

Perplexity VSCode Extension - Detaillierter Projektplan

Phase 1: Projektvorbereitung & Setup (Woche 1-2)

Woche 1: Projektinitialisierung

- **Arbeitsstage:** 5 Tage
- **Geschätzte Stunden:** 40h
- **Verantwortlich:** André + Partner

Tag 1-2: Technische Grundlagen

- **Extension Scaffold erstellen (4h)**
 - Yeoman Generator verwenden: ``yo code``
 - TypeScript Template konfigurieren
 - Basic ``package.json`` und ``tsconfig.json`` Setup
- **Development Environment Setup (4h)**
 - VSCode Extension Development Host konfigurieren
 - Jest Testing Framework integrieren
 - ESLint + Prettier Setup
 - Git Repository initialisieren

Tag 3-4: API Integration Grundlagen

- **Perplexity API Client Prototype (6h)**
 - API Key Management implementieren
 - Basic HTTP Client mit ``axios``
 - Rate Limiting Logik
 - Error Handling Framework
- **MCP Server Grundgerüst (6h)**
 - ``@modelcontextprotocol/sdk`` installieren

- Basic Server Setup nach offizieller Dokumentation
- Tool Registration System
- VSCode Integration Testing

Tag 5: Project Management Setup

- **GitHub Repository Setup (2h)**
 - Issues und Milestones erstellen
 - GitHub Actions für CI/CD vorbereiten
 - Branch Protection Rules
- **Dokumentation Framework (2h)**
 - `README.md` erstellen
 - API Documentation Setup
 - Changelog Template

Deliverables Woche 1:

- ✓ Funktionierendes Extension Scaffold
- ✓ Basic Perplexity API Integration
- ✓ MCP Server Prototype
- ✓ Testing Framework funktional
- ✓ CI/CD Pipeline configured

Woche 2: Core Architecture Implementation

- **Arbeitsstage:** 5 Tage
- **Geschätzte Stunden:** 40h
- **Verantwortlich:** André (Backend) + Partner (Frontend Setup)

Tag 1-2: Backend Services

- **PerplexityClient vervollständigen (6h)**

- Multi-Model Support (sonar-pro, sonar-medium)
- Caching Layer implementieren
- Request/Response Validation
- Comprehensive Error Handling
- ****SecurityManager implementieren (4h)****
- API Key Encryption mit VSCode `SecretStorage`
- Input Validation Framework
- Permission System Grundlagen

Tag 3-4: Frontend Grundlagen

- ****React Webview Setup (6h)****
- Create React App Integration in VSCode Webview
- `@vscode/webview-ui-toolkit` Integration
- Tailwind CSS Konfiguration
- Basic Component Architecture
- ****Webview Communication (4h)****
- Message Passing zwischen Extension Host und Webview
- State Management Setup (Context API)
- Error Boundary Implementation

Tag 5: Integration & Testing

- ****End-to-End Integration (4h)****
- Extension Host ↔ Webview Communication testen
- API Calls von UI triggern
- Basic Error Handling
- ****Testing Suite erweitern (4h)****
- Unit Tests für Core Services
- Integration Tests für API Client
- Webview Testing Setup

****Deliverables Woche 2:****

- ✓ Vollständiger PerplexityClient mit allen Features
- ✓ React Webview mit grundlegender UI
- ✓ Sichere API Key Verwaltung
- ✓ Funktionierende Extension-Webview Kommunikation
- ✓ Umfassende Test Coverage (>80%)

Phase 2: MVP Development (Woche 3-6)

Woche 3: Chat Interface & Basic Search

- ****Arbeitsstage:**** 5 Tage
- ****Geschätzte Stunden:**** 40h
- ****Split:**** André (Backend 24h) + Partner (Frontend 16h)

Backend Tasks (André)

- ****ChatProvider Implementation (8h)**:** Message History, Context Window, Multi-turn Conversation.
- ****SearchProvider Enhancement (8h)**:** Advanced Search, Source Citation Handling, Follow-up Questions.
- ****ContextManager Development (8h)**:** Workspace Analysis, Active File Context, Git Info.

Frontend Tasks (Partner)

- ****ChatInterface Component (8h)**:** Markdown Support, Input Field, Loading States.
- ****ResultsRenderer Component (8h)**:** Rich Content Display (Text, Code), Source Citations UI.

****Deliverables Woche 3:****

- ✓ Funktionale Chat-Oberfläche
- ✓ Perplexity Search Integration
- ✓ Workspace Context Analysis
- ✓ Message History Persistence

Woche 4: MCP Server & Agent Mode

- ****Arbeitsstage:**** 5 Tage
- ****Geschätzte Stunden:**** 40h
- ****Focus:**** Agent Mode Integration

MCP Server Tools Implementation

- ****PerplexitySearchTool (8h)**:** Schema, Validation, Execution.
- ****WorkspaceAnalysisTool (8h)**:** File System Analysis, Symbol Extraction, Dependency Detection.
- ****CodeExplanationTool (8h)**:** Code Context Extraction, Syntax Highlighting, Explanation Generation.

