Week 1: Infrastructure and Baseline Setup

Gautam Agarwal - Environment & Data:

- Set up development environment with GPU access and necessary libraries (Transformers, PyTorch/TensorFlow)
- Research and curate STT datasets focusing on difficult cases (background noise, accents, domain-specific vocabulary)
- Prepare data preprocessing pipelines and standardized evaluation sets
- Create version control repository with proper documentation structure

Shivangi - Model Selection & Deployment:

- Evaluate and select pre-trained STT model (Whisper vs Wav2Vec2) based on cost-effectiveness and performance
- Deploy baseline model in inference environment
- Implement basic inference API for real-time transcription
- Document baseline architecture and dependencies

Kavya - Evaluation Framework:

- Implement WER and CER calculation modules
- Set up benchmarking pipeline with latency and throughput measurements
- Create logging infrastructure for tracking model outputs and performance metrics
- Design database schema for storing transcriptions, corrections, and metadata

Week 1 Deliverable: Functional baseline STT system with established performance benchmarks

Week 2: LLM-Based Correction Agent

Team Member 1 - Agent Integration:

- Integrate LLM agent (GPT-4 or similar) for error detection and correction
- Design prompting strategy for linguistic pattern comparison and error identification
- Implement confidence scoring mechanism for agent corrections
- Test agent on sample transcription errors

Team Member 2 - Data Management Layer:

- Build data management system to store failed cases and corrections
- Implement metadata tracking for performance improvements
- Create fine-tuning dataset preparation pipeline
- Set up data versioning and quality control mechanisms

Team Member 3 - Agent Evaluation:

- Develop metrics for correction accuracy and consistency
- Implement false positive detection for agent corrections
- Create ablation testing framework to isolate agent impact
- Benchmark agent latency and runtime performance

Week 2 Deliverable: Functional correction agent integrated with data management pipeline

Week 3: Automated Fine-Tuning Pipeline

Team Member 1 - Hyperparameter Optimization:

- Implement automated hyperparameter optimization library (learning rate, batch size, epochs)
- Integrate parameter-efficient fine-tuning methods (LoRA) to reduce computational costs

- Create hyperparameter search strategies and configuration management
- Test optimization on small-scale fine-tuning runs

Team Member 2 - Fine-Tuning Orchestration:

- Build automated fine-tuning pipeline triggered after accumulating n error cases
- Implement model validation against baseline using standardized evaluation sets
- Create model versioning and deployment system
- Set up regression testing to prevent model degradation

Team Member 3 - Adaptive Scheduling Algorithm:

- Develop adaptive scheduling mechanism that dynamically adjusts threshold n
- Implement performance-aware logic to increase n when accuracy gains diminish
- Create cost-efficiency tracking for computational resource optimization
- Design overfitting prevention strategies with validation monitoring

Week 3 Deliverable: Complete closed-loop system with automated fine-tuning and adaptive scheduling

Week 4: Integration, Testing, and Generalization

Team Member 1 - System Integration & Testing:

- Integrate all components into unified system architecture
- Conduct end-to-end testing of the full feedback loop
- Perform quantitative analysis with paired t-tests for statistical significance
- Run ablation studies to evaluate individual component contributions

Team Member 2 - Generalization Framework:

- Abstract system architecture for modularity and generalization
- Implement secondary application domain (text generation or translation)
- Demonstrate framework's scalability across different generative tasks
- Document generalization patterns and design principles

Team Member 3 - Documentation & Evaluation:

- Compile comprehensive evaluation across all four dimensions (quantitative, efficiency, ablation, qualitative)
- Create visualizations for WER/CER improvements over iterations
- Measure cost per accuracy gain and convergence time metrics
- Prepare final project report with results, limitations, and future work

Week 4 Deliverable: Complete self-learning Agentic AI system with evaluation results and documentation

Week 5: Report Writing

Coordination Guidelines

Weekly Milestones: End-of-week demos to validate deliverables and adjust plans as needed

Shared Resources: Centralized documentation, shared compute resources, and coordinated API usage to manage costs

Risk Management: Early identification of overfitting, resource constraints, and model degradation through continuous monitoring