13 APIs P1

A Non-Programmer's Introduction to JSON

1. Understanding JSON Structure
    - json stands for javascript object notation
    - data is represented using key–value pairs
    - braces {} hold objects and brackets [] hold arrays
2. JSON Data Types
    - json supports strings, numbers, booleans, null, objects, and arrays
    - keys must be strings
    - values can be nested to form complex structures
3. Objects And Arrays
    - objects store named fields such as {“name”: “Jessica”}
    - arrays store ordered lists such as [1, 2, 3]
    - arrays can contain objects and objects can contain arrays
4. Human And Machine Friendly
    - json is easy for humans to read
    - json is easy for machines to parse
    - json is widely used for apis and web data exchange
5. Consistency And Formatting
    - json requires double quotes for strings and keys
    - trailing commas are not allowed
    - validation ensures json is properly formatted

Understanding URL

1. Purpose Of A URL
    - url identifies the location of a resource on the internet
    - url stands for uniform resource locator
    - clicking or entering it in a browser retrieves the resource
2. Main Components Of A URL
    - scheme specifies the protocol such as http or https
    - domain identifies the server hosting the resource
    - path points to the specific resource or file
3. Optional Components
    - port specifies which server port to connect to
    - query adds key–value data after ? for parameters
    - fragment identifies a specific section of the resource with #
4. URL Encoding
    - spaces and special characters must be encoded
    - encoding converts unsafe characters into % codes

    - browsers and parsers handle encoding automatically
5. Understanding How URLs Resolve
    - browser uses dns to translate domain names to ip addresses
    - the server responds with the requested resource
    - correct url structure ensures successful navigation

Elegant URL Handling With urltools

1. Working With URLs As Data
    - urltools provides tools for parsing and manipulating urls in r
    - urls are treated as structured data instead of plain strings
    - functions help extract, modify, and rebuild components
2. Parsing URL Components
    - url_parse() extracts scheme, domain, port, path, parameter, and fragment
    - output is a dataframe for easy inspection and manipulation
    - vectorized operations work across many urls at once
3. Query Parameter Tools
    - parameters can be extracted with param_get()
    - parameters can be set or modified with param_set()
    - param functions make working with api urls easier
4. URL Encoding And Decoding
    - url_encode() converts unsafe characters to % codes
    - url_decode() does the reverse
    - encoding ensures urls remain valid and standard compliant
5. Domain And Host Utilities
    - functions like suffix_extract() analyze domain structure
    - tools help identify subdomains, tlds, and suffixes
    - useful for web-scraped data and analytics tasks