in the first place, and there was a lack of thorough support and guidance for students in the project. Lastly, it is recommended that ASSISTments stay an inviting work and learning environment where students are able to ask for help and be met with only encouraging and helpful responses.

Original Recommendations for Future Work on ARRS

These recommendations were made before the most recent redesign of ARRS. With a base layout of some of the ARRS pages, there needs to be a way to stress test them with the backend data that the company already has. There is currently no hookup to the real database and if the pages want to be pushed to the actual website; the team suggests that they be tested thoroughly with the actual data. On top of that, other employees were already working on the base design of the vue pages, hence why the team only showed the ARRS features on the pages, so the team advises that the two sides of the pages are linked before going into further development. Additionally, the team was not able to share the completed pages in Vue with test users due to time constraints, though it would be recommended to perform user testing before integrating the pages with the rest of ASSISTments.

Student Velocity Graph

The student velocity graph was a feature that was requested jointly by teachers and Professor Heffernan. The graph allows teachers to quickly gain a summary of their class once they log into ASSISTments. The key concept is that the graph is a measure of the number of late assignments that a student is associated with, which determines the student's velocity. Teachers will have the ability to quickly analyze the graph and identify the students that are not performing well or worsening. The feature was originally intended to be implemented by the team, but there remained a need for the design of the graph to be thoroughly fleshed out before implementation. Aspects such as display, filter options, explanation to new users required further thought. As such, the team recommends that additional feedback be gathered on this feature prior to final implementation in Vue.

Consolidated Class Page

Currently, teachers are only able to access their class page by clicking the link on google classroom. Unfortunately, there is added inconvenience to this method, so it is suggested that a page be included where teachers can view all of their classes on the new ASSISTments website and navigate to other specific classes. This will lessen confusion emanating from the need to go back and forth between URLs and create general, navigational flow for the teachers. The team recommends that this page be mocked up and implemented in a similar fashion to the other pages for ARRS.

REFERENCES

The ASSISTments Foundation. (2020). ASSISTments | Free Education Tool for Teachers & Students. Assistments. https://new.assistments.org/

Cromwell, G., King, J., Capobianco, M., & Kim, R. (2020). Automatic Reassessment and Relearning System in ASSISTments 2.0. Worcester Polytechnic Institute. https://digitalcommons.wpi.edu/mqp-all/7303/

Fain, Paul. (2019, January 16). Online learning fails to deliver, finds report aimed at discouraging. Inside Higher Ed. https://www.insidehighered.com/digital-learning/article/2019/01/16/online-learning-fails-deliver-finds-report-aimed-discouraging

Fielding, R. T., & Taylor, R. N. (2000). Architectural styles and the design of network-based software architectures (Vol. 7). Irvine: University of California, Irvine.

Inside Higher Ed. (2019, January 16). Online learning fails to deliver, finds report aimed at discouraging. https://www.insidehighered.com/digital-learning/article/2019/01/16/online-learning-fails-deliver-finds-report-aimed-discouraging

Organisation for Economic Co-operation and Development. (2020, September 24). Strengthening online learning when schools are closed: The role of families and teachers

in supporting students during the COVID-19 crisis. OECD. http://www.oecd.org/coronavirus/policy-responses/strengthening-online-learning-when-schools-are-closed-the-role-of-families-and-teachers-in-supporting-students-during-the-covid-19-crisis-c4ecba6c/

Schwartz, S. (2020, November 16). Survey: Teachers and Students Are Struggling With Online Learning. Education Week - Teaching Now. https://blogs.edweek.org/teachers/teaching_now/2020/11/survey_teachers_and_students_are_struggling_with_online_learning.html

You, E. (2020). Introduction - Vue.js. Vuejs. https://vuejs.org/v2/guide/

APPENDIX

APPENDIX A - Files Added and Modified before the redesign of ARRS

All files that the team added are on the ARRSdev branch of the TNGVue repository. Add files are new unless labelled otherwise.

src/api/assignmentRecord.api.ts

- Acts as api for simulated assignment data

src/api/classes.api.ts

- Acts as api for simulated class data

src/api/reassessments.api.ts

- Acts as api for simulated reassessment data

src/api/students.api.ts

- Acts as api for simulated student data

src/api/teachers.api.ts

- Acts as api for simulated teacher data

src/assets/arrs_assignment_card.svg

- An image of a RegularCard.vue component, for use in TimelineModal.vue

src/assets/arrs_timeline.svg

- An image of a student landing page's timeline component, for use in TimelineModal.vue

src/components/ProgressIndicator.vue

- A component that displays the sequence of initial assignment, reassessments, and relearnings for an ARRS assignment, for use in RegularCard.vue and on StudentAssignmentReportPage.vue
- Has the option of allowing the user to select between items on the indicator or not

src/components/RegularCard.vue

- A component that holds a ProgressIndicator.vue component

src/components/TimelineModal.vue

- A help button that shows a modal explaining the visuals behind the student landing page's timeline and ProgressIndicator.vue

src/domain/AssignmentRecord.ts

- An interface for how assignment record objects are structured

src/domain/Class.ts

- An interface for how class objects are structured

src/domain/Reassessment.ts

- An interface for how reassessment data objects are structured

src/domain/Relearning.ts

- An interface for how relearning data objects are structured

src/domain/Student.ts

- An interface for how student data objects are structured

src/domain/Teacher.ts

- An interface for how teacher data objects are structured

src/mirage/assignmentRecords.ts

- Simulated data of assignment records for the mirage server

src/mirage/classes.ts

- Simulated data of classes for the mirage server

src/mirage/reassessments.ts

- Simulated data of reassessments for the mirage server

src/mirage/relearnings.ts

- Simulated data of relearnings for the mirage server

src/mirage/server.ts (Modified)

