Pipeline Valuation Web App — Version 1.0 Roadmap

Project Vision

Create a lightweight web application that:

1. Lets a user search for a specific pipeline company from the Form 6 database.

2. Runs three foundational valuation approaches (Cost, Income, Market).

3. Displays a clear results page showing each method’s outcome.

4. Performs basic sanity checks to ensure data exists and looks reasonable.

This version focuses on structure, clarity, and easy deployment rather than detailed accuracy.

Core Components

1. Database Layer

Connects to a local or remote Form 6 SQLite file.

Provides simple read-only queries to list respondents, available years, and high-level financial totals.

2. Backend Logic

Implements three valuation engines:

Approach	Description	Formula
Cost	Approximates replacement value from assets minus depreciation	Value = Gross Assets – Accum. Depreciation
Income	Uses operating income and capitalization rate	Value = NOI / Cap Rate
Market	Compares peers using simple multiples	Value = Peer Multiple × Revenue

3. API / Web Server

- Endpoints: /search, /valuation, /results

- Could use FastAPI or Flask

- Configurable via environment variables

4. Frontend / Interface

Simple HTML pages (no framework yet):

1. Home/Search — search by name or ID, choose year

2. Valuation Setup — adjust assumptions and run

3. Results — show values and average result

5. Data Integrity

Basic checks only:

- Confirm data exists for respondent and year

- Ensure revenues and assets > 0

- If missing, display friendly “Data incomplete” message

Configuration

Use `.env` file for settings:

- DB_PATH

- DEFAULT_CAP_RATE = 0.08

- DEFAULT_DISCOUNT_RATE = 0.09

- DEFAULT_TREND_FACTOR = 1.00

Architecture

pipeline-val/

■■■ app/

■ ■■■ main.py

■ ■■■ valuation/ (cost.py, income.py, market.py)

```
■ ■■■ templates/ (home, valuation, results)
■ ■■■ static/
■■■ .env
■■■ requirements.txt
■■■ README.md
```

Workflow

1. Search for company → select year

2. App fetches data → validates

3. User adjusts inputs → runs valuation

4. System computes Cost, Income, Market → shows results

Deployment Path

1. Develop locally in VS Code with Uvicorn.

2. Deploy on small VPS (Ubuntu + Python).

3. Run via Gunicorn + Nginx, connect same SQLite DB.

Future Enhancements

- Data validation & cleaning
- Custom method weighting
- Trend visualization
- Export PDF/Excel
- Authentication

Summary

Version 1.0:

- Search → Select → Evaluate → View Results
- Minimal checks, simple math, clear outputs
- Designed for transparency and later expansion