1. **Intro to REST —**
   Representational State Transfer, or REST, is discussed in the video. An architectural style for creating networked applications is called REST. It is predicated on a collection of limitations that specify the identification, representation, and manipulation of resources.

   The statelessness of REST, which allows each request to be independent of all previous requests, is one of its main advantages. Because of this, REST applications are highly cacheable and scalable.

   The uniformity of REST, which ensures that all resources are handled consistently, is an additional advantage. This facilitates the learning and use of REST applications.

   Numerous networked applications can be designed using the potent architectural style of REST. When developing a web application, I strongly advise you to take REST into consideration.

2. **HTTP Request & Response Headers —**
   HTTP requests and responses are accompanied by HTTP headers, which are metadata. They offer details about the request or response, including the encoding, content type, and caching guidelines. HTTP headers are necessary for the correct operation of the web because they facilitate communication between the client and the server.

   There are many different HTTP headers, but some of the most common ones include:
   - Accept: This header tells the server what content types the client can accept.
   - Accept-Encoding: This header tells the server what encoding formats the client can accept.
   - Cache-Control: This header tells the server how to cache the response.
   - Content-Type: This header tells the client what type of content the response is.
   - Host: This header tells the server the hostname of the requested resource.
   - User-Agent: This header tells the server the user agent of the client.
   - There are many different tools that can be used to debug HTTP headers. Some of the most popular tools include:

Browser developer tools: Most browsers have built-in developer tools that can be used to view HTTP headers.

- Fiddler: Fiddler is a proxy tool that can be used to capture and inspect HTTP traffic.
- Charles Proxy: Charles Proxy is another proxy tool that can be used to capture and inspect HTTP traffic.

3. **What is a cookie —**

   This video is about cookies. Cookies are little files that hold useful data about your preferences and yourself. A website may save a cookie on your computer when you visit it. It can read the cookie and remember your preferences, including language and the items in your shopping cart, the next time you visit that website.

   Cookies can also be used to track your activity online. For example, if you visit a news website, the ads on that website may be able to identify you and show you ads that you are more likely to be interested in.

   Cookies can be used for both good and bad things, such as tracking your online activities without your consent and remembering your preferences to make your online experience more convenient.

   A website's creators are ultimately responsible for deciding what information they store, what they do not store, and most importantly, how they use it. Cookies are just a tool, and like any tool, they can be used for good or bad.

4. **HTTP Status codes —**

   When a client makes a request to the server, what happens next?
   HTTP status codes tell us whether a request that we made to the server is successful or has failed.
   More about the codes:

   - The 1xx series lets us know about the informational requests
   - The 2xx success messages
   - 3xx redirection
   - 4xx client error
   - 5xx server error

5. **HTTP Proxy —**
   A proxy server obscures the client's IP address or their identity. It acts as an intermediary between you and the websites you visit, protecting your IP address and other personal information.

   This is what a proxy server can do:
   - Protect your identity: When you visit a website, the website can see your IP address, which can be used to track your online activity. A proxy server hides your IP address from the websites you visit, making it more difficult for them to track you.
   - Block malicious traffic: Proxy servers can be used to block malicious traffic, such as viruses and malware. This can help to protect your computer from infection.
   - Block websites: Proxy servers can be used to block websites that you don't want your employees or children to access. This can be helpful for businesses and parents who want to control their users' internet access.
   - Improve performance: Proxy servers can cache content, which means that they can store copies of websites that you have visited. This can make it faster to load websites that you have visited before.

6. **Authentication with HTTP —**
   When we log in to a website, we are essentially providing our username and password to the server. The server then checks to see if the username and password are valid. If they are, the server will create a session token and send it back to us. This session token is a unique identifier that tells the server who we are.
   The next time we make a request to the server, we will need to include the session token in your request. This tells the server that we are the same person who logged in earlier. The server will then use the session token to look up our account information and grant us access to the resources that we are requesting.

   Session tokens are typically stored in cookies, which are small pieces of data that are stored on the computer. This means that we do not need to enter our username and password every time we visit a website. The browser will automatically send the session token to the server for us.

   Session tokens are a secure way to authenticate users because they are difficult to forge. They are also relatively short-lived, so they do not pose a significant security risk if they are compromised.

7. **HTTP basic and digest authentication —**
   This video explains how basic authentication works in a spring-based application. Basic authentication is the simplest HTTP authentication mechanism. It works by sending the username and password in clear text over the network. This is not a secure way to authenticate users, so it should only be used over a secure channel, such as HTTPS.