- Added get functions for obtaining all of the different types of data implemented for this project, specifically functions to obtain all of a student's assignments, reassessments, and relearnings, and to obtain all of a classes' students

src/mirage/students.ts

- Simulated data of students for the mirage server

src/mirage/teachers.ts

- Simulated data of teachers for the mirage server

src/router/index.ts: (Modified)

- Added routes for all seven ARRS pages. These are the student landing page, student assignment report, ARRS settings page, teacher class report, teacher student report, teacher assignment report, and teacher student and assignment report.

src/ARRSStudent/StudentAssignmentReportPage.vue:

- Display a ProgressIndicator.vue component for the selected ARRS assignment, which is extended to allow selection on each part of the progress indicator

src/views/ARRSStudent/StudentLandingPage.vue:

- Display the selected assignment at the top of the page
- Added timeline for this page only
- Showed listing of assignments using RegularCard.vue components
- Included TimelineModal.vue component as a guide for how to interpret progress indicators and timeline

src/views/ARRSTeacher/ARRSSettingsPage.vue:

- Added display of ARRS settings options, and ability locally save current settings

src/views/ARRSTeacher/AssignmentReportPage.vue:

- Added placeholder page

src/views/ARRSTeacher/ClassReportPage.vue:

- Added table of students in a teacher's class showing number of late assignments, and able to filter the columns of the table

src/views/ARRSTeacher/StudentAndAssignmentReportPage.vue:

- Added placeholder page

src/views/ARRSTeacher/StudentReportPage.vue:

- Added placeholder page

src/views/ARRSTeacher/TemplatePage.vue:

- Acted as a template of what ARRS related pages would start from

APPENDIX B - Adaptive Homework Interview Script

Introduction

Hi. My name is _____ and this is _____. I'll be facilitating today's session and _____ will be taking notes as we go. Before we start,would it be OK if we record this session? The recording will not be shared with anyone outside of the ASSISTments team.

Thanks for coming in to talk with us. We're going to be starting this with showing you what you students are going to see as they interact with the adaptive homework feature. This way, hopefully we can answer some of your more general questions before getting to the important part which is you testing the teacher interaction with adaptive homework. Firstly though, we are using the adaptive homework feature as a way for students to opt in to try and answer a similar but not the same question to ones they get wrong on their homework.

Bring up student tutor page first:

- Fill in notes at the bottom of each section
- The tool we will be using for the rest of the day is called figma as we have not actually built this feature yet. Because of that there are some limitations with what you can interact with.
    - Demonstrate in the tutor page how to work Figma
    - Notice how when I click on the page it highlights some parts of the screen, these are the parts of the page you can interact with
- The first thing we are going to show you is how students would be able to use the adaptive homework feature.
- We start by seeing that this student answered a question incorrectly, and instead of the next problem button popping up we have 2 different options.
    - Try again path:
        - The student can click try again and a similar problem will appear, notice how after the student answers the question correctly the red x on problem one in the top left is now a green check.
        - For figma usage notice how in the bottom right corner you can restart the prototype at anytime, just by clicking the button restart. We're going to do this so we can see what happens when a student asks a question instead
    - Ask a question:
        - We didn't want to just make it so that students had to try again even if they really did not understand the material as that can be very discouraging, but we also wanted to encourage students to be actively trying in their homework or atleast actively thinking.

        - Dialogue box pops up and the student asks their question. Notice how after selecting next problem the student still has the red x in the top left corner.
        - The student hopefully may also have that a-ha moment while asking a question and decide to try it again.
- Teacher notes
    -

Bring up Student Report Page:

- Here we are going to look at a student who we will later be calling “Lost” Luke
- In the screen we can see the 2nd hand column with the total credit they have earned for each problem and their assignment grade. In the next we see the class average and finally there answer and grade in the fourth column.
- The fifth column shows their answer for the redo question. Notice how some of the rows are blank in this column. This is because currently we can only grade questions that have definitive answers. We can see that Luke here did not do so hot on some of the initial questions or the redo ones.
- The last column shows which problems they can select to redo and we will generate another assignment for them with similar problems to the ones they selected.
- In this case lets imagine Luke came back to class and his teacher noticed that a lot of people were struggling with those problems, or they answered some of the questions students asked and now Luke thinks that he understands whats going on. He goes home and checks the two problems he answered wrong and tries them again
- We can see we added one column to the table for the new round of redo problems and what he learned in class did help. We can also notice that in the second column for their credit they have now earned full marks for the problem.
- Do you have any questions other questions or comments about this page or the adaptive homework feature before we move on?

- Teacher notes

Send teacher Assign page link:

- The first place every teacher needs to start is in assigning the homework, so thats where we would like you to start. If you click the link you posted in the chat you should be brought to the prototype and a familiar screen if you have worked with 2.0.
- To do that, we’re going to have you carry out certain tasks in some mockups that we’ve built of the functionality. When you carry out these tasks do your best, but don’t be concerned with the results. We’re testing the product not you. The product is still mostly in the form of a mockup, and only a very small subset of the interface actually works. Remember if you don’t know what you can do, just click the page and options your

options will appear. Also, if everything does not fit in the page you can click and drag the page around to see everything.

- As you attempt to complete each task, we'd like you to do something we call "thinking out loud." Essentially, this involves verbalizing whatever is on your mind as you're performing the tasks. By hearing what you're thinking, it will enable us to not only understand what is and isn't easy to use with our interface, but why it is or isn't easy to use.

