Case Study

Spec-Driven AI Development for a Healthcare Education Assessment Platform

Magic EdTech

The Client

The client is a leading provider of healthcare education and assessment tools for nursing and healthcare learning programs.

The Challenge

The client needed to rebuild its legacy Custom Assessment Builder using a modern, modular architecture while preserving complex feature logic. Key workflows, such as case study authoring and delivery, were initially estimated at 80-90 story points due to their complexity. The team also needed to use AI development tools effectively without increasing token costs or introducing unreliable AI-generated outputs.

Critical Success Parameters

- ✓ Reduce story-point effort for complex feature development.
- ✓ Build a modular, reusable application architecture.
- ✓ Eliminate fragmented code and reduce maintenance overhead.
- ✓ Provide AI agents with clear, structured feature context.
- ✓ Reduce AI hallucinations and unnecessary token usage.
- ✓ Ensure accessibility compliance from the start of development.
- ✓ Maintain plain-English feature specifications that technical and non-technical teams could review and update.

Our Approach

- ✓ Created plain-English specification files for each feature to define logic, expected behavior, and development context.
- ✓ Used spec files as persistent context for AI agents to improve output accuracy and reduce hallucinations.
- ✓ Rebuilt the application using a modular micro-frontend architecture for reusable, plug-and-play components.
- ✓ Standardized Angular-based development patterns to ensure fixes and changes could be applied across shared components.
- ✓ Created accessibility guideline files and fed them into AI workflows to support compliance during development.
- ✓ Enabled product owners and managers to review and refine feature logic directly through readable specification files.

Key Result Highlights

Reduced development effort by over **50%** compared with initial estimates.

Lowered AI token consumption through concise, feature-specific context files.

Improved AI-generated code accuracy by grounding outputs in approved specifications.

Modernized the legacy application using a reusable micro-frontend architecture.

Reduced duplicate code and improved maintainability across application views.

Built accessibility requirements directly into AI-assisted development workflows.

Improved collaboration between developers, product owners, and managers through plain-English specifications.