Agile/Scrum in Education

Lily Romano
lilyromano@gatech.edu

Abstract—Agile and Scrum methodologies are becoming increasingly common in the workplace. In order to prepare computer science students for the workforce, Agile concepts can be introduced into the classroom either in a Software Development course or in a capstone course. However current Agile tools do not lend themselves to use in academics. A new tool is necessary to bridge the gap between academics and industry. This tool provides an introduction to Agile, Scrum and other software development methodologies but is targeted towards short term projects. The tool will increase team member responsibility to reduce the friction seen in group projects. The tool will also support instructors managing several disparate concurrent projects.

1 INTRODUCTION

According to the 14th Annual State of Agile Report, "95% of respondents report their organizations practice Agile development methods" (*State of Agile Survey*, 2020). The problem is how to help students benefit from learning industry software development practices but at a scale that is practical for the classroom (Kudikyala & Dulhare, 2015; Machado & Tao, 2007). Current market Agile/Scrum tools can be overly complicated for students as they are geared towards businesses. Raith et al. compiled a list of features with benefits and drawbacks from a survey of industry Agile coaches (Raith et al., 2017). Most of these features are not included as topics of teaching in studies of applying Agile methodologies in the classroom (Bass et al., 2016; Hans, 2017; Koster & College, 2006; Kudikyala & Dulhare, 2015; Rico & Sayani, 2009; Rover et al., 2014).

Students have a limited amount of time to devote to a specific course (Koster & College, 2006). Yet, today's tools feel like a full-time job to learn and maintain which doesn't leave time to focus on the actual project to fulfill the course's learning objective (Krehbiel et al., 2017).

It is important to note that a frictionless environment is not the best environment for learning as friction provides learning opportunities (Reiser, 2004). However, that friction should be strategic when possible (Lamm et al., 2014; Reiser, 2004). For example, if a source of friction in the classroom is from the team dynamics, such as individual accountability of tasks, the learning goals are harder to meet (Decuyper et al., 2010; Oakley et al., 2004).

Finally, a course's primary or secondary learning goal may be to learn various software development frameworks. But, learning the ins and outs of a specific framework, such as Agile and Scrum, is challenging to fit into the course schedule. The instructor may choose a specific framework to utilize on a project parallel to students learning how to work effectively with a team on a larger scale project. Agile is a logical choice, with it being one of the popular frameworks (Rover et al., 2014). Due to Agile not being a primary learning goal, sufficient instruction is not always possible, leading to students haphazardly following an incomplete framework even though the teaching of these methodologies is becoming more prominent (Rico & Sayani, 2009). Students may become frustrated, leaving a "bad taste in their mouth," which may prevent them from learning some of the good habits presented by following a solid model (Krehbiel et al., 2017; Sherrell & Robertson, 2006).

To address these problems, I developed AIECode (Agile in Education Collaborative Development). AIECode works with GitLab's API. The data stored within GitLab is the source of data as well as data storage. The tool may potentially be expanded in the future to other version control repository systems based on educators' needs. AIECode handles the minimal core administrative aspects for software development projects.

2 RELATED WORK

2.1 The framework

While there are many software development frameworks to choose from, I have chosen Agile due to its prevalence in the industry (*State of Agile Survey*, 2020). There are several varieties of Agile. I specifically focused on Scrum.

According to the State of Agile Survey in 2020, Scrum is the most practiced method, with 75% of responses indicating that they used Scrum or some hybrid that included Scrum. From a "tool used in academia" standpoint, there are

several studies that conclude there are benefits to using Scrum in the classroom (Hans, 2017; Kudikyala & Dulhare, 2015; Werner et al., 2012). Hans' research concluded that having a "student Scrum master may have a positive impact on both the quality and the delivery time of the project" (Hans, 2017, p. 4). Kudikyala and Dulhare utilized the Scrum burndown charts (Kudikyala & Dulhare, 2015, pp. 5–6).

2.2 The software

There are many tools out there that have been used in both industry and academic environments. However, each of these tools has some considerable disadvantages. Furthermore, none of the tools were written specifically for academia to address the challenges unique to this environment.

A common tool used by students is Trello. Trello is a board style task tracking system that can be utilized to follow the Scrum Board or Kanban Board workflow tracking portion of software development. At the top of the list of general tool usage in the State of Agile report is Kanban Boards (*State of Agile Survey*, 2020). While Trello is a flexible task tracking tool, this flexibility is a hindrance that requires extra work to set up and maintain, and when not done correctly, it can affect the project's progress (Bass et al., 2016).