- Would you like me to demonstrate what I mean by thinking out loud?
- At this point, I'd like you to share your screen by clicking the green "Share Screen" button in Zoom. Before I do that, though, please make sure that there is nothing personal or proprietary visible on your screen.
- **Awesome, So for this page we would like you to create an assignment where only the first problem gives the student the option to redo the problem. And to explore what options they will have for the redo.**
- Teacher notes
    -

Send teacher Report Page link

- At this point, I'd like you to share your screen by clicking the green "Share Screen" button in Zoom. Before I do that, though, please make sure that there is nothing personal or proprietary visible on your screen.
- We tried to add the adaptive practice to your report screen with as little changes to the screen as possible
- Would you please look at the report and see if you could give a grade for Luke, Liam, and Mattie. As you are doing please remember to think out loud and ask as many questions as you like and comments on what you like and dislike when trying to do this.
- Questions if not mentioned:
    - Was anything unclear to you? Was there anything missing

- Teacher notes

Send teacher Report Page link

- We tried to add the adaptive practice to your report screen with as little changes to the screen as possible
- At this point, I'd like you to share your screen by clicking the green "Share Screen" button in Zoom. Before I do that, though, please make sure that there is nothing personal or proprietary visible on your screen.
- Would you please look at the report and see if you could give a grade for Luke, Liam, and Mattie. As you are doing please remember to think out loud and ask as many questions as you like and comments on what you like and dislike when trying to do this.
- Questions if not mentioned:

    - Base questions
    - Was anything unclear to you? Was there anything missing

- Teacher notes
    -

Send settings page link:

- At this point, I'd like you to share your screen by clicking the green "Share Screen" button in Zoom. Before I do that, though, please make sure that there is nothing personal or proprietary visible on your screen.
- **For this page we would like you to try and configure your personal settings for adaptive homework.** Please think out loud and let us know how you would like your settings configured for each one and if there are any settings you think may be missing. Also if any of the settings are confusing definitely let us know.
- Base questions
- Teacher notes
    -

Send summary report link (if time allows):

- **Task: please view your daily report and tell me whats happening in the class room.**
- Again please think out loud.
- Questions:
    - Is there too much going on or is there too little going on?
    - Is there information you wished we would show and that we aren't
    - Is anything unclear

- Teacher notes
    -

Questions for Each New Page or UX Step

Here are the questions we'll want to take them through after they complete each successful step.

To the best of your ability, could you describe what you're seeing here?

What is least clear?

What do you do on this screen to move forward in the task?

Why is that your choice?

After doing that, what would you expect to see?

To actually move forward in this task what you need to do is this: _____.

Explain what that action accomplishes.

Does that action make sense?

If not

Can you explain what is unclear or confusing about that action?

Have them do the action.

APPENDIX C - Code for the Main Pages that were developed before the redesign of ARRS

