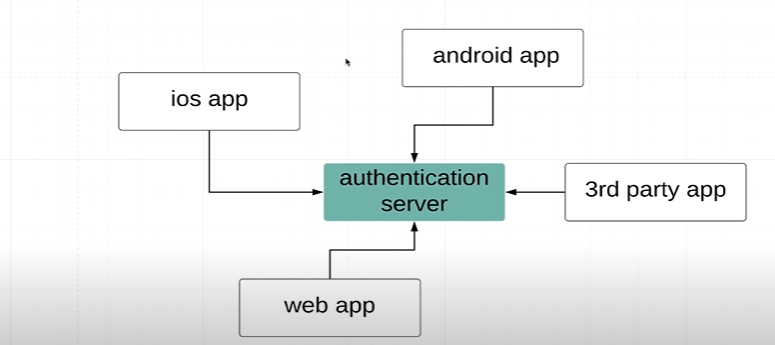
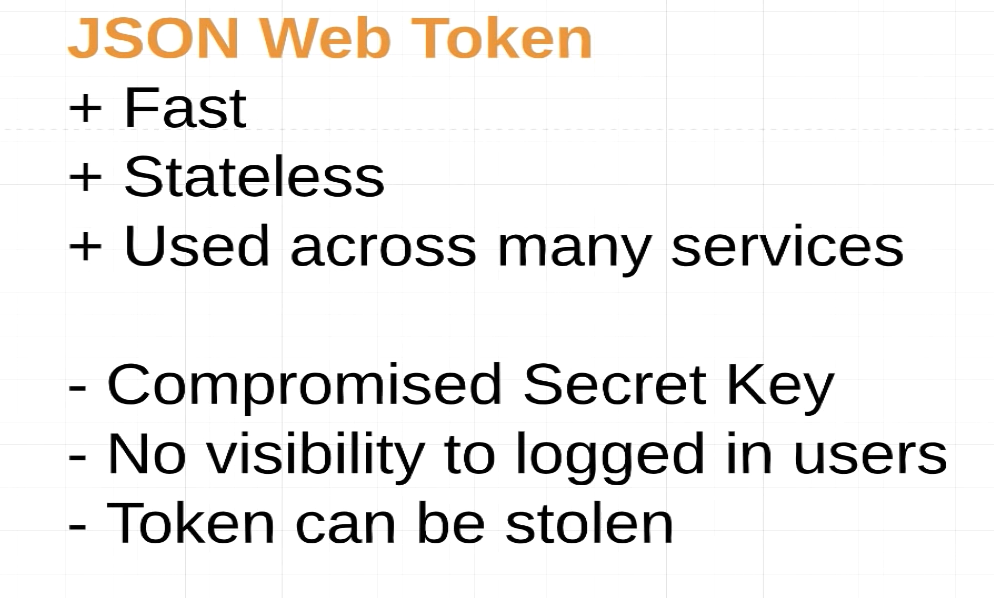