VSCode Agent Mode Integration

- ****Agent Mode Compatibility (8h)**:** VSCode Copilot Chat Integration, Tool Registration.
- ****Testing & Debugging (8h)**:** End-to-End Testing, Performance Optimization.

****Deliverables Woche 4:****

- ✓ Vollständiger MCP Server mit 3 Core Tools
- ✓ VSCode Agent Mode Integration
- ✓ Tool Permission System
- ✓ Comprehensive Tool Testing

Woche 5: Advanced UI Components

- **Arbeitsstage:** 5 Tage
- **Geschätzte Stunden:** 40h
- **Focus:** User Experience Enhancement

Advanced UI Development

- **SettingsPanel Component (10h):** Configuration UI, API Key Wizard, Model Selection.
- **ToolsPanel Component (10h):** Available Tools Display, Execution History.

UX Improvements

- **Theme Integration (8h):** VSCode Theme Compatibility (Dark/Light Mode).
- **Responsive Design & Accessibility (8h):** Multi-Panel Layout, Keyboard Navigation.
- **Performance Optimization (4h):** Lazy Loading, Virtual Scrolling.

Deliverables Woche 5:

- ✓ Professionelle Settings-Oberfläche
- ✓ Tool Management Interface
- ✓ Vollständige Theme Integration
- ✓ Responsive und accessible UI

Woche 6: Testing & Debugging

- **Arbeitsstage:** 5 Tage
 - **Geschätzte Stunden:** 40h
 - **Focus:** Quality Assurance
-
- **Comprehensive Testing (24h):** Unit Test Coverage (>90%), Integration Testing (End-to-End Workflows).

- ****Bug Fixes & Optimization (16h)****: Performance Profiling, Security Audit (Input Validation, XSS).

****Deliverables Woche 6:****

- ✓ 90%+ Test Coverage
- ✓ Performance Benchmarks erfüllt
- ✓ Security Audit bestanden
- ✓ Bug-freie MVP Version

Phase 3: Advanced Features (Woche 7-10)

(Details zu Woche 7-10 gekürzt, folgen dem gleichen Muster wie oben)

Phase 4: Production Readiness (Woche 11-12)

(Details zu Woche 11-12 gekürzt, folgen dem gleichen Muster wie oben)

Optionale Erweiterungsphase (Woche 13-16)

- ****Woche 13-14: Advanced Features****: Team Collaboration, SSO Integration.
- ****Woche 15-16: Community & Growth****: GitHub Community, Documentation Website, Tutorials.

Ressourcenplanung

Personeller Aufwand

- **André (Lead Developer):** Backend (60%), Architecture (20%), QA (15%), Management (5%).
- **Partner (Frontend Developer):** UI/UX (70%), Testing (20%), Documentation (10%).
- **Optional: Third Developer:** Documentation, Testing, DevOps.

Technische Ressourcen

- **Development Tools:** VSCode, Node.js, Git, Figma.
- **Services & APIs:** Perplexity API Credits (~€50/Monat), GitHub Actions, Sentry.

Budget Breakdown (Professional Edition)

- **Development Costs:** €35.000 - €50.000
- **Laufende Kosten (monatlich):** ~€50-€100

Risikomanagement

Technische Risiken

- **Risiko:** Perplexity API Änderungen (Mittel) → **Mitigation:** Multi-Provider Architecture.
- **Risiko:** VSCode API Breaking Changes (Niedrig) → **Mitigation:** API Version Pinning.
- **Risiko:** Performance Probleme (Mittel) → **Mitigation:** Frühzeitige Performance Tests.

Business Risiken

- **Risiko:** Perplexity entwickelt eigene Extension (Hoch) → **Mitigation:** First-Mover Advantage, differentiated Features.

- **Risiko:** Niedrige User Adoption (Mittel) → **Mitigation:** Community Building.
- **Risiko:** Konkurrenz (Hoch) → **Mitigation:** Superior UX, Agent Mode Focus.

Success Metrics

MVP Success Criteria

- ✓ 1.000 Downloads in ersten 30 Tagen
- ✓ 4.0+ Sterne Rating im VSCode Marketplace
- ✓ <500ms durchschnittliche Response Zeit

Growth Targets (6 Monate)

- 📈 10.000+ aktive Nutzer
- 📈 4.5+ Sterne Rating
- 📈 100+ GitHub Stars

Long-term Vision (12 Monate)

- 📈 50.000+ Downloads
- 📈 Premium Feature Adoption: 10%
- 📈 Enterprise Customers: 5+

Nächste Schritte

Sofort umsetzbar (diese Woche)

1. GitHub Repository erstellen.

2. Development Environment Setup.
3. Perplexity API Account einrichten.
4. Project Board in GitHub erstellen.

Kurz-/mittelfristig (nächste 2 Wochen)

1. Extension Scaffold erstellen.
2. Basic Perplexity API Integration.
3. MCP Server Prototype aufsetzen.
4. First Working Demo für Stakeholder.

Projektplan Version: 1.0 | Erstellt am: 2. Oktober 2025 | Nächste Review: 16. Oktober 2025 | Projektleitung: André