

Maximiliano Firtman  
@firt firt.dev

# FullStack Authentication

---



Frontend *Masters*

**mobile+web developer & trainer**

A grayscale world map is shown in the background. The country of Argentina is highlighted in a bright red color. The name "Argentina" is written in a bold, red, sans-serif font directly below the highlighted country.

**Argentina**

Maximiliano Firtman  
@firt firt.dev



PWA

- HTML since 1996
- JavaScript since 1998
- Authored 13 books
- Published 150+ webapps

# Let's Start!



# What we'll cover

Authentication

State of Techniques

Login Forms

New Flows

Data Structure

Web Services

Web Authentication

Passlinks

# Pre-requisites

Questions?

FullStack Authentication

1

# Introduction



Authentication

Authorization

Authentication (Authn)  
verifies the identity of a  
user or service.

We will discuss  
authentication on the  
Web

# Concepts

**Credentials**

**Single Sign On (SSO)**

**2FA (Two-factor authentication)**

**MFA (Multi-factor authentication)**

**OAuth 2.0**

**JWT (JSON Web Token)**

**OTP (One Time Password)**

**Public Key Cryptography**

# Implementing Authentication

Custom Auth

User/pass

WebAuthn

Identity Providers

OpenID

SAML 2.0

Sign in with...

Identity As a Service  
IDaaS



