Frontend: React + TypeScript + Vite + Tailwind CSS

Testing: Vitest + React Testing Library + Cypress

Code Quality: ESLint + Prettier + Husky pre-commit hooks

Build Tool: Vite with hot reload

Package Manager: npm

1. Strong Testing Infrastructure:

- Unit tests with Vitest + React Testing Library

- Integration tests setup

- Pre-commit test validation

- Coverage reporting (80% minimum)

2. Error Handling System:

- Structured error codes (1000-1899 range)

- Toast notification system for user feedback

- ApiErrorHandler class for consistent error processing

- User-friendly error messages (no raw technical errors)

3. Code Quality Enforcement:

- 100% TypeScript implementation

- ESLint with 15+ automated rules

- Prettier for consistent formatting

- Husky pre-commit hooks preventing bad code

- Line length limits (80 characters)

4. Performance & Scalability:

- Component-based architecture

- Clean folder structure with separation of concerns

- Ready for Redis caching integration

- Microservices-ready design

5. CI/CD Pipeline:

- GitHub Actions workflow configured

- Automated testing before deployment

- Security scanning setup

- Bundle analysis included

Structure:

├── src/

│ ├── components/

│ │ └── AuthCard.tsx # Reusable auth container

│ ├── pages/

│ │ ├── Login.tsx # Login form with validation

│ │ └── Register.tsx # Registration with API integration

│ ├── contexts/

│ │ └── ToastContext.tsx # Global notification system

│ └── utils/

│ ├── FormValidator.ts # Comprehensive validation rules

│ └── ApiErrorHandler.ts # Structured error handling

Design Specifications:

- Container: 750px × 400px, white background, rounded-xl, shadow-lg

- Left Panel: 250px, yellow background (#FBC02D), white circle (100px), black text

- Right Panel: 500px, white background, 40px padding, auto-expanding height

- Responsive design with min-h-[400px] to prevent overflow

Color Scheme:

- Primary: Yellow (#FBC02D) for backgrounds and accents

- Secondary: Blue (#1E88E5, blue-600) for buttons and highlights

- Text: Black for primary text, white for button text

- Error: Red (#E53935) for validation messages

- Background: Light gray (#F5F6F7) for page background

Button Styling:

- Auto-sizing buttons with px-6 py-3 padding

- Fully rounded edges (rounded-full)

- Minimum width fitting content (min-w-fit)

- Hover effects with smooth transitions

- Consistent styling across all forms

Form Elements:

- Input fields: 100% width, 48px height, rounded borders

- Focus states: Blue ring (focus:ring-blue-600)

- Error handling: Real-time validation with red error messages

- Spacing: 20px gap between form elements (space-y-5)

1. Authentication System:

✅ Login page with email/password validation

✅ Registration page with password confirmation

✅ API integration (POST to http://localhost:5000/api/register)

✅ Protected routes with authentication guards

✅ Form validation with real-time error feedback

✅ Success/error toast notifications

2. Reusable Components:

✅ AuthCard component for consistent auth UI

✅ ToastContainer for global notifications

✅ FormValidator utility for input validation

✅ ProtectedRoute component for route guarding

3. Navigation & Routing:

✅ React Router setup with BrowserRouter

✅ Routes: /, /login, /register, /dashboard

✅ Automatic redirects based on authentication state

✅ Clean URL structure

4. Development Workflow:

✅ Automated code formatting on save

✅ Pre-commit hooks with quality checks

✅ Real-time linting and error detection

✅ Hot reload development server

Automated Checks:

- npm run quality # Complete quality pipeline

- npm run lint:fix # Auto-fix code issues

- npm run format # Format all code

- npm run type-check # TypeScript validation

- npm run test # Run test suite

- npm run test:coverage # Coverage reporting

Pre-commit Validation:

- TypeScript compilation check

- ESLint error detection

- Prettier formatting

- Test suite execution

- Coverage threshold validation

Continue developing this Petrol Pump Management System with the following context:

ARCHITECTURE: React + TypeScript + Vite + Tailwind CSS with automated development guidelines

AUTHENTICATION: Complete login/register system with AuthCard component and form validation

STYLING: Yellow primary theme, blue accents, auto-sizing buttons with rounded-full edges

QUALITY: Automated testing, linting, formatting with pre-commit hooks

ERROR HANDLING: Toast notifications with structured error codes (1000-1899)

STRUCTURE: Component-based with clean separation of concerns

CURRENT STATE:

- Authentication system fully functional

- Development guidelines automated and enforced

- UI/UX consistent with professional design

- Code quality standards at 100% compliance

- Ready for dashboard and business logic implementation

NEXT STEPS:

- Implement dashboard with petrol pump management features

- Add transaction handling and reporting

- Integrate with backend API for full functionality

- Add role-based access control (Admin vs Manager)

- Implement Redis caching for performance optimization

DESIGN STANDARDS TO MAINTAIN:

- Auto-sizing buttons with px-6 py-3 padding and rounded-full edges

- 750×400px containers with yellow left panels and white right panels

- Consistent color scheme: yellow primary, blue secondary, black text

- Form validation with real-time error feedback

- Toast notifications for all user interactions

- Minimum 80% test coverage requirement

# Start development

npm run dev

# Quality assurance

npm run quality

npm run lint:fix

npm run format

npm run type-check

# Testing

npm run test

npm run test:coverage

npm run test:ui

# Pre-commit validation

npm run pre-commit