Data Schema and Scraping Methodology

Name – Kritnandan

Data Schema

1. The scraped dataset is stored as a CSV file named `zomato_full_data.csv`.
2. The file contains –
   a. Restaurant Name
   b. Address
   c. Status – like currently open or closed
   d. Operating Timing
   e. Contact Number
   f. Location
   g. Menu list with price

Data Collection Methodology

1. Tools and Techniques
   a. `Selenium`: Browser automation for dynamic page interaction
   b. `BeautifulSoup` (assumed present): HTML parsing
   c. `fake_useragent`: Rotation of User-Agent strings
   d. `CSV`: Writing structured data to CSV
   e. Programming language – Python

2. Workflow Summary
   a. Initialisation
      - Defined Zomato restaurant URLs.
      - Configured Chrome in headless mode.
      - Generated random user-agent strings.

   b. Page Interaction
      - Loaded pages using Selenium WebDriver.
      - Simulated scrolling to load dynamic content.
      - Waited for elements using `WebDriverWait`.

   c. Data Extraction
      - Parsed HTML with BeautifulSoup.

- Extracted elements by tag/class (e.g., `<h1>`, `<div>`, `<span>`).
- Matched dish names (<h4>) with prices (`<span>`).

d. Data Quality Handling

- Inserted placeholders for missing elements.
- Logged scraping errors with corresponding URLs.

e. Storage

- Appended records to a list of dictionaries.
- Exported data to CSV using UTF-8 encoding.

Limitation

- Fragility to Layout Changes: Breaks if Zomato Updates Its HTML Structure.
- Basic Error Handling: Logs errors but lacks detailed recovery options.
- Dish-Price Matching Assumption: Assumes a 1:1 order of <h4> and `<span>`, which may not consistently apply.