FullStack Authentication

2

# Authentication on the Web

# Security Risks

**Man in the Middle Attacks**

**Keyloggers**

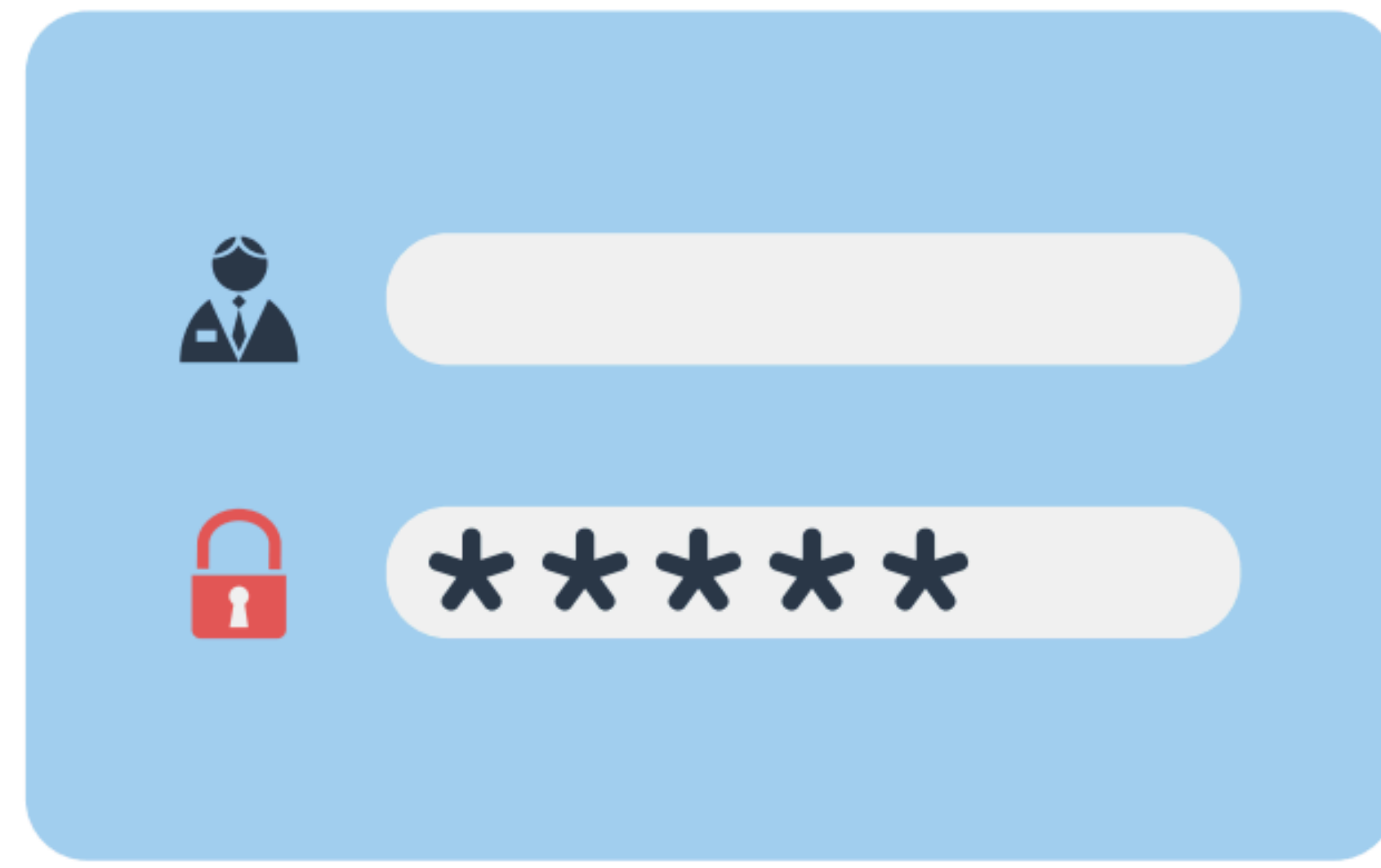
**Easy to guess passwords**

**Web Servers and DBs Attacks**

**Phishing and Social Engineering Attacks**

# HTTP Auth / Login Forms

Browser



Web Server



User



DB



# HTTP Auth / Login Forms

Browser



User

Web Server



DB

# HTTP Auth / Login Forms

Browser



Web Server



User

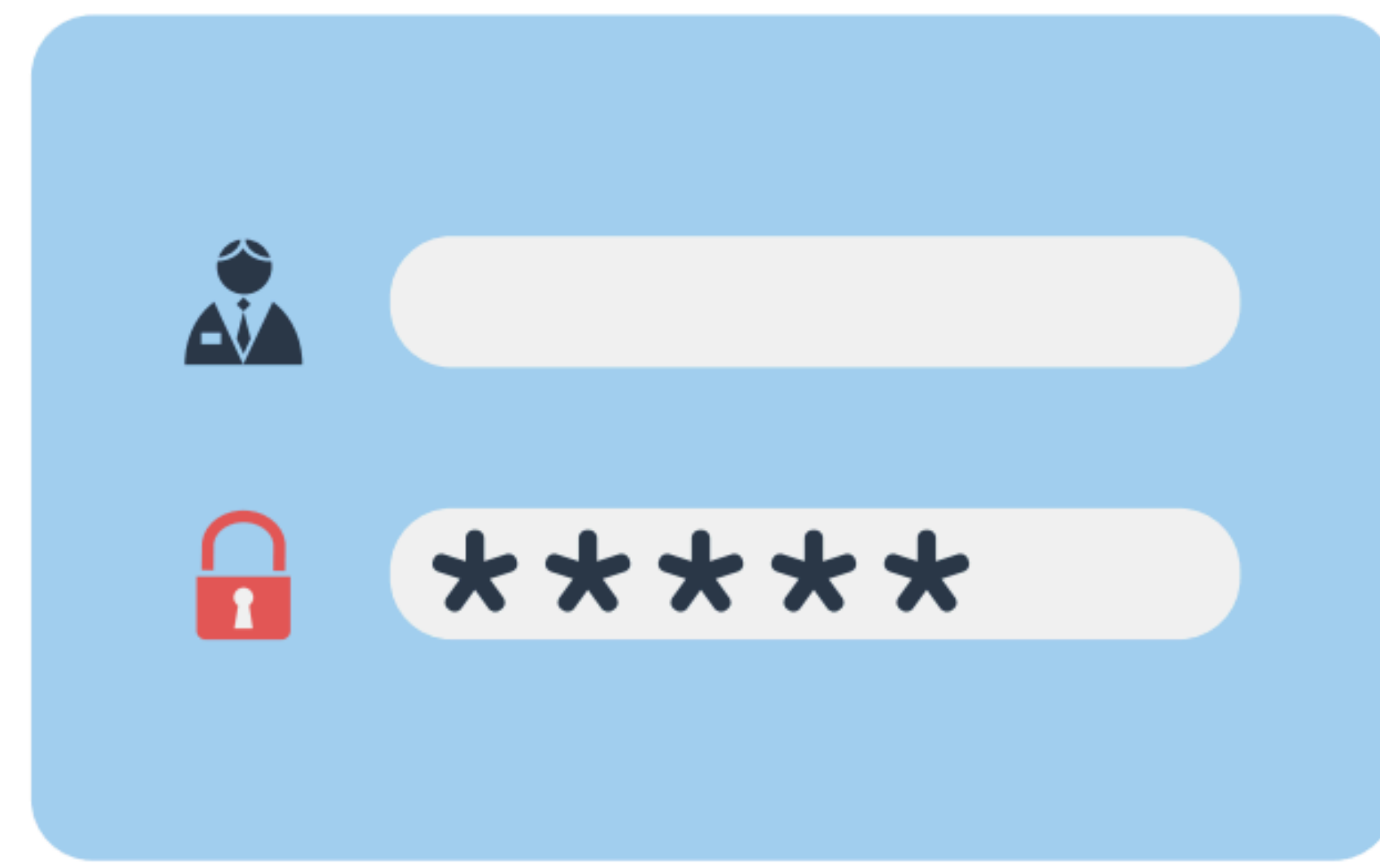


DB

# HTTP Auth / Login Forms

Browser

Web Server



HTTP

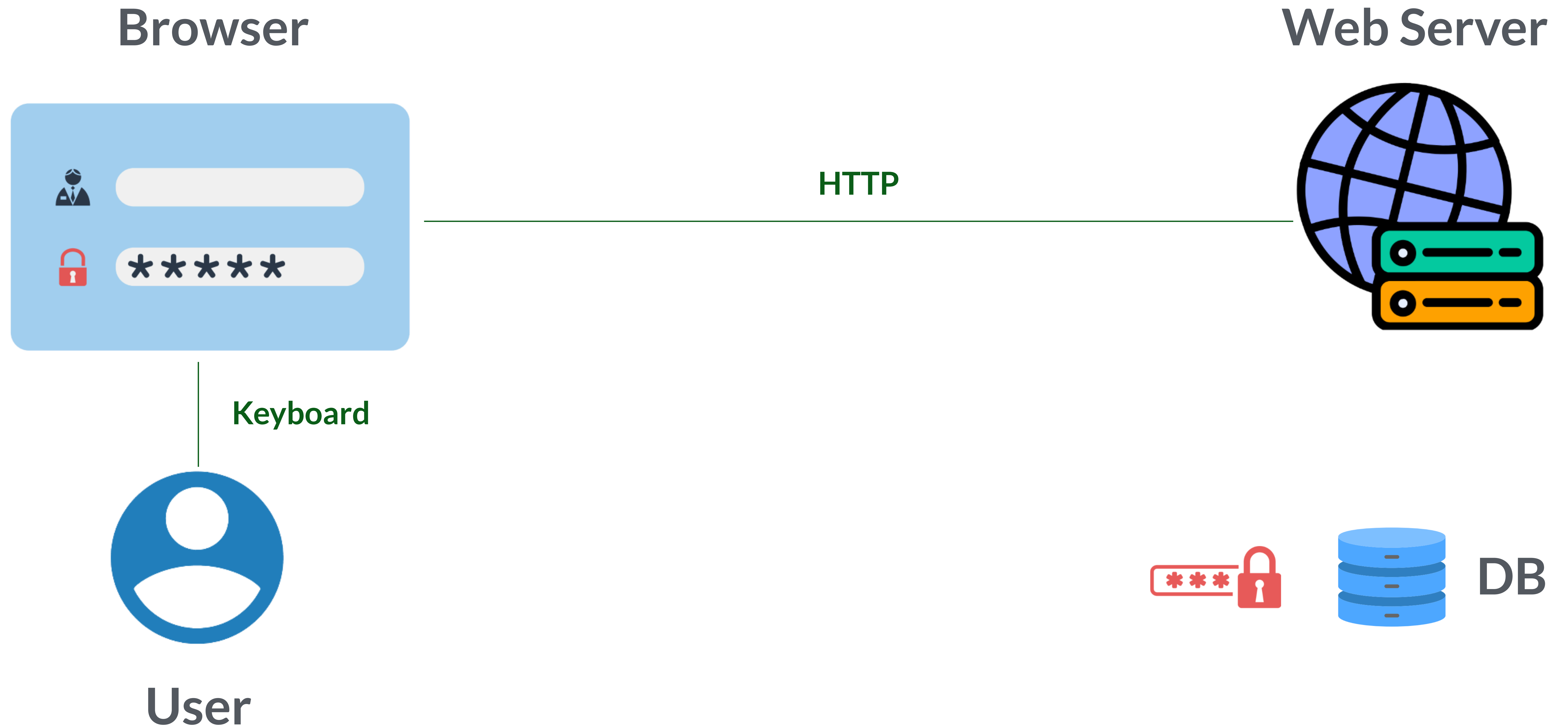


User

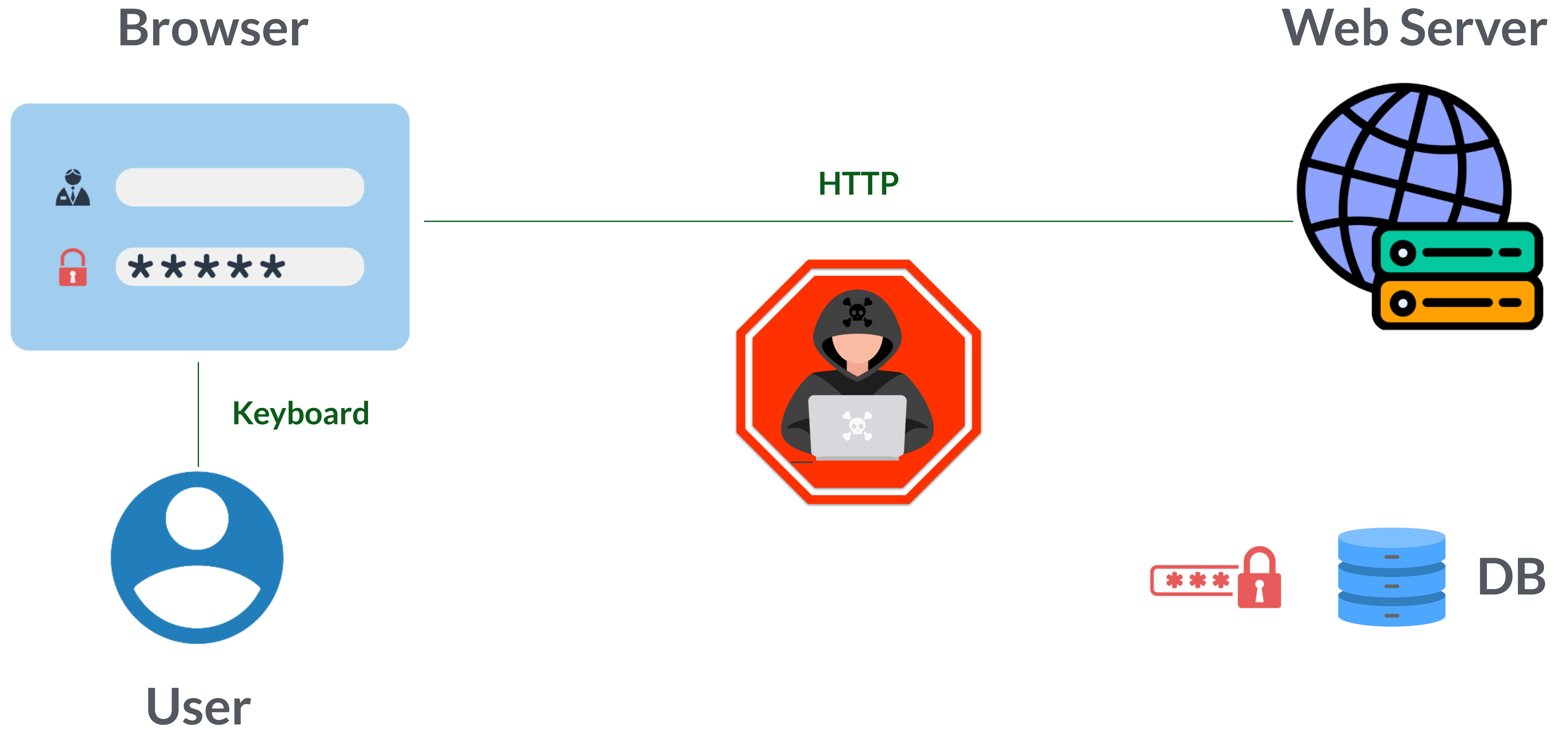


