2 Project Plan

2.1 Task Decomposition

In order to solve the problem at hand, it helps to decompose it into multiple tasks and subtasks and to understand interdependence among tasks. This step might be useful even if you adopt agile methodology. If you are agile, you can also provide a linear progression of completed requirements aligned with your sprints for the entire project. At minimum, this section should have a task dependence graph, description of each task, and a justification of your tasks with respect to your requirements. You may optionally also include sub-tasks.

Have the created problems be autograded accurately in order to make the learning process smooth for the students

Creating questions for exams and homework assignments must be a process that is easy to complete for instructors

Creating courses for students to join and complete assignments filled with questions

Integrate microcontroller emulators into the testing environment to accurately simulate the results of the answers given by the students.

Integrate microcontrollers into the testing environment to accurately simulate the results of the answers given by the students.

- Implement auto grader
    - We need an auto grader to automatically grade submissions. This includes compiling and running c code in a docker environment and evaluating the results.
- Create questions for homework assignments / labs / quizzes / exams
    - Going through the course material and create questions to match course content
        - Binary, Decimal, Hex conversion
        - Digitial logic
        - Clock fuctionality
        - Register bit initializing
        - Datasheet questions
        - C and assembly coding questions
- Integrate emulator into auto grading / testing for labs
    - Create an emulator for the cybot and microcontroller, that will take in uploaded student code and show the results of running the code. This will have to be connected to the auto grader docker environment because the code needs to be compiled

2.2 Project Management/Tracking Procedures

We will be using an agile project management style due to this style being non-linear and incremental. This style fits our project as our goals are determined by decisions made throughout the development process. Many decisions that we make regarding what is to be done is dependant on previous tasks that are completed; however, this process is not linear. We will prioritize the development of different features based on what we feel is the most important at that time. For example, a stretch goal of out project have microcontroller emulators built into the PrairieLearn framework that can be compatible with the autograder. If we are able to achieve this goal. then we are able to continue incrementally building in compatibility with physical microcontrollers in addition to the emulators. This process of microcontrollers and emulators can be done before, during, or after the development of a better autograder or different question types.

What will your group use to track progress throughout the course of this and the next semester. This could include Git, Github, Trello, Slack or any other tools helpful in project management.

We will be using Git through GitLab to track progress, which includes their issue boards. We will also do collaboration through git merge requests and an external Discord server for communication.

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

*** (try to have quantitative goal) ***

What are some key milestones in your proposed project? It may be helpful to develop these milestones for each task and subtask from 2.1. How do you measure progress on a given task? These metrics, preferably quantifiable, should be developed for each task. The milestones should be stated in terms of these metrics: Machine learning algorithm XYZ will classify with 80% accuracy; the pattern recognition logic on FPGA will recognize a pattern every 1 ms (at 1K patterns/sec throughput). ML accuracy target might go up to 90% from 80%.

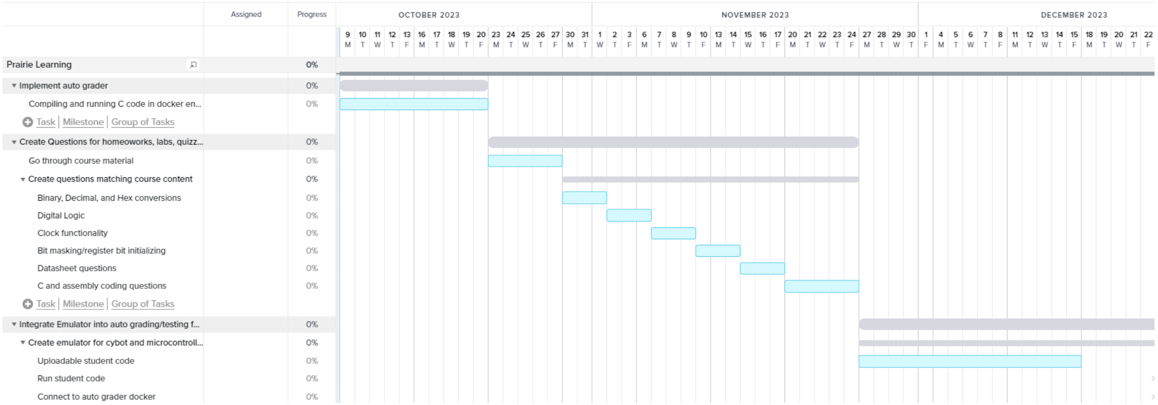
In an agile development process, these milestones can be refined with successive iterations/sprints (perhaps a subset of your requirements applicable to those sprint).

- Improve Auto Grader
  - The Auto Grader must take in questions and output students score with 100% accuracy.
- Interactive Question Development
  - Develop 5-10 new interactive question formats that can be easily changed for each assignment that integrate with the auto grader.
  - Develop a randomization function that can change numbers and other specific parts of a question while keeping the same format. This will integrate with the auto grader with 100% accuracy.
- Implement Emulator
  - Create an Emulator that can take in user code and simulate the real world Cybot.
  - Simulate the driving functionality with 95% accuracy.
  - Simulate the object detection functionality with 80% accuracy
- Create Microcontroller

- ○ Have a working microcontroller that allows user interaction. Further functionality will be determined as the project progresses.