ARRSSettingsPage.vue

```
template>
  <div>
    <h1>
      <td style="width: 33%; padding: 0px 25px">
        ARRS Settings
        <v-btn class="ma-2" color="primary">
          <v-icon dark> mdi-wrench </v-icon>
        </v-btn>
      </td>
    </h1>
    <v-data-table
      :headers="fields"
      :items="items"
      :hide-default-footer="true"
      class="elevation-2"
    >
      <template v-slot:[`item.status`]="{ item }">
        <v-switch
          v-model="item.status"
          :label="`${getStatus(item)}`"
        ></v-switch>
      </template>
    </v-data-table>
    <v-divider></v-divider>
    <table style="width: 100%">
```

```
<tr>
  <td style="width: 33%; padding: 0px 25px">
    <v-text-field
      label="New Assignments will be due how many days after assigning?"
      type="number"
      min="0"
      v-model="data.aDay"
    ></v-text-field>
  </td>
  <td style="width: 33%; padding: 25px">
    <v-select
      :items="hours"
      v-model="data.hour"
      label="Select the Time:"
    ></v-select>
  </td>
  <td style="width: 33%; padding: 25px">
    <v-container fluid>
      <v-select
        v-model="grades.value"
        :items="grades.items"
        label="Select the Class"
        multiple
      >
        <template v-slot:selection="{ item, index }">
          <v-chip v-if="index === 0">
            <span>{{ item }}</span>
```

```
            </v-chip>
            <span v-if="index === 1" class="grey--text caption">
              (+{{ grades.value.length - 1 }} others)
            </span>
          </template>
        </v-select>
      </v-container>
    </td>
  </tr>
</table>
<v-divider></v-divider>
<p v-if="questionSTatus()" style="padding: 25px">
  Reasses an assigned skillbuilder
  <input
    class="input-ARRS"
    label="Reason ?"
    type="number"
    min="0"
    v-model="settings.reassesTime"
  />
  times
  <br />
  <br />
  1st Reassessment is
  <input
    class="input-ARRS"
    label="Reason ?"
```

```
  type="number"
  min="0"
  v-model="settings.reasses1.days"
/>
days after the assigned skillbuilder is
<select class="select-ARRS" v-model="settings.reasses1.status">
  <option value="Completed">Completed.</option>
  <option value="Progress">in .</option>
</select>
<br />
<br />
2nd Reassessment is
<input
  class="input-ARRS"
  label="Reason ?"
  type="number"
  min="0"
  v-model="settings.reasses2.days"
/>
days after previous reassessment.
<br />
<br />
3rd Reassessment is
<input
  class="input-ARRS"
  label="Reason ?"
  type="number"
```

```
    min="0"
    v-model="settings.reasses3.days"
  />
  days after previous reassessment.
  <br />
  <br />
  4th Reassessment is
  <input
    class="input-ARRS"
    label="Reason ?"
    type="number"
    min="0"
    v-model="settings.reasses4.days"
  />
  days after previous reassessment.
  <br />
  <br />
  5th Reassessment is
  <input
    class="input-ARRS"
    label="Reason ?"
    type="number"
    min="0"
    v-model="settings.reasses5.days"
  />
  days after previous reassessment.
</p>
```

```
    <p>
      <v-btn
        :loading="loading3"
        :disabled="loading3"
        color="blue-grey"
        class="ma-2 white--text"
        style="margin-left: 25px !important"
        @click="loader = 'loading3'"
      >
        Cancel
        <v-icon right dark> mdi-cancel </v-icon>
      </v-btn>
      <v-btn class="ma-2" color="primary" dark @click="saveItems()">
        Save
        <v-icon dark right> mdi-checkbox-marked-circle </v-icon>
      </v-btn>
    </p>
    <!--</b-form>-->
  </div>
</template>
<script lang="ts">
import { Component, Vue } from 'vue-property-decorator';
import Vuetify from 'vuetify';

// Install BootstrapVue
Vue.use(Vuetify);
import 'vuetify/dist/vuetify.min.css';
```

```
// import 'bootstrap-vue/dist/bootstrap-vue.css';
// It is likely that these will be needed for every ARRS related page:
//import { getStudents, getStudent } from '@/api/students.api';
//import { Student } from '@/domain/Student';

//<v-row>
// <h3>{{ title }}</h3>
// <!--This settings button could be added to the navbar if desired-->
//<v-btn
//  v-bind:href="'/arrsteacher/' + teacherId + '/arrssettings'"
// class="ma-2"
// color="blue"
// dark
//<
//<v-icon dark> mdi-wrench </v-icon>
// </v-btn>
//</v-row>

@Component
export default class ARRSSettingsPage extends Vue {
  // Elements on the page
  title = 'ARRS Settings';
  items = [];
  fields = [
    { value: 'question', text: 'Feature' },
    { value: 'status', text: 'Status' },
  ];
```

```
public data = {
  aDay: '1',
  aNumber: '1',
  apDay: 1,
  hour: '12:00 AM',
};
public settings = {
  reassesTime: 0,
  reasses1: {
    days: 0,
    status: 'Completed',
  },
  reasses2: {
    days: 0,
  },
  reasses3: {
    days: 0,
  },
  reasses4: {
    days: 0,
  },
  reasses5: {
    days: 0,
  },
};
grades = {
```

```
  items: ['Grade 5', 'Grade 6', 'Grade 7', 'Grade 8'],
  value: [],
};
hours = [
  '12:00 AM',
  '01:00 AM',
  '02:00 AM',
  '03:00 AM',
  '04:00 AM',
  '05:00 AM',
  '06:00 AM',
  '07:00 AM',
  '08:00 AM',
  '09:00 AM',
  '10:00 AM',
  '11:00 AM',
  '12:00 PM',
  '01:00 PM',
  '02:00 PM',
  '03:00 PM',
  '04:00 PM',
  '05:00 PM',
  '06:00 PM',
  '07:00 PM',
  '08:00 PM',
  '09:00 PM',
  '10:00 PM',
```

```
  '11:00 PM',
];
init(): void {
  // Run on page initialization
}
created(): void {
  this.init();
  this.getListData();
}
// Get id from url of teacher to find the right ARRS settings
get teacherId(): string {
  return this.$route.params.teacherId;
}

public getStatus(data: unknown): string {
  const dt = data;
  if (dt['status']) {
    return 'Yes';
  } else {
    return 'No';
  }
}

public getListData(): void {
  this.items = [
    {
      id: 'que1',
```

```
    question: 'Use Assistments reccomended ARRS settings?',
    status: true,
  },
  {
    id: 'que2',
    question: 'Apply these features to all Assignments?',
    status: false,
  },
  {
    id: 'que3',
    question: 'Limit number of Assignments students will recieve in a day?',
    status: false,
  },
  {
    id: 'que4',
    question:
      'Require students to complete ARRS Assignments prior to accessing other assignments?',
    status: false,
  },
  {
    id: 'que5',
    question:
      'Would you like to enable the use of the Student Landing Page?',
    status: false,
  },
 ];
}
```

```
  public questionSTatus() {
    const found = this.items.find(
      (element) =>
        element.question === 'Use Assistments reccomended ARRS settings?'
    );
    return !found.status;
  }

  public saveItems() {
    console.log('data', this.data);
    console.log('grades', this.grades);
    if (this.questionSTatus()) {
      console.log('settings', this.settings);
    }
    console.log(this.items);
  }
}
</script>
<style>
.input-ARRS {
  display: inline;
  width: 50px;
  overflow-x: auto;
  height: 30px;
  padding: 3px;
  border: 3px solid #0661c2;
```

```
}

.select-ARRS {
  display: inline;
  width: auto;
  overflow-x: auto;
  height: 35px;
  padding: 3px;
  border: 3px solid #0661c2;
}
</style>
```