DB

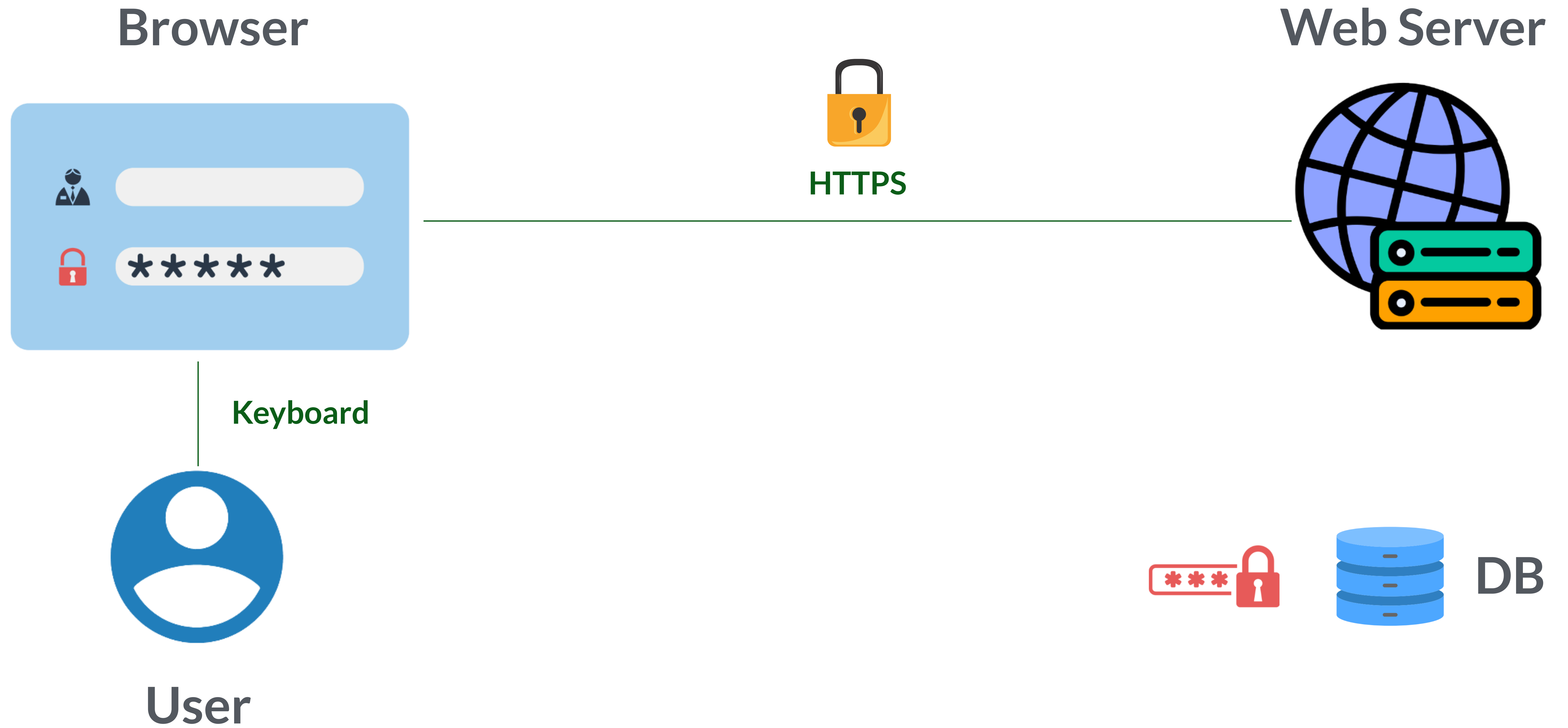
# HTTP Auth / Login Forms



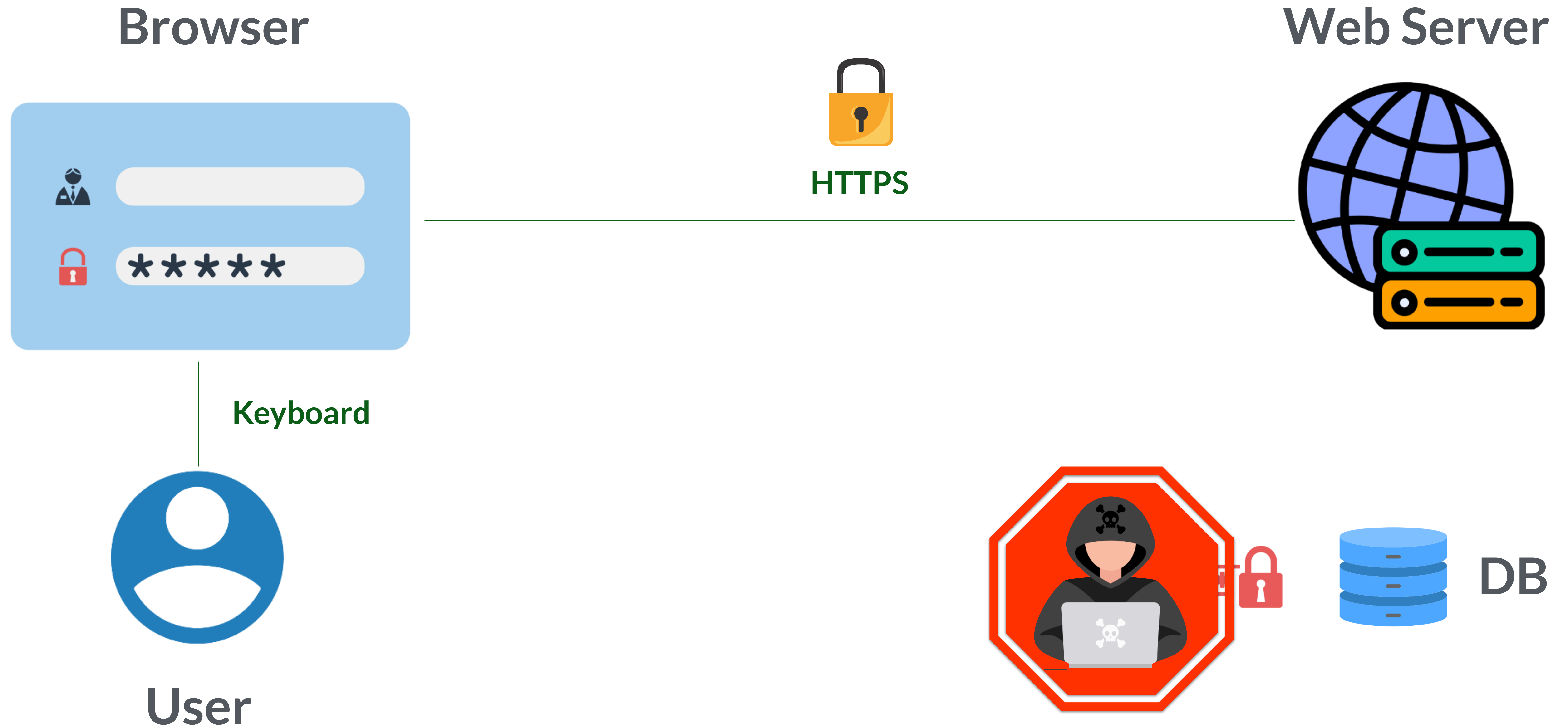
# HTTP Auth / Login Forms



# HTTP Auth / Login Forms



# HTTP Auth / Login Forms



# HTTP Auth / Login Forms





Easy to guess passwords  
and reuse of passwords  
are another risk

Password managers helps  
with both problems but  
still not perfect

Workshop time

**haveibeenpwned.com**

# Multi-factor Authentication

# Multi-factor Authentication

Browser



Web Server



User



DB

# Multi-factor Authentication

Browser



Web Server



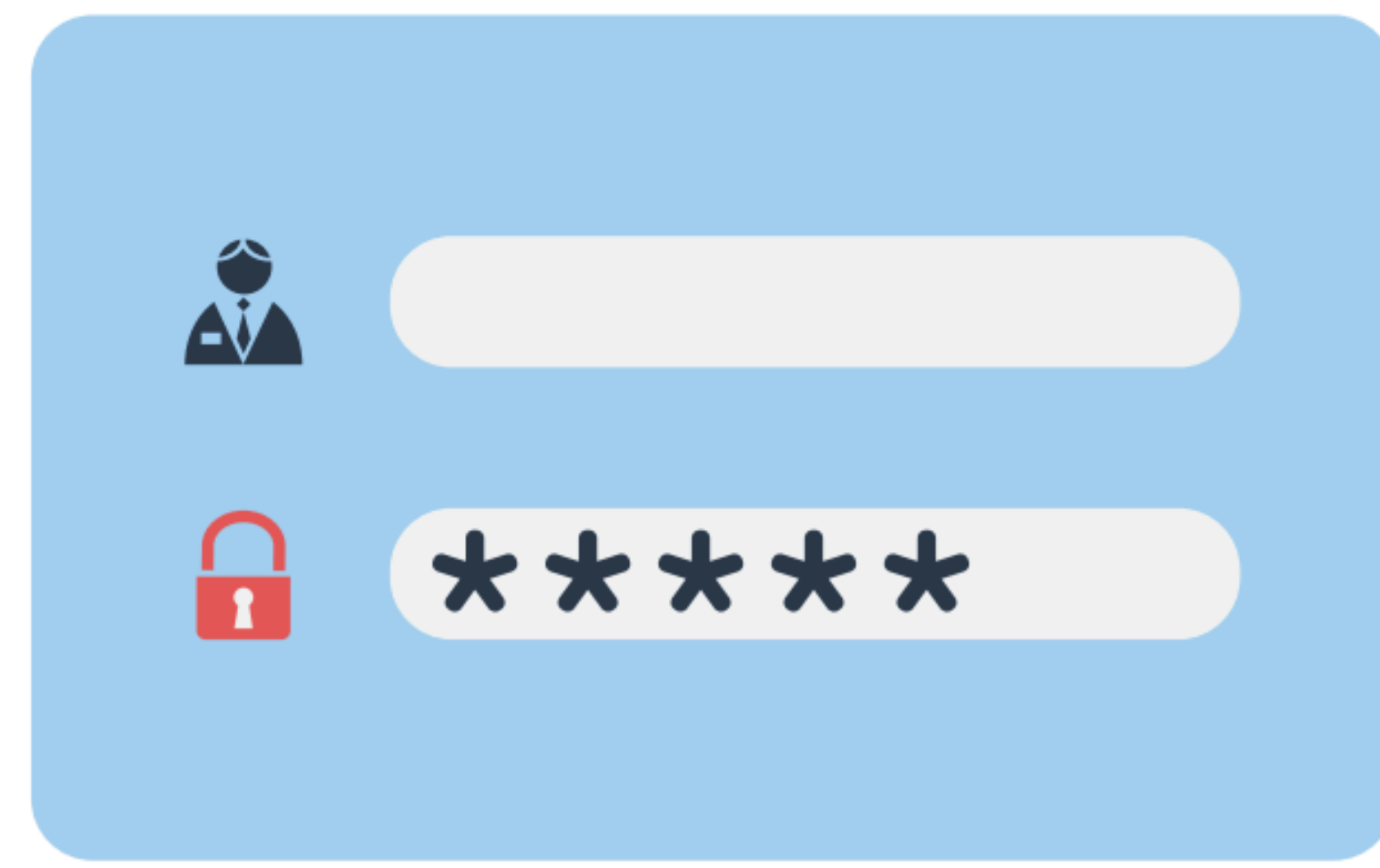
User



DB

