{JS}

Clase 41

JS

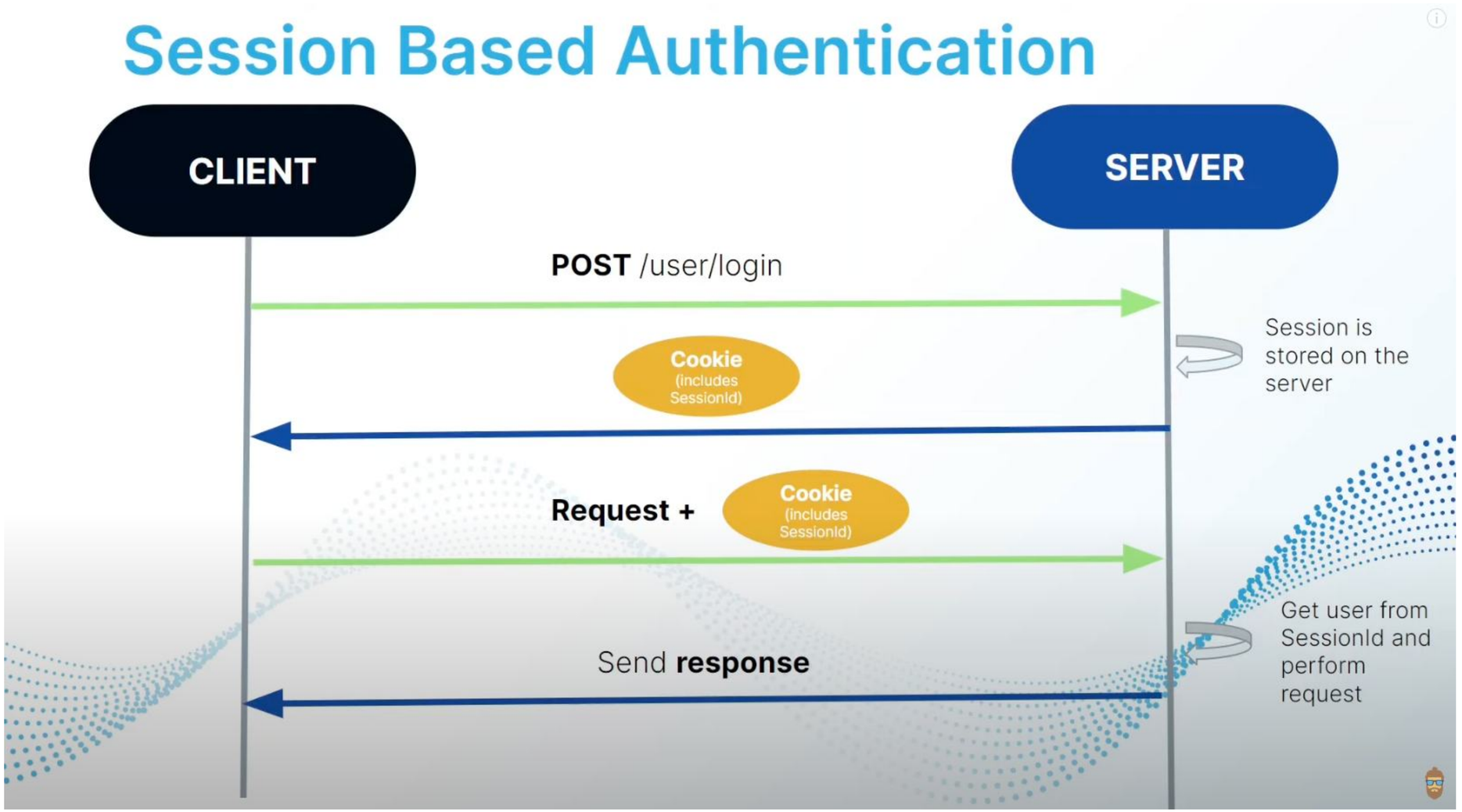
‹Índice›

Índice

Autenticación vs Autorización

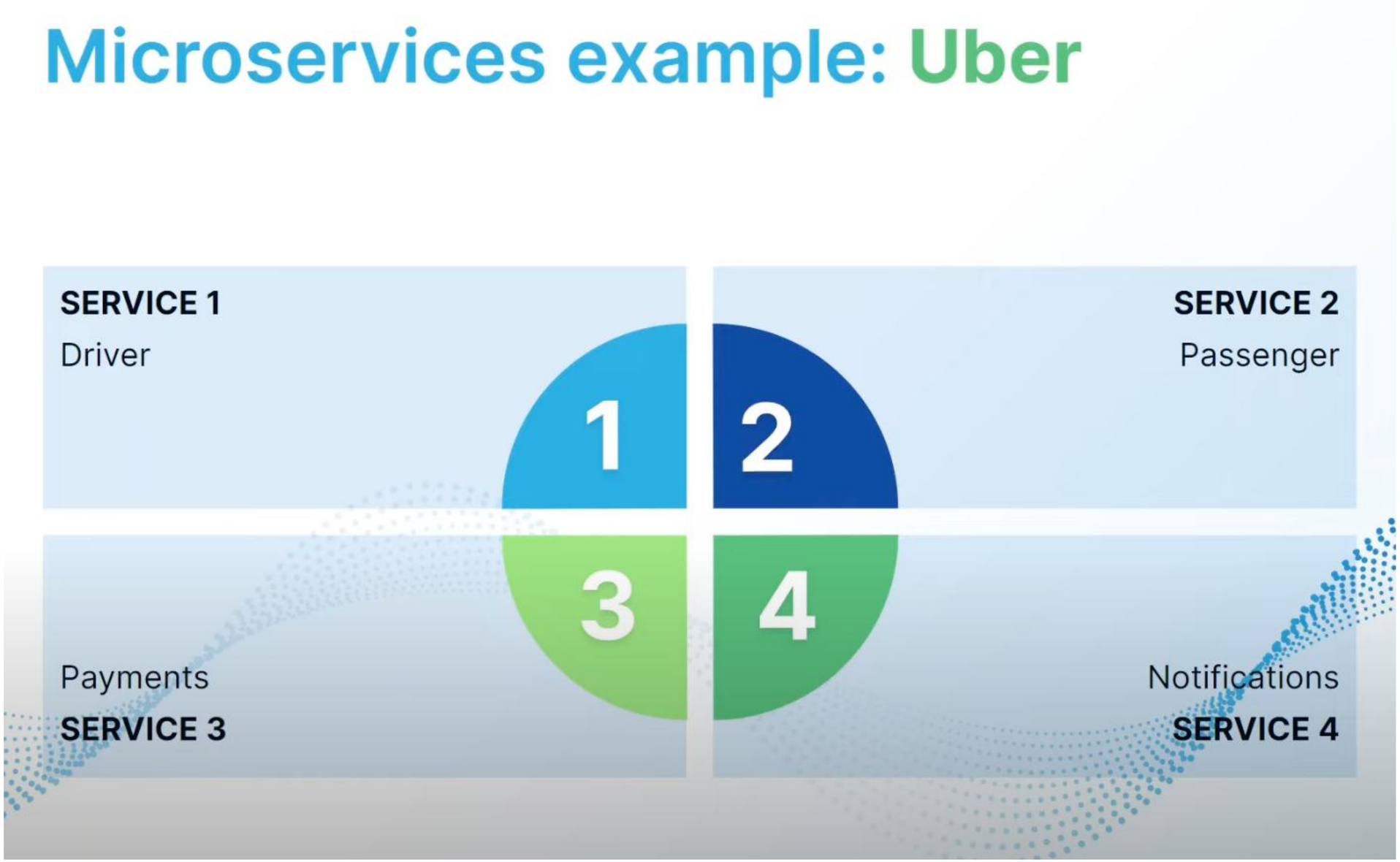