On the opposite end of the spectrum from Trello is Jira. Jira is a feature-rich issue tracking system. While these rich features may be beneficial in the industry, these features may overly complicate classroom projects and unnecessarily burden the student (Umphress et al., 2002).

Somewhere in between Trello and Jira is Pivotal Tracker. This is a project management tool that is less feature-rich than Jira but is more attuned to software project development than Trello. Pivotal Tracker is an industry tool. However, it has been part of an academic study (Werner et al., 2012). While it has fewer features than Jira, students still found it challenging to use.

Finally, ZenHub is a program that extends GitHub to provide features not native to GitHub. These features can be used to follow an Agile/Scrum framework such as task estimates. In a study by Palacios et al., students were allowed to use ZenHub but no one decided to use it, as they felt it would add unnecessary difficulty (Palacios et al., 2020).

2.3 But what about the instructors?

van Leeuwen and Rummel studied two types of dashboards (van Leeuwen & Rummel, 2020). The first type of dashboard was a mirroring dashboard. This dashboard-style provides information and is the type of dashboard that might be found in the previously mentioned tools such as Jira. The second type of dashboard was an advising dashboard which "provide information and alert the teacher to groups that are in need of support" (van Leeuwen & Rummel, 2020, p. 1). The study found that advising dashboards were more beneficial to teachers. None of the reviewed tools above included an advising dashboard for instructors since they are not geared towards academia.

3 THE SOLUTION

3.1 Summary

The goal of AIECode is to address these problems. AIECode will be implemented in Phases.

3.1.1 Phase 1: 2019-2020

The initial proof of concept began in 2019 and was completed in 2020. The primary functionality implemented during this phase was to generate burndown charts and to summarize hours reported to the students.

Figure 1 (left) shows a burndown chart for an actual project completed during the Fall semester of 2019. Hours spent by team member can be seen in Figure 1 (right).

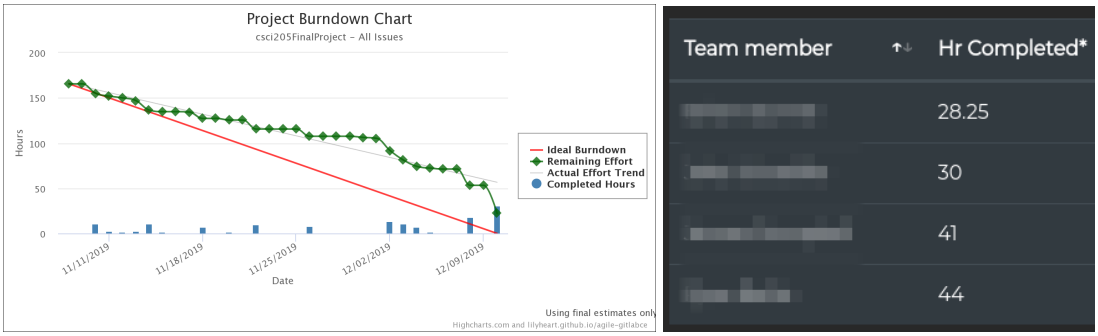


Figure 1—Burndown chart (left) and hours spent (right) from a four week project completed in the Fall of 2019 semester.

The target audience of this phase included a Software Engineering and Design 200 level undergraduate course and a Senior Capstone 400 level undergraduate course. The experience of this student audience ranged from students having their first introduction to Agile and Scrum to students who used the methods in at least one previous course..

3.1.2 Phase 2: Spring 2021 [Current Phase]

AIECode is a high fidelity working prototype of the final tool which handles the minimal core administrative aspects of Scrum for software development projects. In the context of AIECode, the terms "issue" "user story" and "story" are interchangeable. In Agile, they are called stories, while AIECode will track them utilizing GitLab's issue system.

This phase is considered a prototype as it is not fully featured. This implementation does not utilize any backend outside of the information found and stored in GitLab. A backend will be required for various reporting and performance needs.

The target audience for this phase is students who have never been exposed to scrum before. The tool instruments the instruction provided by the course materials. The tool itself will provide instruction and educational opportunities in a future phase.

3.1.3 Future Phases

There are at least two more phases planned.

