**🏦 Bank Reconciliation Agent – Project Roadmap & Executive Summary**

*Last Updated: 2025-07-25*

**✨ Executive Summary**

The **Bank Reconciliation Agent** automates the preparation and transformation of bank statements into Xero-compatible formats. It eliminates manual file formatting by supporting intelligent parsing of various bank statement formats and outputs ready-to-import CSV files. The long-term goal is a fully automated, intelligent reconciliation engine capable of invoice matching, account coding, and Xero posting.

**📊 Key Features**

* Upload interface via FastAPI Swagger UI
* Auto-detection of bank type from filename
* Custom parsers for:
  + NBB (CSV)
  + KFH Account (Excel)
  + KFH Card (Excel & PDF-ready)
* Smart date normalization (all exports: yyyy/mm/dd)
* Currency sanitization and sign detection (debit/credit based)
* Dynamic CSV export with timestamped bank filename
* Export matches custom Xero-compatible template (\*Date, \*Amount, Description)
* File download support via API endpoint

**🚀 Architecture & Tech Stack**

* **Backend**: FastAPI
* **Parsing/Processing**: Pandas
* **Storage**: Local file system (CSV), Google Sheets (future)
* **Integration**: Xero API (future), ApprovalMax (future)
* **Authentication**: None (local testing phase)

**📂 Project Structure (Simplified)**

bank\_reco\_agent/

├── app/

│ ├── main.py # App entry point

│ ├── routes/

│ │ ├── upload.py # Upload handler

│ │ ├── download.py # CSV download endpoint

│ │ └── remittance.py # (future use)

│ ├── services/

│ │ ├── parser.py # All parsing logic

│ │ └── xero\_format.py # Format export CSV

│ ├── templates/

│ │ └── statement\_template.csv # CSV header template

│ └── exports/ # Auto-generated Xero CSVs

**🧠 Core Logic**

* File uploaded via /upload/bank-statement
* Bank name is extracted from filename
* Parser selected based on filename pattern
* Raw data cleaned: currency stripped, signs applied, headers detected
* Dates parsed from various formats (incl. PDF-friendly 15-May-2025)
* Output CSV generated in exports/ using required format
* Link to file returned in API response

**📄 Recommended Architecture (Current Phase)**

| **Component** | **Recommendation** |
| --- | --- |
| File Upload | FastAPI Swagger UI |
| Bank Parsing | Pandas + custom per-bank logic |
| CSV Formatting | Template-driven (auto header match) |
| Export Storage | Local file system (/exports/) |
| Download | /download/{filename} route |
| Automation (Next) | Xero API integration, ML prediction |

**🔄 Visual Flowchart**

User Uploads File

|

v

Detect Bank Type (filename)

|

v

Parse File → DataFrame

|

v

Clean + Normalize (date, amount)

|

v

Map to Template CSV Format

|

v

Save to /exports/ → BANKyyyymmdd.csv

|

v

Return download link to user

**📁 Folder/Code Structure (FastAPI + ML + Chatbot - Future Ready)**

app/

├── routes/

│ ├── upload.py # Handles parsing + export

│ ├── download.py # File download

│ └── remittance.py # To be implemented

├── services/

│ ├── parser.py # Core logic (NBB, KFH, etc.)

│ └── xero\_format.py # Statement formatter

├── templates/ # Xero import template

├── exports/ # Final CSV output

**✅ Completed Steps / Phase 1**

* Set up FastAPI backend and endpoints
* Implement upload route with file-type detection
* Build parser for:
  + NBB CSV
  + KFH Account Excel
  + KFH Card Excel
* Normalize all dates to yyyy/mm/dd
* Format output to match statement\_template.csv
* Generate dynamic export filenames (e.g., KFH20250725.csv)
* Add /download/{filename} endpoint
* Fully tested and imports cleanly into Xero

**⏳ Next Steps / Phase 2**

* Add PDF extraction support for scanned/card statement PDF (e.g., 15-May-2025)
* Add /upload/remittance endpoint for invoice breakdowns
* Parse and map remittance Excel files to invoices
* Implement rules-based invoice allocation logic
* Add optional Google Sheets integration for tracking
* Implement ML model for account coding predictions
* Explore Xero API for direct transaction or journal posting
* Add file preview and tagging dashboard (Streamlit or Chat UI)

**This document should be used as a living roadmap for the Bank Reconciliation Agent.**