ProJor business use-case

Company Background

Alpha-Omega Inc. develops **Hospital Information System** (HIS) software for state-owned hospitals and private healthcare providers. They integrate external data sharing with health insurance companies and pharmaceutical suppliers, based on client needs. The company has heavily invested in healthcare software standard compliance, ensuring rigorous documentation and testing processes.

Their Vision

Alpha-Omega Inc. identified a major bottleneck: each change in system requirements caused significant overhead for the development team.

They aim to become more agile, experimenting freely with new technologies while maintaining full compliance with healthcare software standards.

Their engineering team envisions a **single-source-of-truth** approach, enabling them to apply domain expertise efficiently without sacrificing traceability or system documentation.

Current Environment

The company employs dedicated software architects and test engineers responsible for recording requirements in a standard-compliant way. For every new requirement or modification, a chain of documents is created to maintain traceability and validity.

Substantial resources are invested into verifying the developed system against these design documents before every release.

Business Requirements

- Reduce time-to-market for new features
- Enable flexible technology stack adoption
- Maintain full compliance with healthcare software standards
- Increase system-wide security and stability

Technical Requirements

- Use a model-as-code tool to define system components
- Maintain references to input requirements across all model parts
- Establish a single-source-of-truth for system-managed resources
    - Define database schemas based on resources
    - Define CRUD APIs based on resources
    - Define client libraries and UI components based on resources
    - Define messaging schemas based on resources
- Generate standard-compliant documentation automatically
- Minimize manual boilerplate testing work
- Generate automated test cases

Application of ProJor

ProJor introduces modeling at the requirements level: requirements themselves become model elements.

From these, a **resource model** is created, maintaining references to specific requirements at every level (resource, attribute, association).

From the resource model, ProJor automatically generates:

- ORM entity code
- CRUD controller code
- Client libraries
- User Interface components

Documentation-as-code tools (e.g., Markdown, LaTeX, PlantUML) are used to generate standard-compliant documentation from the same model.

When requirements change, updating the model elements triggers regeneration across all software and documentation layers.

This eliminates redundant manual work and dramatically reduces time-to-market. As a result, development costs decrease and customer demands are realized faster.

Furthermore, by automating as many layers as possible, system stability and security are increased. Each generated layer can be replaced independently by adjusting the code templates, enabling flexible adoption of new technologies without losing model consistency.

Summary

By adopting ProJor, Alpha-Omega Inc. achieves substantial long-term cost savings without compromising stability, security, or compliance with healthcare software standards.

The one-time investment in code and documentation template creation pays off immediately: new features reach customers faster, allowing for more iterations within the same timeframe.

The ability to seamlessly swap technology stacks further future-proofs their HIS platform — a capability previously unattainable.