- Phase 3:
    - This will be a full implementation of the prototype completed during Phase 2. This phase will be completed by the end of summer break 2020 and will be used during a Software Engineering and Design course in an undergraduate program. This is the same course that used the tool discussed in Phase 1.
    - This phase will utilize a backend in order to generate the identified reporting needs found during the current phase. The target audience is the same as Phase 2.
- Phase 4:
    - This phase will extend the tool to provide lessons on Agile and Scrum relying less on the instructors to provide instruction.

- The target audience will be extended to include students with more experience with Scrum. Therefore, the tool will become more customizable to meet the needs of different types of students.
- The tool may be expanded to include GitHub.

Further future plan information can be found in the Future Work section.

3.2 Core features

AIECode handles the minimal core administrative aspects of Scrum for software development projects.

The tool has been kept to a handful of core tenets. These influenced what features to implement and how to implement them.

- Avoids the overly complex setup and maintenance seen in existing tools on the market
- Keeps a specific focus on:
  - Most important parts of Scrum to know and understand
  - Increased team accountability to motivate all team members to participate

Utilizing GitLab's API as the source of data provides key benefits. AIECode mirrors Gitlab's member list (students will have already added their teammates as well as the instructor to the repository so this eliminates having to repeat adding access for everyone in a separate tool). Core data will still be available via GitLab even if AIECode becomes unavailable. Additionally, any externally hosted GitLab that uses the version 4 API can be easily added to AIECode.

3.2.1 User Stories

At the heart of the system are user stories. In the context of this project, the terms "issue" and "story" are interchangeable. In Agile, they are called stories, while AIECode will track them utilizing GitLab's issue system. AIECode allows students to create new stories and edit existing stories. This is similar to functionality on GitLab. When the student adds a new user story, they are automatically prompted to complete the User Story format.

Some Agile/Scrum implementations track effort by time while other track effort by points. As students are new to estimating workload, AIECode keeps with the time method to reduce the amount of new concepts being introduced for the first

time. This mirrors GitLab functionality of time tracking. However, the time tracking in GitLab is handled by very specific syntax comments which are error prone. AIECode removes this ambiguity by restricting inputs to only valid commands and then reformats the comments and posts them to GitLab for the student. Similar to what can be accomplished directly in GitLab, students can edit estimated time and add spent time by date as shown in Figure 2. Additionally, AIECode allows setting the remaining time directly and editing spent time per day, both which is not currently possible on GitLab natively (to edit spent time students must add another /spend command after doing the math about how many hours to add or subtract).

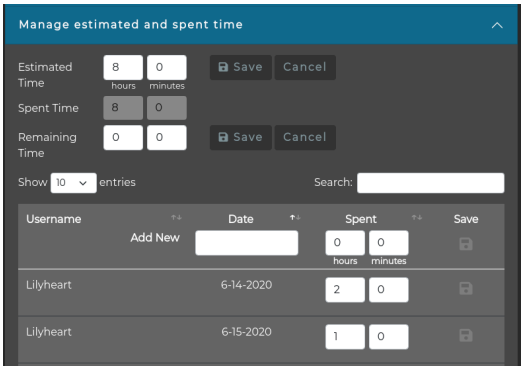


Figure 2—The Manage Estimated and Spent Time section on AIECode.

AIE Code also allows for viewing of User Personas. In the current prototype, students are unable to add new User Personas within the tool, however they can be added to GitLab directly. This will be fully implemented in the new phase.

3.2.2 The sprint cycle

User Stories are the building blocks to the Sprint. AIECode walks the students through the Sprint Cycle.

Once the Sprint is complete, the student can start the next stage of the sprint by clicking the End Sprint button. This will begin the Sprint review. The student is asked some of the key important questions of a Sprint Review. Additionally, issues that AIECode can identify as being problematic for a Review are highlighted as shown in figure 3.

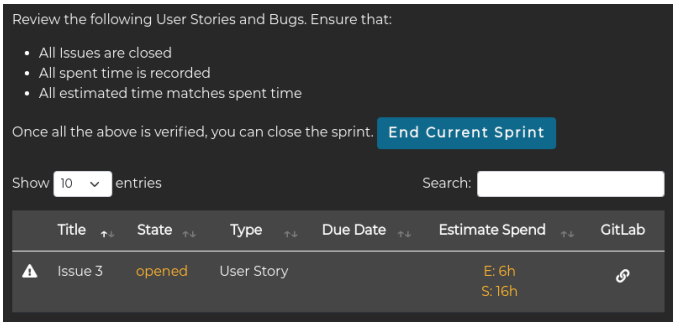


Figure 3—The Sprint review page indicating that there are issues: an issue is still open and the time spent does not match estimated time.