ClassReportPage.vue

```
<template>
  <div>
    <h3>{{ title }}</h3>
    <div>
      <v-data-table
        :headers="fields"
        :items="studentRow"
        :hide-default-footer="true"
        class="elevation-2"
      ></v-data-table>
    </div>
  </div>
</template>
```

```
<script lang="ts">
import { Component, Vue } from 'vue-property-decorator';
import { Student } from '@/domain/Student';
import { Class } from '@/domain/Class';
import { getClass } from '@/api/classes.api';

@Component
export default class ClassReportPage extends Vue {
  // Elements on the page
  // Will dynamically update titles with relevant page info
  title = 'Class Report Page';
  studentArray: Student[] = [];
  selectedClass: Class = {
    id: 0,
    name: 'placeholder',
    teacherID: 0,
    listOfStudents: [],
  };
  fields = [
    { text: 'Student', value: 'name' },
    { text: 'Total Late Originals', value: 'lateOriginals' },
    { text: 'Late Reassessments', value: 'lateReassessments' },
    { text: 'Late Relearnings', value: 'lateRelearnings' },
    { text: 'Total Lates', value: 'totalLate' },
  ];

  studentRow = [];
```

```
init(): void {
  getClass(1).then((response) => {
    this.selectedClass = response;
    this.studentArray = response.listOfStudents;
    let tempArray = [];

    for (let student of this.studentArray) {
      let allLates = this.getLates(student);
      // Push object that holds items of data table
      tempArray.push({
        name: student.name,
        lateOriginals: String(allLates[0]),
        lateReassessments: String(allLates[1]),
        lateRelearnings: String(allLates[2]),
        totalLate: String(allLates[3]),
      });
    }

    this.studentRow = tempArray;
    this.classAssignments = tempArray;
  });
}
created(): void {
  this.init();
}
// Get id from url on class and teacher to find the right settings
get teacherId(): string {
```

```
  return this.$route.params.teacherId;
}
get classId(): string {
  return this.$route.params.classId;
}
/*
  Takes in a student and goes through all types of assignments to
  get the number of lates ones for each type

  @return: an array of numberscontaining lateCount, lateReassessmentCount,
  lateRelearningCount, lateTotal correspondingly.
*/
getLates(student: Student): number[] {
  let record = student.assignmentRecords;
  let lateCount = 0; // Represents the late regular assignments
  let lateReassessmentCount = 0; // Represents the late reassessments
  let lateRelearningCount = 0; // Represents the late relearnings
  let lateTotal = 0; // Total of all the other late variables
  // Parse through each Regular Assignment
  for (let assignment of record) {
    if (assignment.arrs) {
      if (assignment.completionDate == null) {
        // Check if due date has still yet to arrive
        if (new Date().getTime() > Date.parse(String(assignment.dueDate))) {
          lateCount += 1;
          lateTotal += 1;
        }
```

```
} else if (
  Date.parse(String(assignment.completionDate)) >
  Date.parse(String(assignment.dueDate))
) {
  // Add to the count
  lateCount += 1;
  lateTotal += 1;
}
//Check if Reassessments exist
if (assignment.reassessments.length > 0) {
  // Parse through each Reassessment
  for (let reassess of assignment.reassessments) {
    if (reassess.completionDate == null) {
      if (new Date().getTime() > Date.parse(String(reassess.dueDate))) {
        lateCount += 1;
        lateTotal += 1;
      }
    } else if (
      Date.parse(String(reassess.completionDate)) >
      Date.parse(String(reassess.dueDate))
    ) {
      // console.log(reassess.completionDate);
      // console.log(reassess.dueDate);
      lateReassessmentCount += 1;
      lateTotal += 1;
    }
```

```
        if (reassess.relearnings.length > 0) {
          // Parse through each relearning
          for (let relearn of reassess.relearnings) {
            if (relearn.completionDate == null) {
              if (
                new Date().getTime() > Date.parse(String(relearn.dueDate))
              ) {
                lateCount += 1;
                lateTotal += 1;
              }
            } else if (
              Date.parse(String(relearn.completionDate)) >
              Date.parse(String(relearn.dueDate))
            ) {
              lateRelearningCount += 1;
              lateTotal += 1;
            }
          }
        }
      }
    }
  }
}
let lateTotals = [
  lateCount,
  lateReassessmentCount,
  lateRelearningCount,
```

```
      lateTotal,
    ];
    return lateTotals;
  }
}
</script>
```

StudentAssignmentReportPage.vue

```
<template>
  <div>
    <h3>{{ title }} {{ selectedAssignmentName }}</h3>
    <!-- Help button and modal dialog -->
    <TimelineModal :isStudent="true"></TimelineModal>
    <!-- v-on:click="adjustCurrent(assignment.id, 'initial-assignment')" -->
    <v-card class="mx-auto card-spacing" max-width="1000" outlined>
      {{ selectedAssignmentName }}
      <div
        v-for="assignment in selectedStudent.assignmentRecords"
        :key="assignment.name"
      >
        <progress-indicator
          v-if="assignment.arrs && assignment.id == selectedAssignmentId"
          @clicked="adjustCurrent"
          :key="assignment.id"
          :assignment="assignment"
          :arrs-levels="arrsLevels"
          :selectable="true"
```

```
        >
        </progress-indicator>
      </div>
    </v-card>
    <div>{{ currentContents(currentId, currentType) }}</div>
  </div>
</template>

<script lang="ts">
import { Component, Vue } from 'vue-property-decorator';
import { getStudent } from '@/api/students.api';
import { Student } from '@/domain/Student';
import { AssignmentRecord } from '@/domain/AssignmentRecord';
import TimelineModal from '@/components/StudentPage/TimelineModal.vue';
import ProgressIndicator from '@/components/StudentPage/ProgressIndicator.vue';

@Component({
  components: {
    TimelineModal,
    ProgressIndicator,
  },
})
export default class StudentAssignmentReportPage extends Vue {
  // Elements on the page
  // Will dynamically update titles with relevant page info
  arrsLevels = 3;
  title = '    Assignment Report Page - ';
```

```
selectedAssignmentName = '';
selectedAssignmentId = '';
selectedStudent: Student = {
  id: 0,
  name: 'placeholder',
  assignmentRecords: [],
};
currentId = -1;
currentType = 'initial-assignment';

init(): void {
  // Run on page initialization
  getStudent(+this.studentId).then((response) => {
    let resp = response.assignmentRecords.sort(
      (a, b) => new Date(a.dueDate).getTime() - new Date(b.dueDate).getTime()
    );
    response.assignmentRecords = resp;
    this.selectedStudent = response;
    this.selectedAssignmentId = this.assignmentId;
    this.selectedAssignmentName = this.selectedName(
      this.selectedAssignmentId,
      resp
    );
    this.currentId = Number(this.selectedAssignmentId);
  });
}
created(): void {
```

```
  this.init();
}
// Get the name of the selected assignment
selectedName(selectedAssignmentId: string, resp: AssignmentRecord[]): string {
  let name = 'No ARRS Assignment Chosen';
  // Check for the name of the selected assignment:
  resp.forEach(function (record) {
    if (record.id.toString() == selectedAssignmentId && record.arrs) {
      name = record.name;
    }
  });
  return name;
}

// Change which assignment/reassessment/relearning is selected currently on the page
adjustCurrent(assignmentId: number, type: string): void {
  this.currentId = assignmentId;
  this.currentType = type;
}

// Show the contents of the page for the currently selected assignment/reassessment/relearning
currentContents(assignmentId: number, type: string): string {
  console.log(this.selectedAssignmentName === 'No ARRS Assignment Chosen');
  if (this.selectedAssignmentName === 'No ARRS Assignment Chosen') {
    return 'No ARRS assignment selected';
  }
  return 'Placeholder for actual content for ' + type + ': ' + assignmentId;
```

```
  }

  get assignmentId(): string {
    return this.$route.params.assignmentId;
  }
  get studentId(): string {
    return this.$route.params.studentId;
  }
}
</script>
```

