Product Requirements Document: MCP Hub Platform

Version: 1.0

Date: June 6, 2025

Product: MCP Hub - Centralized Model Context Protocol Server Management Platform

---

Executive Summary

The MCP Hub Platform is a centralized marketplace and management system for Model Context Protocol (MCP) servers that enables developers and organizations to discover, deploy, and manage AI-powered integrations through a unified interface. Built with Supabase backend and Google OAuth authentication, the platform addresses the growing need for standardized AI tool integration while maintaining security and user control.

Problem Statement

Current Challenges

1. **Fragmented MCP Ecosystem**: With 500+ MCP servers available, developers struggle to discover and integrate relevant servers for their use cases
2. **Complex Server Management**: Managing multiple MCP servers requires technical expertise and manual configuration
3. **No Centralized Discovery**: Lack of a unified marketplace for browsing, evaluating, and installing MCP servers
4. **Authentication Complexity**: Each MCP server may require different authentication mechanisms
5. **Monitoring Gaps**: No unified monitoring and health management across multiple MCP servers

Market Opportunity

- **Target Market Size**: 50,000+ AI developers and 10,000+ enterprises adopting MCP

- **Growth Rate**: 300% YoY growth in MCP server adoption
- **Revenue Potential**: $10M ARR within 24 months through premium features and enterprise plans

Product Vision

"The App Store for AI Integrations" - Create the definitive platform where developers can discover, deploy, and manage AI-powered integrations through a simple, secure, and scalable interface.

Target Users

Primary Users

1. **AI Application Developers**
2. Building Claude Desktop extensions, VS Code plugins, or custom AI applications
3. Need to integrate multiple data sources and tools
4. Want simplified server management and monitoring
5. **Enterprise Development Teams**
6. Implementing AI workflows across Google Workspace, Slack, GitHub, etc.
7. Require centralized management and security controls
8. Need audit trails and compliance features
9. **MCP Server Publishers**
10. Companies and developers creating MCP servers
11. Want to reach a broader audience
12. Need analytics and usage insights

Secondary Users

1. **System Administrators**
2. Managing enterprise AI infrastructure
3. Responsible for security and compliance
4. **Business Users**
5. Non-technical users who benefit from AI integrations
6. Need simple interfaces to configure AI workflows

Core Features

1. MCP Server Marketplace

1.1 Server Discovery

- **Categorized Browsing**: Browse servers by category (Productivity, Development, Data, Communication, etc.)
- **Search & Filtering**: Advanced search with filters for authentication type, pricing, ratings, compatibility
- **Server Profiles**: Detailed pages with descriptions, capabilities, documentation, and user reviews
- **Trending & Recommendations**: Algorithm-driven suggestions based on user behavior and popular combinations

1.2 Server Information

- **Capability Matrix**: Clear display of supported tools, resources, and prompts
- **Authentication Requirements**: OAuth scopes, API keys, or other auth methods needed
- **Compatibility**: Supported MCP client versions and transport protocols
- **Pricing**: Free, freemium, or paid tiers with transparent pricing

1.3 Installation & Configuration

- **One-Click Install**: Automated server deployment and configuration
- **Configuration Wizard**: Step-by-step setup for complex servers requiring API keys or OAuth
- **Bulk Installation**: Install multiple related servers as packages
- **Version Management**: Update servers and rollback to previous versions

2. Centralized Server Management

2.1 Server Lifecycle

- **Start/Stop Control**: Individual server management with status monitoring
- **Health Monitoring**: Real-time status, uptime tracking, and error reporting
- **Auto-Recovery**: Automatic restart of failed servers with configurable retry policies
- **Resource Usage**: Monitor CPU, memory, and network usage per server

2.2 Configuration Management

- **Unified Configuration**: Manage all server configurations from a single interface

- **Environment Variables**: Secure storage and management of API keys and secrets
- **Configuration Templates**: Pre-built configurations for common use cases
- **Configuration Sync**: Sync configurations across development, staging, and production

2.3 Authentication Hub

- **OAuth Centralization**: Single Google OAuth flow for all compatible servers
- **Credential Management**: Secure storage of API keys, tokens, and certificates
- **Permission Management**: Granular control over server access and capabilities
- **Token Refresh**: Automatic token renewal and expiration management

3. Unified API Gateway

3.1 Single Endpoint Access

- **Unified REST API**: Access all connected servers through a single API endpoint
- **GraphQL Interface**: Query multiple servers and aggregate responses
- **WebSocket Support**: Real-time events and streaming capabilities
- **Rate Limiting**: Configurable rate limits per server and user