# Multi-factor Authentication

Browser



Web Server



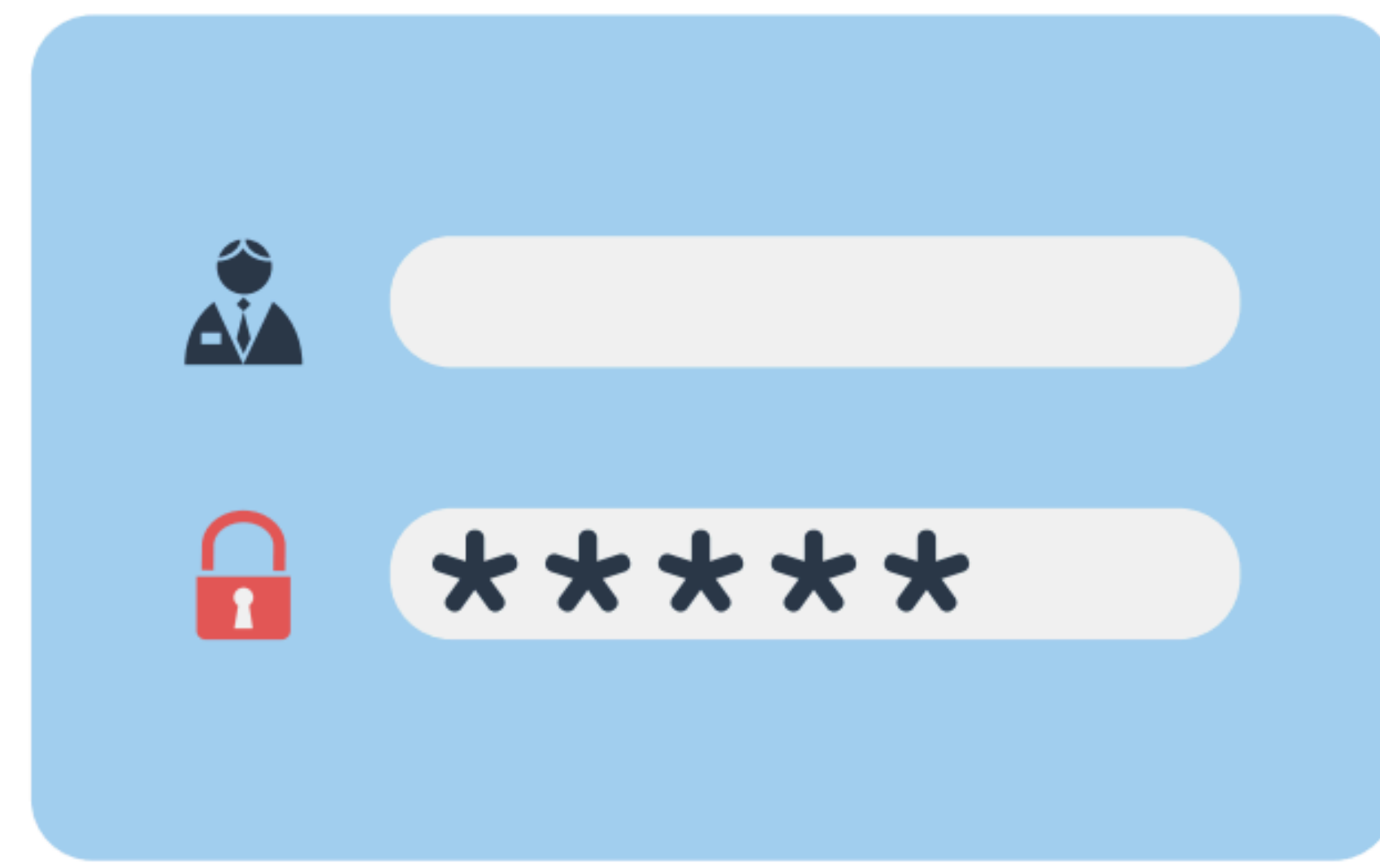
User



DB

# Multi-factor Authentication

Browser



Web Server



User

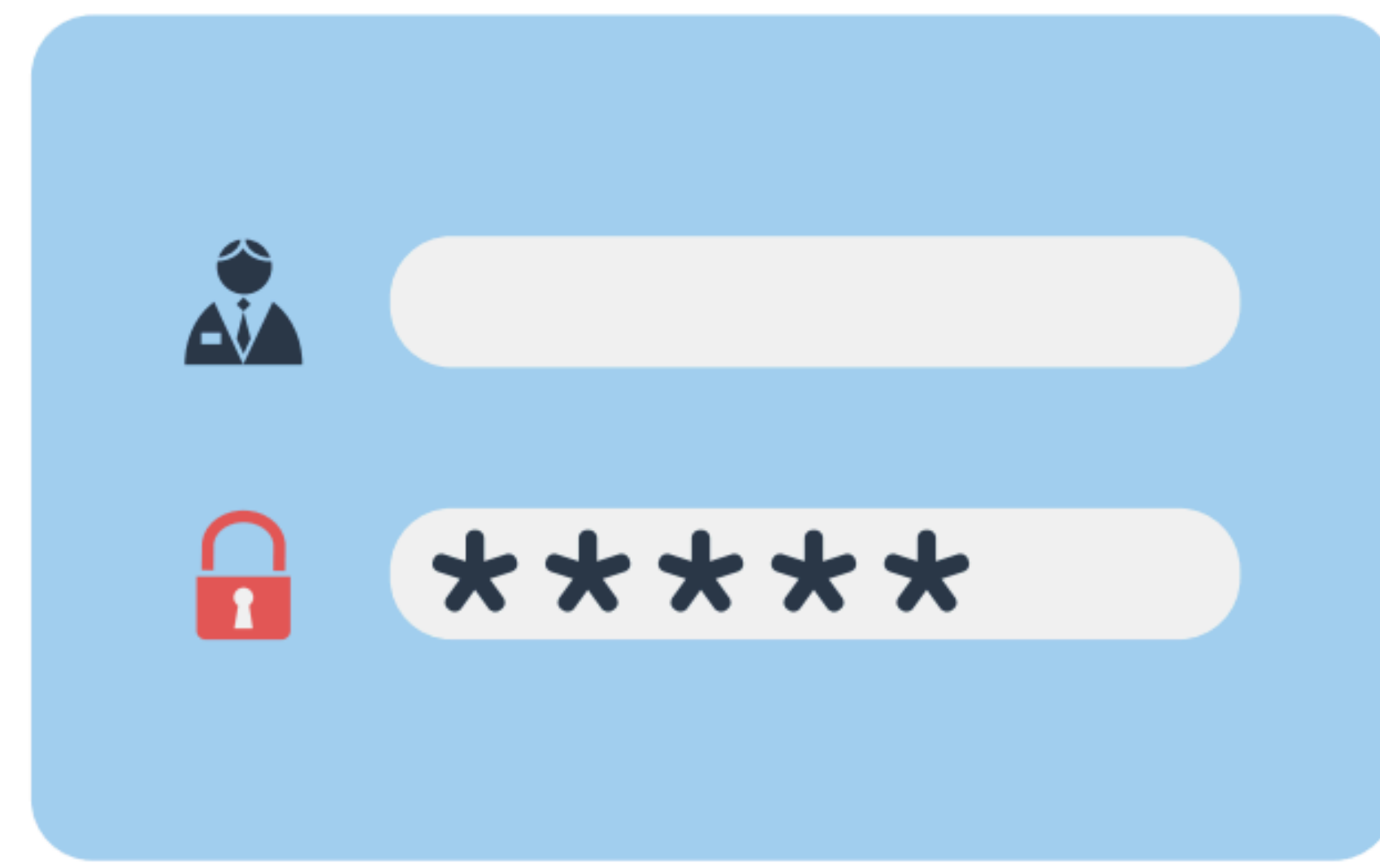


DB



# Multi-factor Authentication

Browser



Web Server



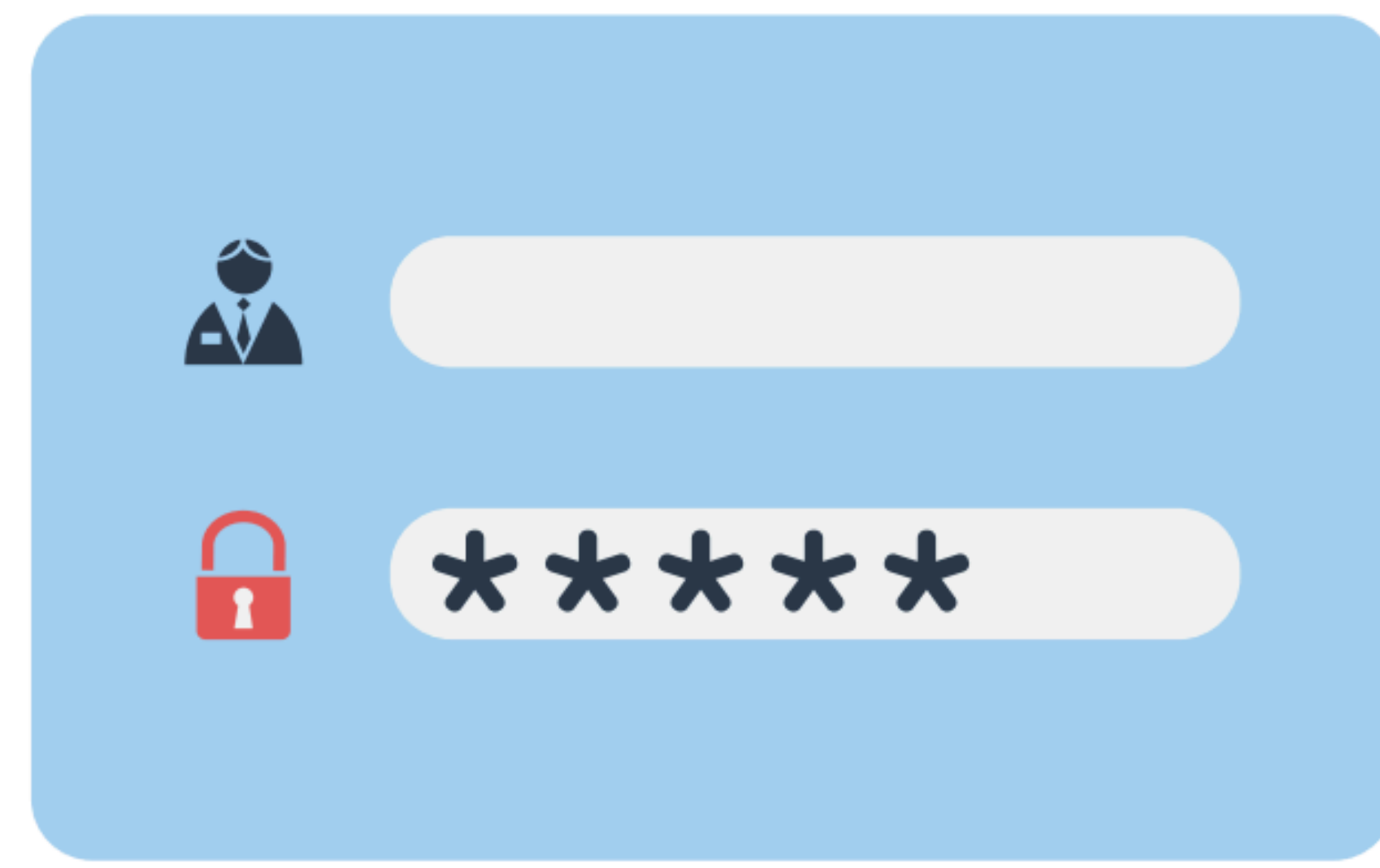
User



DB

# Multi-factor Authentication

Browser



Web Server



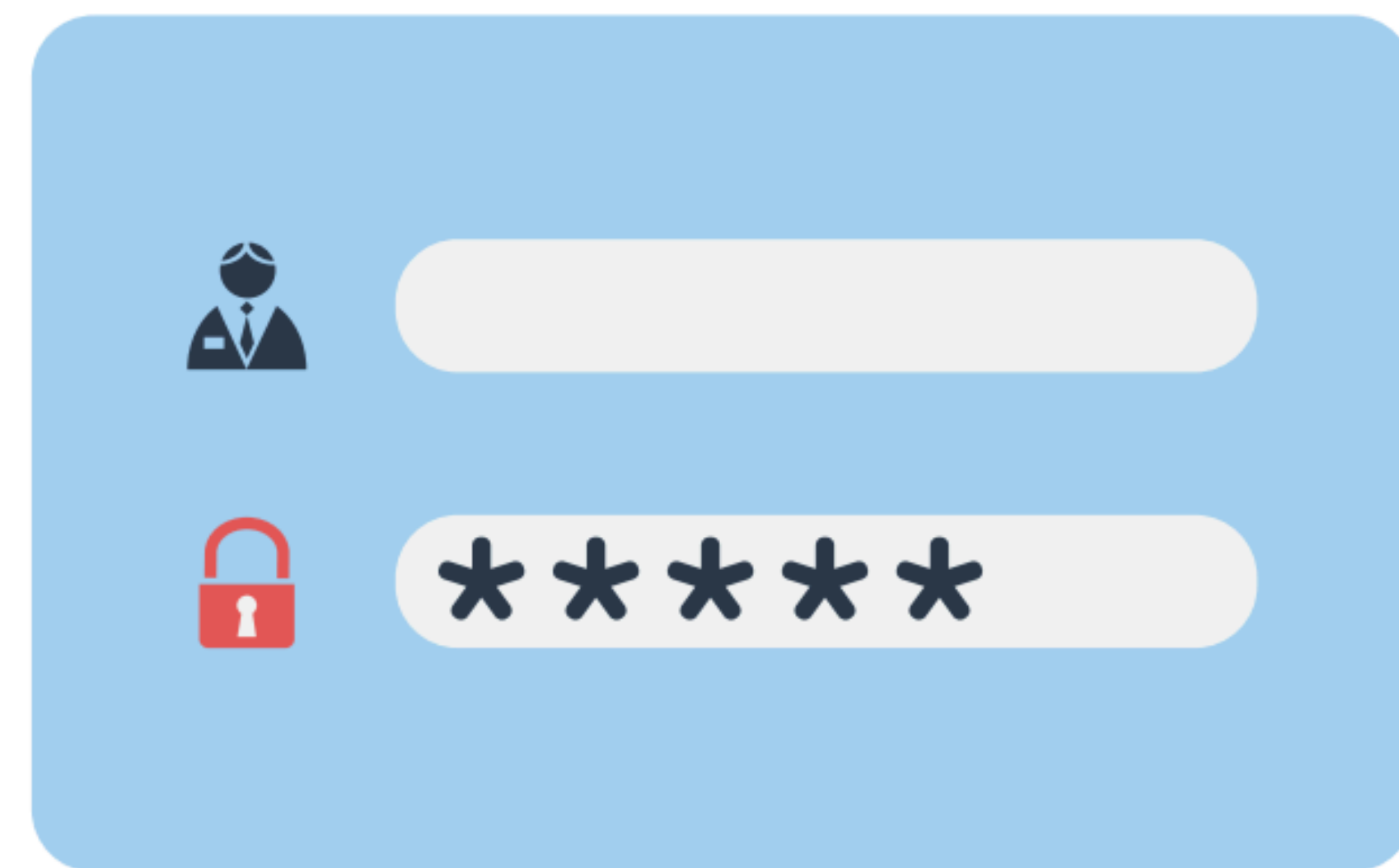
User



DB

# Multi-factor Authentication

Browser



Web Server



User



DB

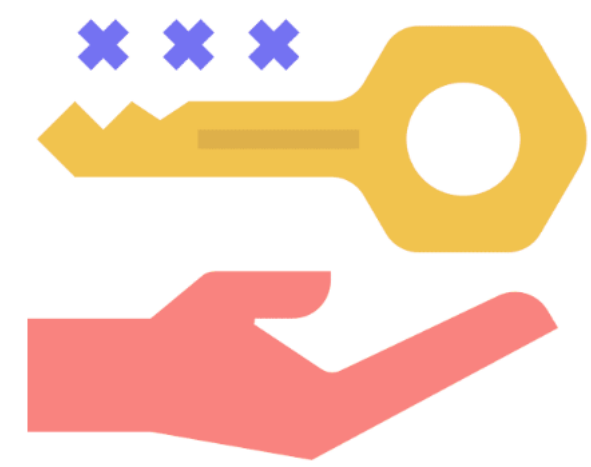
User experience becomes  
a problem and the risk is  
still there are secrets and/  
or MF can be vulnerable

