CS Capstone Final Reflection

This document is intended as a guide for the capstone team to assess its performance in a number of dimensions. You need not answer each question in detail, rather, use the questions as a guide for the kinds of items to assess. Add items you feel are appropriate.

This self-assessment will be one of multiple elements that your advising instructor uses to arrive at an assessment of the team's performance for the term. The other elements that the advising faculty will use include: direct observation of the team, team peer evaluations, sponsor evaluation, and project deliverables. These self-assessments will also be used as part of the CS program's accreditation and curriculum improvement efforts.

To complete this self-assessment the team should carefully consider each of the questions and provide an honest evaluation of the team's performance.

Team: CTRL 5

Project: WASM AS OS

Sponsor: Jacob, Hochstetler

Product

1. Did the team prepare all the documentation artifacts requested by your advising faculty and sponsor? Were these documents carefully inspected prior to delivery? How would you assess the quality of the document artifacts?
   Throughout the semester, we created every document our advisor and sponsor asked for proposals, requirements, design documents, sprint reports, and our final write-ups. In the beginning, we were honestly just trying to figure out the flow of what these documents should look like, so some of the early ones weren't as polished. As the semester went on, we got better at reviewing each other's work before submitting anything. By the end, the quality of our documents improved a lot because we were checking for clarity, consistency, and whether they matched the current state of the project.

2. How well did the team elicit the requirements? What approaches were used to elicit the requirements? Were key requirements missed? What methodology was used to document and validate the project requirements?

We gathered requirements through meetings with our sponsor, follow-up emails, and our own research into Web Assembly and the type of system we were building. At first, we thought we understood everything, but once implementation started, we realized that some requirements needed clarification. We asked more questions during the middle sprints and updated our requirements document to reflect what the sponsor wanted. We tried to keep everything organized using user stories and acceptance criteria so everyone stayed on the same page.

3. Did the team explore the entire design space before arriving at a final design? Have there been many errors found in the design? Was it necessary to make major changes to any part of the design? What were the reasons for the change?

   As a team, we spent time discussing different ways the system could be structured before choosing a direction. We looked at several architectural approaches, considered the strengths and weaknesses of different technologies, and talked through how each design choice would affect the overall project. These conversations helped us understand the design space better and allowed everyone to contribute ideas. Over time, our design naturally evolved as we gained a clearer picture of what would best support the project goals and what the sponsor ultimately wanted. The changes we made along the way weren't about fixing implementation problems but about refining our understanding and aligning the design with the bigger vision of the project.

4. How has the development and implementation progressed? What percentage of the product do you estimate was completed? Is the team providing the documentation within the implementation artifacts?

Over the course of the semester, the team worked consistently toward building out the features and components we committed to at the start. Each sprint allowed us to make steady progress, and we were able to complete the majority of what we originally planned. We estimate that around 80–85% of the intended product was delivered, with the core functionality clearly represented in the final version. Throughout development, we made a point to keep our work organized, documented, and easy for others to follow. We maintained notes, comments, and structured explanations of major components so future developers or even our sponsor can understand how everything fits together.

5. What was the team's testing strategy? Did the team develop a test plan? If so, was it followed? Did the team perform unit testing? Did the team use any test frameworks, such as JUnit? What are the testing results? Were any major defects found during system test? If so, were they fixed? Did the team do regression testing?

From the beginning, we agreed that testing needed to be built into our process rather than treated as an afterthought. We created a test plan outlining how we would verify key behavior and features, then carried out both unit-level and higher-level testing as features were completed. For the backend logic, we wrote tests to confirm that scheduling and task-handling behaved as expected. On the front-facing side of the system, we reviewed the functionality through planned scenarios that mirrored real use cases. Whenever we identified something that could be improved, we adjusted the design or fine-tuned the behavior. By the end of the semester, our testing approach helped us gain confidence that the system aligns with the conceptual goals and functional expectations we set early on.

