DESTINI Estimator — Architecture Overview

DESTINI Estimator is a Windows desktop application (.NET/WPF) that enables real-time collaborative cost estimation. The system operates with a connection-resilient client that tolerates brief disconnects, and will queue changes during those disconnects (~30 seconds), synchronizing all changes bidirectionally through an Azure cloud backend.

Data Flow

1. **Client Application** — Users work in the DESTINI desktop app. All estimation work is performed locally, in-memory with no local persistence. The client maintains a persistent SignalR connection to the Azure backend and can tolerate brief network interruptions without disrupting the user experience.
2. **Real-Time Sync (Azure SignalR Service)** — Every change made by any user is pushed to the SignalR hub via WebSocket. The hub broadcasts changes to all connected clients, ensuring every user always has the latest state of the project.
3. **Persistence (Azure SQL Database)** — From SignalR, all changes are persisted to a non-human readable binary change journal in Azure SQL. This serves as the authoritative source of truth for all project data.
4. **Export Queue** — Every change event triggers a queued process that serializes the complete project state to JSON.
5. **Versioned Storage (Azure Blob Storage)** — The serialized JSON snapshot is written to Blob Storage, creating a full copy of the project at every change point. This provides a complete version history that is both human-readable and machine-consumable.

External Access

- **REST API** — Versioned project snapshots in Blob Storage are available for download via REST endpoints, enabling integrations with external tools and systems.
- **Reporting & Analytics** — The same JSON snapshots serve as the data source for business intelligence dashboards and reporting, providing insights across all project versions.

Key Properties

Property	Description
Connection Resilient	Tolerates brief disconnects (~30s) without interrupting user workflow
Real-Time Collaboration	All changes broadcast to all users instantly via SignalR WebSocket
Versioned Snapshots	Full JSON project export at every change point stored in Blob Storage
Dual Storage	Binary change journal (SQL) for performance; JSON snapshots (Blob) for portability
API-Accessible	Any project version downloadable via REST for integrations and reporting