**Client Stack Tracker – Advanced Functional and Technical Specification Document**

**📄 Comprehensive Requirements Specification**

**✏️ Project Rationale and Strategic Objectives**

The Client Stack Tracker represents a purpose-built, internally-deployed software system engineered specifically for Solutions Architects working in close coordination with Account Executives. Its primary mission is to serve as a centralized intelligence system that consolidates and visualizes a client’s comprehensive technological ecosystem—spanning hardware platforms, software stacks, network layers, and cybersecurity architecture. By structurally documenting and presenting this data, the platform aims to eliminate redundant discovery inquiries, enhance interdepartmental efficiency, and provide contextual scaffolding for more sophisticated, consultative client interactions. Moreover, the system facilitates seamless translation of internal architectural data into client-ready deliverables through an array of professional-grade export formats, promoting operational transparency and strategic alignment.

**📅 Primary Use Case Scenarios**

* Comprehensive input, ongoing maintenance, and refinement of end-to-end client architectural stack configurations.
* Dual-format visualization: a structured, tier-based view and an unstructured, mind map-style graphical display.
* Centralized archival of insights from client engagements for collective reference.
* Chronological tracking of stack evolution through a detailed change log system.
* Multiformat export capabilities—tailored to internal and client-facing audiences.
* Tagging and flagging capabilities to prioritize accounts, identify renewal timelines, and monitor strategic lifecycle events.
* Searchable and filterable database for rapid access to client stack metadata.

**🔹 Data Input Paradigm**

* **Manual Entry Workflow**: Architects manually enter data derived from direct customer engagements, supported by field notes or interview debriefs.
* **Template-Oriented Framework**: A full-stack template based on industry-standard IT architecture (e.g., Presentation, Application, Data, Security, Networking) provides structure. Any irrelevant sections may be suppressed from both the user interface and export files.
* **Freeform Data Flexibility**: Entry fields support loosely structured inputs (e.g., "Palo Alto" under "NextGen Firewall") which can later be edited to reflect changes such as transitioning to "Fortinet."
* **Inline Editing**: Real-time modifications can be made directly within the display framework without navigating away from the page.

**📃 Internal Data Schema (JSON Model)**

{

"client\_id": "client\_123",

"client\_name": "Acme Corporation",

"tags": ["finance", "critical", "enterprise", "legacy-infra"],

"flags": ["Follow-Up Needed", "Renewal Due", "Expansion Planned"],

"stack": {

"tiers": [...]

},

"change\_log": [...],

"last\_modified": "2025-04-09T14:23:00Z"

}

* Tier-based, deeply nestable schema structure
* Tags and operational flags are both extensible
* Change logs include metadata: user identity, time, action description

**📂 Data Durability and Backup**

* **Database**: MongoDB, preferably hosted on Atlas Cloud for scalability
* **Schema Model**: Loosely typed, document-centric JSON
* **Durability**: Persistent and resilient across sessions
* **Backup**: One-click export of entire database in zipped JSON and/or CSV format

**⚖️ Access Control and Security Architecture**

* **User Roles**:
  + **Admin**: Full access, including user management and export authority
  + **Editor**: Can create and update client data records
  + **Viewer**: Read-only privilege
* **Authentication**: Proprietary login mechanism controlled entirely by the application owner
* **Backend Integration**: Secured using Flask-JWT or Flask-Login modules

**🔍 Metadata and Search Intelligence**

* **Global Search**: Real-time querying across client records, OEMs, and stack layers
* **Adaptive Autocomplete**: Triggered after repeated use, allowing faster entry and consistency
* **Flexible Metadata**:
  + Dynamic tagging system for client industry, strategic status, and more
  + Multiple concurrent flags per record for lifecycle and status tracking

**📄 Export Logic and Output Artifacts**

* **Export Types**:
  + Mind Map (PDF format)
  + Tiered List (PDF format)
  + Flat Data Table (CSV format)
* **Visibility Controls**: Any field hidden in the main UI is automatically excluded from exported outputs
* **Custom Export Settings**: Toggle tags, flags, and notes on/off before generation

**🌐 Hosting, Deployment, and CI/CD**

* **Frontend**: Hosted on Netlify using JAMstack architecture
* **Backend**: Deployed via Railway, Render, or self-managed VPS
* **Version Control**: GitHub for source management, CI/CD, and versioning
* **Automation**: Deployments triggered from primary branch merges

**📱 Device Optimization and Responsive Design**

* Mobile-optimized UI architecture
* Custom-calibrated for iPhone 15 Pro Max resolution
* Responsive layouts with touch-first design patterns

**🎨 UX and Visual Design Principles**

* Modular, collapsible component architecture by stack tier
* TailwindCSS-driven styling for lightweight modern appearance
* Persistent dark mode across user sessions
* Context-aware controls and inline status indicators