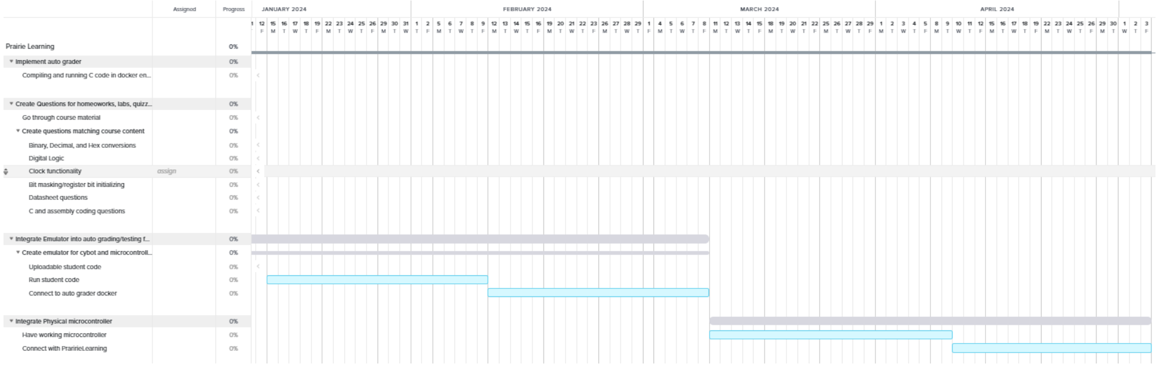
2.4 Project Timeline/Schedule

First Semester Timeline



	Assigned	Progress
Prairie Learning		0%
▾ Implement auto grader		0%
Compiling and running C code in docker en...		0%
▾ Create Questions for homeoworks, labs, quizz...		0%
Go through course material		0%
▾ Create questions matching course content		0%
Binary, Decimal, and Hex conversions		0%
Digital Logic		0%
Clock functionality		0%
Bit masking/register bit initializing		0%
Datasheet questions		0%
C and assembly coding questions		0%
▾ Integrate Emulator into auto grading/testing f...		0%
▾ Create emulator for cybot and microcontroll...		0%
Uploadable student code		0%
Run student code		0%
Connect to auto grader docker		0%

Second Semester Timeline



	Assigned	Progress
Prairie Learning		0%
▾ Implement auto grader		0%
Compiling and running C code in docker en...		0%
▾ Create Questions for homeoworks, labs, quizz...		0%
Go through course material		0%
▾ Create questions matching course content		0%
Binary, Decimal, and Hex conversions		0%
Digital Logic		0%
Clock functionality	assign	0%
Bit masking/register bit initializing		0%
Datasheet questions		0%
C and assembly coding questions		0%
▾ Integrate Emulator into auto grading/testing f...		0%
▾ Create emulator for cybot and microcontroll...		0%
Uploadable student code		0%
Run student code		0%
Connect to auto grader docker		0%
▾ Integrate Physical microcontroller		0%
Have working microcontroller		0%
Connect with PrairieLearning		0%

2.5 Risks And Risk Management/Mitigation

Our project is entirely software based, which means the only risks we need to contend with are performance targets and availability requirements, and the functionality of our tools. Since our project uses docker containers for compiling and simulating code, the server would need to be fast enough to accommodate several classes worth of students at a time. We also need to ensure the server has as little downtime as possible to allow students to work on labs and assignments at all times, especially near due dates. The last risk we could have is PrairieLearn or the emulator could fail to meet specifications and will not work for what we need it to do.

The performance risk is going to be relatively low, probably around .1, since PrairieLearn is a mostly client-based web framework that doesn't take much processing power on the server side beyond auto grading. The availability risk should be around .2 because we can set up multiple servers as redundant backups for when downtime is needed. Finally, the risk for tools not meeting expectations is probably around a .4 considering no one in the team has experience with them. Thanks to that risk, our mitigation plan is to do some market research for tools that may be better suited to our needs. Using that research, if we ever find out PrairieLearn cannot fulfill our needs, we can quickly find a tool to fill that area in.

2.6 Personnel Effort Requirements

Task	Projected Effort (High/Medium/ Low)	Total Person Hours
Improve Auto Grader	Medium	30
Create Questions for Assignments/labs/quizzes/exams	Medium	60
Integrate Emulator	High	120
Integrate Physical Microcontrollers	High	180

The first task to be completed is to improve the existing autograder through the PrairieLearn framework. This is a medium effort task to complete because the auto grader is implemented in the system and needs various improvements. This will take about 20 hours of work to complete and test in the system. The next task is to create questions for multiple assignments, quizzes, labs, and exams for students to utilize. This is a medium effort task for the team because with course material, questions are already created and just need to be implemented. Task 2 will take about 60 hours of total time to complete and include all questions available through the course material. Task 3 involves integrating an emulator for the Cybot and microcontroller. Implementing this requires taking students' code and giving the results of the running code. This is a high effort task because it involves creating an emulator and configuring it to the required needs. Task 3 also has an estimated time of 120 hours for completion because of the large amount of development and testing that will be involved. The final task is to integrate Physical Microcontrollers. The goal is to have a working microcontroller that will allow users to interact with. This is a high effort level because it involves configuring and connecting a microcontroller to work with PrairieLearn. This task will take the most time because it will involve a large amount of development and testing to be completed.

2.7 Other Resource Requirements

Other potential resources we may need for our project are the CPR E 288 course material, example questions, access to the entire PrairieLearn framework, and access to the microcontrollers used in CPR E 288.