6. Products need to be designed within guidelines and constraints appropriate for each project. It is also important to consider the impacts of the products that are designed. In the following categories discuss the constraints and impacts that have a bearing on your project. Note that all of these categories may not have bearing on your project but your project is probably affected by many of them.
   - Economic issues- The use of WebAssembly contributes to a lightweight and cost-efficient execution model, which can be attractive for organizations looking for high performance without heavy infrastructure.
   - Environmental issues- Software projects like this have minimal direct environmental impact, though efficient systems can indirectly reduce resource consumption.
   - Social issues- Our project highlights modern computing concepts that can support educational and research environments, helping others understand new approaches to system design.
   - Political issues- There were no political constraints affecting this project.
   - Ethical issues- We kept ethics in mind by considering safe execution, responsible resource handling, and avoiding unnecessary exposure of internal processes.
   - Health and safety- While not applicable in a traditional sense, the reliability and predictability of execution within Web Assembly can be seen as a safety consideration within the software domain.
   - Manufacturability- For software, this translates to how maintainable, organized, and extensible the system is. We aimed to keep the structure clear so it can be expanded or adapted later.
   - Sustainability- Choosing widely supported, forward-looking technologies like WebAssembly helps ensure the project can evolve over time and remain compatible with broader industry trends.

7. What industry and engineering standards was your project required to adhere to? Were these new standards that the team had to learn? Did your sponsor provide you support for understanding these standards? Did you have to educate your sponsor about these standards?

Because the project involved Web Assembly, we naturally adhered to the conventions and expectations defined by the WASM ecosystem. This included understanding how modules are structured, how execution is handled, and what constraints come with running code in a sandboxed environment. Some of these standards were new to our team, so we spent time researching, learning from documentation, and discussing among ourselves to make sure we approached the project in a way that aligned with established practices. The sponsor provided clarity when needed, but a lot of our understanding came from exploring the technology and building shared knowledge.

Process

1. What was your process methodology? Was the process appropriate for the project? Did you follow the process or modify it as the project progressed? If you could repeat the project, what would you do differently?

As a team, we followed an Agile-inspired process with sprints, regular check-ins, and reflection cycles. This structure helped us stay organized and break the project into manageable phases. Over time, we adjusted our process to fit our working style adding more detailed planning when needed or meeting more frequently during weeks when decisions required group input. Looking back, the methodology suited the project well, though if we were to repeat the experience, we would start certain planning discussions earlier to set an even stronger foundation at the beginning.

2. Was there a large requirement to learn the problem domain? What approach was used to gain domain expertise?

There was definitely a learning curve in understanding the broader domain of Web Assembly and how OS-like behavior could be simulated in that environment. We approached this challenge by dividing up topics, sharing what we learned, referencing documentation, and discussing ideas together. By the midpoint of the semester, the team had built a shared understanding that allowed us to move forward with more confidence.

8. What mechanisms did the team using to track project progress? Did they give the team and sponsor adequate insight into project progress and issues? How well did the team

track its project progress? How often did these artifacts get updated on the department project website?

To keep track of everything throughout the semester, we relied on a combination of tools that worked well for our team. GitHubwas our main space for organizing tasks and keeping our work in one place, but we also used shared Word documents and Google Docs to draft ideas, gather notes, and record decisions. Outside of the official tools, we communicated regularly through Discord and group messaging, which made it easy to ask quick questions, update each other, and stay aligned even outside scheduled meetings. Together, these tools gave us a clear sense of where we were in the project and helped us keep everything moving steadily from sprint to sprint.

9. Did the team conduct effective meetings?

Our meetings were productive and became more structured as the semester went on. Team meetings allowed us to divide tasks, address questions, and make decisions collectively. Meetings with our sponsor were always valuable, they helped us refine our understanding of the project and kept us focused on the intended direction. Over time, the rhythm of these meetings became an important part of our workflow.

10. Did the team meet all project milestones? Which milestones, if any, were missed or were met ahead of schedule? What contributed to schedule changes? What could the team have done differently to ensure that milestones were met?

    We met the majority of our planned milestones. When timelines shifted, it was usually because the team made thoughtful decisions about what needed more attention or clarification. The adjustments we made were based on ensuring that the final product aligned with the goals we had set early in the project. With even earlier planning and more up-front discussions, we believe we could have reached some milestones sooner, but overall, the project moved at a steady and consistent pace.

