LECTURE 1

PLATFORM BASED DEVELOPMENT

Francisco Vilchez @franciscovilchezv

ACKNOWLEDGEMENTS

THANKS

- Thanks to all web pages whose pictures I am using without asking for permission. Sorry 😅

GITHUB LECTURES

Francisco Vilchez @franciscovilchezv

Q&A

TODAY'S LECTURE

@FRANCISCOVILCHEZV

- Modelo Cliente - Servidor (Client - Server Model)
- Páginas Web (Web Pages) & Aplicaciones Web (Web Applications)
- Components in a Web Application
  - Front End
  - Back End
  - Database

CLIENT - SERVER MODEL

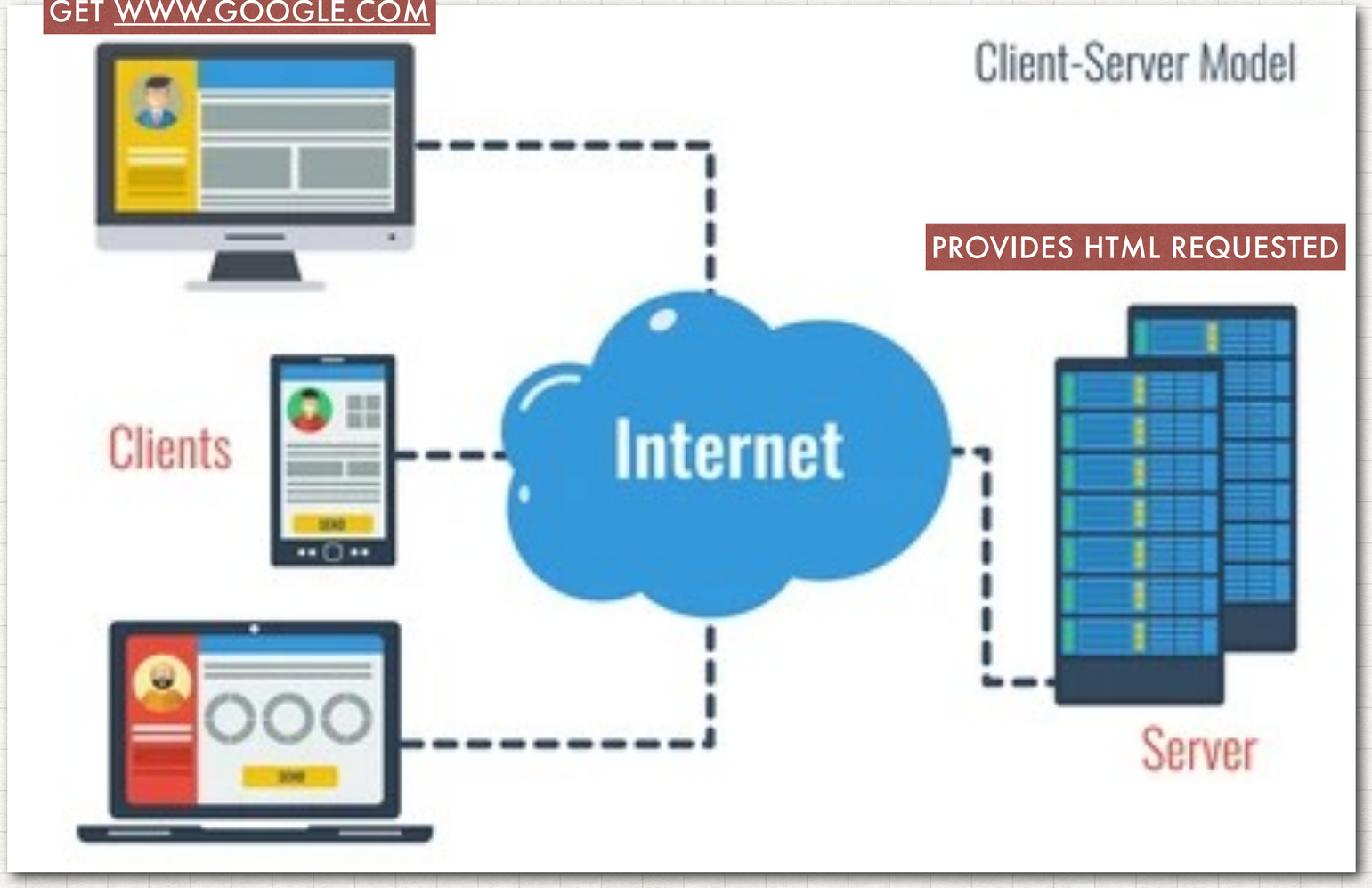
CLIENT - SERVER MODEL

- Es una estructura de aplicaciones que divide las tareas en dos partes:
  - El que provee los recursos (servicios o data), llamados servers.
  - El que solicita los recursos, llamado clients.
- Ejemplos de aplicaciones que utilizan el modelo Cliente - Servidor son:
  - Correo Electrónico (Mail Servers)
  - Servidores de archivos (File Servers)
  - World Wide Web (Web Servers)