Autenticación basada en sesiones

API

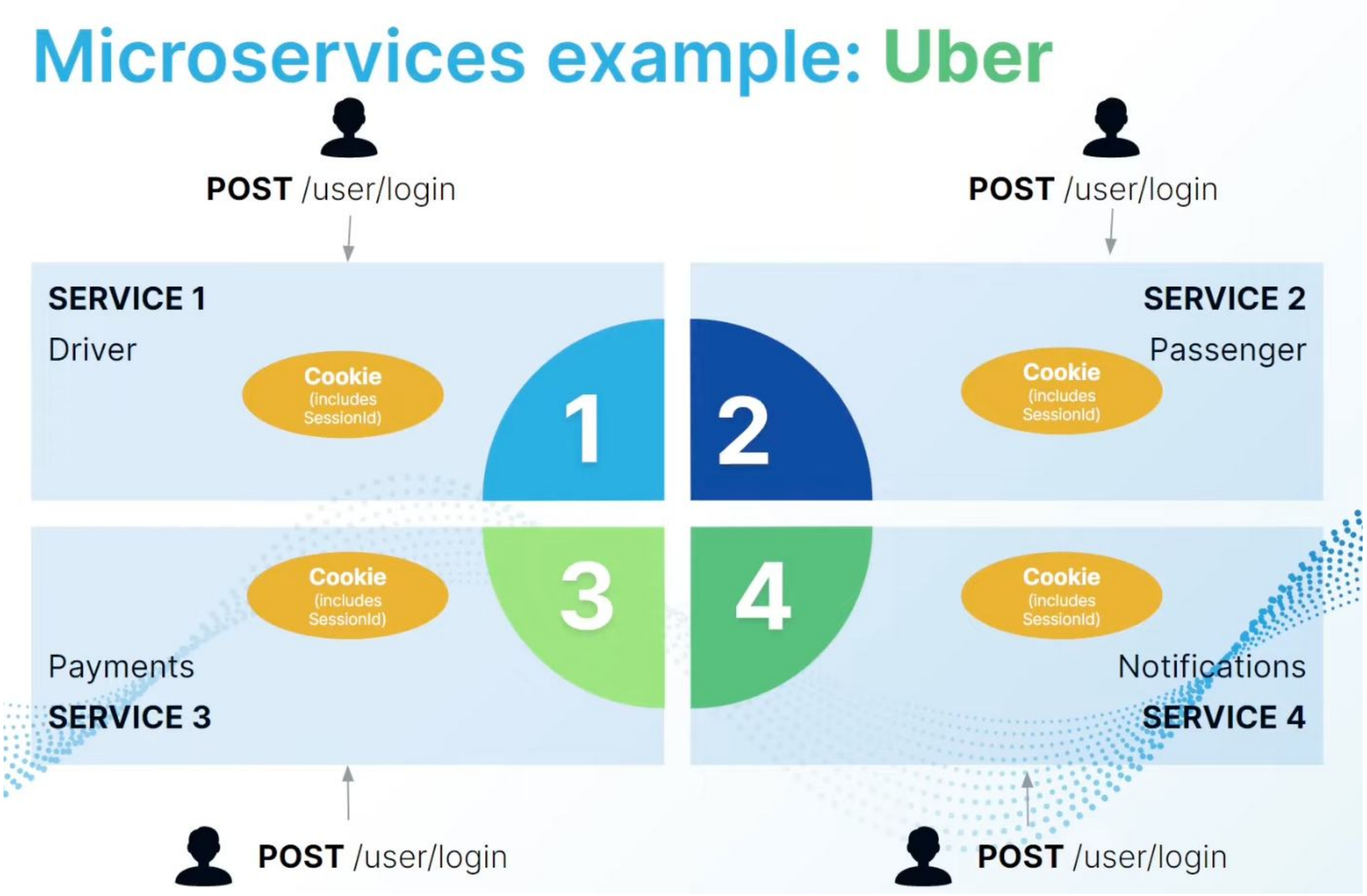

Session Based Authentication
CLIENT
SERVER
POST /user/login
Cookie
(includes
SessionId)
Session is
stored on the
server
Request +
Cookie
(includes
SessionId)
Get user from
SessionId and
perform
request
Send response

