Sure! Here's an expanded table including **technologies, libraries, and models** used in your resume screener project, along with their **purpose**:

|  |  |
| --- | --- |
| **Technology / Library / Model** | **Purpose** |
| **Python** | Main programming language used for application logic. |
| **Streamlit** | For creating the web-based UI for resume upload, job preference input, and result display. |
| **sentence-transformers** | Python library to work with BERT-based models for sentence/document embeddings. |
| **BERT (e.g., all-MiniLM-L6-v2)** | Pretrained model from Sentence Transformers used to generate semantic embeddings of resumes and job preferences. |
| **scikit-learn (sklearn)** | Specifically, cosine\_similarity is used to compute similarity between embeddings. |
| **PyMuPDF (fitz)** | To extract text from PDF files. |
| **docx2txt** | To extract text from Word (.docx) files. |
| **zipfile** | To extract uploaded zipped folders of multiple resumes. |
| **os / shutil / glob** | For file management – locating, copying, and handling uploaded resume files. |
| **tempfile** | To manage temporary directories for storing extracted resumes. |
| **pandas** | Used to store, sort, and filter resume similarity scores in tabular format. |
| **numpy** | Numerical operations for array-based computations. |
| **typing** | Used for type hinting in functions (List, Tuple, etc.) for cleaner code. |
| **streamlit.components.v1** | For adding advanced custom elements in the Streamlit interface. |
| **Regular Expressions (re)** | For cleaning or parsing specific patterns in extracted resume text. (If used) |