

The Agile Dev Studio

From Concept to MVP

CTE Pathway: Programming & Software Development

School: _____

Teacher: _____

Project Dates: _____

Grade Level: 9–12

Duration: 4–6 weeks

Instructional Model: Project-Based Learning (PBL)

Project Overview

In this project, students work as a junior software development team to solve a real-world digital problem for a community partner. The goal is to transform an initial idea into a functional **Minimum Viable Product (MVP)** using professional workflows and industry-inspired practices.

Students focus not only on writing code, but also on planning, collaboration, documentation, and reflection. The project is fully teacher-guided and designed for mixed-ability classrooms.

What You Will Build

You will design and develop a **working web application** that solves one clearly defined user need.

Examples include:

- A school club sign-up and management tool
- A simple inventory or tracking system
- A basic service request or form-based application

The scope is intentionally limited to ensure depth, clarity, and success.

Final Deliverables

Each team will submit:

- A **functional web application** with a user interface and backend logic
- A **GitHub repository** containing:

- A professional README with setup instructions and technical explanation
 - A clear commit history showing individual contributions
 - A **Kanban board** documenting planning, task ownership, and progress
 - An **individual post-mortem reflection** analyzing challenges, decisions, and growth
-

Project Phases (Sprint Cycle)

Phase 1: Discovery & Design

Students analyze the problem space before writing code.

You will:

- Identify the target user and their needs
- Write simple user stories
- Create wireframes or interface sketches
- Map data inputs, storage, and outputs

Checkpoint (Required): Design review with teacher feedback

Phase 2: Development Sprints

Teams build the application in short, structured sprints.

You will:

- Implement core features incrementally
- Commit work regularly using Git
- Update the Kanban board to track progress
- Participate in peer check-ins and feedback

Checkpoint (Required): Sprint demo and progress review

Phase 3: Quality Assurance (QA)

Students test and improve each other's work.

You will:

- Test features created by another team or teammate
- Identify bugs and usability issues
- Provide constructive technical feedback
- Refine your own work based on testing results

Checkpoint (Required): QA test plan and fixes completed

Phase 4: Demo Day

Teams present their final product.

You will:

- Demonstrate the working application
- Explain design decisions and challenges
- Reflect on team collaboration and workflow

Checkpoint (Required): Final demo and presentation

Team Roles & Collaboration

Teams use **role-based collaboration** to support different strengths and skill levels. Roles may rotate or be combined.

Project Manager

- Organizes tasks and timelines
- Maintains the Kanban board
- Acts as the main point of contact for requirements

Backend Lead

- Implements data logic and server-side functionality
- Manages validation and core features
- Supports integration between components

UX/UI Lead

- Designs the user interface
- Improves usability and accessibility

- Applies basic WCAG accessibility principles

Technical Writer

- Documents setup and usage clearly
 - Writes and maintains the project README
 - Translates technical decisions into user-friendly language
-

Skills You Will Develop

Technical Skills

- Full-stack fundamentals (input, storage, output)
- Guided version control using Git
- System design and data modeling
- Responsible use of AI tools for debugging and documentation

Professional & Career Skills

- Agile-inspired workflows and stand-ups
 - Task management and accountability
 - Technical communication
 - Collaboration and professional reflection
-

Assessment Overview

Your grade reflects **both the product and the process**.

Criteria	Weight
Product Reliability	40%
Professional Workflow (Kanban + Git)	30%
Documentation & Reflection	20%
Collaboration & Participation	10%

AI Use Policy (Student-Facing)

AI tools may be used **as learning supports**, not shortcuts.

Allowed

- Debugging help and error explanations
- Syntax clarification and documentation research
- Drafting small code snippets with understanding

Not Allowed

- Copy-pasting code without understanding
- Using AI to complete entire features independently
- Submitting AI-generated work as original

Required

- Document any AI tool used and its purpose
- Be able to explain AI-assisted code line by line
- Reflect on how AI supported your learning

Teacher Support & Scaffolding

The teacher will provide:

- Starter templates and example project structures
- A Kanban board template
- Checklists and clear weekly expectations
- Ongoing feedback at each checkpoint

This project is designed to balance **structure and student ownership**, ensuring all students can succeed.