API

Microservices example: Uber

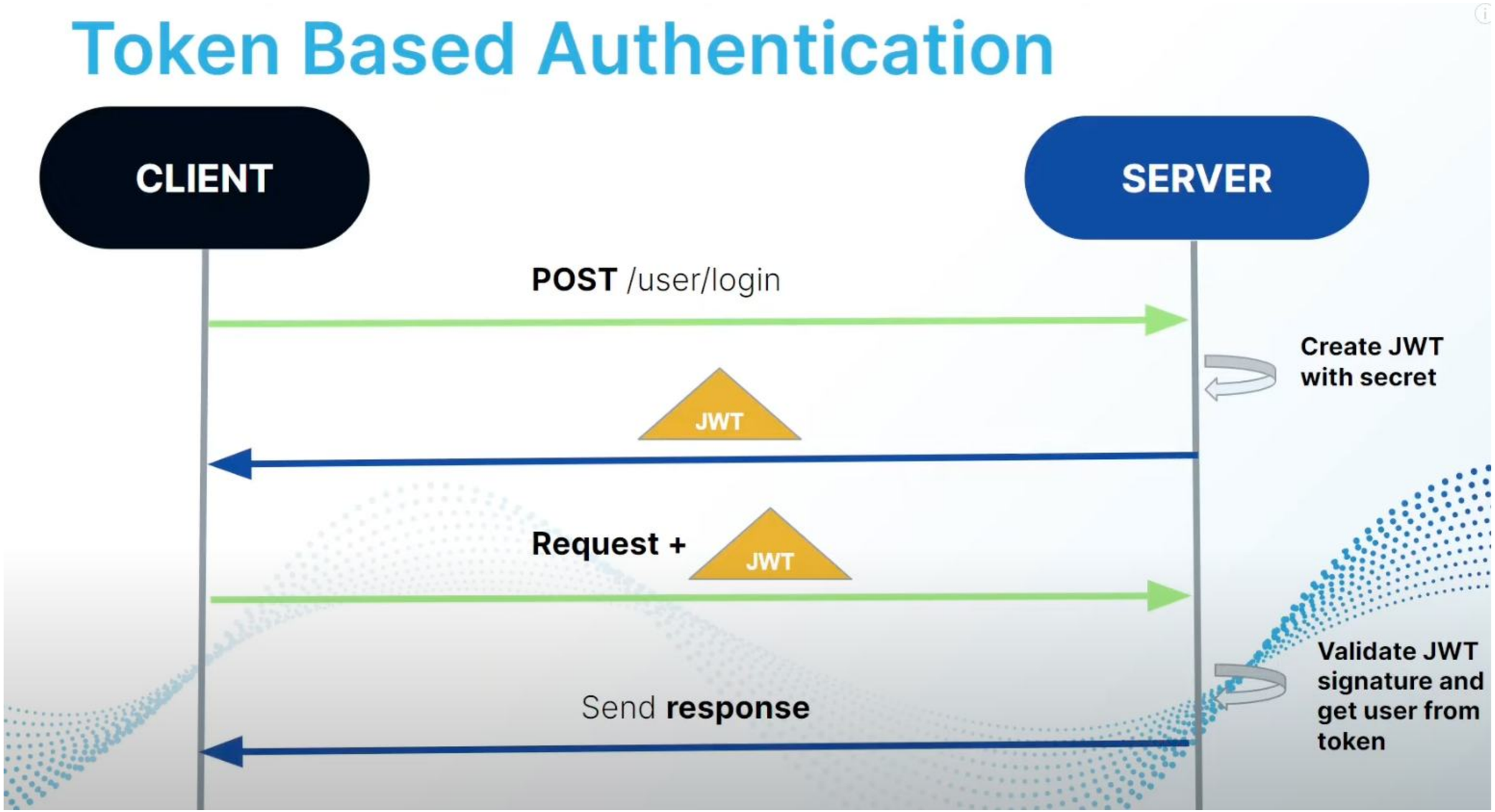


API


Microservices example: Uber
POST /user/login
POST /user/login
SERVICE 1
Driver
Cookie
(includes
SessionId)
1
2
SERVICE 2
Passenger
Cookie
(includes
SessionId)
Cookie
(includes
SessionId)
3
4
Cookie
(includes
SessionId)
Payments
SERVICE 3
Notifications
SERVICE 4
POST /user/login
POST /user/login