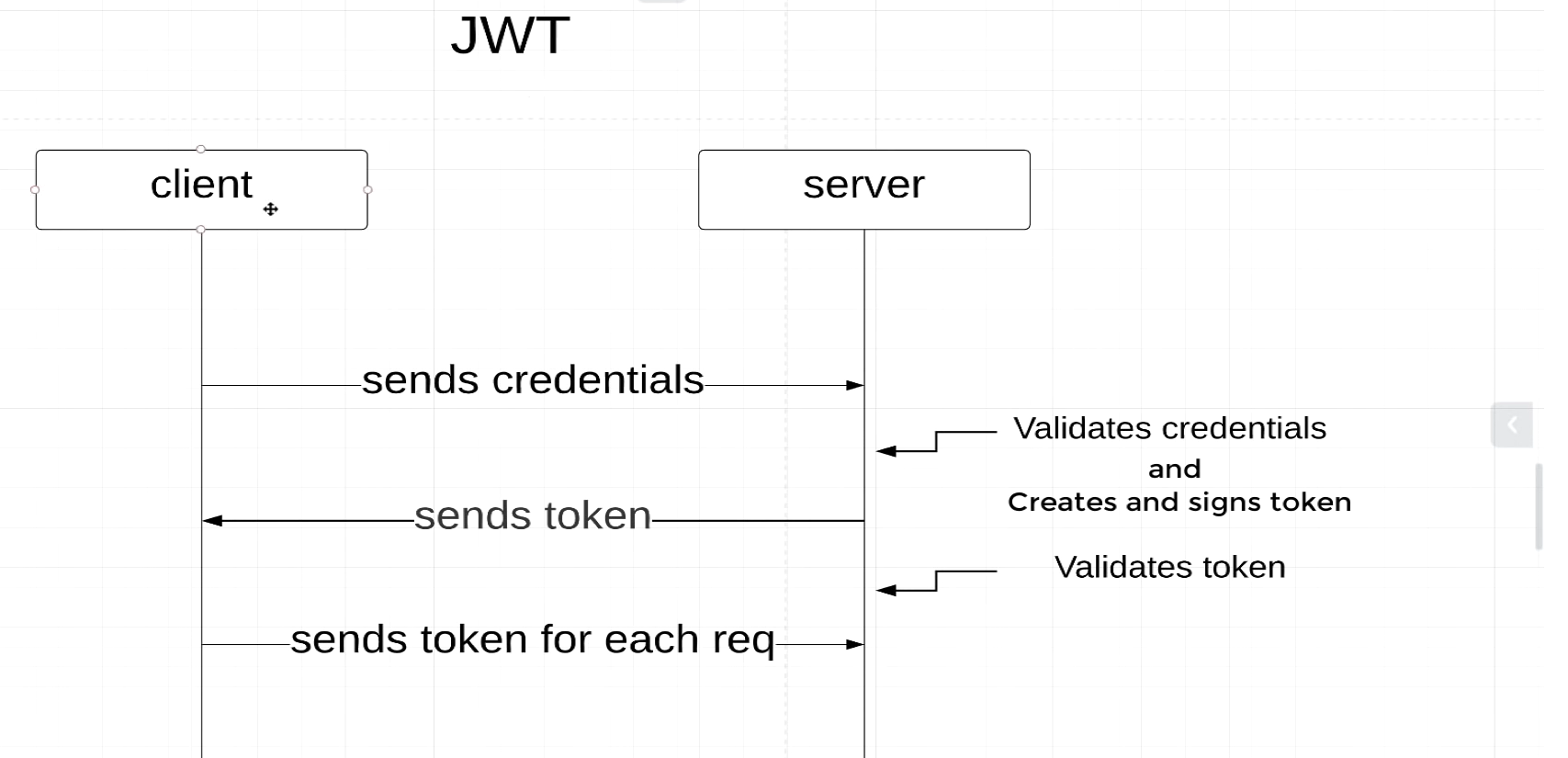
-----------------------------X--------------------------------------