# Passwordless

**DB**

# Passwordless

Browser



Public Key



User



Private Key

Web Server



DB

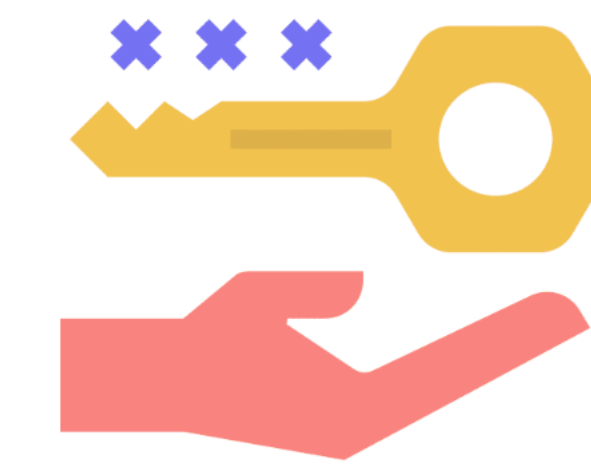
# Passwordless

Browser

Web Server



User



DB

# Passwordless

Browser

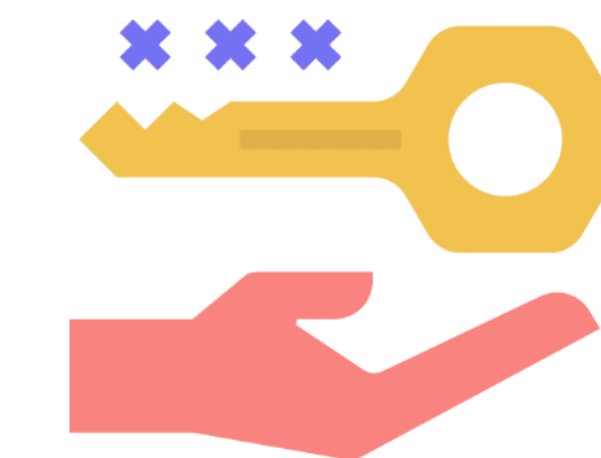
Web Server



User



Private Key



Public Key



DB



# Passwordless

Browser

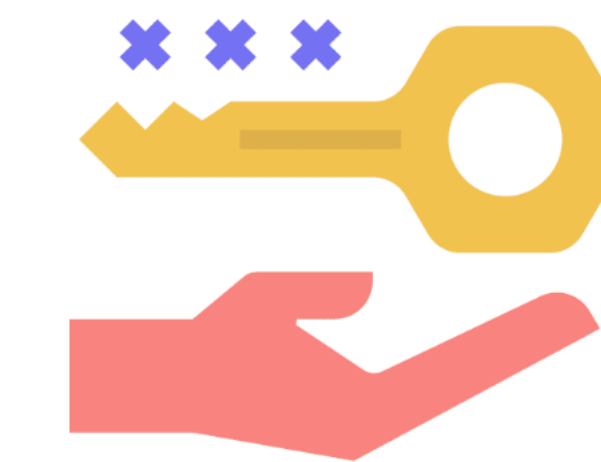
Web Server



User



Private Key



Public Key



DB

# Passwordless

Browser

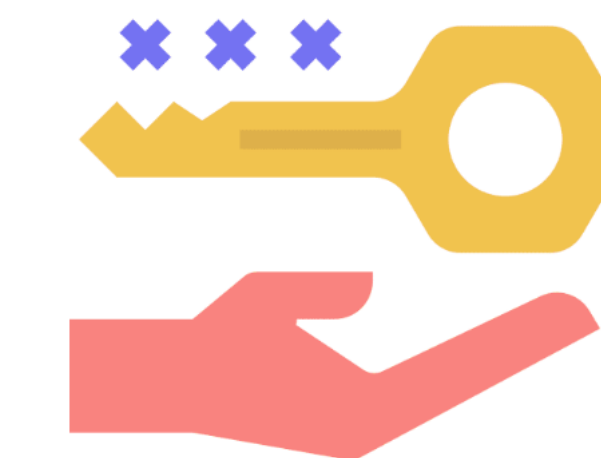
Web Server



User



Private Key



Public Key



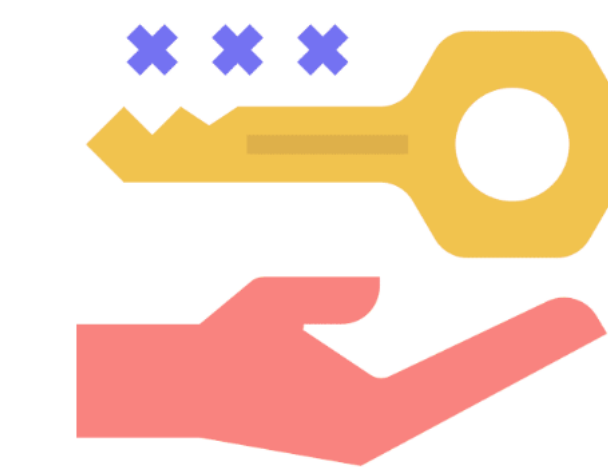
DB



# Passwordless

Browser

Web Server



DB

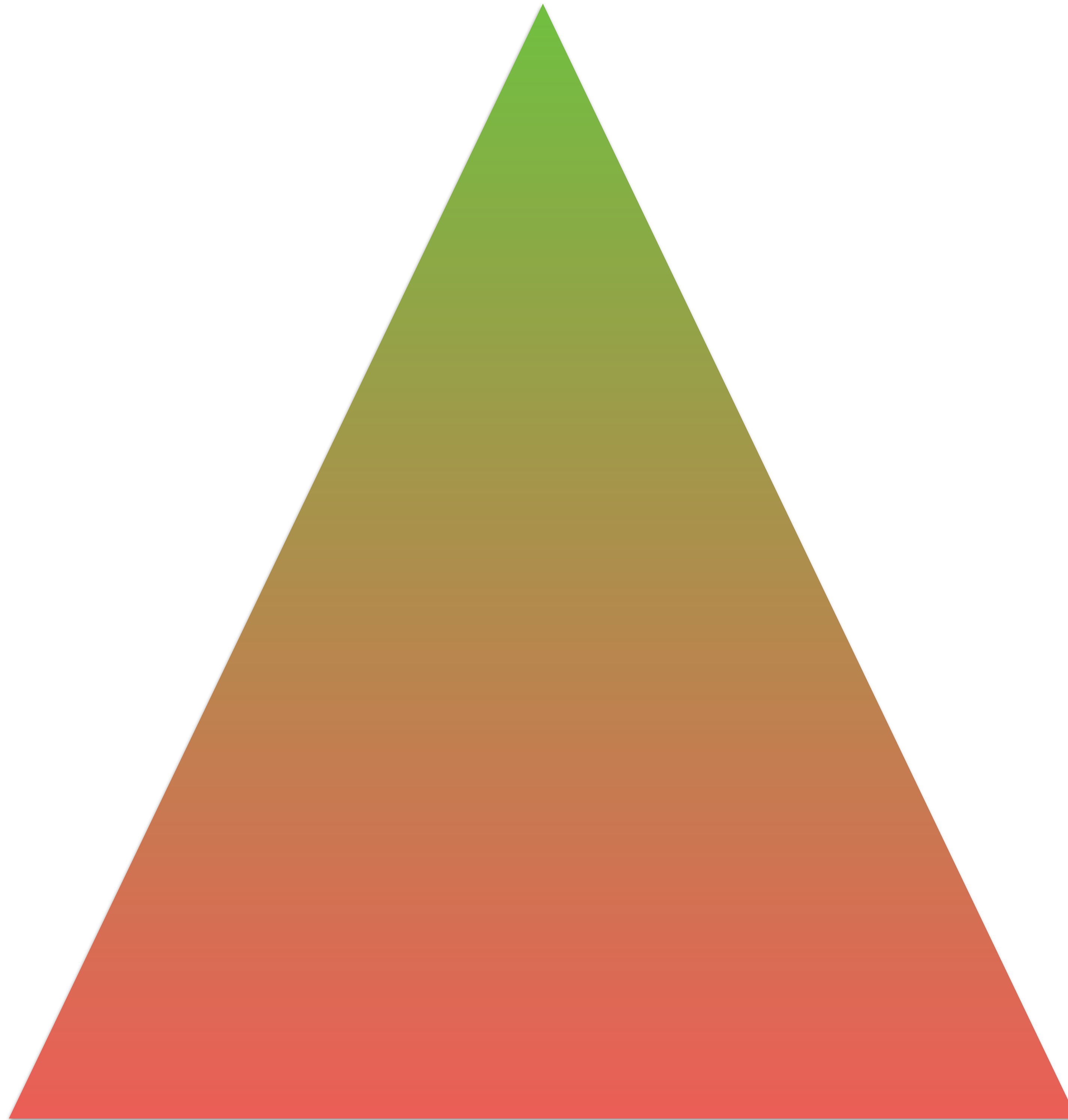
User

Private Key

Public Key



# Status



**Passwordless Auth**

**WebAuthn**

**Multi-factor Auth**

**Form-based Auth**

FullStack Authentication

3

# Classic Login Flow

# Login Form Flow

Registration

Login

Recover Password

# Enhancing Login Forms

**Connected Labels for each element**

**Don't use placeholder as labels**

**Using HTML semantics**

**On SPAs, form names different for registration and login forms**

**Let the user make the password visible**

**Help Password Managers with autocomplete HTML attributes**

**Help Accessibility with aria-describedby attribute for instructions**

**On SPAs, use submit form event and submission will be triggered by a pushState**

# Enhanced Login Form

index.html

```
<input type="password" autocomplete="new-password">
```

```
<input type="password" autocomplete="current-password">
```

```
<input type="text" autocomplete="name">
```

```
<input type="email" autocomplete="username">
```



Workshop time

FullStack Authentication

4