Autenticación basada en tokens

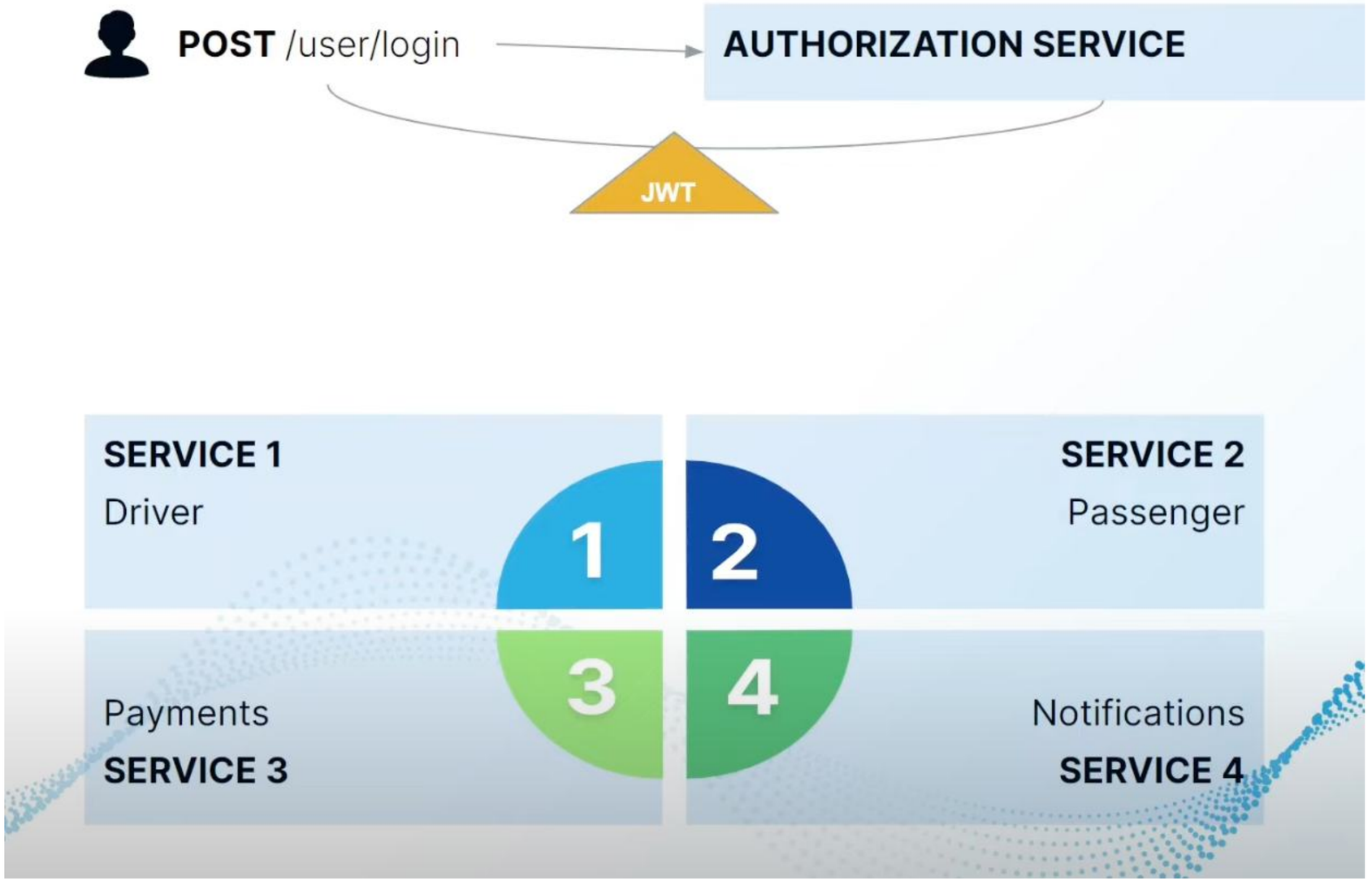
API


Token Based Authentication
CLIENT
SERVER
POST /user/login
Create JWT with secret
JWT
Request +
JWT
Validate JWT signature and get user from token
Send response