CLIENT - SERVER MODEL

WEB SERVER

GET WWW.GOOGLE.COM



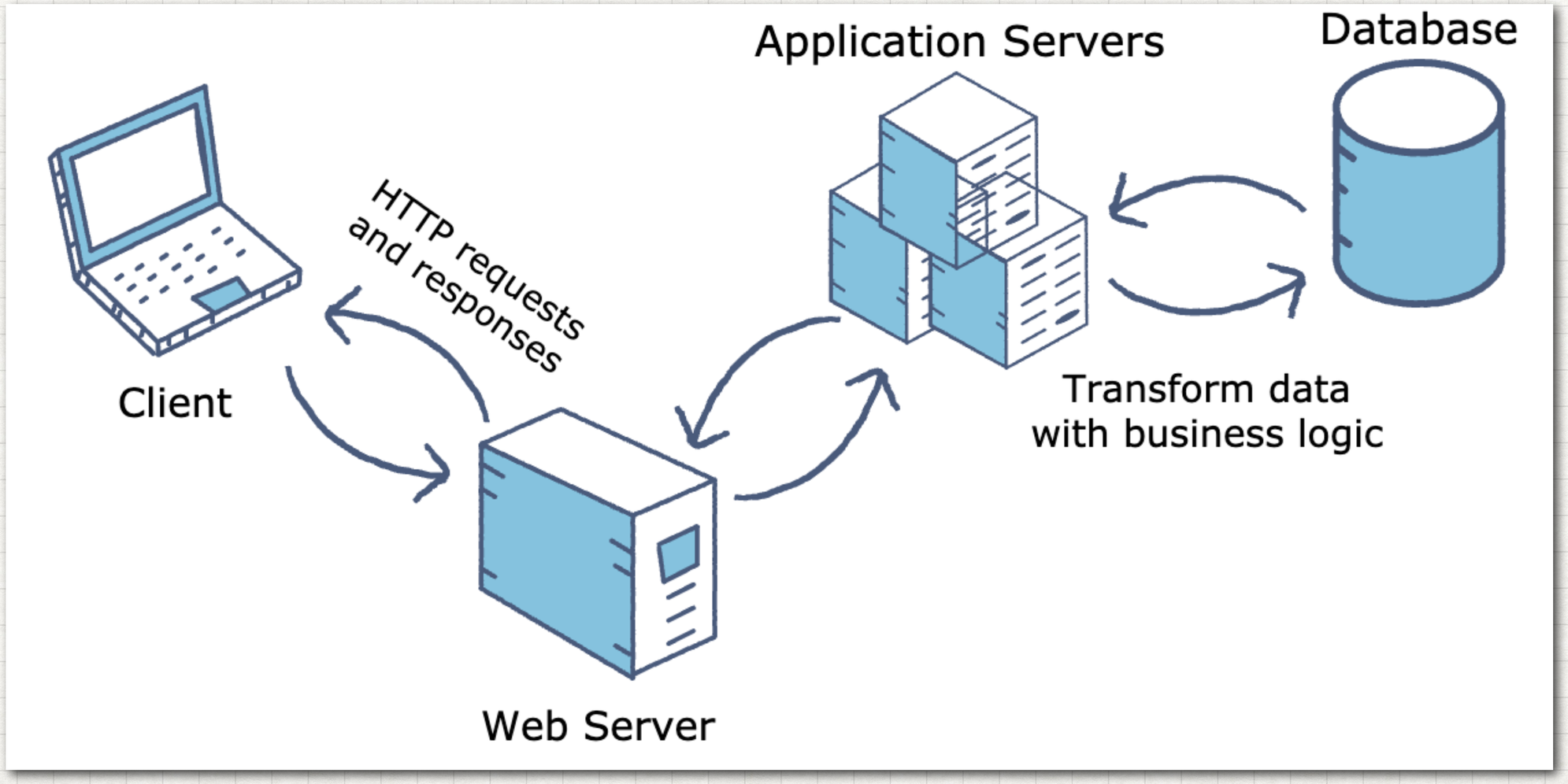
WEB PAGES & WEB APPLICATIONS

Francisco Vilchez @franciscovilchezv

WEB PAGES & WEB APPLICATIONS

- Software que es ejecutado en un Web Server, e.g. Google Apps, Office 365, Tinder, etc…
- La principal diferencia entre una Página Web (Web Page) y una Aplicación Web (Web Application) es que el contenido de la **Página Web** es generalmente **estático**, mientras la **Aplicación Web** tiene contenido **dinámico**.
- Las Web Application generalmente necesitan un **servidor** adicional (Application Server), encargado de procesar la información ingresada desde el **cliente**.