A Sprint Retrospective was not included in this phase. The questions that are answered as a part of the retrospective are a part of the conversation the team has with the instructor at the end of each sprint. Adding them to AIECode would duplicate work.

Next AIECode assists the student with sprint planning by displaying a kanban style board of the product backlog and the new sprint. The student can drag and drop stories between the two. The total number of hours of total work is displayed at the top of each of the boards as shown in figure 4. When a story is dragged between boards, they are immediately updated on GitLab to indicate the new sprint (called Milestones on GitLab).

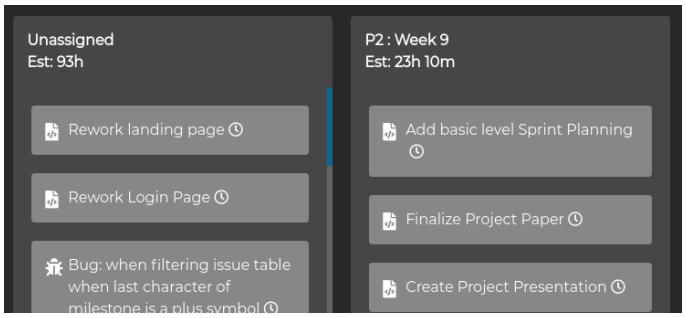


Figure 4—The Sprint planning board allows the students to quickly plan the next sprint by drag issues between the boards with updated total estimated time.

3.2.3 The reports

The reports can be divided into two broad categories. Student and Instructor. This phase only implemented student dashboards and reports. Future phases will introduce instructor reporting.

There are two primary views the student can see reports: current sprint and project overview. Both views show the same charts (See *Appendix 10.1: Report charts* for example images)

- Health Bar: A single bar that indicates the ratio of 1) unstarted hours 2) Remaining hours of stories that have some logged time 3) Spent Hours
- Burndown Chart: A combined line and bar graph that shows a line remaining effort hours by day and completed hours per day in vertical bars.
- Assignee Chart: A variable pie chart that gives a visual indicator of relative workload per team member. The size of the pie slice around the circumference shows the relative amount of work each student has been assigned. The size of the slice from the center to the edge shows the amount of work that has been completed

Additionally, the sprint view has a table that indicates the current issues, their status, assignee, remaining and total spent time.

4 METHODOLOGY

Due to the scope and timing of the project, evaluation during this semester is difficult. However, I will continue this project beyond the semester. Therefore, the project's success will be evaluated by the following criteria in the Fall of 2021.

- The software will be tested by an undergraduate class that teaches software engineering and design.
- A core set of Scrum principles are implemented.
- The primary features of the tool can be utilized directly within the AIECode tool (i.e., students will not have to jump back and forth between AIECode and GitLab).
- Student success:
    - Students are able to see what work has been done
    - Students are able to see what works still needs to be done
    - System is not overly burdensome to use (e.g., daily time tracking)

    - Students have a good understanding of the core features of Agile/Scrum
    - Students can answer why those core features are important
- Instructor success:
    - Instructors are able to view all current projects with minimal effort
    - Instructors are able to see teams that may potentially have issues (e.g., haven't tracked spent time recently or missing daily meetings if they are done virtually)

5 THE RESULTS

The success of the tool cannot be fully measured until it is implemented in a classroom environment. However, we can evaluate the feature list of the current version against the proposed success indicators.

5.1 The needs are met

A core set of Scrum principles are implemented. While not all scrum principles are implemented within the tool, such as sprint retrospectives, the key set of principles that are important for the target audience of students who have never been exposed to Scrum principles before have been implemented as outlined in section 3.

Students are able to see what work has been done. There are several ways students are able to see this information; the closed issues tab, the project or sprint tab burndown chart and the project or sprint assignee chart.

Students are able to see what works still needs to be done. This can be viewed in several ways including the sprint view for current sprint only tasks, and the project backlog table.

System is not overly burdensome to use (e.g., daily time tracking). The system is much quicker to manage estimated and spent time than it is directly within GitLab.

5.2 Mixed results

The primary features of the tool can be utilized directly within the AIECode tool (i.e., students will not have to jump back and forth between AIECode and GitLab). For the most part, this was a success. Notable exceptions are:

- Students are unable to create user personas within the tool, however they can view and edit them. They must be created in GitLab first.
- Students are unable to view reports for previous sprints such as burndown charts. However, they are able to view the table of issues in a sprint via the Closed Issues table and filtering by the desired sprint.

