App.py Technical Documentation

Overview

app.py is a Streamlit-based interactive web application that visualizes job posting data and associated skills across different geographical locations. The application provides two distinct viewing modes: a **Global Map** view showing all job postings, and a **Coverage Map** view showing primary/secondary city coverage areas with detailed skills breakdowns.

Table of Contents

Application Architecture

The application follows a modular architecture with clear separation of concerns:

```
app.py
├── Configuration & Styling (lines 16-131)
├── Helper Functions (lines 154-250)
├── Data Loading Functions (lines 266-569)
├── Data Processing & Caching (lines 574-650)
├── Map Visualization Functions (lines 879-1244)
└── Main Application Logic (lines 1247-1275)
```

Dependencies

Required Python Packages

Listed in requirements.txt:

- **streamlit**: Web application framework
- **pydeck**: WebGL-powered visualization library for large-scale data
- **pandas**: Data manipulation and analysis
- **numpy**: Numerical computing
- **python-calamine**: Fast Excel file reading (primary engine)
- **openpyxl**: Excel file reading (fallback engine)

Standard Library Imports

- **re**: Regular expression operations
- **json**: JSON parsing for city configuration
- **hashlib**: Hashing functions (imported but not actively used)
- **io**: I/O operations for CSV downloads
- **pathlib**: Object-oriented filesystem paths
- **typing**: Type hints for better code documentation

Configuration

Page Configuration (lines 19-24)

- **Wide layout** for better map visibility
- **Collapsed sidebar** for cleaner interface
- Earth emoji (🌍) as page icon

Visual Constants (lines 136-150)

Constant	Value	Purpose
UNKNOWN_SHEET_NAME	"Unknown_Country"	Sheet name to filter out from data
DEFAULT_COUNTRY_MIN_RADIUS	400m	Minimum bubble size for country view
DEFAULT_COUNTRY_MAX_RADIUS	10,000m	Maximum bubble size for country view
DEFAULT_GLOBAL_MIN_RADIUS_ALL	4,000m	Minimum bubble size for global view
DEFAULT_GLOBAL_MAX_RADIUS_ALL	300,000m	Maximum bubble size for global view
DEFAULT_FALLBACK_RADIUS_KM	10 km	Default radius when data is missing

Data Files

Required Files

1. **primary_secondary_cities.json**

   Location: Project root directory
   Format: JSON file containing city definitions
   Required fields per city: country, City/city, Latitude/latitude, Longitude/longitude, Radius/radius
   Purpose: Defines major cities and their coverage areas

2. **mapping/data/all_OJAs_aggregated.xlsx**

   Location: mapping/data/ subdirectory
   Format: Multi-sheet Excel workbook (one sheet per country)
   Required columns per sheet: City, Latitude, Longitude, Count
   Purpose: Contains aggregated job posting counts by city

3. **mapping/data/cities_skills.xlsx**

   Location: mapping/data/ subdirectory
   Format: Multi-sheet Excel workbook (one sheet per country)

Required columns per sheet: Skill, SkillType (ST1=Hard, ST2=Soft), City, Latitude, Longitude, count
Purpose: Contains detailed skills data per city (optional)

Core Components

1. CSS Styling (lines 27-131)

The application includes extensive custom CSS for a modern, polished appearance with hidden Streamlit branding, gradient-styled radio buttons, modern data table styling, and responsive column layouts.

Color Scheme:

- Primary gradient: Purple (#667eea to #764ba2)
- Success gradient: Teal (#11998e to #38ef7d)
- Accent colors for interactive elements

2. Data Normalization (lines 174-214)

Text Normalization Functions:

- **_canonicalize_columns(df)**: Maps column name variations to standard names
- **_normcase(x)**: Converts text to case-folded, whitespace-normalized form
- **_strip_weird_whitespace(s)**: Removes irregular whitespace
- **_clean_punctuation(s)**: Removes leading/trailing punctuation
- **_parse_int_series(s)**: Converts series to integers safely
- **_safe_to_numeric(s)**: Converts to numeric with error handling

3. Geographic Calculations (lines 224-234)

Haversine Distance Formula:

Vectorized implementation calculating great-circle distances between points. Input: Arrays of latitudes/longitudes and a reference point. Output: NumPy array of distances in kilometers. Uses Earth Radius of 6371.0088 km and is optimized with NumPy for batch calculations.

4. Visual Scaling (lines 216-221)

The _scale_radius_by_global_sum function proportionally scales circle radii based on data values by calculating proportion of each value relative to total, mapping proportions to specified radius range, and ensuring visual differentiation while maintaining readability.

Data Loading

All data loading functions use @st.cache_data decorator for performance optimization.

1. Aggregated Job Data (lines 267-315)

Function: load_aggregated_all_sheets(path, mtime)

Attempts to use calamine engine (faster), falls back to openpyxl. Reads all sheets from the Excel workbook and for each sheet normalizes column names, validates required columns exist, cleans city names and coordinates, and converts counts to integers. Returns dictionary: {country_name: DataFrame}.

2. Primary/Secondary Cities (lines 319-367)

Function: load_primary_secondary_cities(path)

Loads JSON file with city definitions, handles both dictionary format and direct array format, normalizes column names, validates required columns, cleans and normalizes text fields, converts coordinates to numeric values, and applies fallback radius where missing.