WEB APPLICATION

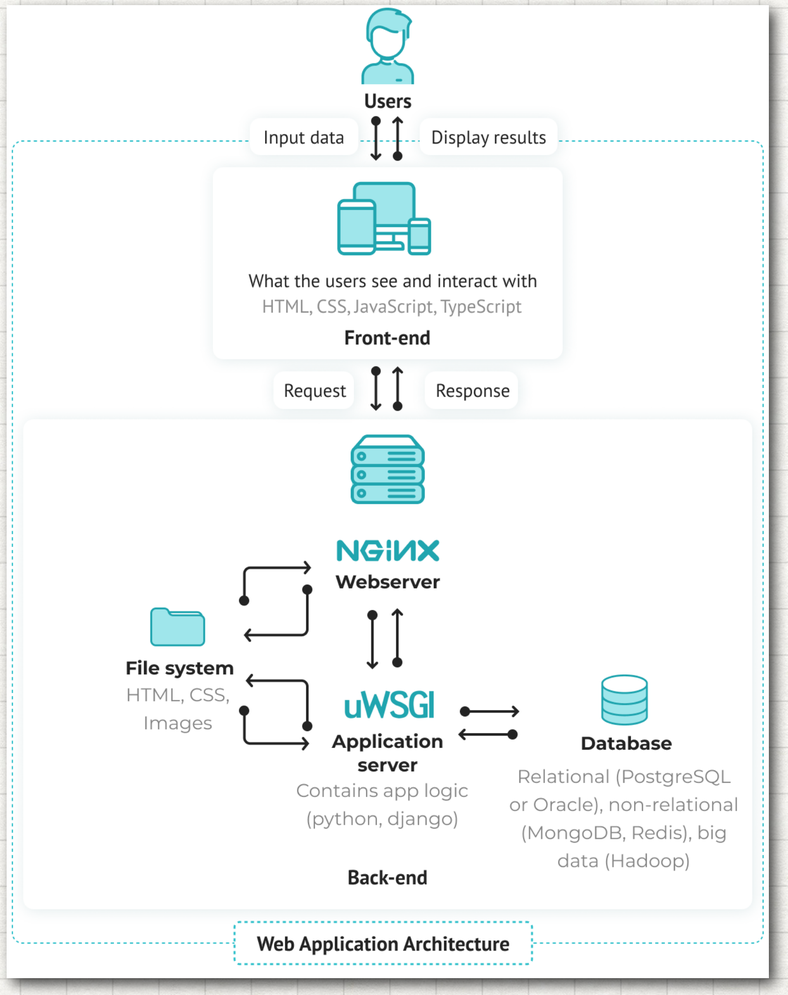


WEB APPLICATION COMPONENTS

WEB APPLICATION COMPONENTS

- Un proyecto web puede empezar a crecer rápidamente, en términos de funcionalidad y de código.
- Por esa razón, existe una división muy utilizada para dividir las tareas realizadas en la creación una aplicación web, la cual consiste en 2 principales grupos:
  - Front End: Incluye lo que el usuario ve y con lo que interactúa, e.g. html, js, css.
  - Back End: Incluye la parte lógica de la aplicación, encargada de procesar la data, almacenarla, realizar operaciones matemáticas, etc. e.g. python, java, etc...
- Cabe mencionar, que si el Back End necesita almacenar data y usarla posteriormente, probablemente usará una Base de Datos (Database).

