

Automaspec

AI-Powered Test Specification & Automation

Student: Roman Radchenko (Group JS-22)

Supervisor: Volha Kuzniatsova

Date: January 7, 2026

“  **One platform to manage all your test specs, track coverage, and generate requirements with AI – instead of juggling Jira, Confluence, and Slack.** ”

1. The Problem & Business Impact

The Problem

- Docs and code get out of sync over time
- Too many sources of truth: Jira, Confluence, Slack
- Test results hidden in CI/CD logs

Business Impact

- ✗ Slow time-to-market
- ✗ Increased regression risks
- ✗ Team frustration from manual sync
- ✗ Poor visibility into test coverage

2. Goal & Scope

Goal: Build a unified platform for QA teams to:

- Store all test specs in one place
- Track test status from CI/CD in real-time
- Use AI to help write specs faster

Scope: Web app with auth, multi-org support, GitHub Actions sync

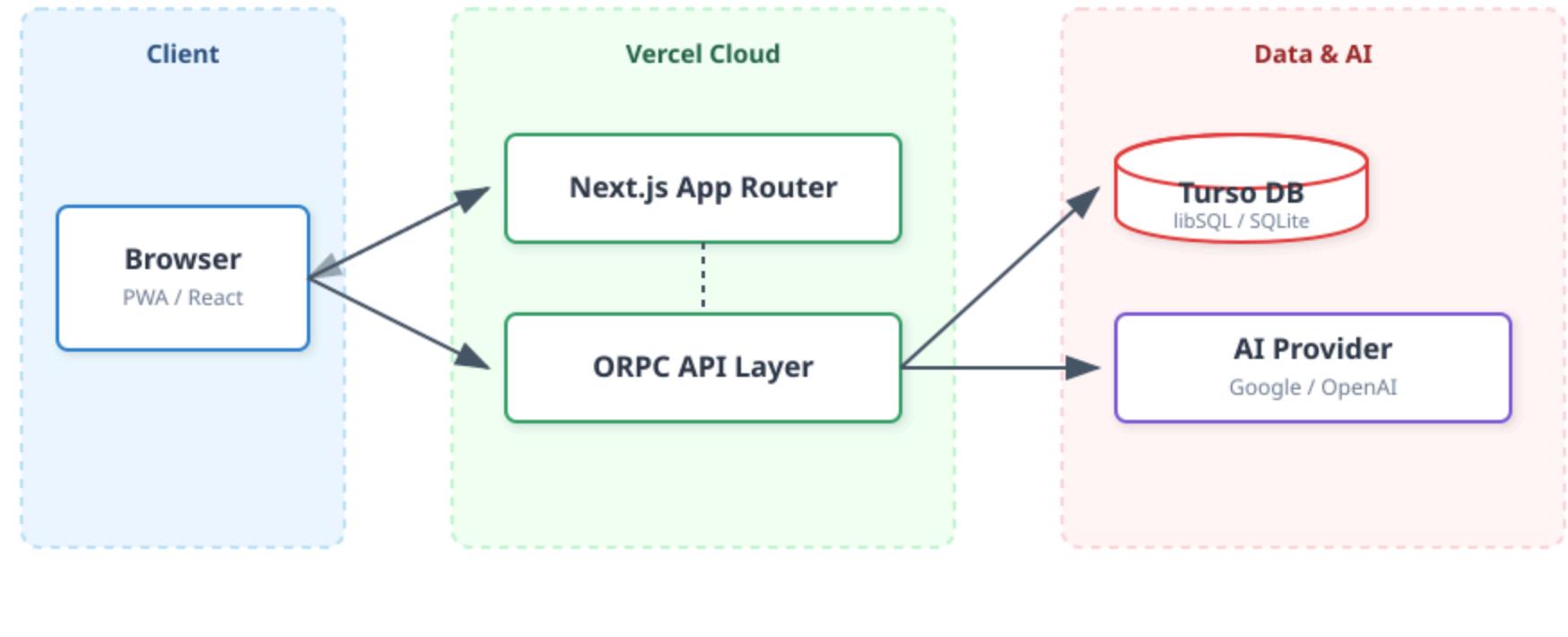
3. The Solution: Unified QA Engine

- **Single Source of Truth:** Centralized specification management.
- **Context-Aware AI:** LLM fed with real project requirements.
- **Live CI/CD Sync:** Status mapped directly to business specs.
- **Multi-Tenant:** Secure isolation for multiple organizations.