3.2 Request Routing

- **Intelligent Routing**: Route requests to appropriate servers based on capability
- **Load Balancing**: Distribute requests across multiple instances of the same server
- **Failover**: Automatic failover to backup servers or alternative implementations
- **Caching**: Intelligent caching of responses to improve performance

3.3 Response Aggregation

- **Multi-Server Queries**: Execute tools across multiple servers in a single request
- **Response Merging**: Combine and normalize responses from different servers
- **Conflict Resolution**: Handle conflicting responses with configurable strategies
- **Format Standardization**: Consistent response formats across all servers

4. Developer Experience

4.1 Dashboard & Analytics

- **Usage Dashboard**: Real-time metrics on server usage, performance, and costs
- **Error Monitoring**: Centralized error tracking and alerting
- **Performance Analytics**: Response times, success rates, and bottleneck identification

- **Cost Tracking**: Monitor usage-based costs across all connected services

4.2 Development Tools

- **API Explorer**: Interactive API testing and documentation
- **Server Testing**: Built-in testing tools for validating server functionality
- **Debugging Tools**: Request/response logging and debugging capabilities
- **SDK Generation**: Auto-generated SDKs for popular programming languages

4.3 Integration Support

- **Client Libraries**: Official libraries for Python, TypeScript, and other languages
- **Webhooks**: Event notifications for server status changes and usage alerts
- **CI/CD Integration**: GitHub Actions and other CI/CD pipeline integrations
- **Documentation**: Comprehensive guides, tutorials, and API reference

5. Enterprise Features

5.1 Team Management

- **Organization Accounts**: Multi-user accounts with role-based access control
- **Team Collaboration**: Shared server configurations and collaborative management
- **Audit Logs**: Comprehensive logging of all user actions and system events
- **Compliance**: SOC 2, GDPR, and other compliance certifications

5.2 Security & Governance

- **Private Server Registry**: Host private MCP servers within organization boundaries
- **Network Isolation**: VPC and private network support for enterprise deployments
- **Security Scanning**: Automated security scanning of server configurations
- **Policy Enforcement**: Configurable policies for server usage and data access

5.3 Advanced Monitoring

- **Custom Metrics**: Define and track custom business metrics
- **SLA Monitoring**: Track and alert on service level agreement violations
- **Capacity Planning**: Predictive analytics for resource planning
- **Integration Monitoring**: End-to-end monitoring of AI workflow performance

Technical Architecture

Backend Infrastructure (Supabase)

Database Schema

```sql
 -- Organizations
organizations (
  id uuid primary key,
  name text not null,
  slug text unique not null,
  plan text not null default 'free',
  created_at timestamp with time zone default now()
);

-- Users
users (
  id uuid primary key references auth.users(id),
  email text not null,
  full_name text,
  avatar_url text,
  google_refresh_token text encrypted,
  created_at timestamp with time zone default now()
);

-- Organization Members
organization_members (
  id uuid primary key,
  organization_id uuid references organizations(id),
  user_id uuid references users(id),
  role text not null default 'member',
  created_at timestamp with time zone default now()
);

-- MCP Servers Registry
mcp_servers (
  id uuid primary key,
  name text not null,
  slug text unique not null,
  description text,
  category text not null,
  publisher_id uuid references organizations(id),
  repository_url text,
  documentation_url text,
  icon_url text,
  pricing_model text not null default 'free',
  auth_type text not null,
  capabilities jsonb not null,
  configuration_schema jsonb,
  is_verified boolean default false,
  download_count integer default 0,
  rating_average decimal(3,2),
  rating_count integer default 0,
  created_at timestamp with time zone default now(),
  updated_at timestamp with time zone default now()
);

-- User Server Installations
user_server_installations (
  id uuid primary key,
  user_id uuid references users(id),
  organization_id uuid references organizations(id),
  server_id uuid references mcp_servers(id),
  configuration jsonb,
  status text not null default 'stopped',
  last_health_check timestamp with time zone,
  created_at timestamp with time zone default now(),
  updated_at timestamp with time zone default now()
);

-- Server Usage Metrics
server_usage_metrics (
  id uuid primary key,
  installation_id uuid references user_server_installations(id),
  metric_type text not null,
  value numeric not null,
  timestamp timestamp with time zone default now()
);

-- API Keys
api_keys (
  id uuid primary key,
  user_id uuid references users(id),
  organization_id uuid references organizations(id),
  name text not null,
  key_hash text not null,
  permissions jsonb not null,
  last_used_at timestamp with time zone,
  expires_at timestamp with time zone,
  created_at timestamp with time zone default now()
);
```