StudentLandingPage.vue

```
<template>
  <div>
    <h3>{{ title }} {{ selectedAssignmentName }}</h3>
    <!-- Help button and modal dialog -->
    <TimelineModal :isStudent="true"></TimelineModal>
    <!-- Selected Assignment-->
    <div class="container-card">
      <v-card class="mx-auto" max-width="1000" outlined>
        <h3 class="card-header">Your Selected Assignment:</h3>
        <div
          v-for="assignment in selectedStudent.assignmentRecords"
          :key="assignment.name"
        >
          <div v-if="assignment.id == selectedAssignment">
            <RegularCard :assignment="assignment" :arrs-levels="3">
            </RegularCard>
```

```
      </div>
    </div>
  </v-card>
</div>
<!-- Timeline -->
<div class="container-card">
  <v-card class="mx-auto" max-width="1000" outlined>
    <v-list-item three-line>
      <v-list-item-content>
        <h3 class="card-header">Assignment Timeline:</h3>
        <template id="progress-indicator-template">
          <div class="indicator">
            <div class="step-indicators">
              <!-- Overdue Assignments -->
              <div class="overdue">
                Late:
                <div v-for="assignment in overdue" :key="assignment.id">
                  <div
                    v-if="assignment.id != selectedAssignment"
                    class="step-indicator initial-assignment"
                    :class="'late'"
                  ></div>
                  <div
                    v-else-if="assignment.completionDate"
                    class="step-indicator initial-assignment"
                    :class="'selected complete'"
                  ></div>
```

```
  <div
    v-else
    class="step-indicator initial-assignment"
    :class="'selected'"
  ></div>
</div>
<!-- Overdue Reassessments -->
<div
  v-for="reassessment in overdueReassessmentArray"
  :key="reassessment.id"
>
  <div
    v-if="
      reassessment.recordID != selectedAssignment &&
      reassessment.releaseDate != null &&
      !reassessment.completionDate
    "
    class="step-indicator reassessment"
    :class="'late'"
  ></div>
  <div
    v-else-if="
      reassessment.recordID == selectedAssignment &&
      reassessment.releaseDate != null &&
      !reassessment.completionDate
    "
    class="step-indicator reassessment"
```

```
    :class="'selected'"
  ></div>
  <div
    v-else-if="
      reassessment.recordID == selectedAssignment &&
      reassessment.releaseDate != null &&
      reassessment.completionDate
    "
    class="step-indicator reassessment"
    :class="'selected complete'"
  ></div>
</div>
<!-- Overdue Relearnings -->
<div
  v-for="assignment in selectedStudent.assignmentRecords"
  :key="assignment.name"
>
  <div v-if="assignment.arrs">
    <div
      v-for="reassessment in assignment.reassessments"
      :key="reassessment.id"
    >
      <div
        v-if="
          reassessment.relearnings != undefined &&
          reassessment.relearnings.length > 0
        "
```

```
>
  <div
    v-if="
      reassessment.recordID == selectedAssignment &&
      Date.parse(
        String(
          reassessment.relearnings[
            reassessment.relearnings.length - 1
          ].dueDate
        )
      ) < Date.parse(String(new Date())) &&
      !reassessment.relearnings[
        reassessment.relearnings.length - 1
      ].completionDate
    "
    class="step-indicator relearning"
    :class="'selected'"
  ></div>
  <div
    v-else-if="
      !(
        reassessment.recordID == selectedAssignment
      ) &&
      Date.parse(
        String(
          reassessment.relearnings[
            reassessment.relearnings.length - 1
```

```
                ].dueDate
              )
            ) < Date.parse(String(new Date())) &&
            !reassessment.relearnings[
              reassessment.relearnings.length - 1
            ].completionDate
          "
          class="step-indicator relearning"
          :class="'late'"
        ></div>
      </div>
    </div>
  </div>
</div>
<!-- Assignments for each day in the next 2 weeks -->
<div class="day" v-for="index in days" :key="index">
  {{ watchedDates[index - 1].toDateString() }}
  <div
    v-for="assignment in selectedStudent.assignmentRecords"
    :key="assignment.id"
  >
    <div
      v-if="
        assignment.id != selectedAssignment &&
        watchedDates[index - 1].toDateString() ==
          new Date(assignment.dueDate).toDateString() &&
```

```
    assignment.releaseDate != null
  "
  class="step-indicator initial-assignment"
  :class="'released'"
></div>
<div
  v-else-if="
    assignment.id == selectedAssignment &&
    new Date(assignment.dueDate).toDateString() ==
      watchedDates[index - 1].toDateString() &&
    assignment.releaseDate != null &&
    assignment.compltionDate
  "
  class="step-indicator initial-assignment"
  :class="'selected complete'"
></div>
<div
  v-else-if="
    assignment.id == selectedAssignment &&
    new Date(assignment.dueDate).toDateString() ==
      watchedDates[index - 1].toDateString() &&
    assignment.releaseDate != null
  "
  class="step-indicator initial-assignment"
  :class="'selected'"
></div>
<!-- Reassessments -->
```

```
<div v-if="assignment.arrs">
  <div
    v-for="reassessment in assignment.reassessments"
    :key="reassessment.id"
  >
    <div
      v-if="
        reassessment.recordID != selectedAssignment &&