4. Tech Stack

Layer	Technology	Purpose
Frontend	Next.js 16, React 19, TailwindCSS v4	App Router, modern UI
UI Components	Shadcn/ui	Accessible, customizable components
Backend API	oRPC	Contract-first, type-safe RPC
ORM	Drizzle	Type-safe SQL queries
Database	Turso	Distributed SQLite, edge-ready
Auth	Better Auth	Session management, RBAC
AI	Vercel AI SDK	LLM integration (OpenRouter, Gemini)
Hosting	Vercel	Serverless deployment
CI/CD	GitHub Actions	Automated testing & deployment

5. High-Level Architecture



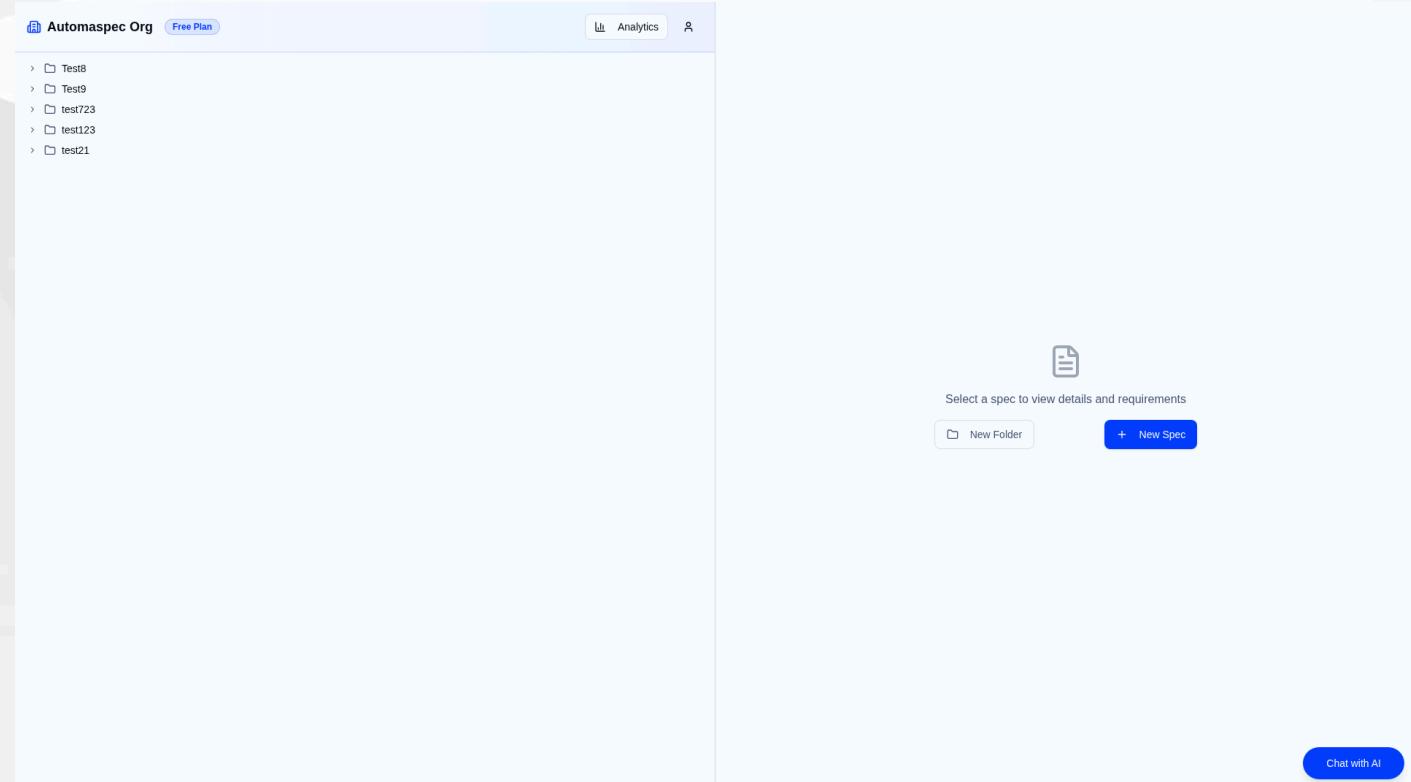
6. Demo: Home & Authentication

- **Landing Page:** Professional overview of capabilities.
- **Auth Flow:** Secure login via Better Auth.
- **Organization:** Seamlessly switch between workspaces.

The screenshot shows the AutomaSpec landing page. At the top, there's a navigation bar with links for Features, Pricing, Go to App (which is highlighted in blue), and DE. Below the navigation, a banner says "Supporting Vitest" and features the heading "Testing Made Simple". A subtext explains: "Transform your test specifications into executable code. Manage, run, and track your tests with a modern dashboard built for developers, testers, and managers." Below this is a "Go to App" button. The main section is titled "Everything you need for testing" with the subtext: "From test specification to execution, manage your entire testing workflow in one place." It lists six features in cards: "Code Generation" (with a code icon), "Real-time Execution" (with a lightning bolt icon), "Role-based Access" (with a shield icon), "Team Collaboration" (with a people icon), "Git Integration" (with a git icon), and "Cross-browser Testing" (with a globe icon). Each card has a brief description below it.

7. Demo: Testing Dashboard

- **Hierarchy:** Navigate folders and specs with ease.
- **AI Side Panel:** Create specifications and organize requirements with AI assistance.
- **Live Status:** Real-time results from GitHub Actions.



8. Analytics Dashboard

- **Test Status Distribution:** Visual breakdown of passed/failed/pending tests.