Real-time Features

- **Supabase Realtime**: Live server status updates and usage metrics
- **Database Triggers**: Automatic notifications for server state changes
- **Row Level Security**: Fine-grained access control for multi-tenant architecture

Authentication & Authorization

- **Supabase Auth**: Google OAuth integration with custom claims
- **JWT Tokens**: Secure API access with role-based permissions
- **Row Level Security**: Database-level security policies

Frontend Architecture

Technology Stack

- **Framework**: Next.js 14 with App Router
- **UI Library**: Tailwind CSS + Shadcn/ui components
- **State Management**: Zustand for client state, React Query for server state
- **Real-time**: Supabase Realtime subscriptions
- **Charts**: Recharts for analytics dashboards

Key Components

- **Server Marketplace**: Browse and search MCP servers
- **Installation Manager**: Configure and deploy servers
- **Dashboard**: Monitor server health and usage
- **API Explorer**: Test and debug server capabilities

MCP Hub Server (Core Engine)

Technology Stack

- **Runtime**: Node.js with TypeScript
- **Framework**: Fastify for high-performance API
- **MCP Integration**: Official MCP TypeScript SDK
- **Process Management**: PM2 for server lifecycle management
- **Monitoring**: Prometheus metrics with Grafana dashboards

Core Responsibilities

- **Server Orchestration**: Manage lifecycle of multiple MCP servers
- **Request Routing**: Route API requests to appropriate servers
- **Authentication Proxy**: Handle OAuth flows and token management
- **Health Monitoring**: Monitor server health and performance
- **Event Streaming**: Real-time events via Server-Sent Events

User Stories

Developer Stories

As an AI application developer, I want to:
1. Browse available MCP servers by category so I can find relevant integrations
2. Install servers with one click so I can quickly prototype AI workflows
3. Test server capabilities through an API explorer so I can understand functionality
4. Monitor server performance and costs so I can optimize my application
5. Manage authentication centrally so I don't need to handle OAuth for each server

As an enterprise developer, I want to:
1. Deploy servers in my private cloud so I can maintain data sovereignty
2. Set up team access controls so I can manage who can install which servers
3. Monitor usage across my organization so I can track costs and compliance
4. Audit all server interactions so I can meet regulatory requirements
5. Create custom server packages so I can standardize our AI toolchain

Business User Stories

As a business user, I want to:
1. Configure AI workflows through a visual interface so I don't need technical skills
2. Connect my Google Workspace to AI tools so I can automate document processing
3. Set up notifications for important events so I can stay informed
4. Share AI workflows with my team so we can collaborate effectively
5. Track the ROI of AI integrations so I can justify the investment

Publisher Stories

As an MCP server publisher, I want to:
1. Submit my server to the marketplace so I can reach more users
2. Track download and usage metrics so I can understand adoption
3. Collect user feedback and ratings so I can improve my server
4. Monetize my server through the platform so I can generate revenue
5. Receive notifications about issues so I can provide support

Success Metrics

User Adoption

- **Monthly Active Users**: 10,000 MAU within 12 months

- **Server Installations**: 100,000 total installations within 18 months
- **User Retention**: 70% monthly retention rate
- **Time to First Value**: <5 minutes from signup to first successful server installation

Platform Health

- **Server Uptime**: 99.9% availability SLA
- **API Response Time**: <200ms average response time
- **Error Rate**: <0.1% error rate across all server interactions
- **Support Resolution**: <24 hours average response time

Business Metrics

- **Revenue Growth**: $1M ARR within 18 months
- **Customer Acquisition Cost**: <$50 CAC for self-serve users
- **Customer Lifetime Value**: >$500 LTV for paid users
- **Net Promoter Score**: >50 NPS from active users

Ecosystem Growth

- **Server Catalog**: 1,000+ verified servers within 24 months
- **Publisher Adoption**: 500+ active publishers
- **Enterprise Customers**: 100+ enterprise accounts
- **API Usage**: 10M+ API calls per month

Monetization Strategy

Pricing Tiers

Free Tier

- Up to 5 server installations
- 10,000 API calls per month
- Basic monitoring and analytics
- Community support
- **Target**: Individual developers and small projects

Pro Tier ($29/month)

- Up to 25 server installations
- 100,000 API calls per month
- Advanced monitoring and alerting

- Priority support
- Custom integrations
- **Target**: Professional developers and small teams

Team Tier ($99/month)

