**Workspace**

Collecting workspace information

This project is a **Church Invoice Analyzer** system designed to automate the extraction of payment/contribution data from scanned or photographed church payment forms, and to generate professional receipts and reports. It supports desktop, web, and mobile interfaces, and is optimized for both iOS and Android devices.

**Key Components**

* **invoiceanalyzer.py**: Main logic for analyzing uploaded images/PDFs using Azure AI, extracting structured data, and exporting to CSV.
* **web\_invoice\_app.py**: Flask web application providing:
  + File upload and analysis
  + Manual entry form
  + Mobile-optimized interface (/mobile)
  + REST API for mobile apps (/api/upload)
  + Receipt generation and download
  + Weekly and summary report generation
* **printreceipt.py**: Generates PDF/JPG receipts from extracted or manually entered data.
* **mobile\_upload.html**: Standalone mobile web interface for camera/gallery upload, editing, and receipt generation.
* **templates/**: HTML templates for web UI (upload, results, reports, etc.).
* **uploads/**: Stores uploaded files and analysis results.
* **Receipts store**: Centralized folder for all generated receipts and reports.

**Main Features**

* **Multi-platform**: Works as a desktop script, web app, and mobile web/PWA.
* **Azure AI Integration**: Uses Azure Content Understanding to extract fields like Name, Amounts, Dates, Comments, etc.
* **Mobile Support**: Camera capture, gallery selection, touch-friendly UI, PWA install, WhatsApp sharing.
* **Receipt Generation**: Creates PDF/JPG receipts with all contribution details.
* **Reports**: Generates weekly, summary, and offertory reports in PDF/Excel/CSV.
* **Security**: File type/size validation, secure filenames, environment variable configuration, CORS for mobile API.
* **Offline/PWA**: Service worker and manifest for offline use and home screen installation.

**Typical Workflow**

1. **Upload**: User uploads a payment form (image/PDF) via web or mobile.
2. **Analysis**: Azure AI extracts structured data (amounts, names, etc.).
3. **Review/Edit**: User reviews and edits extracted data if needed.
4. **Receipt Generation**: User generates a receipt (PDF/JPG) and downloads or shares it.
5. **Reporting**: Admin can generate weekly/summary reports from stored receipts.

**Mobile Experience**

* /mobile route and mobile\_upload.html provide a touch-optimized, camera-enabled interface.
* PWA features: installable, offline support, home screen icon.
* API endpoint /api/upload for mobile and app integration.

**File Structure**

See README.md and [SECURITY\_README.md](vscode-file://vscode-app/c:/Users/might/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) for details.

**Deployment**

* Can be run locally ([python web\_invoice\_app.py](vscode-file://vscode-app/c:/Users/might/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)) or deployed to cloud platforms (Azure, Heroku, etc.).
* Environment variables for credentials and configuration.

**For more details, see:**

* README.md
* MOBILE\_COMPATIBILITY\_REPORT.md
* PRODUCTION\_DEPLOYMENT.md
* [SECURITY\_README.md](vscode-file://vscode-app/c:/Users/might/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)