Please find my answers for the questions below.

1.Do some research on the Python requests module. Explain what it is, and how it is used to make HTTP requests.

Ans: **Definition and usage**

The requests module allows you to send HTTP requests using Python. The HTTP request returns response object with all the response data (content, encoding, status, etc.)

The Python requests module is a library for making HTTP requests in Python. It is inbuilt library within python ecosystem, and it simplifies communication with web servers and APIs by providing an easy-to-use interface for sending requests and handling responses.

It allows users to send various types of HTTP requests—GET, POST, PUT, DELETE, PATCH etc. to web servers. GET requests retrieve data, POST requests submit data, PUT requests update resources, and DELETE requests remove resources.

As the python requests third party packages, users need to install their machine. The module abstracts away the complexity of HTTP communication, handling details like URL formatting, headers, and response parsing automatically. This means you can make requests with just a few lines of code instead of dealing with low-level networking libraries.

Rather than dealing with Python's lower-level networking tools, requests provide a clean, fast and straight forward way to communicate with web servers and APIs. It avoids the complexity of HTTP communication. This makes HTTPS requests in python very user friendly and most efficient.

Salient Features of Python HTTP request

The requests module is highly valued for its clarity, flexibility, and reliability in performing web operations with Python.

References:

https://realpython.com/python-requests/

https://www.w3schools.com/python/module_requests.asp

https://requests.readthedocs.io/en/latest/

2. Document your independent findings on the JSON and XML data formats and what they are used for. List at least four advantages and four disadvantages of each data format.

Ans: **JSON Definition and usage**

JSON (JavaScript object notation) is a lightweight, text-based data format that uses key-value pairs and arrays to represent structured data. It is language-independent and has become the standard format for data exchange in modern web applications.

XML Definition and usage

XML is a markup language designed to store and exchange structured data in a platform-independent manner. Unlike JSON, XML uses a tag-based structure where data is enclosed in user-defined tags, making the format highly extensible and descriptive. XML has been a cornerstone technology for data interchange in enterprise systems for decades.

Both **JSON (JavaScript Object Notation)** and **XML (Extensible Markup Language)** are text-based data formats used for structuring and exchanging data between applications, systems, or platforms.

Below are the advantages and disadvantages of JSON and XML files.

Advantages of JSON

a. Efficient and easy to parse, ensuring fast data exchanges
b. Lightweight and less verbose than XML, resulting in smaller file sizes
c. Easily readable by humans, making debugging and development simpler
d. Supported natively in most modern programming languages and platforms

Disadvantages of Json

a. Limited data types—no support for complex structures like attributes or comments
b. Does not support schema validation or error handling mechanisms by default
c. Can pose security risks if used with untrusted sources or services
d. Not ideal for highly hierarchical or self-describing data models

Advantages of XML

a. Platform and language independent for broad interoperability

b. Supports schema validation using DTD and XSD for robust error checking
c. Flexible, self-describing format accommodates complex, hierarchical structures
d. Unicode support enables data sharing in multiple languages

Disadvantages of XML

a. Verbose syntax results in much larger file sizes than JSON
b. Less readable and harder to debug compared to JSON
c. Inefficient for storage and transmission, especially for binary data
d. More complex to parse and process, requiring additional tools and knowledge

Here is a quick summary of **JSON against XML.**

Feature	**JSON**	**XML**
File Size	Smaller, lightweight	Larger, verbose
Parsing Speed	Fast	Slower
Readability	Human-readable, concise	Human-readable, verbose
Schema Support	Limited (external tools required)	Robust (XSD, DTD)
Data Types	Basic types only	Flexible but not native
Extensibility	Limited	Highly extensible
Comments	Not supported	Supported
Best For	Web APIs, mobile apps, real-time data	Enterprise systems, complex data validation

Conclusion

The choice between JSON and XML depends upon project or business need, legacy system compatibility, and the nature of data being exchanged etc.

References:

https://www.geeksforgeeks.org/html/difference-between-json-and-xml/

https://www.turing.com/kb/what-is-json

https://beginnersbook.com/2018/10/advantages-and-disadvantages-of-xml/

3. Provide a brief explanation of what a RESTful API is, how it works and what it is used for. List at least four advantages and four disadvantages of RESTful APIs.

Ans: **Restful API and working principles**

A RESTful API is a web service that follows the principles of REST (Representational State Transfer), an architectural style for building modern web applications. REST uses standard HTTP methods to enable communication between clients and servers over the web.

Post, Get, Put, Patch and Delete are most commonly used methods used in Rest APIs. RESTful APIs operate by having clients send HTTP requests to specific endpoints (URLs) through server. Each endpoint represents a resource (like a user, product, or document). The server processes the request and returns data, typically in JSON or XML format.

Advantages of RESTful APIs

1. **Simplicity and Ease of Use:** REST is less complex than older alternatives (like SOAP) and directly leverages the HTTP protocol, making it intuitive for developers to learn and implement.
2. **Scalability and Independence:** The **stateless** nature ensures that requests are self-contained. This allows for easy load balance and separate, independent development of the client and server.
3. **Performance (Caching):** REST explicitly supports **caching** of responses. If a resource hasn't changed, clients can retrieve the data from a local cache, which significantly reduces server load and latency.
4. **Flexibility in Data Format:** REST is **format-agnostic**, meaning it can support JSON, XML, or plain text, though JSON is the dominant choice for modern use due to its lightweight nature.

Disadvantages of RESTful APIs

1. **Over- and Under-fetching:** Clients often receive more data than they need (over-fetching) or must make multiple requests to get all necessary related data (under-fetching) because the API endpoint's response structure is fixed.
2. **Stateless Overhead:** The requirement for statelessness means that every request must carry authentication details and session context, slightly increasing the size of requests compared to stateful protocols.
3. **Lack of State/Intent:** Mapping complex, non-CRUD business operations (e.g., "Transfer Funds") to the simple HTTP verbs can be challenging and sometimes unintuitive.

4. **Inefficiency for Real-Time:** The strict request-response cycle is less suitable for real-time applications or push notifications, where other architectures (like WebSocket's) are preferred.

References

https://www.oreilly.com/library/view/hands-on-restful-web/9781838643577/becffd48-bca9-4c3a-9d89-0a490c5dce49.xhtml

https://www.techtarget.com/searchapparchitecture/definition/RESTful-API

https://www.geeksforgeeks.org/node-js/rest-api-introduction/