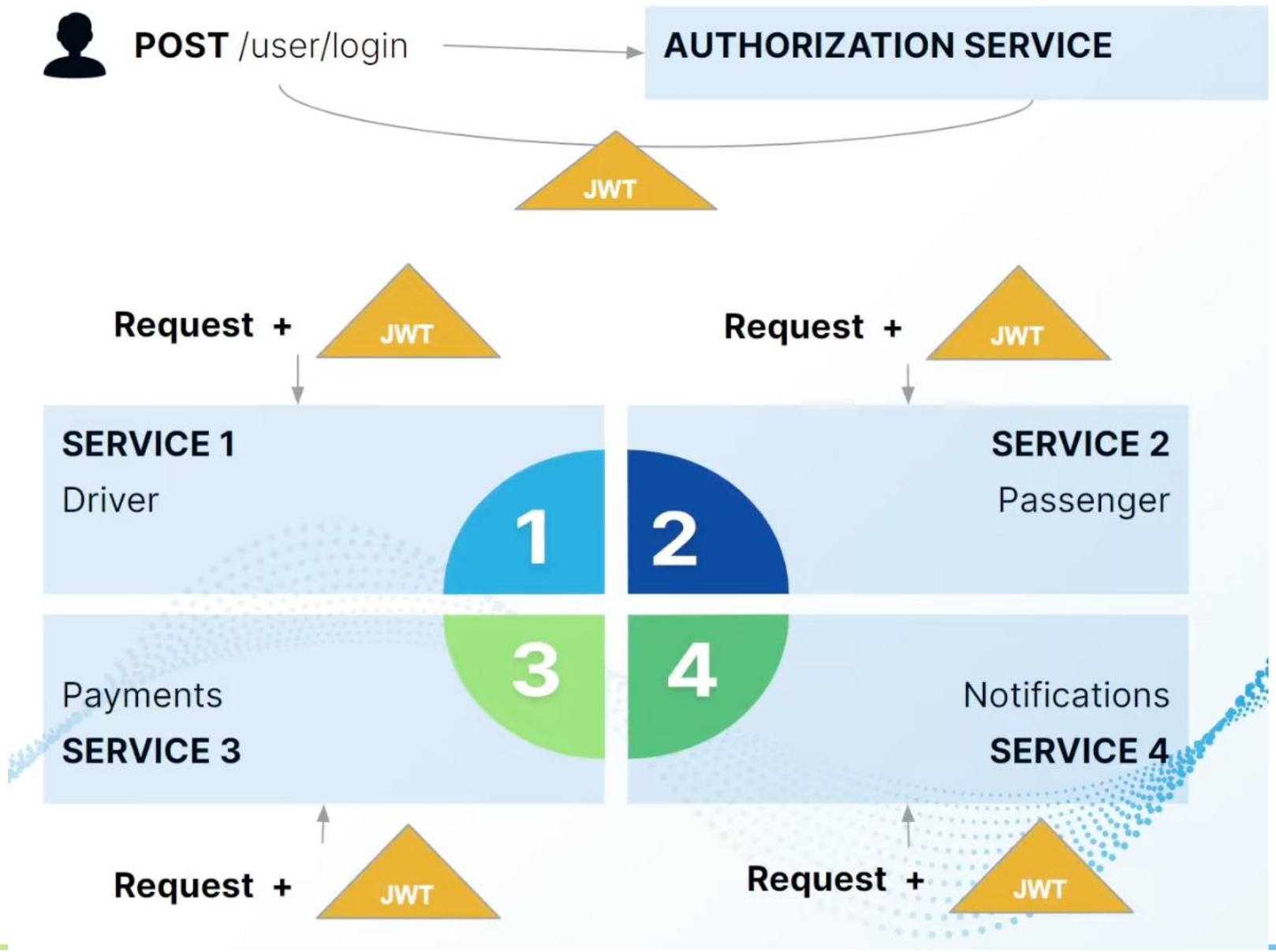
API

Microservices example: Uber

API


POST /user/login
AUTHORIZATION SERVICE
JWT
SERVICE 1
Driver
SERVICE 2
Passenger
1
2
3
4
Payments
SERVICE 3
Notifications
SERVICE 4