11. Was the team required to adopt new technologies? What were these technologies? What approach did the team use for selecting the appropriate technology for the project? Did the sponsor provide any support for learning these technologies? How well did the team ramp up on the new technologies and begin to apply them effectively?

Because this project involved Web Assembly and related tooling, we naturally had to adopt several new technologies. Each team member took time to learn different aspects some focused on runtime concepts, others on scheduling behavior, API design, or frontend structure. We shared knowledge frequently, and the learning process itself became one of the most rewarding parts of the project. As the semester progressed, we became noticeably more comfortable with the technologies and used them more effectively.

12. How well did the team maintain quality control over the project artifacts? Have all artifacts been reviewed for adherence to quality standards? What was the review process used by the team?

Quality control was something we developed gradually over the semester. We reviewed each other's documents, discussed design decisions openly, and kept our work organized in a way that made it easier to follow. As our understanding grew, we became more intentional about ensuring clarity and consistency in our artifacts. This helped maintain a level of quality that we feel proud of as a team.

13. Did the team have any issues with configuration management? How were these problems solved? What percentage of project artifacts is under configuration control?

We used GitHub to manage our work and keep everything versioned. Over time, we developed a rhythm for how we organized branches, handled updates, and maintained our files. By the end of the semester, nearly all of our project artifacts were properly maintained under configuration control, and our workflow as a team became smooth and predictable.

Communication and Interaction

1. How well did the team communicate project progress to the sponsor? What regular communication did the team have with the sponsor? Did the team been maintaining this communication to the satisfaction of the sponsor? Were any adjustments needed in the communication over time? Were these changes initiated by the team or the sponsor?

Our communication with the sponsor was straightforward and aligned with what the project required. We met with him twice during the semester. The first meeting was an introduction where he explained his vision for the project and gave us a clear overview of what he expected the system to accomplish. After that, most of our communication happened through updates rather than frequent meetings. Toward the end of the sprint cycle, we gave updates, walk through what we had built, and confirm that the project direction matched what he originally described. Even though the meetings were limited, they were productive and gave us enough clarity to move confidently throughout the semester.

14. Did the team need to provide technical input to the sponsor? How well did the team educate the customer in these areas? What mechanism did the team use?

Whenever technical topics came up, we made it a point not to assume our sponsor was familiar with the deeper details of the technologies we were using. Instead, we explained our decisions and system behavior in a clear, straightforward way. We focused on breaking things down so he could easily understand how the pieces fit together and why certain choices made sense for the project. Even though we didn't have many meetings, the explanations we provided were simple, organized, and tied back to his original goals. This helped make sure everyone stayed on the same page without overwhelming him with unnecessary technical detail.

15. Was this an effective team? What has been contributing to and detracting from the team's effectiveness? What are the team's weak points? What are the team's strong points? What changes could the team have made to make it more effective?

We consider ourselves an effective team. Each member contributed in their own way, and we found a balance between individual strengths and group collaboration. Some of the things that contributed to our effectiveness were communication, willingness to learn, and openness to feedback. If we were to improve anything, it would simply be establishing certain routines earlier in the semester. Overall, the team dynamic was positive and supportive.

Achieving Customer Satisfaction

1. In the team's opinion did the work satisfy the project sponsor? Were there any weak spots in this regard?

Based on our interactions and the feedback we received, we believe our sponsor was satisfied with the work we produced. The core goals of the project were represented in the final product, and we maintained transparency throughout the process.

Achieving Team Satisfaction

1. Did the project satisfy the team's expectations for learning? Were there any weak spots in this regard? What could have been done differently to improve the team's learning experience?

As a team, we feel the project met and, in many ways, exceeded our expectations for learning. We gained hands-on experience with unfamiliar technologies, improved our collaboration skills, and developed a clearer understanding of what it means to plan and execute a long-term technical project. If there is one thing we would change, it would be giving ourselves even more time early on to explore concepts before jumping into structured work. Other than that, we are proud of what we accomplished and grateful for the experience.