**Detailed Steps**

**A. File Preparation & Naming**

* Standardize Excel filenames (e.g., "META Model 12-31-24.xlsx") to consistently parse company names
* Store all Excel models in a single data/ directory
* Save PowerPoint template in the same directory or a known path

**B. Data Extraction from Excel**

**Goal**: Retrieve two numbers:

* Fully Diluted Share Count
* Share Price

**Strategy**:

* Load each workbook and iterate over all sheet names
* Identify the correct sheet using keywords like "Simple Model", "Valuation", or "Case"
* Search row-by-row for relevant keywords using string matching or regex (e.g., "Shares Outstanding", "Share Price")
* Use pandas.to\_numeric() to extract and clean the first numerical value on the matching row
* If multiple matches, choose the most reasonable (e.g., the first non-null number after the keyword)

**Edge Cases**:

* Handle formatting errors (merged cells, headers, missing values)
* Skip sheets that do not match known patterns
* Gracefully fail if values are not found, and return "Error" or None

**C. Compute Market Cap**

* Multiply: Share Count × Share Price
* Round to 2 decimal places
* Add company name (parsed from filename) and result to a dict

{

"Company": "META",

"Market Cap (USD)": 653200000000

}

**D. Aggregate Results**

* Collect all such dict results in a list:

[

{"Company": "META", "Market Cap (USD)": 653200000000},

{"Company": "MSFT", "Market Cap (USD)": 2498300000000},

...

]

**E. Create PowerPoint Slide with Table**

* Open existing .pptx file using python-pptx
* Use a predefined slide layout (e.g., slide\_layouts[1])
* Add title: "Market Capitalization Summary"
* Add a table with n+1 rows (n = number of companies) and 2 columns: Company, Market Cap
* Format headers and data cells:
  + Header in bold
  + Currency values formatted as strings (e.g., $1,234,567,890)
* Save the output as a new .pptx file

**F. Build the User Interface (Optional but Ideal)**

* Use Streamlit or Flask for UI
* Provide a button labeled: "Generate Market Cap Slide"
* On click:
  + Trigger data extraction and calculation
  + Generate output .pptx
  + Show a "Download" button for the user to retrieve the result

**Testing & Validation**

* Confirm extracted values match manual Excel checks
* Check presentation output formatting
* Test with intentionally corrupted/missing data
* Ensure program does not crash; instead logs appropriate warnings

A screenshot of a computer

Description automatically generatedA screenshot of a report

Description automatically generated