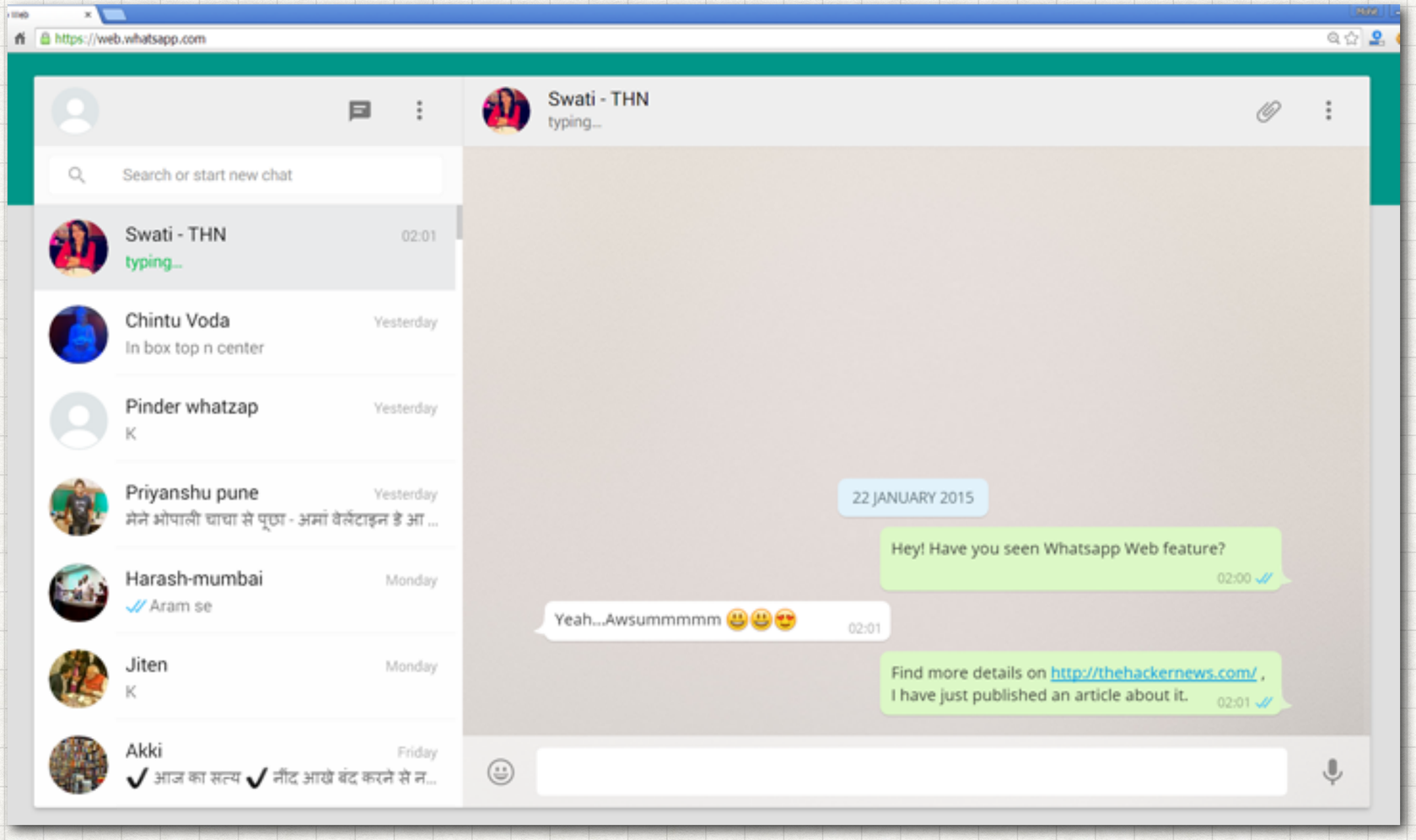
WEB APPLICATION COMPONENTS



FRONTEND & BACKEND EXAMPLES

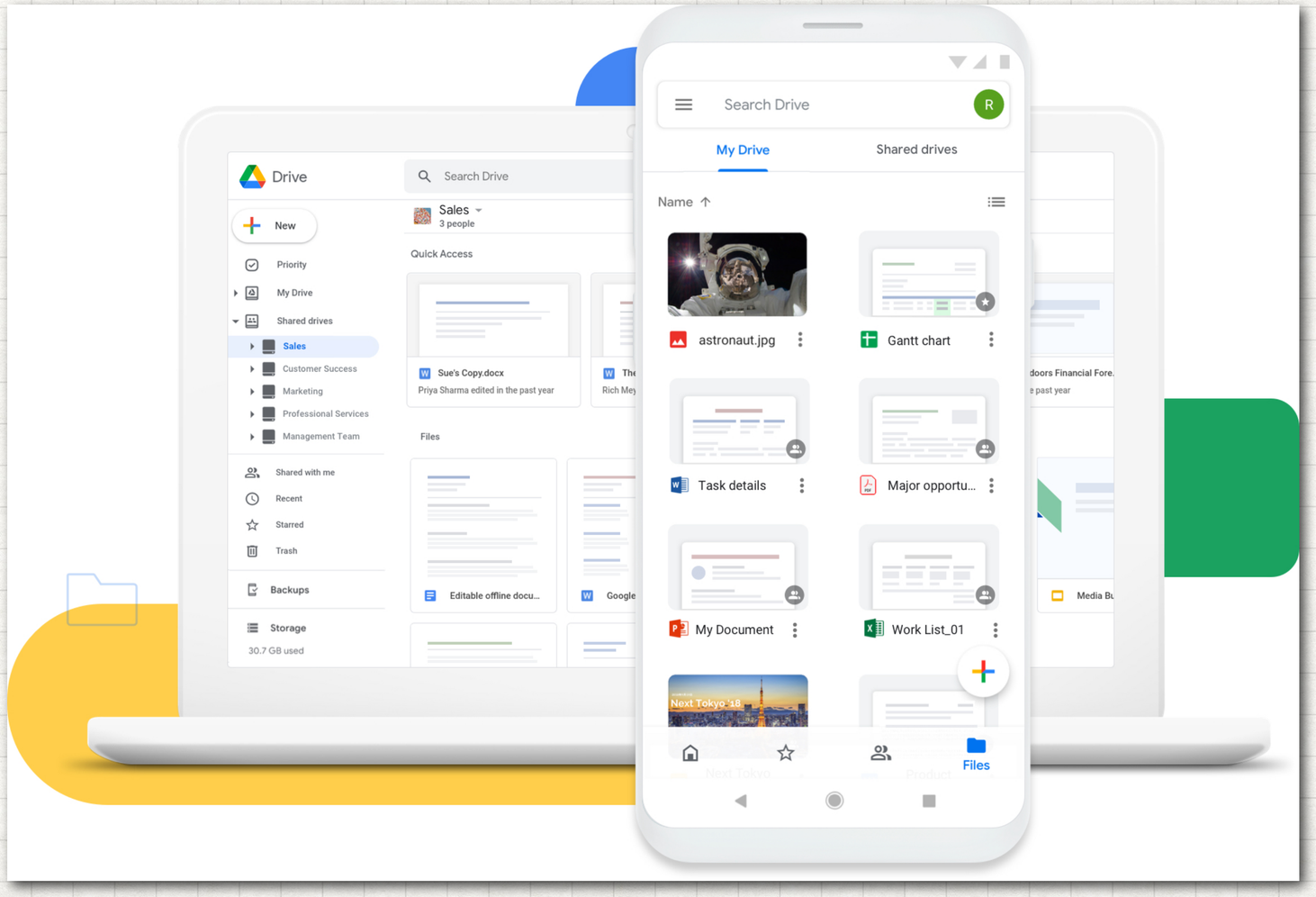
WEB APPLICATION COMPONENTS

EXAMPLE (1)



