## Team Documentation Template for LMForge

This template is designed to guide each team in documenting their specific contributions, lessons learned, and strategic recommendations for the LMForge project.

## Team Summary & Accomplishments

- **Team Name**: Fine Tuning
- **Team Leads**: Natalie Hilliker and Nikolaus Dobozy
- **Term & Year**: Fall 2025

## Key Challenges

Describe the 2-3 most significant hurdles encountered during your work on LMForge and how your team successfully navigated them.

1. **Challenge 1**: Connecting our devices to the LM forge
    a. **Resolution**: Have members who have successfully connected show demos to product owners and the scrum master
2. **Challenge 2**: Keeping our team on track with implementation over research
    a. **Resolution**: We sought advice from our professor to take the tilt our team needed.

## Lessons Learned

Based on your experience, what are the 2-3 most critical takeaways? This could be technical, process-related, or collaborative–what future teams should know.

- **Lesson 1**: Learning to work virtually with other team members and classmates as things come up. We did a good job at collaborating, even when other circumstances got in the way of going to class in person.
- **Lesson 2**: Working together with other teams to implement and successfully get the software up and running was integral to getting things to work. Being able to ask for help and know how to ask it is vital for a role like this.

## Future Recommendations

This section is paramount for the long-term success of the LMForge project. Provide clear, actionable advice and suggestions for the next iteration of your team's component and the class structure itself.

### Technical and Development Recommendations

1. **Required Technical Debt Resolution**: [Detail any unfinished features, suboptimal code, or known bugs that must be addressed next.]
   One thing we would like to implement is an easier way to enter the LMForge, especially for the business majors within the class.
   *The UI (Fine-Tuning) is having Encoder implementation. The form is connected to encoder_training.py (which has the bert code for SQuAD)*
2. **Proposed Feature Expansions**: Further implementation of the design, as well as connecting it to the software itself. Working with some kind of UI team to help create a page and implement the pages created from fine-tuning. Also working through the implementation of the fine-tuning modules to the interface created will be the next step going forward. We can also add graphs below the log terminal in the UI for better understanding.
   If we could integrate Bert and expand on more encoders

3. Are there any **confusing parts of the UI or workflow** that you think future teams should redesign? It looks good now, simple and not confusing

**Recommendations for Future AI/ET Classes**

These suggestions should focus on improving the class structure, knowledge transfer, or project management process for future teams working on LMForge.

- **Onboarding and Knowledge Transfer**:
- Show up to as many classes as possible (However, don't work yourself sick)
- Don't modify provided hyperparameters more than + or - 50% you will regret it.
- Don't be afraid to ask questions (or for help)
- If LMForge doesn't successfully install the first time, don't worry, it took a lot of trial and error for everyone to get it up and running.
    - Ideally the tutorial video will be updated by then
-

**Teaching & Curriculum Feedback**

- Topics that should be taught earlier to unblock project work (e.g., Docker, APIs, Git workflows, prompt engineering, evaluation).
- Any gaps between the lecture content and what the project actually required.
- Suggestions for labs, mini-projects, or examples that would have helped (e.g., "small example of an end-to-end LLM pipeline").

Having a day to work together as a class