Digital Twin MCP Server

AI-Powered Professional Profile Assistant

Leveraging RAG, Vector Databases, and Model Context Protocol

20/11/2025

Nashib Rana Magar

Project Overview

What

An MCP server that creates an AI-powered digital twin of a professional profile

Why

Enables intelligent Q&A about career history, skills, and experience using RAG

Tech Stack

- Next.js 15
- TypeScript
- Upstash Vector
- Groq/LLaMA
- MCP Protocol

Problem Statement

Poor Recall

Job seekers struggle to remember specific project details, technologies used, and quantified achievements from their career history

Quantifying Impact

Difficulty in articulating measurable results and business impact of their work in convincing, data-driven ways

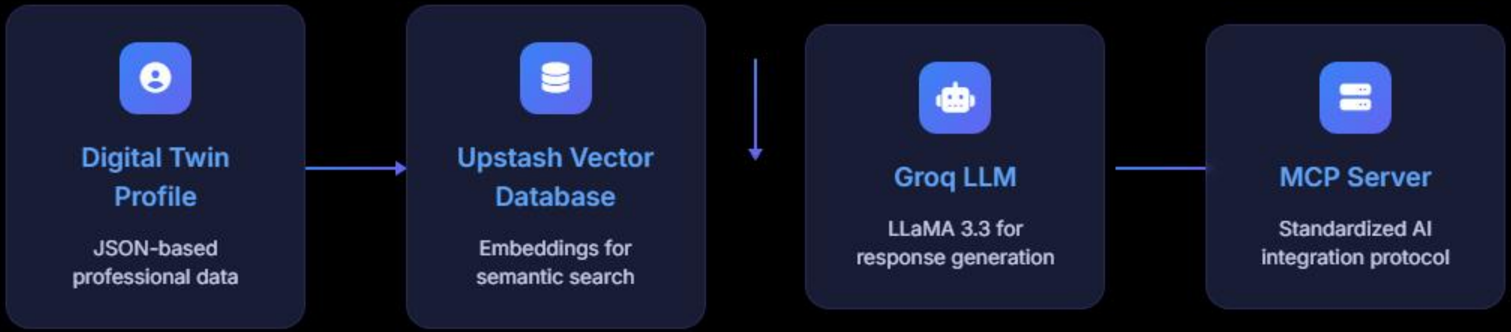
Tailoring Responses

Challenges in customizing answers for different roles, companies, and interview contexts while maintaining consistency

AI Opportunity

Use AI to maintain and query a comprehensive professional profile that can be accessed conversationally, ensuring consistent and detailed responses across all interview scenarios

Solution Architecture


Digital Twin Profile
JSON-based professional data
Upstash Vector Database
Embeddings for semantic search
Groq LLM
LLaMA 3.3 for response generation
MCP Server
Standardized AI integration protocol

Key Features

Semantic Search

Search across professional experience using vector embeddings for contextual understanding

Natural Language Q&A

Ask questions about skills, projects, and achievements using natural language

MCP Integration

Seamless integration with Claude Desktop and GitHub Copilot via MCP protocol

Interview Preparation

Real-time interview practice with AI-powered simulations and feedback

Job Analysis

Analyze job postings and assess fit based on your professional profile

Technical Implementation

Framework

Next.js 15.5.3+ with TypeScript
Full-stack React framework with built-in API routes
Enterprise-ready with excellent TypeScript support

Vector DB

Upstash Vector with embedding models
Serverless vector database for semantic search
Redis-compatible with global edge deployment

LLM

Groq API with LLaMA 3.3 70B
Ultra-fast inference for real-time responses
Advanced reasoning capabilities for complex queries

MCP Integration

JSON-RPC 2.0 protocol
Standardized AI tool integration
Seamless connection with Claude Desktop and GitHub Copilot

Deployment

Vercel (serverless) with Docker support
Edge functions for global performance
Automatic scaling and zero-config deployment

Code Quality

TypeScript with strict mode
ESLint and Prettier configuration
Comprehensive error handling and logging

Interview Simulation System

Multiple Interviewer Personas

HR, Technical, Hiring Manager, Executive

STAR-Format Answer Coaching

Structured response guidance for behavioral questions

Real-time Feedback & Scoring

Performance tracking and improvement metrics

Gap Analysis & Profile Improvement

Identify weaknesses and recommendations

Practice Across 10+ Job Postings

Tailored interviews for different roles and industries

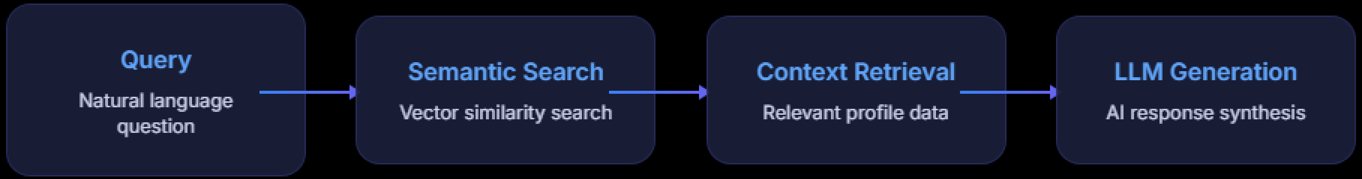
AI-Powered Question Generation

Dynamic questions based on job requirements

Timed Interview Sessions

Realistic interview time constraints

Data Architecture



Profile Structure

- digitaltwin.json with STAR-formatted projects
- Experience, skills, achievements, certifications
- Quantified metrics and results
- Metadata: salary, location, preferences