        watchedDates[index - 1].toDateString() ==
          new Date(reassessment.dueDate).toDateString() &&
        reassessment.releaseDate != null &&
        !reassessment.completionDate
      "
      class="step-indicator reassessment"
      :class="'released'"
    ></div>
    <div
      v-else-if="
        reassessment.recordID == selectedAssignment &&
        watchedDates[index - 1].toDateString() ==
          new Date(reassessment.dueDate).toDateString() &&
        reassessment.releaseDate != null &&
        !reassessment.completionDate
      "
      class="step-indicator reassessment"
      :class="'selected'"
    ></div>
```

```
    <div
      v-else-if="
        reassessment.recordID == selectedAssignment &&
        watchedDates[index - 1].toDateString() ==
          new Date(reassessment.dueDate).toDateString() &&
        reassessment.releaseDate != null &&
        reassessment.completionDate
      "
      class="step-indicator reassessment"
      :class="'selected complete'"
    ></div>
  </div>
</div>
<!-- Relearnings -->
<div v-if="assignment.arrs">
  <div
    v-for="reassessment in assignment.reassessments"
    :key="reassessment.id"
  >
    <div
      v-if="
        reassessment.relearnings != undefined &&
        reassessment.relearnings.length > 0
      "
    >
      <div
        v-if="
```

```
    reassessment.recordID == selectedAssignment &&
    new Date(
      reassessment.relearnings[
        reassessment.relearnings.length - 1
      ].dueDate
    ).toDateString() ==
      watchedDates[index - 1].toDateString() &&
    !reassessment.relearnings[
      reassessment.relearnings.length - 1
    ].completionDate
  "
  class="step-indicator relearning"
  :class="'selected'"
></div>
<div
  v-else-if="
    !(
      reassessment.recordID == selectedAssignment
    ) &&
    new Date(
      reassessment.relearnings[
        reassessment.relearnings.length - 1
      ].dueDate
    ).toDateString() ==
      watchedDates[index - 1].toDateString() &&
    !reassessment.relearnings[
      reassessment.relearnings.length - 1
```

```
                ].completionDate
              "
              class="step-indicator relearning"
              :class="'released'"
            ></div>
          </div>
        </div>
      </div>
    </div>
  </template>
 </v-list-item-content>
 </v-list-item>
</v-card>
</div>

<v-switch
 v-model="showCompleted"
 label="Show Completed Assignments"
></v-switch>
<div>
 <v-tabs>
  <v-tab title="Skillbuilders" active>Skillbuilders</v-tab>
  <v-tab-item>
   <template v-for="assignment in selectedStudent.assignmentRecords">
```

```
    <div
      v-if="
        assignment.arrs &&
        assignment.id != selectedAssignment &&
        (!assignment.reassessments.length > 0 ||
          !assignment.reassessments[assignment.reassessments.length - 1]
            .completionDate ||
          showCompleted)
      "
      :key="assignment.id"
    >
      <RegularCard
        :key="assignment.name"
        :assignment="assignment"
        :arrs-levels="3"
      >
      </RegularCard>
    </div>
  </template>
</v-tab-item>

<v-tab title="Regular Assignments">Regular Assignments</v-tab>
<v-tab-item>
  <template v-for="assignment in selectedStudent.assignmentRecords">
    <div
      v-if="
        (showCompleted || !assignment.completionDate) &&
```

```
            !assignment.arrs &&
            assignment.id != selectedAssignment
          "
          :key="assignment.id"
        >
          <RegularCard
            :key="assignment.id"
            :assignment="assignment"
            :arrs-levels="3"
          >
          </RegularCard>
        </div>
      </template>
    </v-tab-item>
  </v-tabs>
 </div>
</div>
</template>

<script lang="ts">
import { Component, Vue } from 'vue-property-decorator';
import RegularCard from '@/components/StudentPage/RegularCard.vue';
import { getStudent } from '@/api/students.api';
import { Student } from '@/domain/Student';
import { AssignmentRecord } from '@/domain/AssignmentRecord';
import { Reassessment } from '@/domain/Reassessment';
import ProgressIndicator from '@/components/StudentPage/ProgressIndicator.vue';
```

```
import TimelineModal from '@/components/StudentPage/TimelineModal.vue';
//import { BootstrapVue } from 'bootstrap-vue';

// Install BootstrapVue
//Vue.use(BootstrapVue);
//import 'bootstrap/dist/css/bootstrap.css';
//import 'bootstrap-vue/dist/bootstrap-vue.css';

@Component({
  components: {
    ProgressIndicator,
    RegularCard,
    TimelineModal,
  },
})
export default class StudentLandingPage extends Vue {
  // Elements on the page
  title = 'Landing Page - ';
  selectedAssignmentName = '';
  studentArray: Student[] = [];
  selectedStudent: Student = {
    id: 0,
    name: 'placeholder',
    assignmentRecords: [],
  };
  selectedAssignment = '';
  latestDate = new Date();
```

```
watchedDates: Date[] = [];
overdue: AssignmentRecord[] = [];
overdueReassessmentArray: Reassessment[] = [];
days = 15;
showCompleted = false;
init(): void {
  // Query for the student specified in url
  getStudent(+this.studentId).then((response) => {
    let resp = response.assignmentRecords.sort(
      (a, b) => new Date(a.dueDate).getTime() - new Date(b.dueDate).getTime()
    );
    response.assignmentRecords = resp;
    this.selectedStudent = response;
    this.selectedAssignment = this.assignmentId;