3. Major Centers with Posts (lines 371-439)

Function: build_major_centers_with_aggregated_posts(centers_df, aggregated_by_country_norm)

Combines city definitions with actual job posting data to calculate coverage statistics. Groups centers by country, and for each center city retrieves own job posting count, calculates distances to all other cities using Haversine formula, identifies cities within coverage radius, sums job postings from nearby cities, and calculates total posts.

4. Skills Data (lines 492-568)

Function: load_country_skills(path, mtime, country)

Opens Excel file using cached handle, parses specific country sheet, canonicalizes column names, cleans text fields, converts coordinates to numeric, parses count as integers, and filters invalid entries.

Map Visualizations

Version 1: Global Map (lines 879-1006)

Purpose: Displays all job posting locations as red circles scaled by posting volume.

Features:

- Single dropdown for country selection (All or specific country)
- Red circles for job posting locations
- Size scaling proportional to posting count
- Hover tooltips available only for priority cities
- Text labels display names for priority cities

Version 2: Coverage Map (lines 1011-1244)

Purpose: Shows primary/secondary cities with coverage areas and detailed skills.

Features:

- Two dropdowns: Country and City selection
- Green circles showing coverage areas (radius = Radius_km_max)
- Click interaction to view skills
- Skills panel with detailed breakdown when city selected
- Adaptive zoom based on city radius

Skills Panel (lines 740-822)

Purpose: Displays top 50 hard and soft skills for selected city and coverage area.

Features:

- Scope selection: City only vs. coverage area
- Dual columns: Hard skills (ST1) and Soft skills (ST2)

- Top 50 ranking with most frequent skills displayed
- CSV download for analysis
- Modern styling with gradient headers and clean tables

User Interface

Main Layout (lines 1247-1275)

The interface consists of a title header, radio button selection between Global Map and Coverage of Main Locations, view-specific description text, the main map visualization area, and a skills panel when applicable.

Radio Button Selection

Two mutually exclusive views: "Global Map" shows Version 1 visualization with red circles for job post counts, while "Coverage of Main Locations" shows Version 2 visualization with green circles showing coverage radii.

Session State Management

Version 1 State Variables

v1_country: Selected country in Global Map view (Default: "All")

Version 2 State Variables

v2_country: Selected country in Coverage Map view (Default: "All")
v2_city: Selected city in Coverage Map view (Default: "All", updates when country changes)

Click Interaction State (lines 848-868)

Uses a pending selection mechanism: user clicks city on map, _extract_clicked_center extracts city/country from PyDeck event, _set_dropdown_selection stores in temporary state, st.rerun() is triggered, and _ingest_pending_selection() applies pending values. This two-step process ensures the country dropdown updates first so city choices refresh based on the new country.

Usage Guide

Initial Setup

1. **Prepare Data Files:** Place primary_secondary_cities.json in project root, create mapping/data/ directory, add all_OJAs_aggregated.xlsx, and optionally add cities_skills.xlsx
2. **Install Dependencies:** Run pip install -r requirements.txt
3. **Run Application:** Run streamlit run app.py

Navigation Guide

Viewing Global Distribution

1. Select "Global Map" radio button

2. Choose country from dropdown or keep "All" for global view
3. Hover over red circles to see city details
4. Observe relative posting volumes by circle size

Exploring City Coverage

5. Select "Coverage of Main Locations" radio button
6. Choose country from first dropdown
7. Optionally choose specific city from second dropdown
8. Green circles show coverage areas
9. Click any green circle to view that city's skills

Analyzing Skills

1. In Coverage view, select a specific city
2. Wait for skills panel to load below map
3. Review top 50 hard skills (left column)
4. Review top 50 soft skills (right column)
5. Click "Download CSV" to export data

Technical Details

Excel Engine Selection

The application uses calamine (Rust-based, extremely fast for large files) as the primary engine and falls back to openpyxl (pure Python, universal compatibility). Auto-detection ensures the best available engine is used.

Color Coding System

Global Map (Version 1):

Red (#DC3232) for job posting locations with 70% opacity for overlapping visibility.

Coverage Map (Version 2):

Forest Green (#228B22) for standard coverage areas, Bright Green (#00B400) for selected city, and Dark Green (#006400) for circle borders. Opacity varies from 80-180/255 based on selection state.

Known Limitations

1. File Dependencies: Application stops with error if required files are missing
2. Sheet Names: Must exactly match country names in city configuration
3. Memory Usage: Large Excel files are fully loaded into memory
4. Skills Panel: Requires exact country/city match in skills workbook
5. Label Overlap: Text labels may overlap in dense areas
6. Unknown Countries: Sheet named "Unknown_Country" is filtered out

Error Handling Strategy

Fatal Errors (using st.error() + st.stop()):

Missing primary_secondary_cities.json, invalid JSON format, missing required columns, missing all_OJAs_aggregated.xlsx, or no valid country sheets found.

Graceful Degradation:

Skills file missing shows info message and continues without skills. Empty data for country shows info message. No skills for city shows info message. Invalid/missing coordinates are filtered out silently.

Conclusion

This application provides a comprehensive, interactive visualization platform for exploring job posting and skills data across geographical locations. Its dual-view architecture balances global overview with detailed local analysis, while efficient caching and modern UI design ensure a responsive user experience.

The modular codebase facilitates maintenance and extension, with clear separation between data loading, processing, and visualization concerns. Extensive validation and error handling ensure robustness across diverse data scenarios.