What is JWT?

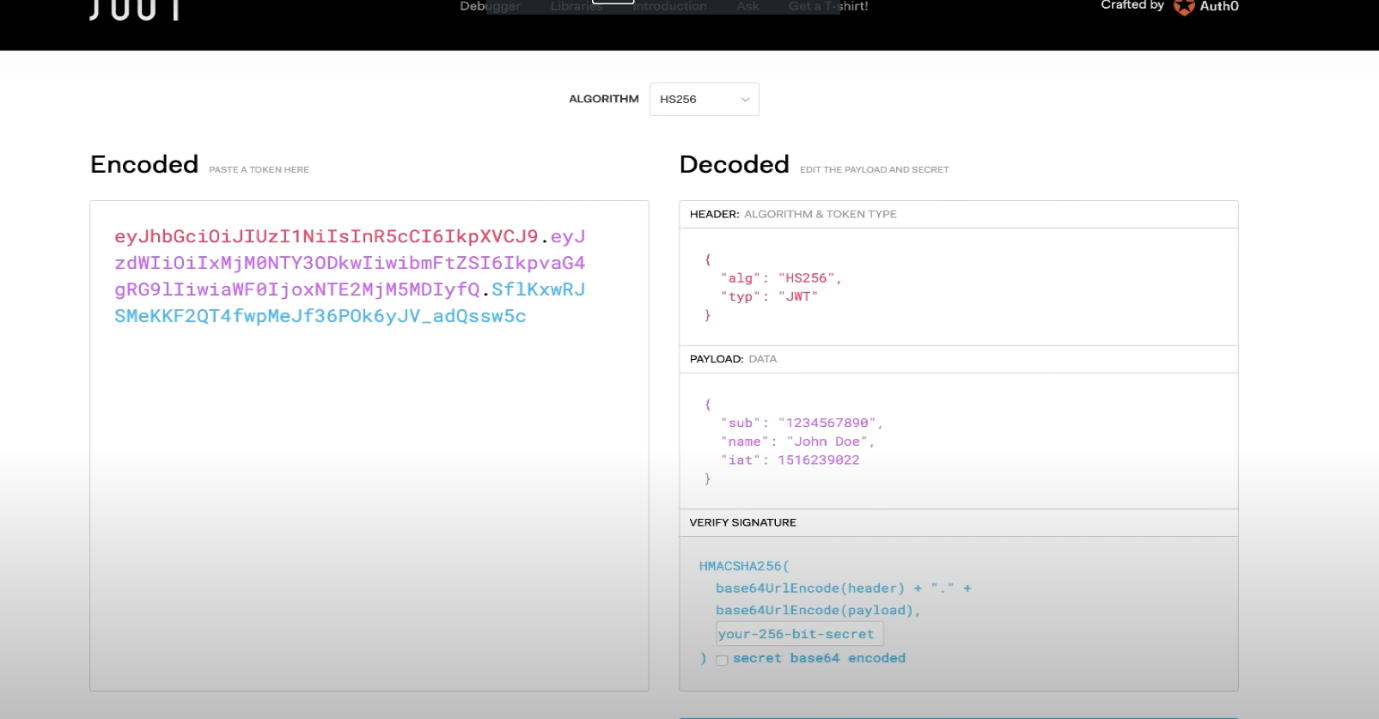
Ans-







<https://jwt.io/introduction>



* Currently we want to connet to real DB.
* Using outh qualifieres

-----------------------------------X---------------------------------

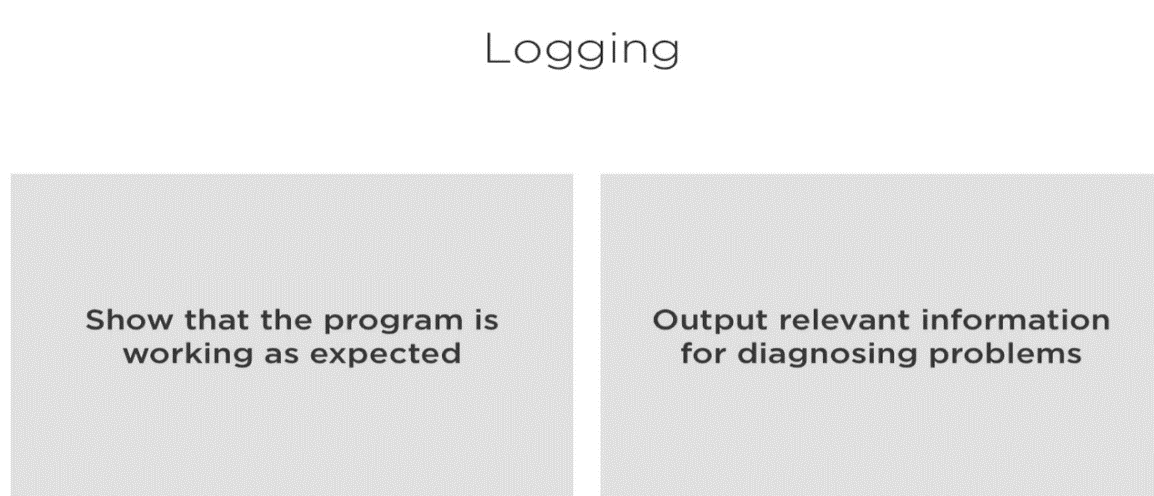
What is CSRF?