WEB APPLICATION COMPONENTS

EXAMPLES (2)



Francisco Vilchez @franciscovilchezv

MOBILE APPLICATION COMPONENTS

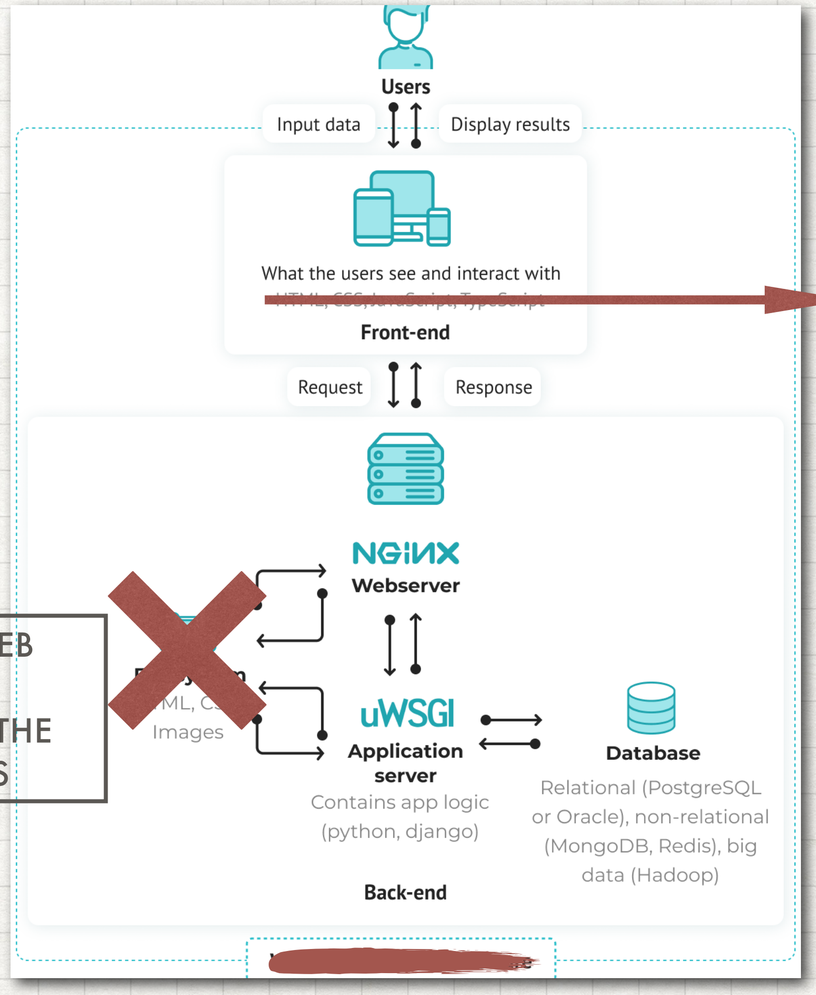
Francisco Vilchez @franciscovilchezv

MOBILE APPLICATIONS COMPONENTS

- La división en una Mobile Application no difiere mucho de la división en una Web Application.
- La aplicación mobile, al estar instalada en el dispositivo, tiene almacenada la lógica del lado Front End.
- Por esta razón, archivos como HTML, JS, CSS no necesitan ser descargados continuamente de un Web Server. Principalmente porque otras tecnologías son usadas para desarrollar el lado Front End.

MOBILE APPLICATION COMPONENTS

USUALLY...



TECHNOLOGIES USED MAY BE JAVA (ANDROID), SWIFT (IOS), OR EVEN HTML + JS + CSS. HOWEVER, THESE FILES ARE STORED IN THE DEVICE, I.E. THEY ARE NOT DOWNLOADED FROM A WEB SERVER WHILE USING THE APP

SOME APPS MAY USE A WEB SERVER TO STORE SOME COMPONENTS NEEDED IN THE FRONT-END, LIKE IMAGES

COURSE DIVISION (WEB & MOBILE)

FRONTEND (WEB)

FRONTEND (WEB)

- Como mencionamos, incluye lo que el usuario ve e interactúa.
- Para el desarrollo de el lado frontend se utilizan generalmente tecnologías como:
  - HTML
  - JS
  - CSS

FRONTEND

HTML (1)

- HTML es un lenguaje de marcado (markup language). Te permite codificar un documento utilizando etiquetas (tags).
- Es utilizado para definir la estructura de la página web.
- La versión inicial de HTML (llamada HTML 0.a) contaba con 18 etiquetas para poder usar, como <h1>, <h2>, <p>, etc…
- La versión mas actualizada (HTML5) has approximately 110 tags.

