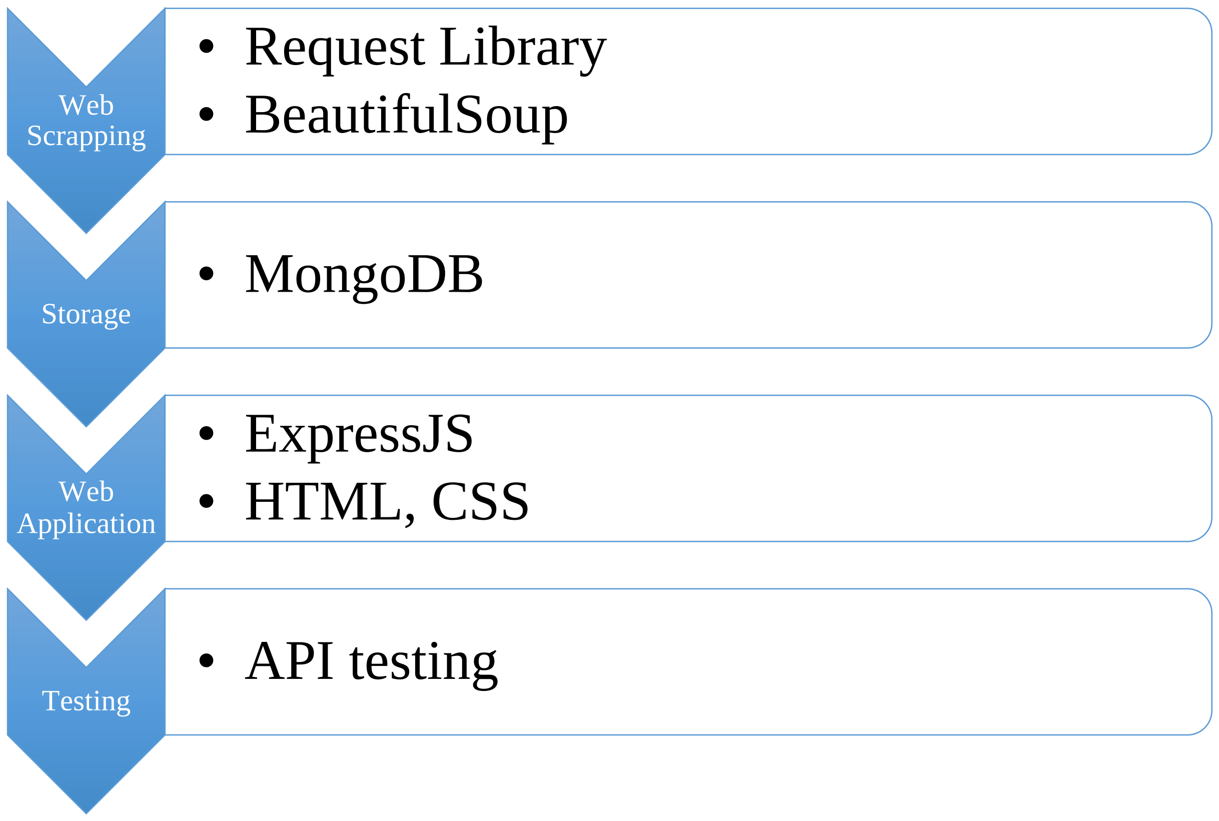
APPROACH

NYC Taxi Trip Data (February and March 2019) Web Application

This web application is designed to facilitate access to and analysis of NYC Taxi Trip data from February and March 2019. Built with Node.js and Express, it connects to a MongoDB database where trip records are stored. The application offers a RESTful API with various endpoints that enable users to retrieve all trips, filter by VendorID, calculate average fares, and analyze tolls and fares by different criteria.



1. Web Scraping

- **Fetch Data**: Uses the `requests` library to fetch the NYC TLC trip record data page.
- **Parse HTML**: Utilizes `BeautifulSoup` to parse the HTML content.
- **Extract Links**: Identifies and extracts links to Parquet files for February and March 2019.

2. MongoDB Connection

- **Connect to MongoDB**: Establishes a connection to a local MongoDB instance using `pymongo`.
- **Define Database**: Sets up a database and collection for storing the data ('records_data' and 'trip_records').

3. Parquet File Ingestion into MongoDB

- **DataFrame Creation**: Reads the Parquet file into a Pandas DataFrame using `pyarrow`.
- **Data Conversion**: Converts the DataFrame to a list of dictionaries suitable for MongoDB insertion.
- **Insert Records**: Inserts the records into the designated MongoDB collection.

4. Web Application:

- **Framework and Database**: Built with Node.js and Express, connected to a MongoDB database for storing NYC Taxi Trip data.
- **API Functionality**: Provides RESTful endpoints for retrieving trips, filtering by VendorID, calculating average fares, and analyzing tolls and fares.
- **Data Analysis**: Enables users to identify popular routes, peak usage times, and search for trips based on fare amounts or date ranges.
- **User Interaction**: Designed for easy data access and analysis, facilitating insights into taxi trip patterns and financial metrics.

5. API Testing

1. Retrieve Trips

- **Get All Trips**: Fetches all trips (limited to 19 by default).
- **Filter by VendorID**: Allows filtering trips based on VendorID.
- **Sort Trips**: Enables sorting by a specified field.
- **Total Trips**: Returns the total number of trips.

2. Calculate Statistics

- **Average Fare**: Retrieves average fare amount for a specific VendorID.

- **Tolls and Fares**: Calculates total tolls and fares grouped by trip distance (by VendorID).
- **Payment Type**: Computes total tolls and fares for a specified payment type.

3. High Fare Trips

- **Fetch High Fares**: Gets trips with fare amounts exceeding a specified value.

4. Date Range

- **Retrieve by Date**: Allows retrieving trips within a specified date range (requires startDate and endDate in YYYY-MM-DD format).