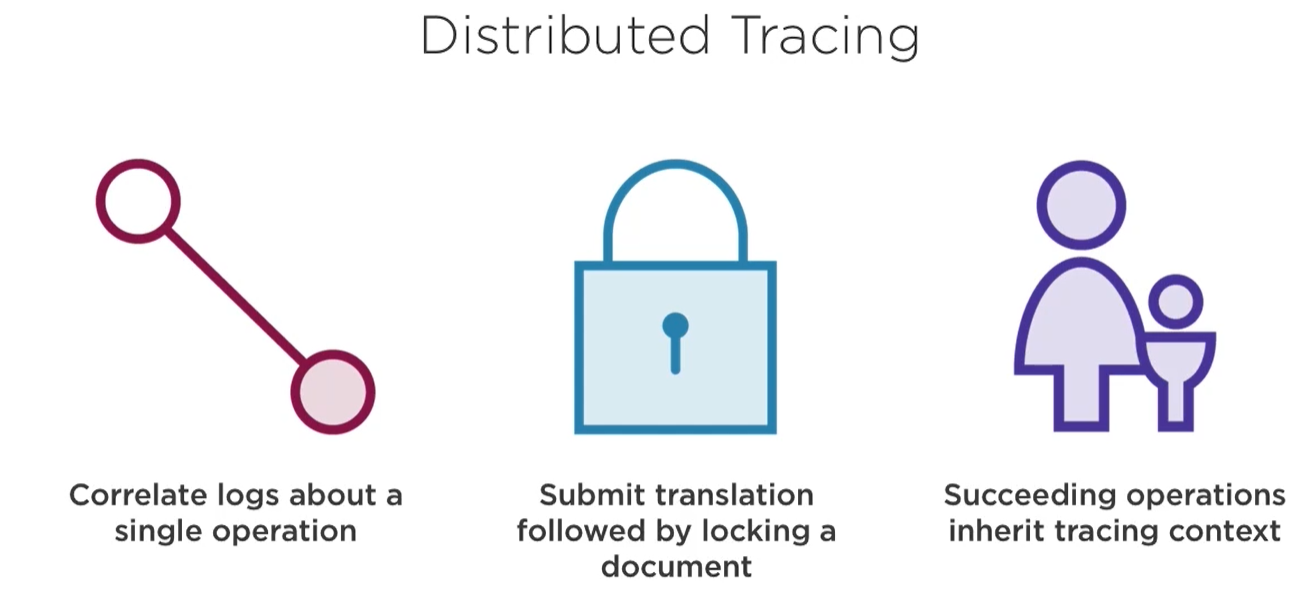
Ans-

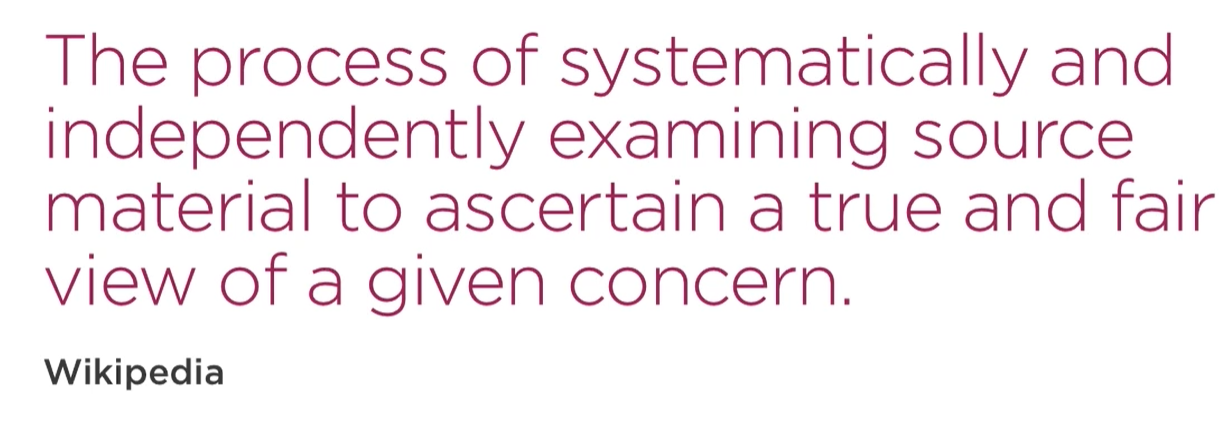
----------------------------------------X--------------------------------

**What is Logging and Distributed tracing and Audit?**

Ans – This helps developer understand code and follow a request for complex service to service communication





