PROYECTO DE SOFTWARE

Cursada 2021

TEMARIO

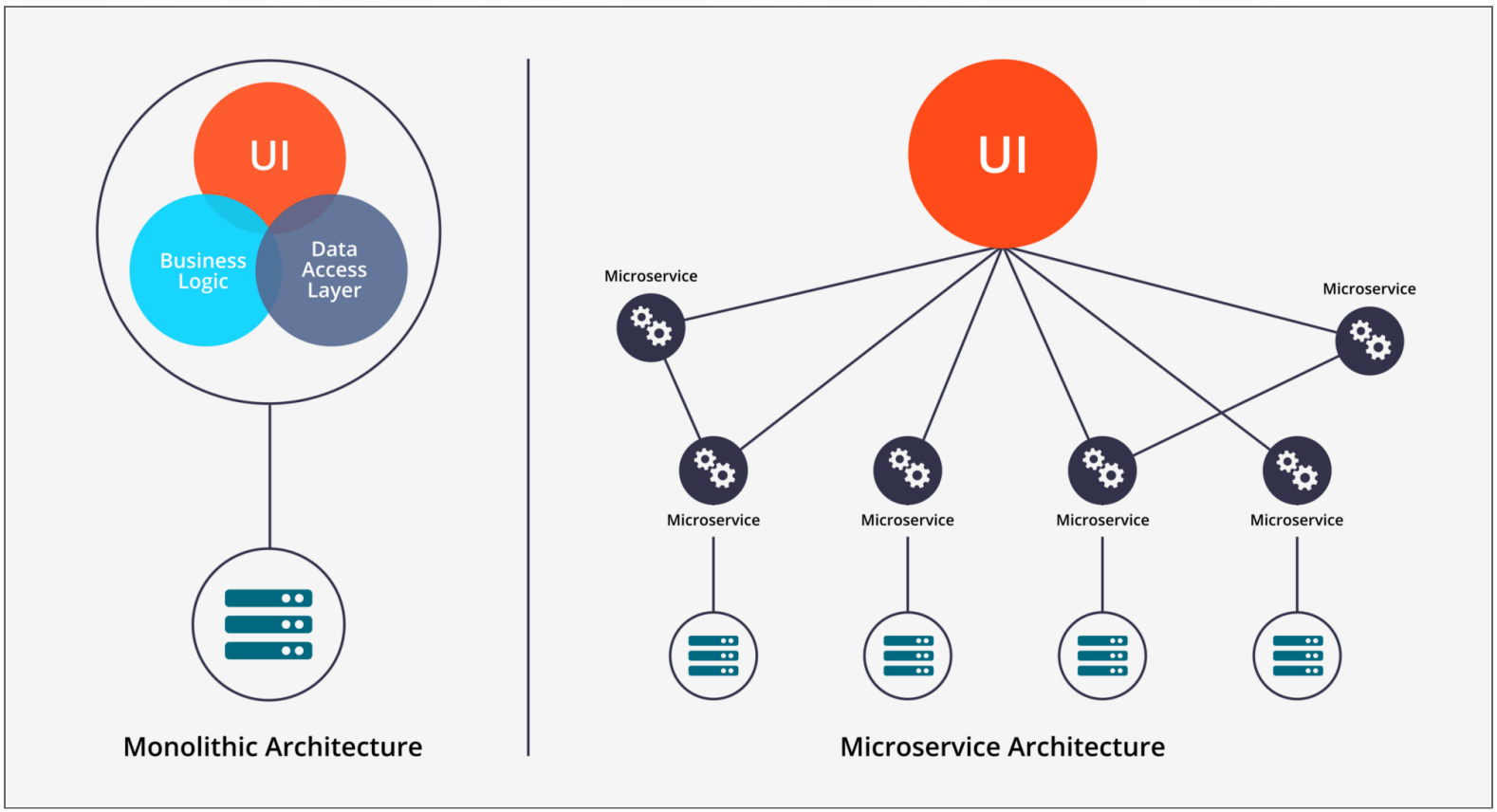
- Microservicios
- Frameworks JS
- Intro Vue

ARQUITECTURA DE MICROSERVICIOS

MICROSERVICIOS

- Las aplicaciones se dividen en sus componentes más pequeños, independientes entre sí.
- A diferencia del enfoque tradicional y monolítico de las aplicaciones, en el que todo se encuentra en una única pieza.
- Los microservicios funcionan en conjunto para llevar a cabo las mismas tareas que la aplicación monolítica.
- Los microservicios facilitan la escalabilidad de todo el sistema, se despliegan según se vayan necesitando.
- Pueden tener distintas tecnologías entre sí.
- Al ser más pequeños, son mas simples de mantener y actualizar.