# Data Storage and Auth APIs

# Data Storage

**E-mail or Username**

**Name**

**Password (hashed)**

**TODO:**

**Last time logged in**

**Last 99 login timestamps**

**Last 99 IP addresses**

**Code for recovering passwords**

**Multi-factor information**

API endpoints

**We will use HTTPS REST APIs using POST  
with a JSON body**

**/register**

**/login**

Our code won't be  
production ready

There are much more to  
do in terms of security  
and data consistency! :)

Workshop time

FullStack Authentication

5

# Identifier-First Flow


# Identifier-First Flow

**Instead of a login form with username and password:**

- 1- We first ask for identity (username)**
- 2- We ask the browser about the login options for that user**
- 3- We offer the user enter a password or login with other options**



# Identifier-First Login Forms




## Welcome

Log in to Travel0 to continue to Partner Portal

Email Address

[Continue](#)

Don't have an account? [Sign up](#)



## Enter Your Password

[Edit](#)

Password

[Forgot your password?](#)

[Continue](#)

Don't have an account? [Sign up](#)

Workshop time

# Credential Management

**It let us save and retrieve in the browser's password manager:**

- \* Credentials (username/password)**
- \* Federated Credentials**
- \* Public/Private Keys**

**It let us implement auto login safely.**

**For credentials, it's Chromium-only.**

**Safari supports only public/private keys, used for WebAuthn API.**

# Credential Management API

script.js

```
const credentials = new PasswordCredential({
  id: "admin",
  password: "12345"
});

await navigator.credentials.store(credentials);
```

# Credential Management API

script.js

```
const credentials = await navigator.credentials
    .get({password: true});
```

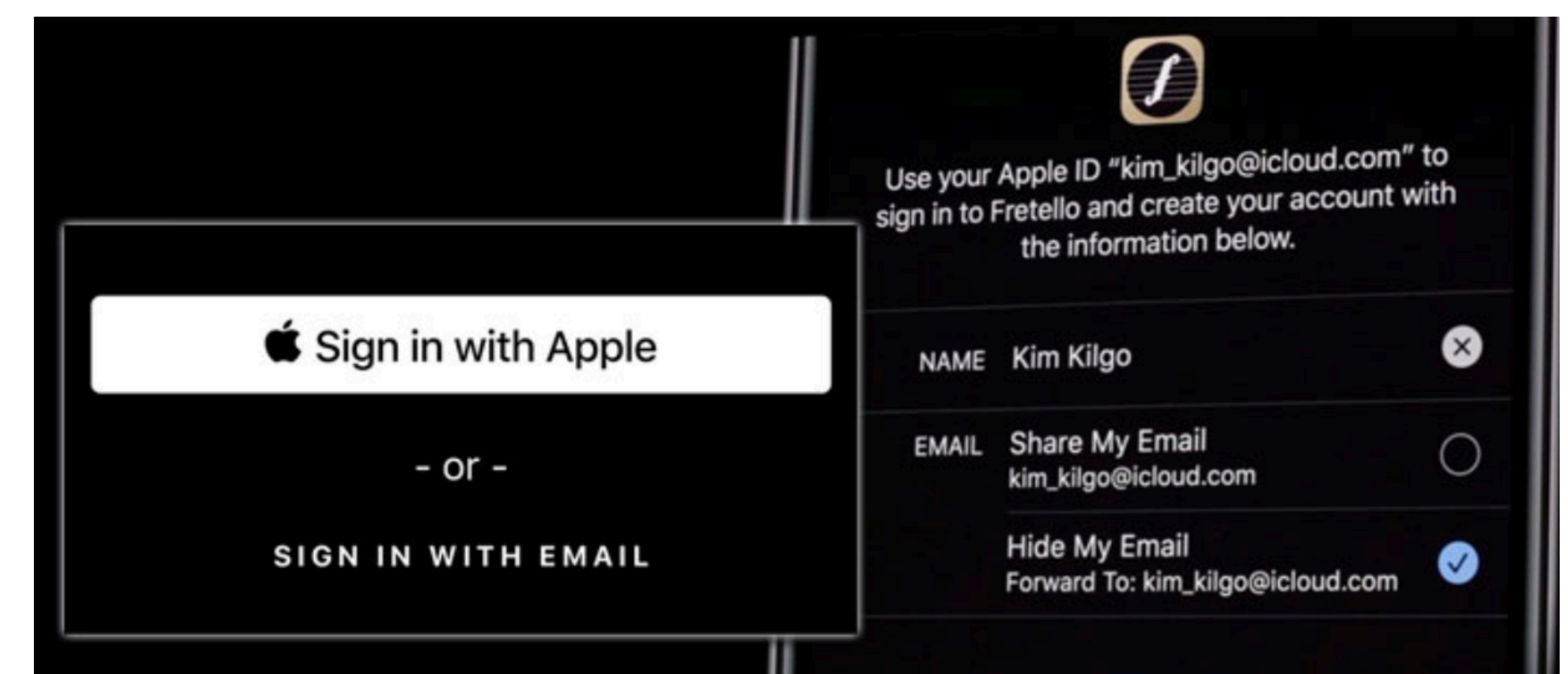
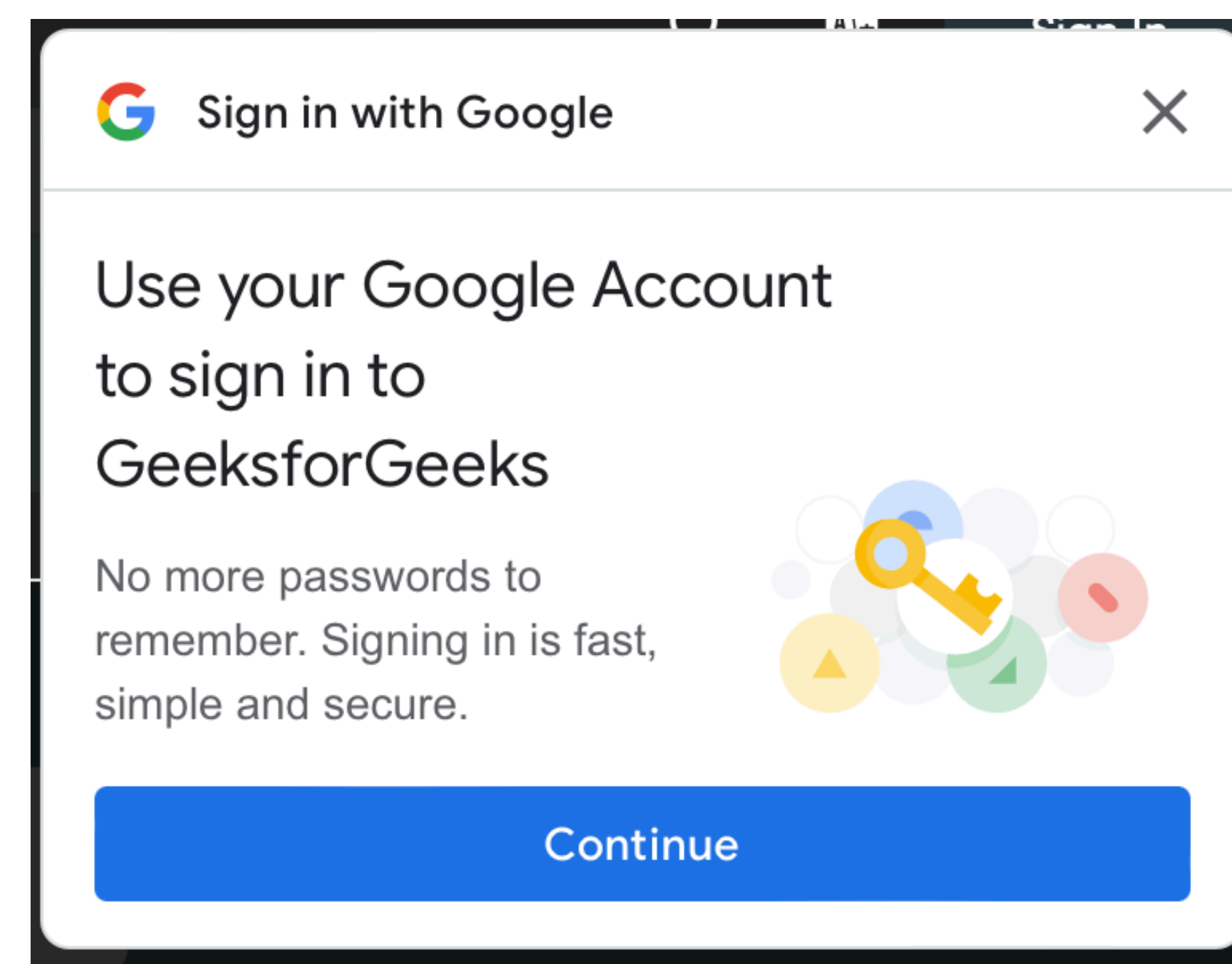
Workshop time