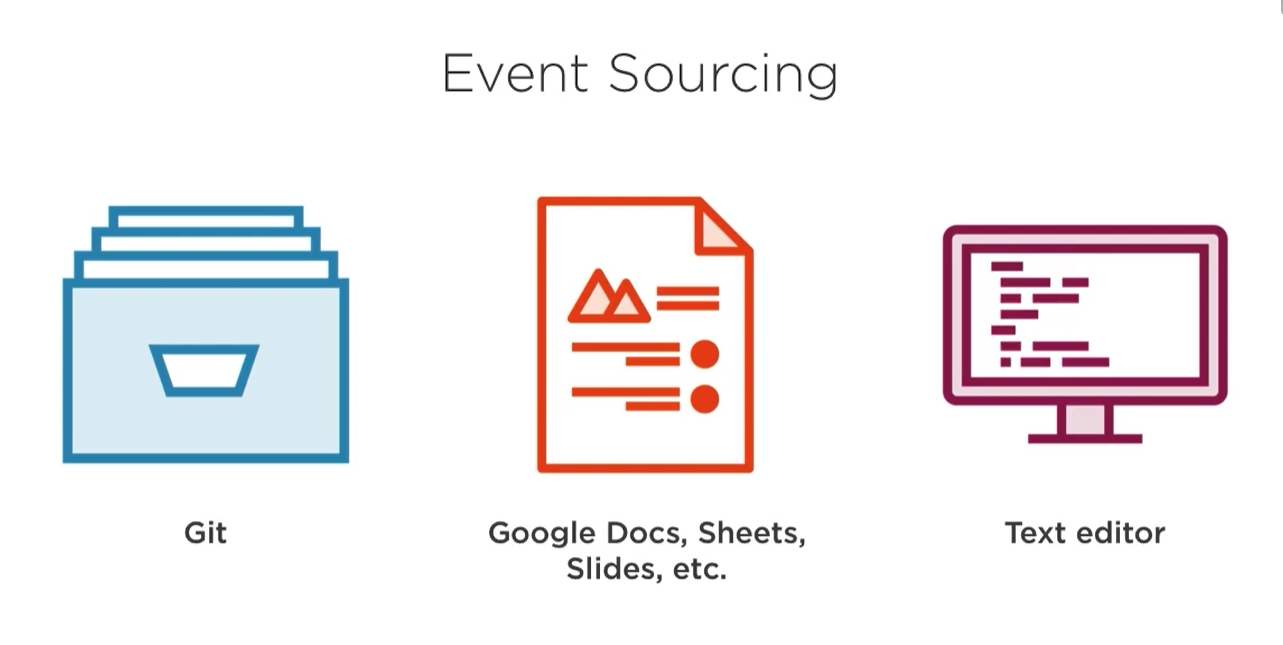


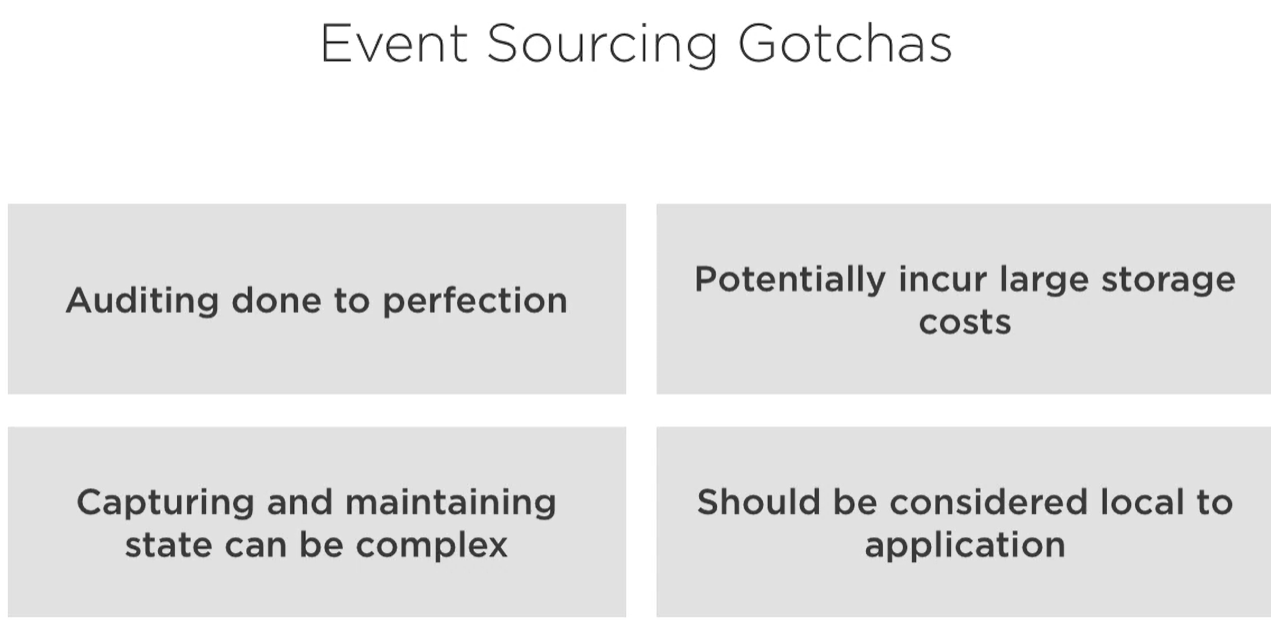