5.3 Not met in this phase

Students have a good understanding of the core features of Agile/Scrum. Educational opportunities will be added in a later phase.

Students can answer why those core features are important. Educational opportunities will be added in a later phase.

Instructor success: Instructors are able to view all current projects with minimal effort and Instructors are able to see teams that may potentially have issues (e.g., haven't tracked spent time recently or missing daily meetings if they are done virtually). Both of these require an additional backend which was not feasible for this phase.

6 LIMITATIONS

The early decision to not add an additional backend hamped the desired feature list more heavily than originally anticipated.

The tool's current target audience limits what students would benefit from the tool. Until the tool is further expanded this can hamper the students in unexpected ways such as having to scramble to find new tools for projects beyond their initial introduction with AIECode as the tool might not be flexible enough to meet their needs for projects lasting more than a portion of a semester.

Additionally, the tool is limited in that it entirely relies on the instructor to explain the key concepts and why they are important.

7 CONCLUSION

A well designed project development tool should be seamless where possible with well planned friction to promote learning. The tool should have a feature set specific to academics without being overly complex. The tool should provide enough advantages to implementing Agile/Scrum that students prefer to use it over GitLab as the tool doesn't add any unnecessary difficulty. As students may

end up being required to use project development tools once they enter the workforce, the tool should serve as a stepping stone between academics and industry.

The current project was proposed as a high fidelity working prototype for a more fully featured project that would take longer than the time available for this phase. Most of the desired features were implemented in this working prototype.

8 FUTURE WORK

There are many areas ripe for further work. Specific to AIECode, the core tenets mentioned in section 3.1 should be expanded as follows:

- Maintains a free model to ensure students can utilize the tool:
    - Without cost
    - Without having to request special education exceptions
- Keeps a specific focus on:
    - Provide feedback to students to increase knowledge and skills (such as feedback regarding time estimates)
    - Future increased team accountability to motivate all team to participate to avoid what I call the Group Project Pareto: 80% of the work is done by 20% of the team
    - Ease the burden on the educator to track, maintain, provide feedback on, and grade many concurrent projects
    - Provides education moments to teach the reason behind the aspects of the Scrum framework

Many other features would benefit the tool and still stay within the vision and requirements for the tool.

- Project Burnup chart, especially for the instructor to judge changes in scope over time.
- Tracking of Epics and Spikes (XP concept)
- Bass et al. found that the daily scrum meetings may have helped reduce team conflict (Bass et al., 2016, p. 4). Kudikyala and Dulhare found it difficult to implement daily meetings (Kudikyala & Dulhare, 2015, pp. 5–6).

- Sprint Retrospectives and more advanced Sprint Planning overall to make the tool more relevant to more advanced students (See *Appendix 10.2: Advanced sprint retrospective and planning flow* for a potential flow.)
- Kanban boards
- Connecting user personas and user stories
- Allowing for story instead of time
- Instructor Reports
    - Mirror dashboards to provide metrics of the project
    - Advising dashboards that automatically highlight teams that may require additional intervention.

Adding a GitHub implementation will also add additional concerns such as GitHub doesn't handle effort in time natively.

9 REFERENCES

1. Bass, R. B., Pejcinovic, B., & Grant, J. (2016). Applying Scrum project management in ECE curriculum. *2016 IEEE Frontiers in Education Conference (FIE)*, 1–5. https://doi.org/10.1109/FIE.2016.7757568
2. Decuyper, S., Dochy, F., & Van den Bossche, P. (2010). Grasping the dynamic complexity of team learning: An integrative model for effective team learning in organisations. *Educational Research Review*, *5*(2), 111–133. https://doi.org/10.1016/j.edurev.2010.02.002
3. Hans, R. T. (2017). Work in Progress - The Impact of the Student Scrum Master on Quality and Delivery Time on Students' Projects. *2017 International Conference on Learning and Teaching in Computing and Engineering (LaTICE)*, 87–90. https://doi.org/10.1109/LaTiCE.2017.22
4. Koster, B., & College, M. (2006). Agile methods fix software engineering course. *Journal of Computing Sciences in Colleges*, *22*(2), 131–137.
5. Krehbiel, T. C., Salzarulo, P. A., Cosmah, M. L., Forren, J., Gannod, G., Havelka, D., Hulshult, A. R., & Merhout, J. (2017). Agile Manifesto for Teaching and Learning. *Journal of Effective Teaching*, *17*(2), 90–111.
6. Kudikyala, U. K., & Dulhare, U. N. (2015). Using Scrum and Wikis to manage student major projects. *2015 IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE)*, 15–20. https://doi.org/10.1109/MITE.2015.7375279
7. Lamm, M. H., Dorneich, M., & Rover, D. T. (2014, October 14). Team-Based Learning in Engineering Classrooms: Feedback Form and