- **Coverage Metrics:** Track test coverage across specifications.
- **Trend Analysis:** Monitor testing progress over time.

The screenshot displays the 'Analytics Dashboard' for the 'Automaspec Org' account. The top navigation bar includes links for 'Dashboard', 'Automaspec Org', and 'Analytics'. The main dashboard area features several key metrics and data cards:

- Total Tests:** 0 (All tests in organization)
- Requirements:** 7 (Total requirements defined)
- Specifications:** 20 (Test specifications)
- Team Members:** 1 (Active organization members)

Below these, there are two main sections:

- Tests Growth:** New tests created over time. Message: "No test growth data available for this period".
- Test Status Distribution:** Tests grouped by their current status. Message: "No test status data available".

At the bottom, there is a section titled "⚠ Stale Tests" which lists specifications that have not been updated in the last 30 days:

Specification Name	Last Updated
New Test	about 1 month ago
New Test	about 1 month ago
New Test	about 1 month ago

9. Business Analysis

Key User Stories Implemented

Epic	User Story	Priority
Auth & Orgs	Sign up, create organization, invite team members	Must
Test Hierarchy	Create nested folders, specs, requirements, drag-and-drop	Must
AI Assistant	Create specs with AI, get structure suggestions	Must
Status Tracking	View test status (passed/failed/pending), aggregated stats	Must
CI/CD	Connect GitHub repo, automatic test result sync	Must

KPIs: ↓ 20-30% test creation time • ↑ 40% coverage visibility • 80% adoption in 2 months

10. Backend Architecture

Layered Architecture



Key Benefits

- **Type Safety:** Contract-first with Zod validation
- **Separation of Concerns:** Clear layer boundaries
- **Middleware:** Auth, logging, error handling
- **Scalability:** Modular route structure
- **Auto Docs:** Generated API documentation

11. Database Engineering

Design: 3NF • Multi-tenant isolation • Self-referential folders • Turso (distributed SQLite) • Drizzle ORM