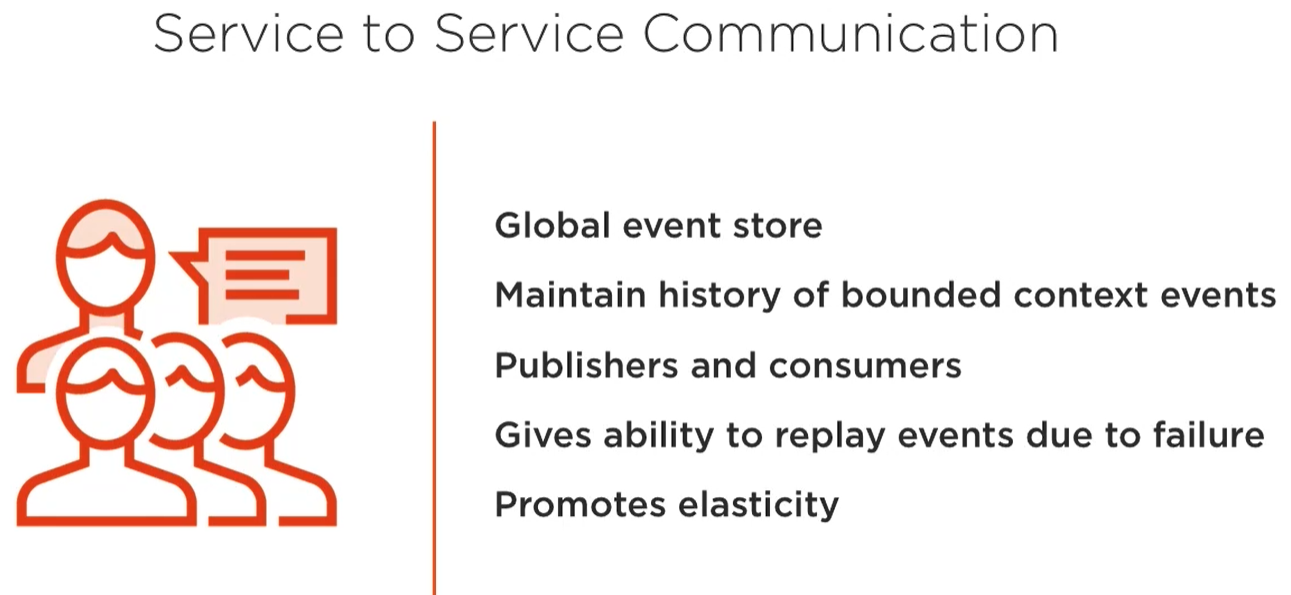
------------------------------------X-------------------------------------------