Content Adds Value to the Learning Experience. *2014 ASEE North Midwest Section Conference: Engineering Something More*. Engineering Something More, Iowa City, Iowa, USA. https://doi.org/10.17077/aseenmw2014.1026

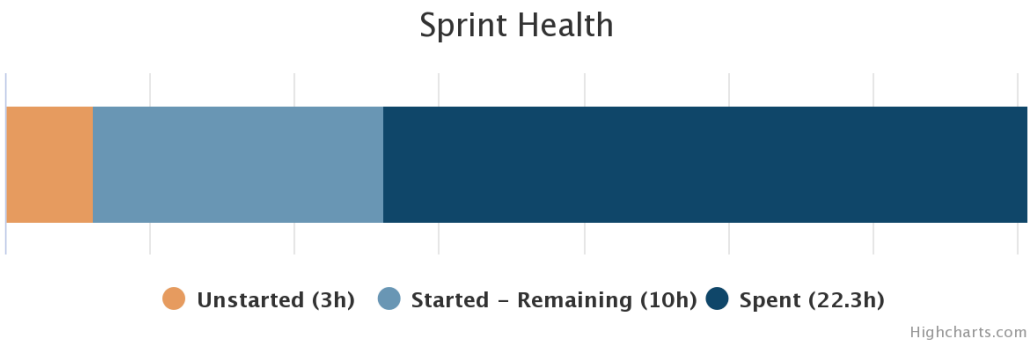
8. Machado, M., & Tao, E. (2007). Blackboard vs. moodle: Comparing user experience of learning management systems. *2007 37th Annual Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports*, S4J-7-S4J-12. https://doi.org/10.1109/FIE.2007.4417910
9. Oakley, B., Felder, R. M., Brent, R., & Elhajj, I. (2004). Turning student groups into effective teams. *Journal of Student Centered Learning*, *2*(1), 9–34.
10. Palacios, R. C., Casado-Lumbreras, C., Samuelsen, T., & Larrucea, X. (2020). Students' selection of teamwork tools in software engineering education: Lessons learned. *The International Journal of Engineering Education*, *36*(1), 309–316.
11. Raith, F., Richter, I., & Lindermeier, R. (2017). How Project-management-tools are used in Agile Practice: Benefits, Drawbacks and Potentials. *Proceedings of the 21st International Database Engineering & Applications Symposium*, 30–39. https://doi.org/10.1145/3105831.3105865
12. Reiser, B. J. (2004). Scaffolding Complex Learning: The Mechanisms of Structuring and Problematizing Student Work. *Journal of the Learning Sciences*, *13*(3), 273–304. https://doi.org/10.1207/s15327809jls1303_2
13. Rico, D. F., & Sayani, H. H. (2009). Use of Agile Methods in Software Engineering Education. *2009 Agile Conference*, 174–179. https://doi.org/10.1109/AGILE.2009.13
14. Rover, D., Ullerich, C., Scheel, R., Wegter, J., & Whipple, C. (2014). Advantages of agile methodologies for software and product development in a capstone design project. *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, 1–9. https://doi.org/10.1109/FIE.2014.7044380
15. Sherrell, L. B., & Robertson, J. J. (2006). Pair programming and agile software development: experiences in a college setting. *Journal of Computing Sciences in Colleges*, *22*(2), 145–153.
16. *State of Agile Survey*. (2020). https://stateofagile.com/#ufh-i-615706098-14th-annual-state-of-agile-report/7027494

17. Umphress, D. A., Hendrix, T. D., & Cross, J. H. (2002). Software process in the classroom: the Capstone project experience. *IEEE Software*, *19*(5), 78–81. https://doi.org/10.1109/MS.2002.1032858
18. van Leeuwen, A., & Rummel, N. (2020). Comparing teachers' use of mirroring and advising dashboards. *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, 26–34. https://doi.org/10.1145/3375462.3375471
19. Werner, L., Arcamone, D., & Ross, B. (2012). Using Scrum in a quarter-length undergraduate software engineering course. *Journal of Computing Sciences in Colleges*, *27*(4), 140–150.