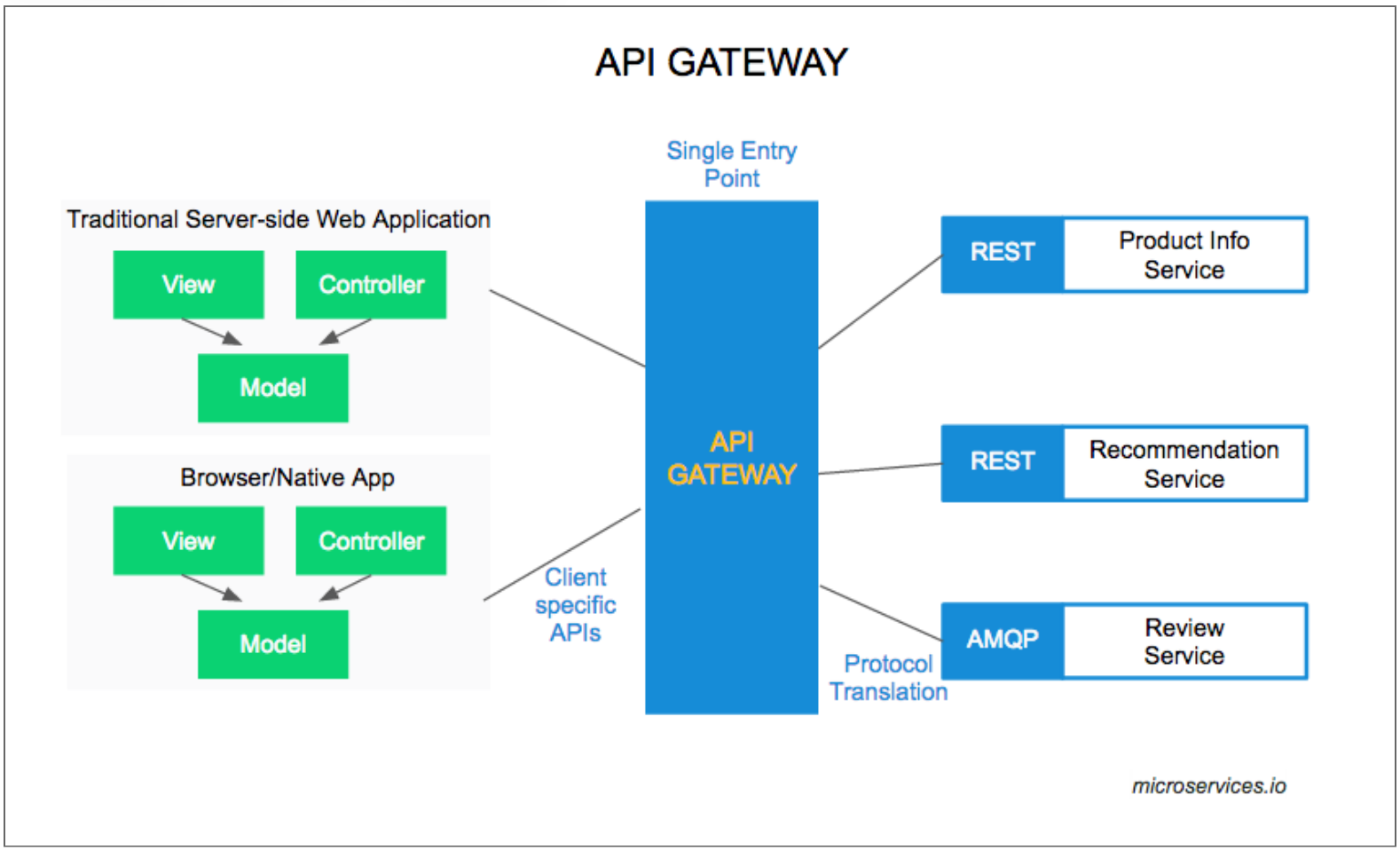
ARQUITECTURA MONOLÍTICA VS ARQUITECTURA DE MICROSERVICIOS



¿CÓMO COMUNICO UN SERVICIO CON OTRO?

- Actualmente la opción más utilizada es mediante APIs HTTP/REST con JSON.
- Incluso puede centralizarse la comunicación utilizando un API Gateway.
- En dicho API Gateway puede implementarse una capa de seguridad, que ante una petisión verifique si el cliente tiene permisos de acceso.

MICROSERVICIOS: API GATEWAY



PARA SEGUIR LEYENDO DE MICROSERVICIOS

- AWS: **https://aws.amazon.com/es/microservices/**
- Redhat: **https://www.redhat.com/es/topics/microservices**
- Video en español: **https://www.youtube.com/watch?v=9R2hFwIPGnQ**

FRAMEWORKS JS

¿QUÉ ÉS UN FRAMEWORK JS?

