The approach taken involves creating a chatbot using Streamlit for the user interface and integrating it with various libraries and tools to handle PDF data, perform text processing, manage conversation history, and generate responses. Here’s a summary of the approach:

1. **Extract Text from PDF**:
   - **Library Used**: PyPDF2
   - **Purpose**: Extract raw text from PDF files containing wine-related information.
2. **Text Processing**:
   - **Library Used**: langchain.text_splitter.CharacterTextSplitter
   - **Purpose**: Split the extracted text into chunks suitable for embedding and vector search.
3. **Vector Store Management**:
   - **Library Used**: langchain_community.vectorstores.FAISS, langchain_community.embeddings.HuggingFaceEmbeddings
   - **Purpose**: Convert text chunks into vector embeddings and store them in a FAISS index for efficient similarity search.
4. **Conversational AI**:
   - **Library Used**: langchain_google_genai.ChatGoogleGenerativeAI
   - **Purpose**: Handle user queries and generate responses based on the vector store and conversation history.
5. **State Management**:
   - **Framework Used**: Streamlit's session state
   - **Purpose**: Manage conversation history and vector store state between interactions.
6. **User Interface**:
   - **Framework Used**: Streamlit
   - **Purpose**: Build the web interface where users can input queries and receive responses. Customize the display of the chatbot conversation.