- Up to 100 server installations
- 1M API calls per month
- Team collaboration features
- Role-based access control
- Advanced analytics
- **Target**: Development teams and growing companies

Enterprise Tier (Custom pricing)

- Unlimited server installations
- Unlimited API calls
- Private server registry
- Dedicated support
- Custom SLAs
- On-premise deployment options
- **Target**: Large enterprises and organizations

Revenue Streams

1. **Subscription Revenue**: Primary revenue from tiered pricing plans
2. **Usage-Based Billing**: Additional charges for API calls beyond plan limits
3. **Marketplace Commission**: 10-20% commission on paid server sales
4. **Professional Services**: Custom integration and consulting services
5. **Enterprise Licensing**: On-premise and private cloud licensing

Go-to-Market Strategy

Phase 1: Developer Community (Months 1-6)

- **Target**: Individual developers and AI enthusiasts
- **Channels**: GitHub, Reddit, AI/ML communities, developer conferences
- **Content**: Technical blog posts, tutorials, open-source contributions
- **Partnerships**: Anthropic, MCP SDK maintainers, AI tool creators

Phase 2: Professional Developers (Months 6-12)

- **Target**: Professional developers and small teams
- **Channels**: Developer publications, webinars, product demos
- **Content**: Case studies, best practices, integration guides
- **Partnerships**: Cloud providers, development tool vendors

Phase 3: Enterprise Adoption (Months 12-24)

- **Target**: Enterprise development teams and IT departments
- **Channels**: Enterprise sales, partner channels, industry events
- **Content**: ROI calculators, security whitepapers, compliance guides
- **Partnerships**: System integrators, enterprise software vendors

Risk Assessment

Technical Risks

1. **MCP Specification Changes**
2. **Risk**: Breaking changes in MCP protocol

3. **Mitigation**: Close collaboration with MCP maintainers, version compatibility matrix

4. **Server Reliability**

5. **Risk**: Third-party server failures affecting platform reputation

6. **Mitigation**: Health monitoring, automatic failover, server certification program

7. **Scalability Challenges**

8. **Risk**: Platform performance degradation under load
9. **Mitigation**: Horizontal scaling architecture, performance testing, CDN usage

Business Risks

1. **Market Competition**
2. **Risk**: Large tech companies building competing platforms

3. **Mitigation**: First-mover advantage, strong developer community, unique features

4. **Regulatory Changes**

5. **Risk**: New AI regulations affecting platform operations

6. **Mitigation**: Compliance-first design, legal monitoring, adaptable architecture

7. **Economic Downturn**

8. **Risk**: Reduced enterprise spending on AI tools
9. **Mitigation**: Strong free tier, focus on ROI, cost-effective pricing

Security Risks

1. **Data Breaches**
2. **Risk**: Unauthorized access to user data or credentials

3. **Mitigation**: End-to-end encryption, regular security audits, incident response plan

4. **Malicious Servers**

5. **Risk**: Compromised or malicious MCP servers in marketplace
6. **Mitigation**: Server verification process, security scanning, user reporting system

Implementation Roadmap

Phase 1: MVP (Months 1-3)

- Basic server marketplace with search and filtering
- Google OAuth authentication
- Simple server installation and management
- Core API gateway functionality
- Basic monitoring dashboard

Phase 2: Enhanced Platform (Months 4-6)

- Advanced server management features
- Real-time monitoring and alerting
- API explorer and testing tools
- User reviews and ratings
- Basic team collaboration

Phase 3: Enterprise Features (Months 7-9)

- Organization accounts and RBAC
- Private server registry
- Advanced analytics and reporting
- Audit logs and compliance features

- Enterprise security controls

Phase 4: Advanced Capabilities (Months 10-12)

- Multi-server workflow orchestration
- Custom server development tools
- Advanced marketplace features
- Mobile applications
- International expansion

Conclusion

The MCP Hub Platform represents a significant opportunity to become the central infrastructure for AI integration, similar to how npm transformed JavaScript development or Docker Hub revolutionized container distribution. By focusing on developer experience, enterprise security, and ecosystem growth, we can build a sustainable platform that captures value from the rapidly growing AI integration market.

The combination of Supabase's robust backend infrastructure, Google's trusted authentication, and the growing MCP ecosystem provides a strong foundation for building this platform. With proper execution, the MCP Hub can become an essential tool for any organization implementing AI-powered workflows.

Next Steps:
1. Validate core assumptions through user interviews
2. Build MVP with key stakeholders
3. Establish partnerships with major MCP server publishers
4. Secure initial funding for development and go-to-market
5. Begin development with focus on developer experience