**Project Finder Application Tech Stack**

This document outlines the core technologies and frameworks that will be used in the development of the Project Finder application, based on the provided requirements.

**1. Development Environment**

* **Operating System:** Windows 11
* **Shell:** PowerShell (exclusive use, no Bash syntax)
* **Virtual Environment:** Python venv
* **Package Management:** pip (with requirements.txt for dependencies)
* **Linting:** flake8 (configured for PEP 8, E501, E302, E305, W292, F401, E128)
* **Testing Framework:** pytest (with fixtures, docstrings, Arrange-Act-Assert pattern)

**2. Backend**

* **Language:** Python 3.11
* **Framework:** FastAPI
* **ASGI Server:** uvicorn
* **Data Validation:** Pydantic
* **Database ORM:** SQLAlchemy
* **Database:** SQLite (default, with flexibility for other systems via SQLAlchemy)
* **API Key Management:** .env file for secure storage
* **Documentation:** Swagger/OpenAPI for all endpoints
* **Logging:** Configurable log levels, structured logging, log rotation, multiple handlers (configured via config.json)
* **CORS Handling:** Magnanimous CORS setup for frontend requests

**3. Frontend**

* **Framework:** React
* **Build Tool:** Vite
* **Language:** TypeScript
* **Styling:** Modern design for 1980x1020 screen
* **User Interface (UI) Features:**
  + Loading indicators
  + Error notifications
  + Success messages
  + Confirmation dialogs

**4. External APIs / Services**

* **AI for Skill Embedding:** OpenAI API (text-embedding-3-large model)
* **AI for Data Extraction:** Mistral API (mistral-large-latest model)

**5. Web Scraping & Data Processing**

* **Web Scraper:** Selenium (headless mode)
* **WebDriver Management:** Proper WebDriver management with anticipation of anti-bot measures (retries, exponential backoff)
* **Configuration:** config.json for website scraping logic and CSS classes
* **Concurrency:** Parallel processing for level 3 scans and project detail analysis.

**6. Database Schema (Key Tables)**

* **projects:** Stores project data (title, description, release\_date, start\_date, location, tenderer, project\_id, requirements, rate, URL, embedding fields, budget, duration, last\_scan)
* **skills (or requirements-vdb):** Stores semantic vector embeddings for project and employee skills.
* **employees:** Stores employee information (employee name, skill list, embedding fields, experience tracking)
* **AppState:** For managing application state like "Live/Test Mode" and "Overwrite" toggle, and last\_scan tracking.

**7. Key Functionality & Algorithms**

* **Project Matching:**
  + Cosine similarity
  + Threshold-based filtering
  + Weighted skill scoring
  + Multiple skill matching
* **Date Handling:** Multi-language support, German relative dates, multiple format support.
* **Error Handling:** Specific exception types, proper logging, appropriate HTTP status codes, descriptive error messages.
* **Data Validation:** Pydantic models for input data validation on backend.

**8. Configuration**

* Multiple configuration locations
* Environment variable overrides
* Server and database specific configurations