Bhashasutra Documentation

Required Libraries : - [reuirements.txt](file:///D:\Bhashasutra\requirements.txt)

|  |  |  |
| --- | --- | --- |
| Front End | Back End | Functions |
| streamlit | Fastapi | scikit-learn |
| streamlit-authenticator | Uvicorn | Xgboost |
| extra-streamlit-components | Starlette | lightgbm |
| Altair | bcrypt | nltk |
| Plotly | argon2-cffi | textblob |
| seaborn | passlib | wordcloud |
| Pydeck | argon2-cffi-bindings | pdfminer.six |
| Matplotlib | python-jose | Pdfplumber |
| wordcloud | redis | python-docx |
|  | celery | docx2txt |
|  | kombu | pypdf |
|  | amqp | PyPDF2 |
|  | httpx | pypdfium2 |
|  | httpcore | regex |
|  | aiohttp | huggingfacehub |
|  | aiosignal | datasets |
|  | anyio | numpy |
|  | async-lru | scipy |
|  | APScheduler | pandas |
|  | pydantic | joblib |
|  | pydantic | threadpoolctl |
|  | python-dotenv | dill |
|  | loguru | multiprocess |
|  | fastjsonschema | tqdm |
|  | idna | pyspellchecker |
|  | requests |  |
|  | postgres |  |
|  |  |  |
|  |  |  |
|  |  |  |

Description of required Libraries :-

**Frontend Libraries (Streamlit & Visualization)**

**1. streamlit**

* A Python framework for building interactive web applications with minimal code.
* Allows easy integration of widgets, charts, and user inputs for data-driven apps.

**2. streamlit-authenticator**

* Provides authentication functionality for Streamlit applications, including login and signup.
* Supports various authentication methods like hashed passwords and OAuth.

**3. extra-streamlit-components**

* A collection of additional UI components to enhance Streamlit apps.
* Includes features like toggle switches, dropdown menus, and progress bars.

**4. Altair**

* A declarative statistical visualization library based on the Vega and Vega-Lite frameworks.
* Enables easy creation of complex, interactive visualizations with minimal code.

**5. Plotly**

* A powerful library for creating interactive and dynamic charts, including 3D and animated plots.
* Compatible with web-based applications and integrates well with Streamlit.

**6. seaborn**

* A statistical data visualization library built on Matplotlib with a focus on aesthetics.
* Provides high-level functions for visualizing complex datasets, including heatmaps and pair plots.

**7. Pydeck**

* A library for large-scale geospatial data visualization in Python.
* Supports interactive maps with layers, tooltips, and 3D visualizations.

**8. Matplotlib**

* A foundational visualization library in Python for static, animated, and interactive plots.
* Highly customizable and supports various chart types like histograms, bar charts, and scatter plots.

**9. wordcloud**

* A library for generating word clouds from text data to visualize word frequency.
* Provides customization options for colors, shapes, and font styles.

**Backend Libraries (With Two-Line Descriptions)**

**1. FastAPI**

* A modern, high-performance web framework for building APIs with Python based on type hints.
* Provides automatic validation, serialization, and OpenAPI documentation generation.

**2. Uvicorn**

* A lightning-fast ASGI server for running FastAPI applications with support for WebSockets.
* Utilizes uvloop and httptools for efficient asynchronous request handling.

**3. Starlette**

* A lightweight ASGI framework that serves as the foundation for FastAPI.
* Offers essential middleware, routing, and WebSocket support for scalable applications.

**4. bcrypt**

* A secure password hashing library that uses the Blowfish-based bcrypt algorithm.
* Helps protect user credentials against brute-force and rainbow table attacks.

**5. argon2-cffi**

* Implements the Argon2 password hashing algorithm with enhanced security features.
* More resistant to brute-force attacks due to memory-hard properties.

**6. passlib**

* A robust password hashing library supporting multiple hashing algorithms like bcrypt and Argon2.
* Simplifies password security by providing automatic hash upgrades and verification methods.

**7. argon2-cffi-bindings**

* Low-level CFFI-based bindings for Argon2 password hashing.
* Serves as the core cryptographic implementation for argon2-cffi.

**8. python-jose**

* A lightweight implementation of JSON Web Tokens (JWT) for authentication.
* Supports JSON Web Signature (JWS) and JSON Web Encryption (JWE).

**9. redis**

* An in-memory key-value store used for caching, session management, and background tasks.
* Frequently paired with Celery for distributed task queues and real-time data processing.

**10. celery**

* A powerful asynchronous task queue that enables background job processing.
* Works with message brokers like Redis and RabbitMQ to manage distributed tasks.

**11. kombu**

* A messaging library that provides an abstraction layer for different message brokers.
* Used by Celery to send and receive messages across various transports.

**12. amqp**

* A low-level AMQP protocol implementation for Python.
* Used for direct communication with RabbitMQ and other AMQP-compatible brokers.

**13. httpx**

* A modern HTTP client for making asynchronous requests with HTTP/1.1 and HTTP/2 support.
* Provides connection pooling, timeouts, and authentication features.

**14. httpcore**

* A low-level networking library that powers httpx.
* Handles connection pooling, request retries, and efficient I/O operations.

**15. aiohttp**

* An asynchronous HTTP client/server framework for non-blocking request handling.
* Commonly used in FastAPI applications for API calls and microservices communication.

**16. aiosignal**

* A lightweight library for managing asynchronous event hooks.
* Often used with aiohttp to trigger event-driven actions.