FRONTEND

HTML (2)

FRONTEND

HTML (3)

- La versión inicial de HTML (llamada HTML 0.a) contaba con 18 etiquetas para poder usar, como <h1>, <h2>, <p>, etc...
- La versión mas actualizada (HTML5) tiene aproximadamente 110 tags.

FRONT END

CSS

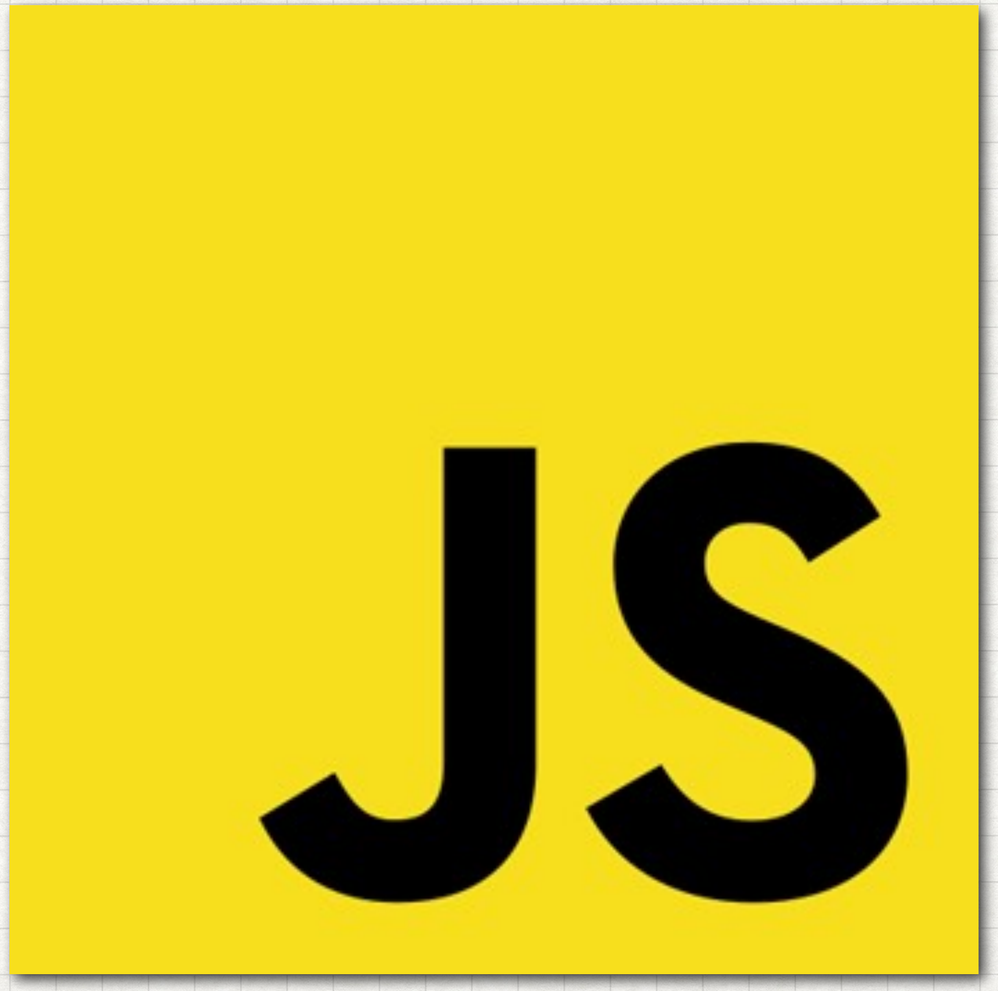
- CSS es un lenguaje de estilos (style sheet language).
- Es utilizado para personalizar la presentación de un documento escrito en lenguaje de marcado, como HTML.
- Te permite personalizar atributos de algún tag (e.g <h1>, <h2>, <p>) como el tamaño, color, posición, etc…
- La útilma versión de CSS, llamada CSS3, provee diversas opciones para incluir incluso animaciones a algún tag (e.g. modo de aparición, tiempo de duración del efecto, etc…)

```
<style>
    h1 {
        color: blue;
    }
</style>
```

FRONTEND

JAVASCRIPT (1)

- Javascript no tiene NADA que ver con Java.
- Javascript es utilizado para hacer las páginas web interactivas.
- Declarar funciones. Modificar valores en el HTML de acuerdo a alguna lógica. Cambiar propiedades de alguna parte de tu HTML (e.g. cambiarlo de “visible” a “oculto”). Cambiar la URL en la que te encuentras en este momento. ETC



