Initial Project Plan: AI-Powered Code Converter

Date: October 8, 2025

Author: Vitaliy Sviridyuk

This document outlines the initial plan for the AI-Powered Code Converter senior project. It currently reflects the team's current progress as of mid-semester 1 and the strategic decision to focus on the reverse-engineering feedback loop for the remainder of the first semester.

1. Project Goals and Scope Statement

Goals

The primary goal of this project is to design, build, and deploy an experimental application that uses AI/ML to automate the process of converting legacy code to a modern platform. The objective is to create a reusable **process and tool**, not just to perform a one-time conversion.

Scope

In Scope:

- A web-based application for users to initiate and manage conversion projects.
- An automated backend pipeline built on AWS.
- A "sandbox" environment to run automated tests against generated code.
- Complete project documentation for the tool itself.
- Complete documentation of generated modern applications.

Out of Scope:

- Our team manually converting entire legacy applications.
- A production-ready, highly-available system.
- A real-time conversational interface with the AI.

2. Planned Milestones and Major Deliverables

This timeline has been updated to reflect current progress and the refined goals for the end of the first semester from the timeline loosely discussed in previous meetings.

Milestone	Target	Major Deliverables	Status
Project Foundation	Mid-Oct 2025	Finalized Project Plan, Domain Model, and	✅ **Near Complete**

		Metrics. Core AWS infrastructure (VPC, EKS) and a basic, functional backend API.	
Reverse Engineering MVP	Mid-Nov 2025	**(End of Semester 1 Goal)** A functional UI connected to the backend, enabling users to upload legacy code, trigger the AI to generate planning artifacts (specs, user stories), and provide feedback on those artifacts.	🏃 **In Progress**
Code Generation Pipeline	End of Feb 2026	**(Semester 2 Goal)** The backend is extended to generate modernized code. An automated pipeline (EventBridge, Step Functions, CodeBuild) is built to containerize and deploy the generated code to the EKS sandbox.	📝 **Planned**
Feature Complete	Mid-Mar 2026	Full end-to-end functionality. The UI can display sandbox test results, and the user feedback loop for generated code is implemented.	📝 **Planned**
Final Delivery	End of Apr 2026	Final presentation and demo to sponsor, project website complete, final technical report submitted.	📝 **Planned**

3. Initial Work Breakdown Structure (Epics & Stories)

This WBS reflects work completed, in progress, and planned for the near future based on the current Jira board. All epics and stories are tracked in Jira under the ACC project key (e.g., ACC-1, ACC-2).

Epics:

- **Establish Core AWS Infrastructure:** Covers the setup of the VPC, EKS Cluster, ECR, and other core networking components. *(Largely complete)*
- **Develop Backend Service and API Endpoints:** Encompasses the creation of the FastAPI application, its endpoints, and business logic. *(In Progress)*
- **Integrate with Bedrock for Artifact Generation:** Includes all tasks related to prompt engineering and connecting to the Bedrock API. *(In Progress)*
- **Develop and Connect Frontend User Interface:** Covers all frontend development, from initial mockups to a functional, connected UI. *(In Progress)*
- **Build the Post-Generation Pipeline:** Includes the setup of CodeBuild, the EKS Sandbox, and the orchestration logic to connect them. *(Deferred to Semester 2)*

Sample User Stories (Semester 1 Focus):

- (Done) As a developer, I need a basic REST API backbone so that future endpoints can be implemented.
- (Done) As a user, I want to be able to have all my various sessions visible on the side and be able to swap between them
- (Backlog) As a user, I can submit a piece of legacy code via an API call so that the AI generates a functional specification for it.
- (Backlog) As a user, I want the frontend hosted on CloudFront and S3 so I can access it via a public URL.
- (Backlog) As a user, I want to authenticate via Cognito so that access to the application is secure.
- (Backlog) As a user, I can view the generated planning documents in the UI and provide textual feedback on their quality.

4. Communication and Stakeholder Management Plan

Clear and consistent communication is essential for project success.

Stakeholder	Communication Method	Frequency	Purpose
Project Sponsor	Scheduled Video Call & Demo	Weekly	Provide status updates, demonstrate progress, and receive feedback.
Faculty Coach	In-person/Video	Weekly	Discuss progress,

	Meeting		address team dynamics, and resolve roadblocks.
Development Team	Weekly Standup (Jira/Discord)	Weekly	Synchronize on daily tasks and identify blockers.
Development Team	Sprint Planning & Retrospective	Per Sprint (2 weeks)	Plan upcoming work and reflect to improve processes.

5. Risk Management Plan

Our team's process for mitigating risks is to identify, discuss, and track them as part of our regular sprint cycle. Risks will be identified during sprint planning and retrospectives and documented here. For each risk, we will assess its likelihood and potential impact and agree on a proactive or reactive mitigation strategy. The table below serves as a snapshot of the living document of the key risks we are currently tracking.

Risk	**Likelihood**	**Impact**	**Mitigation Strategy**
AI Model Performance is Poor	Medium	High	**Proactive:** Design the backend service to be model-agnostic. **Reactive:** Experiment with different models in Bedrock and invest heavily in prompt engineering. Our "Manual Intervention Rate" metric will track progress here.
AWS Infrastructure Complexity	High	Medium	**Proactive:** Rely on Infrastructure as Code (eksctl YAML) to ensure the setup is repeatable and documented. Dedicate team members to

			become subject matter experts. *(This risk is being successfully managed, as the core EKS setup is complete.)*
Underestimated Task Complexity	High	Medium	**Proactive:** Break down large user stories into smaller tasks. Track sprint velocity to improve future estimations. **Reactive:** The project timeline has already been re-scoped to focus on a realistic Semester 1 goal (the reverse-engineering loop), deferring the code generation pipeline to Semester 2.
Scope Creep	Medium	High	**Proactive:** Maintain a prioritized backlog. All new requests must be framed as user stories and added to the backlog to be prioritized by the Proxy Product Owner, not added mid-sprint.
Team Member Becomes a Blocker	Low	Medium	**Proactive:** Encourage pair programming and regular code reviews. Ensure all work is committed to the central repository and not kept on local machines.