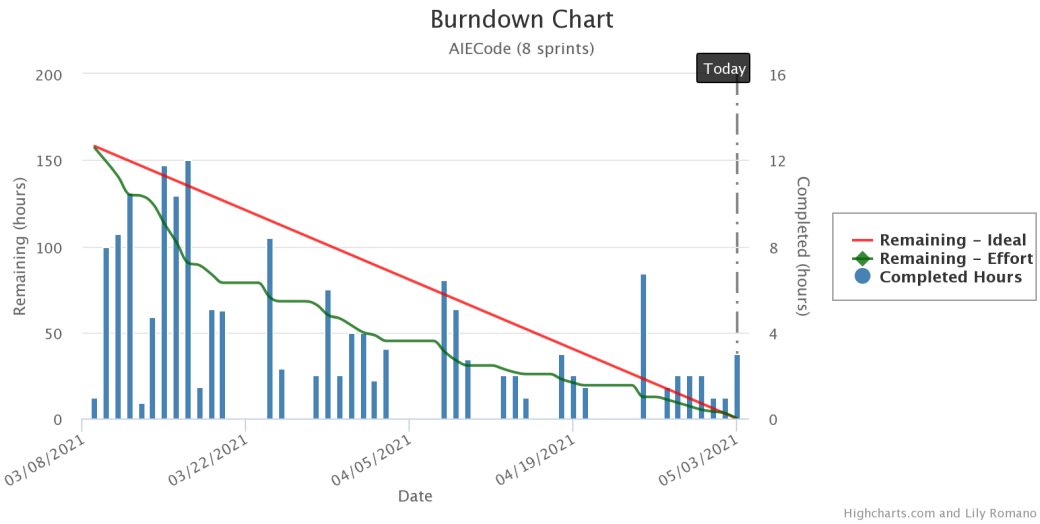
10 APPENDICES

10.1 Report charts

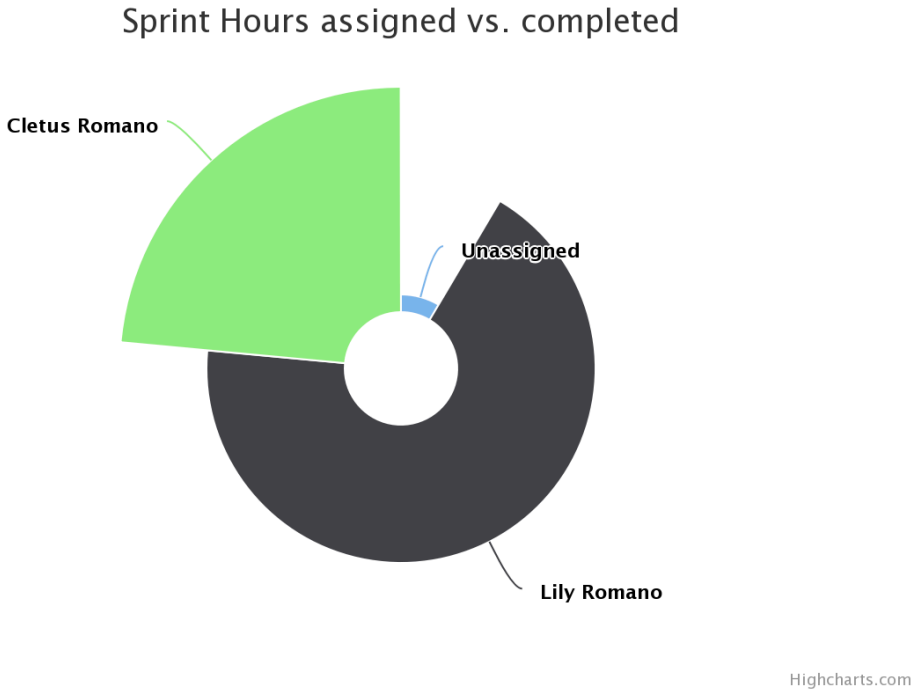
10.1.1 Sprint Health Bar



10.1.2 Burndown Chart



10.1.3 Sprint hours assigned vs completed.



10.2 Advanced sprint retrospective and planning flow