FRONTEND

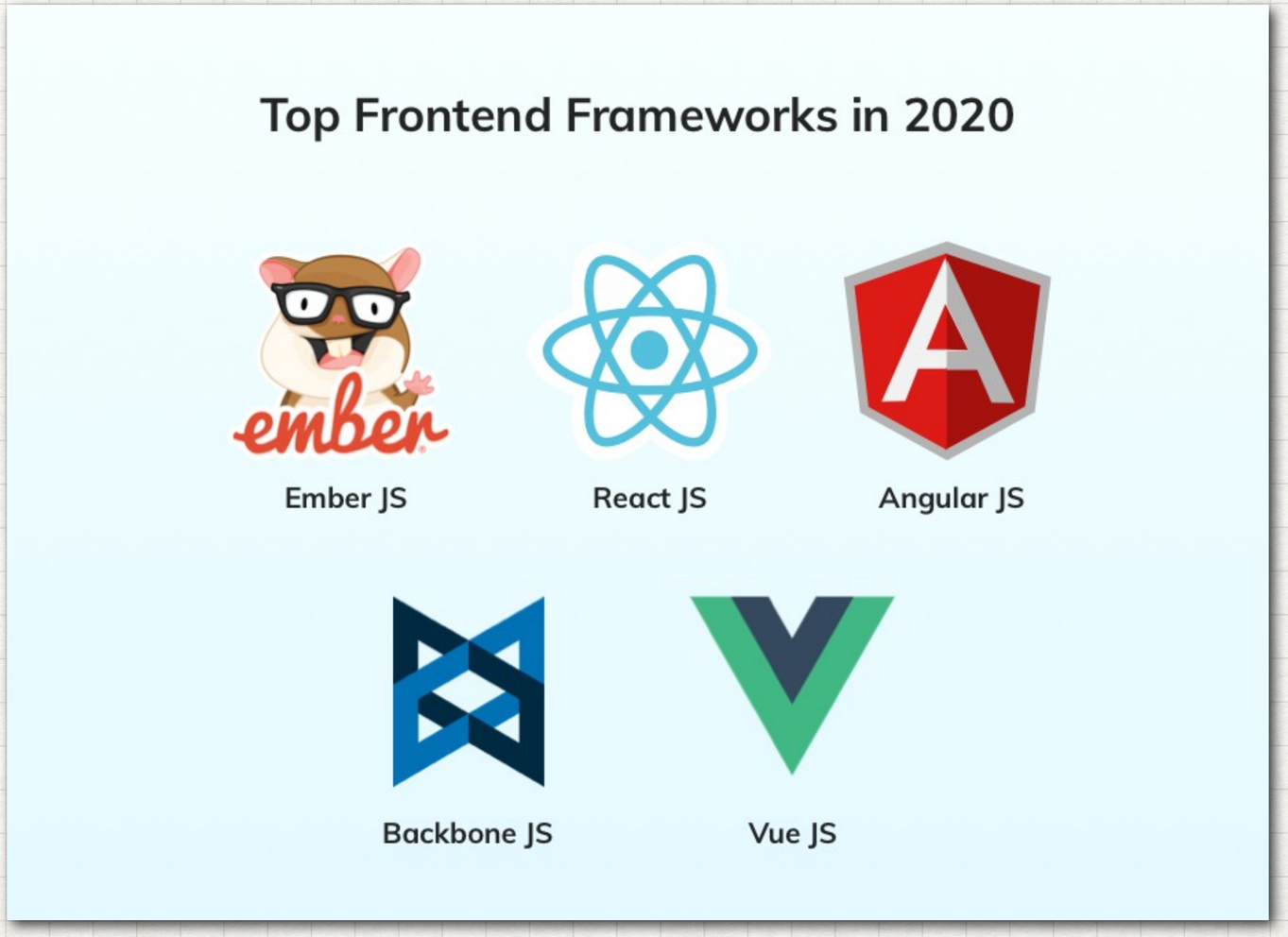
JAVASCRIPT (2)

```
<p>Before the script...</p>

<script>
  alert( 'Hello, world!' );
</script>

<p>...After the script.</p>
```

FRONTEND FRAMEWORKS

FRONTEND
FRAMEWORK

- Un framework te da un proyecto base sobre el cual desarrollas la aplicación.
- Te provee “funciones” ya creadas para las tareas que vas a necesitar realizar durante tu proyecto.
- Un framework frontend puede darte funciones para:
  - Navegación
  - Intercambio de datos entre frontend y backend, o entre vistas.
  - Estilos para tus tags HTML.



BACKEND

BACKEND

- Como mencionamos, se encarga de realizar la lógica necesaria para obtener los datos que se van a mostrar en el Frontend.
- Algunos lenguajes utilizados para realizar el Backend de una aplicación son:
  - Java
  - Python
  - C#
  - Javascript (Ojo, es el mismo lenguaje de programación usado en el Frontend, pero se encuentra en un proyecto diferente)

