Required Elements of Milestone	Evidence
User Story (no epics, proper format, acceptance criteria)	Make sure each user story is clearly stated. Acceptance criteria must be about the user story. Testing is not acceptance criteria. Ensure user stories selected for grading are small. If there are connections between user stories please note them, especially if a user story is part of an epic.
User Story deliverable	Provide a short video that demonstrates each user story running. Make sure the video is clearly linked from the user story description
User Story repeatable testing	Provide a link to the automated test cases for this user story. Scripts for a human operator to follow are not acceptable testing for this milestone. Coverage reports are not required for this portion of the milestone.
User Story refactored	Provide evidence that the code implementation for the user story is clean and refactored. Evidence could include comparisons between earlier versions, discussion about code smells that have been eliminated (and how), evidence that the code meets your team's coding conventions. It isn't enough just to say it is clean, you need to provide some evidence.
A3 Optional Elements:	
Refactoring: Identify code improvements between M1 and M3. Explain the benefits and process for refactoring done in at least 3 sections of the code.	Ensure that the discussion provides links to the code base as well as to the tickets. Refactoring should be at the method, module, or class level, not individual blocks of code.
Architectures: Explain the architecture used for this project. Identify another possible architecture and explain how it could be accomplished.	Include diagrams that clearly show how the classes from your project map to the elements of the architectures you are discussing. Justify the choice of architecture. Explain the high level changes that would need to be made to change to a different architecture (a specific one).
Design Patterns: Identify and document the existence of any three design patterns in your codebase	Provide an overview and a simple structural diagram of each design pattern. Ensure that the mapping between the design pattern and your code is very clear. Ensure that the discussion provides links to the code base as well as to the tickets. Explain why these design patterns are appropriate for the problem being solved in this part of the code.
Team Choice Options: You may submit one of these that has not been previously submitted by your group	
Testing: Evidence of a full suite of unit tests with coverage reported and a test plan for integration testing.	Evidence of a full suite of unit tests for your entire code base. Coverage report is required and should be explained. Code coverage must be 80% or greater for full marks. Explain how your team plans to conduct integration testing. Identify all testing tools used, or planned for and justify your choice of tools.
DevOps: CI/CD employed for linter, unit tests. Merge rules require pipeline success. Single-issue branch/merge strategy in place for project.	Evidence that CI/CD is employed for merges to master as a minimum. Describe the merge rules for your team. Provide a link or a sample to your team's merge template. Provide a link or screen shot of the merge discussion and resolution as evidence of the merge process being followed for at least 6 merges. Discuss how CI/CD could be used for deployment.
SOLID: Discussion and evidence of adherence project wide to at least 3 of the 5 SOLID principles. Discussion should include reasons why any of the 5 are not included.	Ensure that the discussion provides links to the code base as well as to the tickets
Scrum Processes: Evidence of extra attention to scrum process including well-developed scrum boards, a groomed backlog, use of issue tracking (including regular issue updates, and documentation in wiki on GitLab)	Should include screenshots of the scrum board in different states over the sprint. Issue tracking should include regular updates by developers and provide a clear, well documented picture of how the ticket was implemented and tested. Connections between issues should be clear. Your team wiki should include information about the team members, the overall vision for the product, and information that would be useful to anyone joining the team such as coding standards and merge processes.
Code Smells: Discussion of how any 3 code smells have been avoided or refactored out. Discussion should include definitions and reasons why those code smells are likely/common to this type of project.	Ensure that the discussion provides links to the code base as well as to the tickets
Agile Development: Clear description of the roles for each team member for this sprint. Summary/Minutes from all meetings.	May not choose this one for Sprint Three
Agile Development: Evidence of team processes including merge review, coding standards, use of a linter, build tools (CI is optional)	May not choose this one for Sprint Three