AI-Powered Development POC

Objective

- Showcase how AI-assisted development accelerates delivery
- Build a working prototype of Brands Hub using Cursor AI
- Validate feasibility of integrating AI into UI, backend, and APIs

What is the POC?

- Home Page: Ticket widget listing tickets
- APIs: Serve ticket data dynamically
- All implemented using Cursor AI with minimal manual effort

How the POC Was Achieved

- Fed the plain layout image to stitch.withgoogle
- Achieved business requirement designs in stitch.withgoogle using prompt
- Sent the final home page design from stitch.withgoogle to Cursor A along
- Tuned the requirements with more prompts in Cursor AI and achieved the result
- Hosted the final UI to GitHub Pages by auto-generating GitHub Action code

Tech Stack

- Frontend (UI): Next.js
- Backend (APIs): Spring Boot
- AI Tool: Cursor AI (latest model)
- Database: Dummy / Example DB

AI Model Details

- Cursor AI (latest model) used for frontend & backend code generation
- API contracts and test case generation
- Documentation & architectural suggestions
- Reduced manual effort by ~70–80%

Implementation Highlights

- AI-generated UI components & APIs
- Auto-generated documentation & test cases
- End-to-end working prototype in 2–3 days vs 8–10 days manually

Pros of AI-Assisted Development

- No manual writing of code, including test cases
- Quickly find solutions without searching multiple websites
- AI suggests optimized code, reducing manual performance tuning
- Promotes best practices and modern standards
- Helps in designing project structure and making architectural decisions
- Enables easy comparison of frameworks, tools, and plugins
- Highlights potential design flaws or anti-patterns early
- Makes documentation easier with auto-generated summaries

Cons / Challenges

- AI-generated code may need review & refactoring
- Potential data leaks if prompts include sensitive info
- Can reduce developers' deep understanding of the code
- AI-generated logic may be harder to debug
- May require extra effort to enforce org coding guidelines
- Enterprise AI tools and API usage can add recurring expenses
- AI may misinterpret incomplete or ambiguous requirements
- Models may suggest outdated libraries or incompatible APIs

Time & Effort Comparison

- Traditional development: ~8–10 days
- AI-assisted development: ~2–3 days
- Coding: 1–2 hrs vs 6–7 hrs
- Documentation: 15 mins vs 1 hr
- Testing & Review: 45 mins vs 1–2 hrs

Next Steps

- Expand to automated test generation & CI/CD
- Evaluate other AI tools (Copilot, Codeium, etc.)
- Define AI governance & security policies (DLP, data privacy)
- Plan controlled rollout in more projects

Q&A