# Federated Login

Using OAuth you can use many providers

Sign In with Google

Sign In with Apple



Workshop time



FullStack Authentication

6

**WebAuthn**

# Web Authentication



**A multi-vendor effort**

**FIDO Alliance and W3C**

**Store safely private keys while sending public keys for the server**

**It can work with FIDO2 and platform authenticators**

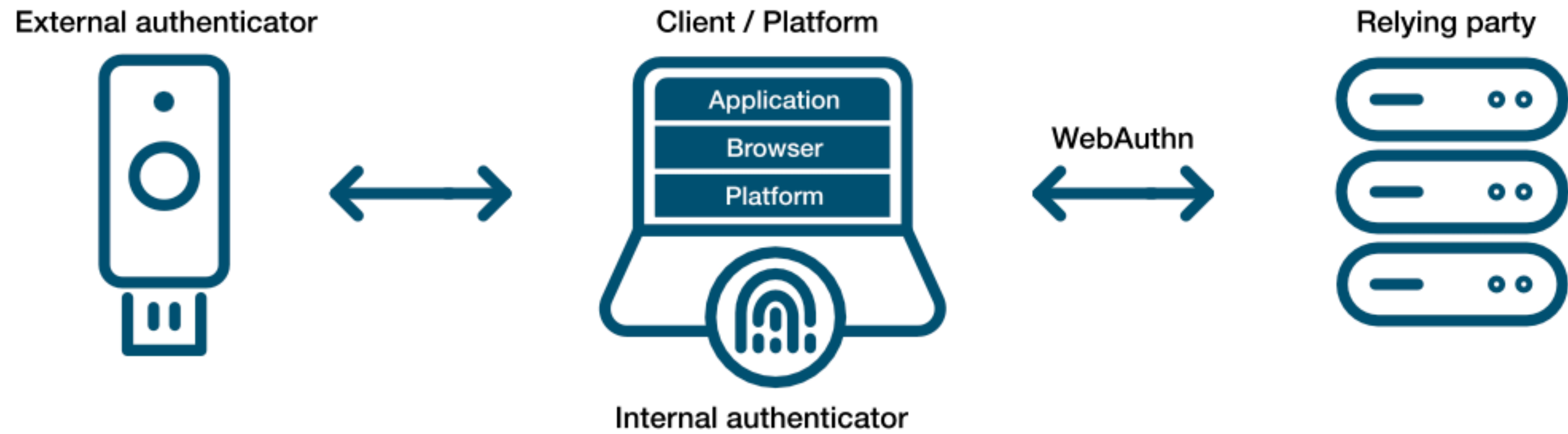
**It's typically used as a 2FA**

**It works on every browser with different techniques and abilities:**

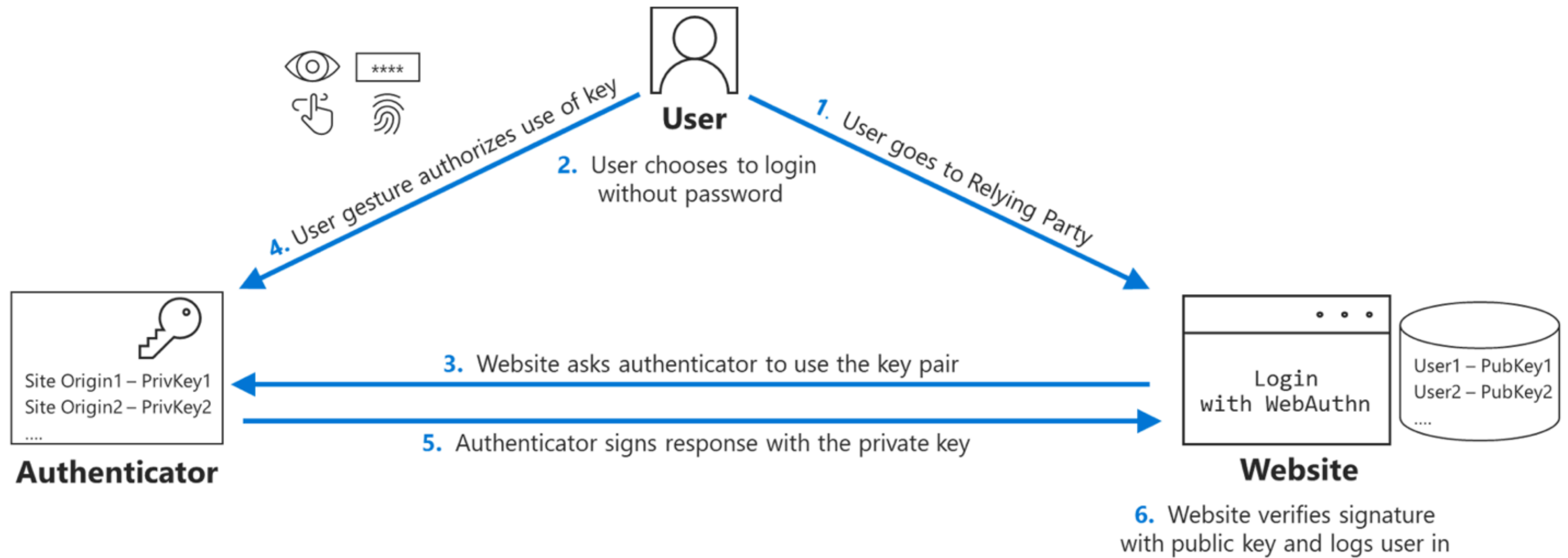
- \* **PIN-based keys**
- \* **FIDO2 USB keys** (such as Yubico)
- \* **Biometric Authenticators**

(TouchID, FaceID, Windows Hello, Android)

# WebAuthn



# WebAuth



# WebAuth Flows

Registration

Authentication  
(Login)

Library

We will be using  
<https://simplewebauthn.dev>

It has a server and a client library

Workshop time

**<https://webauthn.io/>**

Workshop time



FullStack Authentication

7

**Going Passwordless**

# Passwordless Options

Magic Links

OTP

Passkeys

# Passkeys

**It's the new DNA of WebAuthn**

**The idea is to use WebAuthn as a First Factor**

**Authenticators are saving the keys (known as passkeys) in the cloud, so you can use them on different devices, and even share with other users**

**It includes a Conditional UI that lets the browser autocomplete a login form without the user typing anything**

Workshop time

**<https://passkeys.io>**

Workshop time

FullStack Authentication

8

**Best Practices**

# Native Apps

**There are ways to connect your login credentials between the web and native**

**For password managers**

**For WebAuthn and Passkeys**

**Handshake between your site and native app**

# Ideas

**Keep user's destination after login**

**Always confirm user emails**

**Ask the user about autologin**

**Check Privacy's legislation**

**Security is too important**

**Test your login UX flows**



# Project TODO

**Validations**

**Better database integrity**

**Forget Password Flow**

**Confirm email on registration**

**Sign in with Apple**

**Magic Link login**

**Passkeys conditional UI**

A blurred high-speed train in motion on a track, with a stone bridge in the background.

hi@firt.dev

@firt

Fot