Embedding Process

- Text chunking and preprocessing
- Vector embedding generation
- Upsert to Upstash Vector database
- Metadata indexing for filtering

Use Cases

1 **Interview Preparation**
Practice with AI interviewers tailored to specific roles

2 **Resume Enhancement**
Identify gaps and add quantified achievements

3 **Salary Negotiation**
Data-driven compensation discussions

4 **Career Planning**
Assess fit for different opportunities

5 **Skills Gap Analysis**
Identify areas for professional development

Demo Workflow

1

Update Profile

User updates digitaltwin.json with professional information

2

Embed Data

Profile is embedded into Upstash Vector database

3

Connect MCP

MCP server starts and connects to Claude Desktop

4

Ask Question

User asks: \"What are my strongest projects for a full-stack role?\"

5

Generate Response

System retrieves relevant context and generates tailored response

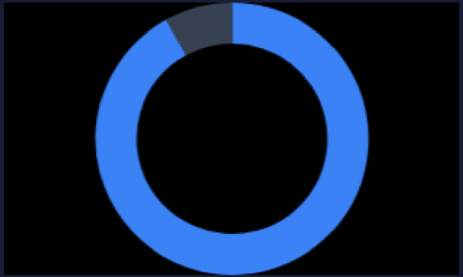
6

Practice Interview

User practices interview with AI personas

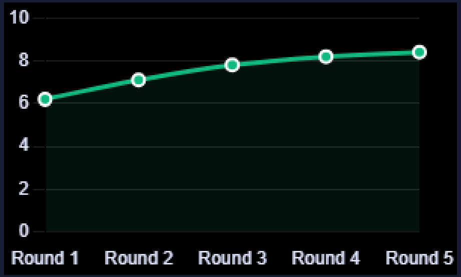
Results & Metrics

Profile Completeness



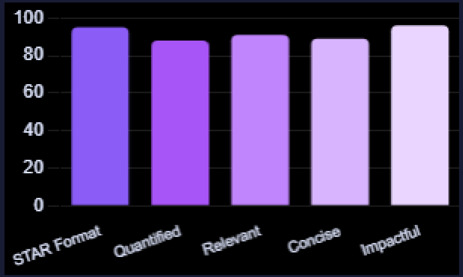
92%

Interview Scores



8.4/10

Response Quality



94%

100%
MCP Integration Success

6
Interviewer Personas

10
Job Postings Analyzed

Technical Challenges & Solutions

Challenges

Maintaining Context

Keeping context across long conversations

Generic Responses

Avoiding generic interview responses

Integration Complexity

Complex AI tool integration requirements

Profile Data Structure

Structuring professional profile data

Solutions

Vector DB + Semantic Search

Vector databases enable powerful semantic search for context maintenance

Personalized RAG

RAG with actual profile data for personalized responses

MCP Standardized Protocol

MCP protocol simplifies AI tool integration

STAR Format

STAR format with quantified results for structured data

Future Enhancements

Multi-modal Support

Resume PDFs, LinkedIn integration for comprehensive profile data

Job Market Analysis

Real-time job market analysis and matching opportunities

Automated Generation

Automated cover letter and resume generation

Video Practice

Interview video practice with speech analysis

Team Profiles

Team/organizational digital twin profiles

Key Learnings

Structured Data Importance

STAR format provides crucial context for AI understanding

Vector Embeddings Power

Enable powerful semantic search across professional experience

MCP Protocol Simplifies Integration

MCP protocol simplifies AI tool integration

Quantified Achievements Matter

Measurable results significantly improve interview performance

Iterative Refinement is Key

Profile refinement based on feedback improves AI responses

Conclusion

Built Functional MCP Server

Built functional MCP server for professional profile AI assistant

Integrated Cutting-Edge Technologies

Integrated cutting-edge technologies (Vector DB, LLM, MCP)

Created Interview Preparation System

Created practical interview preparation system

Demonstrated RAG Architecture

Demonstrated real-world application of RAG architecture

Production Ready

Ready for production deployment and scaling

Real-World Implementation

Showcased practical AI application for career development

Technical Appendix

GitHub Repository

Source Code

Complete implementation with documentation

Documentation

- `agents.md` - Implementation guides
- `README.md` - Setup instructions
- `docs/` - API documentation

Environment Variables

```
# Vector Database
UPSTASH_VECTOR_REST_URL=https://...
UPSTASH_VECTOR_REST_TOKEN=... # LLM API
GROQ_API_KEY=groq_...
```

Deployment Options

Vercel Docker Railway

TypeScript Example

```
// MCP Server Setup import { Server }
from
'@modelcontextprotocol/sdk/server.js';
const server = new Server({ name:
'digital-twin-mcp', version: '1.0.0',
});
```

Python Integration

```
# Vector Database Query
from upstash_vector
import Vector vector =
Vector(url=UPSTASH_URL,
token=UPSTASH_TOKEN)
results = vector.query(
vector=query_vector,
top_k=5 )
```

CLI Commands

```
# Install
dependencies npm
install # Start
development server
npm run dev # Build
for production npm
run build
```