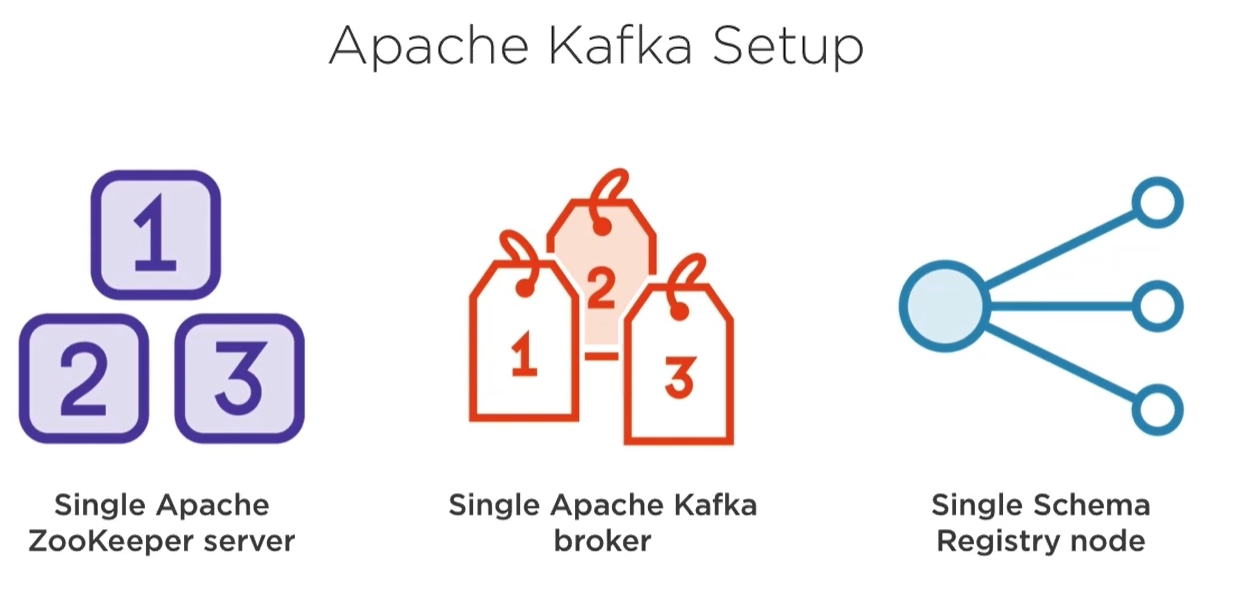
**What is Event Sourcing?**

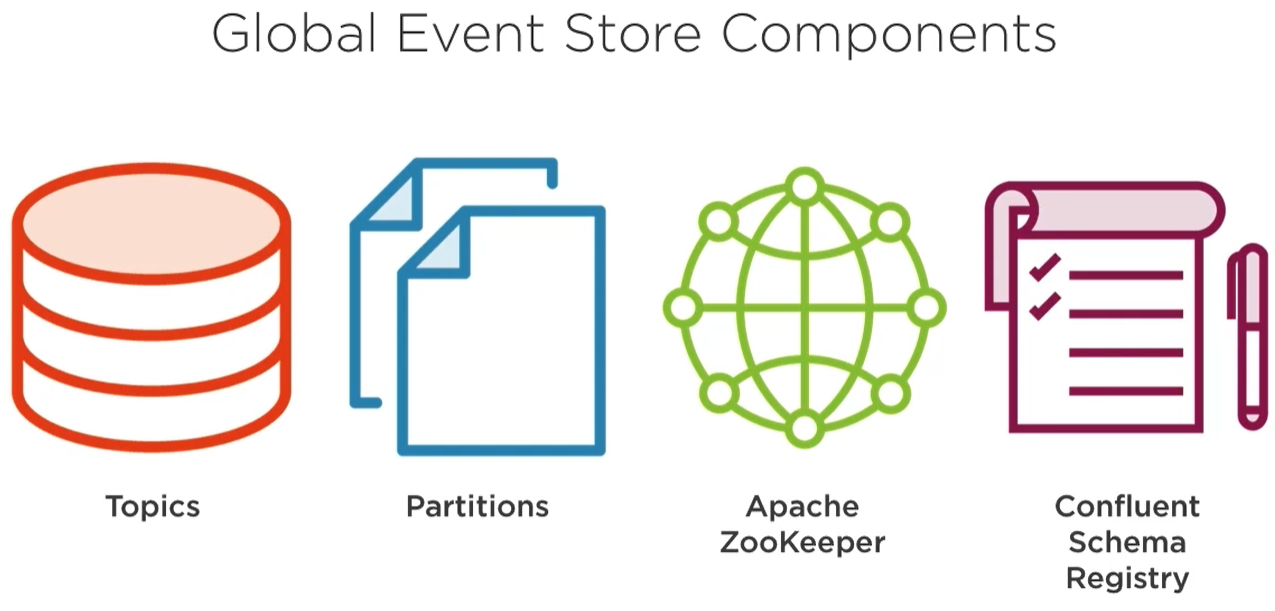
**Ans-**

****

****

****

****



* **We can store domain base event in domain base topics**
* **In case of heave traffic in schema registry and we will do the cashing for load balancing.**

-------------------logstach.cong file-----------------------

## **ture of a config file**[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A Logstash config file has a separate section for each type of plugin you want to add to the event processing pipeline. For example:

# This is a comment. You should use comments to describe

# parts of your configuration.

input {

...

}

filter {

...

}

output {

...

}

Each section contains the configuration options for one or more plugins. If you specify multiple filters, they are applied in the order of their appearance in the configuration file.

### Plugin configuration[edit](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

The configuration of a plugin consists of the plugin name followed by a block of settings for that plugin. For example, this input section configures two file inputs:

input {

file {

path => "/var/log/messages"

type => "syslog"

}

file {

path => "/var/log/apache/access.log"

type => "apache"

}

}

In this example, two settings are configured for each of the file inputs: path and type.

The settings you can configure vary according to the plugin type. For information about each plugin, see [Input Plugins](https://www.elastic.co/guide/en/logstash/current/input-plugins.html), [Output Plugins](https://www.elastic.co/guide/en/logstash/current/output-plugins.html), [Filter Plugins](https://www.elastic.co/guide/en/logstash/current/filter-plugins.html), and [Codec Plugins](https://www.elastic.co/guide/en/logstash/current/codec-plugins.html).

### Value types[edit](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A plugin can require that the value for a setting be a certain type, such as boolean, list, or hash. The following value types are supported.

