The goal of the group project is to dive deeper into language design and implementation by developing a significant extension to MyPL. The deliverables and due dates for the project are as follows.

- **Mar 19**: Project proposal due
- **Apr 2**: Status update 1 due
- **Apr 14**: Status update 2 due
- **Apr 28**: Source code due for evaluation
- **Apr 28–May 7**: Demo and Interview

Each of the deliverables is further described below.

**Project Proposal (Due Mar 19)**. Your proposal must contain:

1. The names of each team member. All but one team should have 4 students.
2. Select a topic. A list of potential topics are given below. You may select a topic not on the list as well.
3. A brief list of what you plan to accomplish (i.e., the scope).
4. A breakdown of what each student will roughly be responsible for. Note that even if you will be working together, each student should be responsible for some aspect of the work.
5. A breakdown of general tasks you’ll need to complete and a timeline of when they will be completed. Note that you must also include tests and test results with our code drop.

The following is a list of possible topics for your project.

- **A MyPL language extension**. Examples could include support for classes (member variables and methods), support for lists (e.g., like in Python), a module system and a standard library (for example, to allow for programs made up of multiple files), support for pattern matching (guards), or support for higher-order functions. If you would like to extend the language, you might also look for inspiration in other languages.

- **MyPL implementation extensions**. Examples could include adding garbage collection (for structured types), allowing overloaded functions, improving type inference (e.g., or function return types).

- **Improving MyPL performance**. Examples could include writing a translator from MyPL to C or assembly, using the LLVM framework or compiling directly to bytecode (e.g., for the JVM), or employing basic optimizations for the interpreter implementation. For compilation to LLVM or the JVM, getting a small subset working for MyPL would likely be sufficient for this project. If you choose this type of project, you must do basic performance evaluation to show how your improvements perform compared to the pure AST implementation.
• **Build MyPL tooling support.** Examples could include writing a REPL (read-eval-print-loop like for Python), developing an editor for MyPL with syntax checking and highlighting, or building a MyPL debugger.

**Status Updates (April 2 and April 14).** You must provide a one-page status update on April 2nd and another on April 14. Each status update should describe the progress you have made as a team, the contributions made by each team member, the work remaining, and a revised timeline as needed. You must also discuss any challenges or issues you encounter.

**Final Code Submission (April 28).** You must submit your final project via GitHub Classroom for review by the TA and the instructor. Your final code submission must include:

- All source code required to build and run your final project. Your source code must be commented, organized, and not obviously “hacked” together.
- Instructions on how to run your project, with example test cases for us to use (think bare-bones tutorial).
- The test cases and results you obtained from running your tests.
- A 1–2 page write up describing what was implemented and the contributions of each team member. Be sure to explain any features that are not fully implemented and any known bugs.

**Demo Session and Interview (April 28–May 7).** Both the TA and instructor will meet individually with each team to discuss their project. The team should be prepared to give a brief demo of their project and to answer questions regarding their implementation. These meetings will be approximately 20 minutes, which will include time for a 5 minute demo and 15 minutes of Q & A. A sign up sheet with possible times will be provided prior to the meetings.

**How you will be graded.** Your grade will be determined based on the following criteria.

1. Difficulty of your project (relative to what you accomplished)
2. Quality of your implementation
3. Completeness of your implementation
4. Quality of your tests and write ups
5. Individual contributions to your team (including from peer evaluation)

Please be sure to ask the instructor about project requirements and expectations if you have questions.