    // Process assignment records for this student
    this.selectedStudent.assignmentRecords;
    this.selectedAssignmentName = this.selectedName(
      this.selectedAssignment,
      resp
    );
    this.overdue = this.overdueAssignments(
      this.selectedAssignment,
      resp,
      this.latestDate
    );
    this.overdueReassessmentArray = this.overdueReassessments(
      this.selectedAssignment,
```

```
      resp,
      this.latestDate
    );
  });
}
created(): void {
  this.watchedDates = this.getDates();
  this.init();
}
get assignmentId(): string {
  return this.$route.params.assignmentId;
}
get studentId(): string {
  return this.$route.params.studentId;
}
// Get the name of the selected assignment
selectedName(selectedAssignment: string, resp: AssignmentRecord[]): string {
  let name = 'No Assignment Chosen';
  // Check for the name of the selected assignment:
  resp.forEach(function (record) {
    if (record.id.toString() == selectedAssignment) {
      name = record.name;
    }
  });
  return name;
}
// Filter out past completed assignments and collect all unfinished overdue assignments
```

```
// With the exception of the selected assignment, which is always kept
overdueAssignments(
  selectedAssignment: string,
  resp: AssignmentRecord[],
  now: Date
): AssignmentRecord[] {
  let overdue: AssignmentRecord[] = [];
  resp.forEach(function (record) {
    // For some reason, record.dueDate is a date object but
    // record.dueDate.getTime() returns undefined
    // So interpret it as a string then parse it instead
    // (This seems like the wrong way to do this but it works)
    if (Date.parse(String(record.dueDate)) < Date.parse(String(now))) {
      if (
        record.completionDate == null ||
        record.id.toString() == selectedAssignment
      ) {
        // Record is overdue
        overdue.push(record);
      }
    }
  });
  return overdue;
}
// Get overdue reassessments
// Like above, keep reassessments if they belong to the selected assignment
overdueReassessments(
```

```
  selectedAssignment: string,
  resp: AssignmentRecord[],
  now: Date
): Reassessment[] {
  let overdue: Reassessment[] = [];
  resp.forEach(function (record) {
    if (record.arrs && record.reassessments != undefined) {
      record.reassessments.forEach(function (reassessment) {
        if (
          Date.parse(String(reassessment.dueDate)) < Date.parse(String(now))
        ) {
          if (
            reassessment.completionDate == null ||
            reassessment.recordID.toString() == selectedAssignment
          ) {
            // Record is overdue
            overdue.push(reassessment);
          }
        }
      });
    }
  });
  return overdue;
}
// Get the next two weeks worth of time for the timeline
getDates(): Date[] {
  let nextWeeks = [];
```

```
  nextWeeks.push(new Date(this.latestDate.getTime()));
  for (let i = 1; i < 15; i++) {
    nextWeeks.push(
      new Date(this.latestDate.getTime() + i * 24 * 60 * 60 * 1000)
    );
  }
  return nextWeeks;
}
// Get released assignments due on the same day as the given date
// Don't show nonreleased assignments on the timeline
getAssignments(date: Date): AssignmentRecord[] {
  let records: AssignmentRecord[] = [];
  // This is not an optimal solution (checks all assignments), perhaps speed it up somehow?
  for (let i = 0; i < this.selectedStudent.assignmentRecords.length; i++) {
    if (
      date.toDateString() ==
        new Date(
          this.selectedStudent.assignmentRecords[i].dueDate
        ).toDateString() &&
      Date.parse(
        String(this.selectedStudent.assignmentRecords[i].releaseDate)
      ) <= Date.parse(String(this.latestDate))
    ) {
      records.push(this.selectedStudent.assignmentRecords[i]);
    }
  }
  return records;
```

```
  }
}
</script>

<style lang="scss" scoped>
.card-header {
  text-align: center;
}

.container-card {
  margin-bottom: 10px;
}

.indicator {
  display: block;
  padding: 20px 0 20px 0;
}

.step-indicators {
  position: relative;
  display: flex;
  justify-content: space-between;
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Helvetica, Arial,
    sans-serif, 'Apple Color Emoji', 'Segoe UI Emoji', 'Segoe UI Symbol';
}

.overdue {
```

```
  font-size: 12px;
  z-index: 2;
}

.day {
  font-size: 10px;
}

.step-indicator {
  border: 2px solid;
  /*border-radius: 50%;*/
  height: 32px;
  width: 32px;
  display: flex;
  justify-content: center;
  align-items: center;
  font-weight: 700;
  z-index: 2;
  background-color: white;
  box-sizing: border-box;
}

.initial-assignment {
  border-radius: 0;
  margin-top: 15px;
}
```

```
.reassessment {
  border-radius: 50%;
  margin-top: 15px;
}

.relearning {
  height: 25px;
  width: 25px;
  margin: 0.45em;
  border-radius: 0;
  transform: rotate(45deg);
  margin-top: 15px;
}

.selected {
  border-color: orange;
}

.not-released {
  border-color: gray;
}

.released {
  border-color: lawngreen;
}

.complete {
```

```
  background-color: lawngreen;
}

.on-time {
  border-color: lawngreen;
}

.late {
  border-color: red;
}

.step-indicators-line {
  position: absolute;
  height: 2px;
  top: 50%;
  left: 24px;
  right: 24px;
  transform: translateY(-50%);
  z-index: 1;
  background: rgb(223, 231, 239);
}

.modal-container {
  float: left;
  margin-right: 10px;
  margin-left: 10px;
}
```

```
.test {
  margin-top: 10px;
}
</style>
```