Technical Implementation Plan

Phase 1: Project Setup & Core Infrastructure (Weeks 1-2)

1. Environment Setup

- Initialize project with npm/yarn
- Configure webpack/Vite for bundling
- Set up development server with hot-reloading
- Configure ESLint and Prettier for code quality

2. Base Game Engine Setup

- Install and initialize Phaser.js
- Create basic game configuration
- Implement main game loop
- Set up scene manager

3. Offline Infrastructure

- Implement Service Worker for offline caching
- Set up IndexedDB via localForage.js
- Create data storage/retrieval utilities
- Build offline detection system

Phase 2: Core Game Mechanics (Weeks 3-4)

1. Map & Navigation System

- Design and create tilemap for school layout
- Implement collision detection system
- Build camera system that follows player
- Create room transition system

2. Player Systems

- Implement player character with animations
- Create movement controls (keyboard/touch)
- Build interaction system for objects
- Design and implement player inventory

3. Interactive Objects

- Create base interactive object class

- Implement resource objects (books, computers)
- Build door/portal system between rooms
- Create NPC templates for future expansion

Phase 3: Educational Content & Resources (Weeks 5-6)

1. Resource Management System

- Design resource data structure
- Implement resource loader for different types
- Create resource viewer UI
- Build progress tracking system

2. Quiz System

- Design quiz data structure
- Implement quiz UI
- Create scoring and feedback system
- Build progress tracking for quizzes

3. Content Creation Tools

- Create content authoring utilities
- Build content validation tools
- Implement content packaging system
- Create content import/export functionality

Phase 4: UI & Experience (Weeks 7-8)

1. User Interface

- Design and implement main menu
- Create HUD for gameplay
- Build settings/options menu
- Implement help/tutorial system

2. Chat System

- Design chat data structure
- Implement local chat storage
- Create chat UI components
- Build message processing system

3. Progress & Achievement System

- Design achievement system
- Implement progress visualization
- Create reward/feedback mechanics
- Build student dashboard

Phase 5: Data Synchronization & Refinement (Weeks 9-10)

1. Online Synchronization

- Design sync protocol
- Implement sync manager
- Create conflict resolution system
- Build sync UI and notifications

2. Performance Optimization

- Analyze and optimize asset loading
- Implement lazy loading for resources
- Optimize rendering performance
- Reduce memory usage

3. Testing & Refinement

- Conduct user testing
- Fix bugs and issues
- Refine user experience
- Optimize for different devices

Phase 6: Deployment & Documentation (Weeks 11-12)

1. Packaging & Deployment

- Create production build pipeline
- Implement versioning system
- Build deployment automation
- Create update mechanism

2. Documentation

- Create developer documentation
- Write user guides
- Document content creation process
- Create API documentation

3. Handover & Launch

- Conduct final testing
- Prepare launch materials
- Train administrators
- Official release