API


POST /user/login
AUTHORIZATION SERVICE
JWT
Request +
JWT
Request +
JWT
SERVICE 1
Driver
1
2
SERVICE 2
Passenger
3
4
Payments
SERVICE 3
Notifications
SERVICE 4
Request +
JWT
Request +
JWT

API

Cifrado

Sesiones VS JWT

Sesiones VS JWT

Entendamos cómo funciona HTTP

Sesiones VS JWT

La autenticación por sesiones y la autenticación con tokens JWT (JSON Web Tokens) son dos métodos comunes para mantener el estado de la sesión de un usuario en aplicaciones web. Ambos tienen sus propias características, ventajas e inconvenientes. Aquí te explico las principales diferencias entre ambos métodos

Sesiones VS JWT

Diferencias principales

Sesiones VS JWT

Almacenamiento de estado

Sesiones VS JWT

- Sesiones: La autenticación por sesiones mantiene el estado del usuario en el servidor. Se crea un archivo o un registro en una base de datos para cada sesión donde se almacena la información del usuario. El cliente solo almacena un identificador de sesión (generalmente en una cookie), que el servidor utiliza para recuperar el estado almacenado.

Sesiones VS JWT

- JWT: La autenticación JWT es stateless (sin estado). No se almacena información del usuario en el servidor. En cambio, el servidor genera un token que contiene todos los datos necesarios, codificados y posiblemente cifrados, que el cliente envía en cada solicitud. El servidor lee y verifica este token para obtener la información del usuario.

Sesiones VS JWT

Escalabilidad

Sesiones VS JWT

- Sesiones: Como el estado se almacena en el servidor, la escalabilidad puede ser un problema, especialmente en aplicaciones distribuidas donde se necesitan mecanismos como el sticky session o bases de datos de sesión compartidas.

Sesiones VS JWT

- JWT: Es más escalable en sistemas distribuidos ya que el servidor no necesita mantener el estado de la sesión. Cada solicitud contiene toda la información necesaria en el token.

Sesiones VS JWT

Seguridad

Sesiones VS JWT

- Sesiones: Son relativamente seguras si se implementan correctamente, como configurar las cookies de sesión para que sean HttpOnly (no accesibles por JavaScript) y Secure (transmitidas solo a través de HTTPS).

Sesiones VS JWT

- JWT: Puede ser vulnerable si no se maneja adecuadamente. La información sensible en el token debe ser cifrada, no solo codificada. Los tokens son susceptibles a ataques si se interceptan, ya que contienen toda la información necesaria para autenticarse.

Sesiones VS JWT

Ventajas de las Sesiones

Sesiones VS JWT

- Fácil de implementar con muchas frameworks y bibliotecas.
- Más control sobre la sesión, ya que el servidor puede invalidar sesiones fácilmente.
- La información del usuario no se expone al cliente, solo se almacena un ID de sesión.

Sesiones VS JWT

Inconvenientes de las sesiones

Sesiones VS JWT

- Requiere más recursos del servidor, ya que necesita almacenar información de la sesión.
- Menos eficiente en aplicaciones distribuidas a menos que se implementen soluciones adicionales para la gestión de sesiones.

Sesiones VS JWT

Ventajas de JWT

Sesiones VS JWT

- No requiere almacenamiento de estado en el servidor, lo que simplifica la arquitectura en sistemas distribuidos.
- Facilita el control de acceso y la autorización en diferentes servicios y microservicios.
- Puede ser utilizado en diferentes tipos de clientes, como aplicaciones móviles, web y de escritorio.

Sesiones VS JWT

Inconvenientes de JWT

Sesiones VS JWT

- Requiere un manejo cuidadoso de la seguridad, especialmente en la generación y almacenamiento del token.
- Los tokens no pueden ser invalidados fácilmente. Una vez emitidos, son válidos hasta que expiren.
- La información del token puede volverse obsoleta si los datos del usuario cambian y el token aún no ha expirado.

https://jwt.io/

¿Probamos?

‹Despedida›

Email

bienvenidosaez@gmail.com

Instagram

@bienvenidosaez

Youtube

youtube.com/bienvenidosaez