Table	Purpose	Source
user	User accounts	Better Auth
session	Active sessions	Better Auth
account	OAuth providers	Better Auth
verification	Email verification	Better Auth
organization	Multi-tenant root	Better Auth
member	Org membership & roles	Better Auth
invitation	Team invites	Better Auth
test_folder	Hierarchical folders (self-ref)	Custom
test_spec	Test specifications	Custom
test_requirement	Requirements per spec	Custom
test	Individual test cases	Custom

12. Testing Strategy

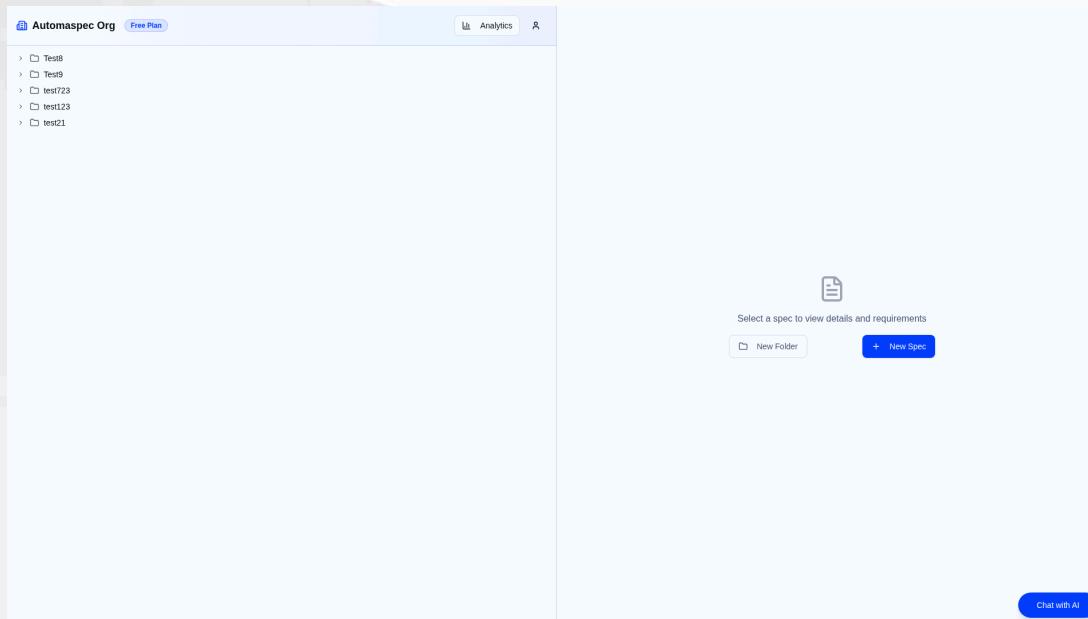
- **Quality Gates:** ≥70% coverage enforced in CI.
- **Playwright E2E:** Critical flow (Auth, Tree Ops) validation.
- **Vitest:** Logic testing & oRPC procedure verification.

Result: A self-documented, high-reliability platform.

13. AI Assistant

How it works

- AI helps write specs & requirements
- Streaming: see results in real-time
- Context-aware suggestions
- Multiple LLM providers (OpenRouter, Gemini)
- Prompt engineering for quality output



14. Technical Challenges

Challenge	Solution	Impact
AI Accuracy	Structured context injection	High quality suggestions
Hierarchy	Self-referential Drizzle schemas	Unlimited nesting
CI/CD Sync	Secure webhook integration	Real-time status
Multi-Tenancy	Organization-level isolation	Secure data separation

15. Results & Future Work

For Users

- Less sources of truth for test documentation
- Tracking finalized requirements in one place
- AI-assisted specification creation
- Real-time CI/CD status tracking
- Multi-organization support

Technical Achievements

- **70%+** Code Coverage
- **5 Criteria** Implemented
- **Production Ready:** automaspec.vercel.app
- Type-safe end-to-end architecture
- Comprehensive API documentation

16. Q&A

Roman Radchenko (JS-22)

- **Repo:** github.com/automaspec/automaspec
- **App:** automaspec.vercel.app
- **Docs:** </rpc/docs>