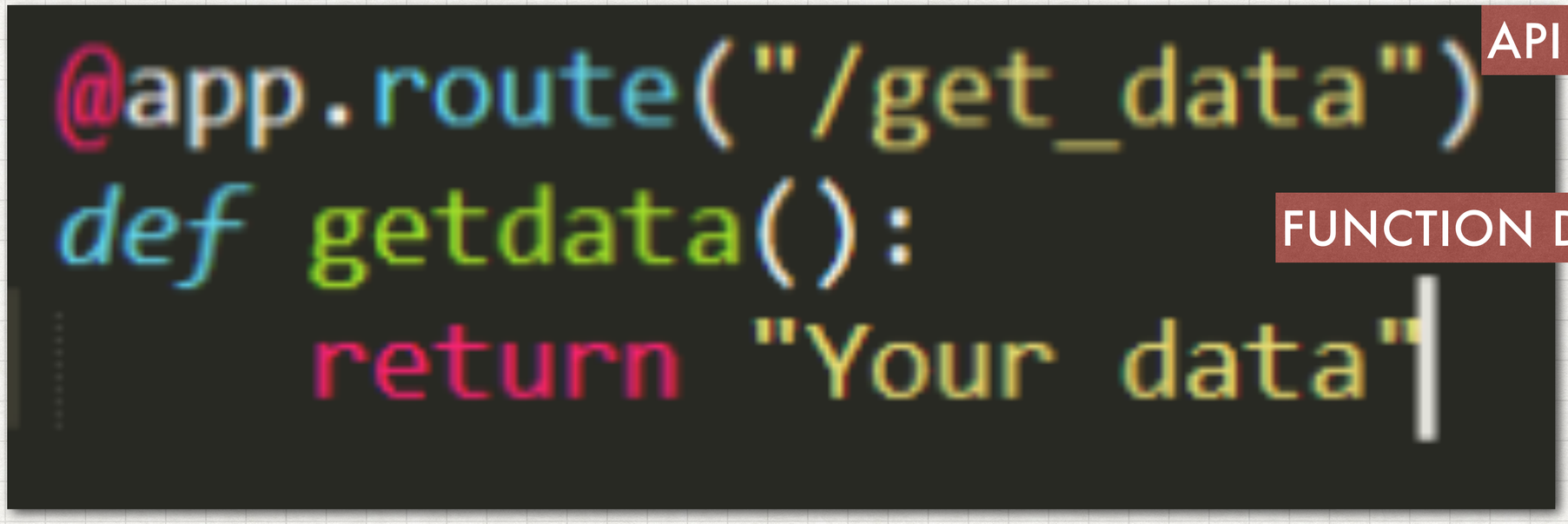
BACKEND

API (1)

- Las funciones creadas en el Backend deben de alguna manera ser “expuestas” para que el Frontend pueda “llamarlas” (o invocarlas) cuando lo necesite.
- La definición de las funciones expuestas y la manera en que deberán ser “llamadas” se definen en la API.
- API significa Interfaz de Programación de Aplicaciones (Application Programming Interface), e indica las funciones de tu proyecto que pueden ser invocadas por otros softwares.

BACKEND

API (2)



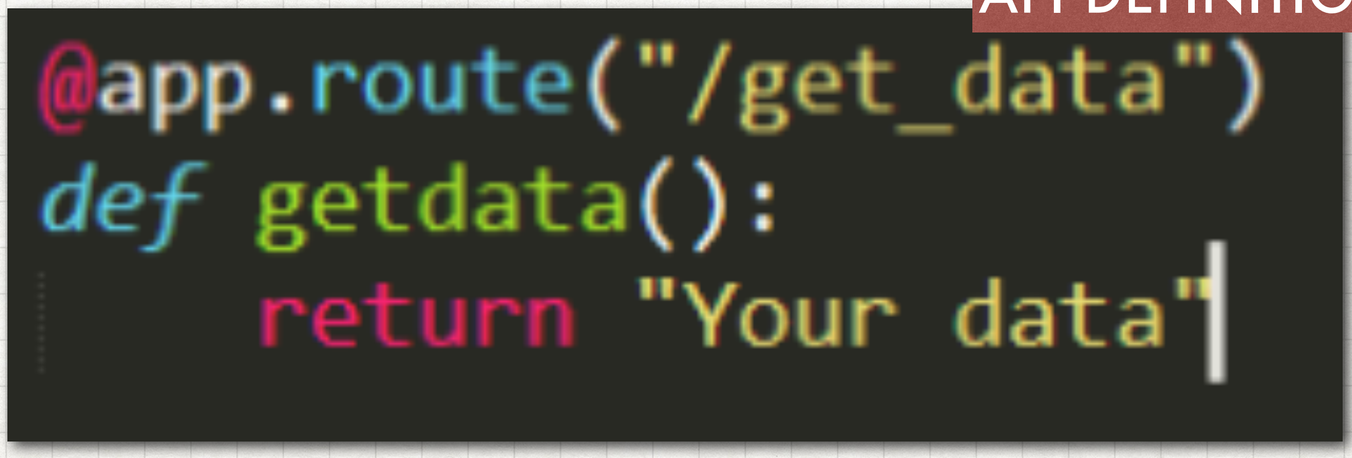
BACKEND FRAMEWORKS

BACKEND

FRAMEWORKS

- Un framework backend de igual manera de provee funciones que hacen mas sencillo el desarrollo de tu proyecto.
- En el ejemplo mostrado, la definición de la API con Python toma muchas líneas de código. El ejemplo utiliza un framework llamado Flask, lo cual lo simplifica a una sola línea de código.
- Algunos ejemplos de Frameworks para el backend son:
  - Flask (Python)
  - KoaJS (Javascript)
  - Springboot (Java)

API DEFINITION



RESOURCES

THANKS

- https://flyaps.com/blog/difference-front-end-back-end-development-in-different-applications/