### Array[edit](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

This type is now mostly deprecated in favor of using a standard type like string with the plugin defining the :list => true property for better type checking. It is still needed to handle lists of hashes or mixed types where type checking is not desired.

Example:

users => [ {id => 1, name => bob}, {id => 2, name => jane} ]

#### Lists[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

Not a type in and of itself, but a property types can have. This makes it possible to type check multiple values. Plugin authors can enable list checking by specifying :list => true when declaring an argument.

Example:

path => [ "/var/log/messages", "/var/log/\*.log" ]

uris => [ "http://elastic.co", "http://example.net" ]

This example configures path, which is a string to be a list that contains an element for each of the three strings. It also will configure the uris parameter to be a list of URIs, failing if any of the URIs provided are not valid.

#### Boolean[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A boolean must be either true or false. Note that the true and false keywords are not enclosed in quotes.

Example:

ssl\_enable => true

#### Bytes[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A bytes field is a string field that represents a valid unit of bytes. It is a convenient way to declare specific sizes in your plugin options. Both SI (k M G T P E Z Y) and Binary (Ki Mi Gi Ti Pi Ei Zi Yi) units are supported. Binary units are in base-1024 and SI units are in base-1000. This field is case-insensitive and accepts space between the value and the unit. If no unit is specified, the integer string represents the number of bytes.

Examples:

my\_bytes => "1113" # 1113 bytes

my\_bytes => "10MiB" # 10485760 bytes

my\_bytes => "100kib" # 102400 bytes

my\_bytes => "180 mb" # 180000000 bytes

#### Codec[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A codec is the name of Logstash codec used to represent the data. Codecs can be used in both inputs and outputs.

Input codecs provide a convenient way to decode your data before it enters the input. Output codecs provide a convenient way to encode your data before it leaves the output. Using an input or output codec eliminates the need for a separate filter in your Logstash pipeline.

A list of available codecs can be found at the [Codec Plugins](https://www.elastic.co/guide/en/logstash/current/codec-plugins.html) page.

Example:

codec => "json"

#### Hash[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A hash is a collection of key value pairs specified in the format "field1" => "value1". Note that multiple key value entries are separated by spaces rather than commas.

Example:

match => {

"field1" => "value1"

"field2" => "value2"

...

}

# or as a single line. No commas between entries:

match => { "field1" => "value1" "field2" => "value2" }

#### Number[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

Numbers must be valid numeric values (floating point or integer).

Example:

port => 33

#### Password[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A password is a string with a single value that is not logged or printed.

Example:

my\_password => "password"

#### URI[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A URI can be anything from a full URL like [*http://elastic.co/*](http://elastic.co/) to a simple identifier like foobar. If the URI contains a password such as [*http://user:pass@example.net*](http://user:pass@example.net/) the password portion of the URI will not be logged or printed.

Example:

my\_uri => "http://foo:bar@example.net"

#### Path[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A path is a string that represents a valid operating system path.

Example:

my\_path => "/tmp/logstash"

#### String[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A string must be a single character sequence. Note that string values are enclosed in quotes, either double or single.

#### Escape sequences[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

By default, escape sequences are not enabled. If you wish to use escape sequences in quoted strings, you will need to set config.support\_escapes: true in your logstash.yml. When true, quoted strings (double and single) will have this transformation:

|  |  |
| --- | --- |
| Text | Result |
| \r | carriage return (ASCII 13) |
| \n | new line (ASCII 10) |
| \t | tab (ASCII 9) |
| \\ | backslash (ASCII 92) |
| \" | double quote (ASCII 34) |
| \' | single quote (ASCII 39) |

Example:

name => "Hello world"

name => 'It\'s a beautiful day'

#### Field reference[**edit**](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

A Field Reference is a special [String](https://www.elastic.co/guide/en/logstash/current/configuration-file-structure.html#string) value representing the path to a field in an event, such as @timestamp or [@timestamp] to reference a top-level field, or [client][ip] to access a nested field. The [Field References Deep Dive](https://www.elastic.co/guide/en/logstash/current/field-references-deepdive.html) provides detailed information about the structure of Field References. When provided as a configuration option, Field References need to be quoted and special characters must be escaped following the same rules as [String](https://www.elastic.co/guide/en/logstash/current/configuration-file-structure.html#string).

### Comments[edit](https://github.com/elastic/logstash/edit/7.15/docs/static/configuration.asciidoc)

Comments are the same as in perl, ruby, and python. A comment starts with a # character, and does not need to be at the beginning of a line. For example:

# this is a comment

input { # comments can appear at the end of a line, too

# ...

}