**17. anyio**

* A compatibility layer that enables seamless async programming across asyncio, trio, and curio.
* Facilitates writing cross-platform asynchronous applications.

**18. async-lru**

* Provides an asynchronous Least Recently Used (LRU) cache decorator.
* Optimizes performance by caching expensive function calls in async applications.

**19. APScheduler**

* A job scheduling library that enables periodic task execution.
* Supports cron-like scheduling and background task management.

**20. Pydantic**

* A data validation and serialization library used extensively in FastAPI.
* Ensures API request and response data conform to predefined schemas using type hints.

**21. python-dotenv**

* Loads environment variables from a .env file into the application environment.
* Helps manage sensitive configuration data like API keys and database credentials.

**22. loguru**

* A modern logging library with simple syntax and rich formatting capabilities.
* Supports asynchronous logging, automatic rotation, and structured logs.

**23. fastjsonschema**

* A fast JSON schema validator for validating API request payloads.
* Used in FastAPI for enforcing strict data validation in request bodies.

**24. idna**

* Implements Internationalized Domain Names (IDN) encoding and decoding.
* Ensures proper handling of non-ASCII domain names in web applications.

**25. requests**

* A widely used HTTP library for making synchronous API requests.
* Provides an intuitive API for sending GET, POST, and other HTTP methods.

**26. postgres**

* PostgreSQL database adapter for connecting Python applications to a PostgreSQL database.
* Enables interaction with PostgreSQL using SQL queries and ORM libraries.

**Machine Learning, NLP, and Data Processing Libraries (With Two-Line Descriptions)**

**1. scikit-learn**

* A comprehensive machine learning library for classification, regression, clustering, and dimensionality reduction.
* Provides efficient tools for model training, evaluation, and hyperparameter tuning.

**2. XGBoost**

* An optimized gradient boosting library designed for speed and performance in predictive modeling.
* Supports parallel computation and handles missing values effectively.

**3. LightGBM**

* A fast, distributed, and scalable gradient boosting framework developed by Microsoft.
* Uses histogram-based learning to improve efficiency on large datasets.

**4. NLTK**

* A powerful NLP toolkit providing text processing utilities like tokenization, stemming, and POS tagging.
* Supports linguistic research and building machine learning models for text analysis.

**5. TextBlob**

* A simple NLP library built on NLTK and Pattern, offering easy-to-use APIs for text analysis.
* Supports sentiment analysis, noun phrase extraction, and translation.

**6. wordcloud**

* A visualization library for generating word clouds from text data.
* Useful for identifying frequently occurring words in textual datasets.

**7. pdfminer.six**

* A Python library for extracting text, images, and metadata from PDF documents.
* Supports advanced text parsing, layout analysis, and character recognition.

**8. pdfplumber**

* Built on pdfminer.six, it provides additional functionalities for extracting structured data from PDFs.
* Ideal for parsing tables and extracting formatted text from PDFs.

**9. python-docx**

* A library for reading, creating, and modifying Microsoft Word .docx files.
* Allows programmatic insertion of text, tables, and images into Word documents.

**10. docx2txt**

* A lightweight library to extract text from .docx files while maintaining formatting.
* Primarily used for simple text extraction without advanced document manipulation.

**11. pypdf**

* A modern library for manipulating PDF files, including merging, splitting, and extracting text.
* Provides an intuitive API for handling complex PDF operations.

**12. PyPDF2**

* A legacy PDF processing library used for reading, writing, and modifying PDF files.
* Supports encryption, metadata extraction, and text manipulation.

**13. pypdfium2**

* A Python wrapper for PDFium, offering high-performance PDF rendering and manipulation.
* Supports converting PDFs to images and extracting structured text efficiently.

**14. regex**

* An advanced regular expression library providing extended pattern-matching functionalities.
* Supports Unicode-aware regex operations and complex text processing tasks.

**15. Hugging Face Hub**

* A platform for sharing and downloading pre-trained NLP and ML models.
* Provides seamless integration with transformers and datasets libraries for deep learning applications.

**16. datasets**

* A library for easy access to large-scale machine learning datasets.
* Optimized for streaming, multi-processing, and efficient data loading.

**17. numpy**

* A foundational numerical computing library for handling arrays and matrices.
* Provides high-performance mathematical operations used in scientific computing and ML.

**18. scipy**

* A scientific computing library offering advanced mathematical functions, optimization, and statistics.
* Essential for numerical integration, signal processing, and scientific simulations.

**19. pandas**

* A powerful library for data manipulation and analysis with DataFrame support.
* Enables efficient handling of structured data and integrates well with ML workflows.

**20. joblib**

* A library for efficient memory caching and parallel execution of Python functions.
* Commonly used in machine learning pipelines for model persistence and computation optimization.

**21. threadpoolctl**

* Controls the number of threads for libraries like NumPy, SciPy, and scikit-learn.
* Helps optimize performance by managing parallelism in ML workloads.

**22. dill**

* An enhanced serialization library that extends Python's pickle module.
* Supports serializing complex objects, including functions and closures.

**23. multiprocess**

* A multiprocessing library extending Python’s multiprocessing module.
* Enables parallel execution of complex computations and ML workflows.

**24. tqdm**

* A fast, extensible progress bar library for tracking loop iterations.
* Often used in data processing, ML training, and file downloads.

**25. pyspellchecker**

* A simple spell-checking library using a Levenshtein distance algorithm.
* Supports identifying and correcting spelling errors in text data.