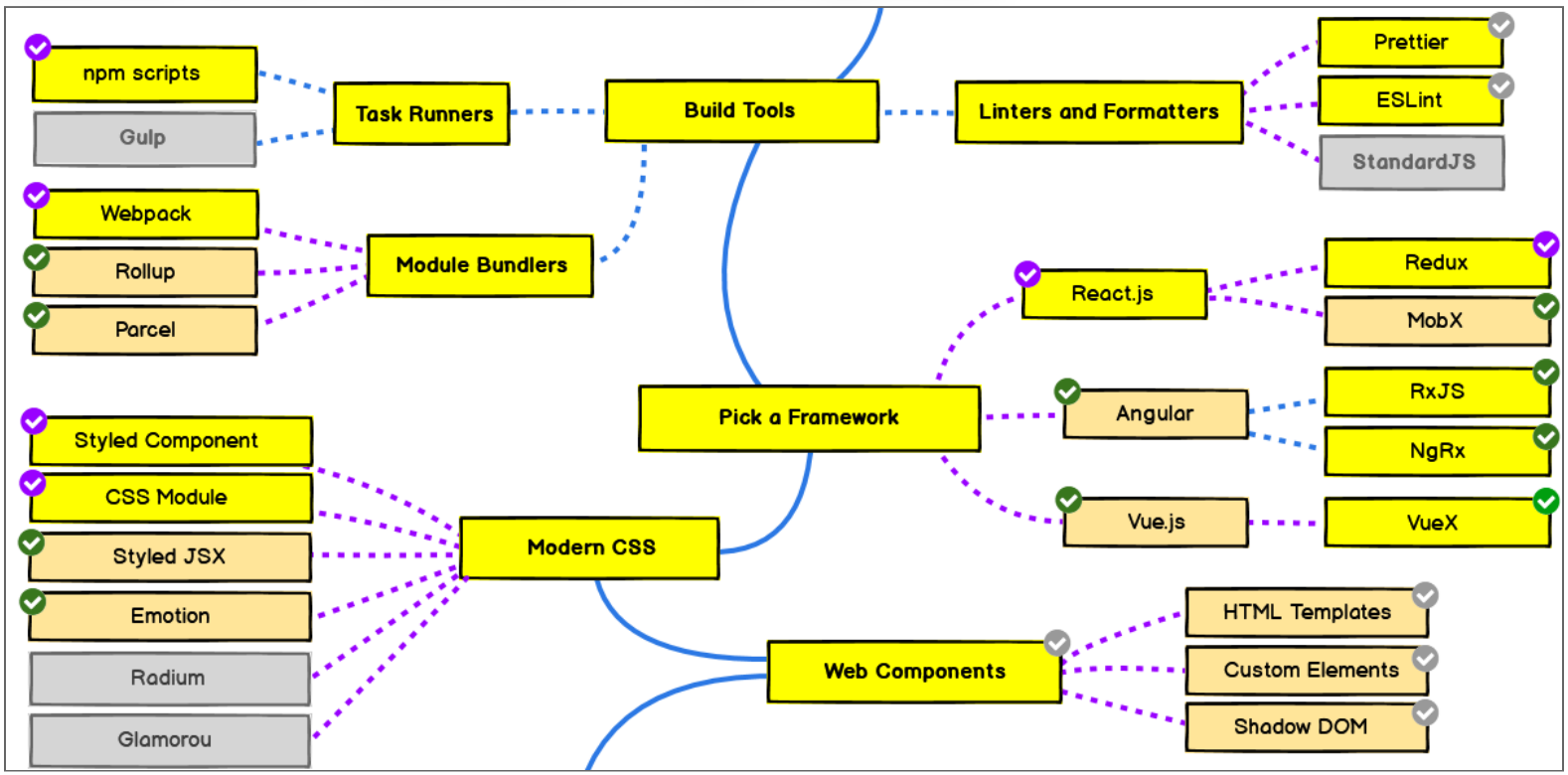
- Trabajar con **JS pelado** es complejo. Por eso nació **jQuery**(2006) para facilitar el desarrollo.
- jQuery sigue estando **muy extendido**.
- El poder de cómputo de los clientes web **aumenta día a día**.
- Las aplicaciones web implementan **cada vez más funcionalidades y complejidad**, con lo que jQuery se queda corto.

DEMASIADAS HERRAMIENTAS Y FRAMEWORKS JS:

https://en.wikipedia.org/wiki/List_of_JavaScript_libraries


JS
REQUIRE·JS
npm
TS
BABEL
JS
REQUIRE·JS
npm
TS
REQUIRE·JS

ELECCIÓN DE UN FRAMEWORK JS: FRONT-END PATH



[Developer Roadmap]

¿POR QUÉ EXISTEN LOS FRAMEWORK JS? LA VERDADERA RAZÓN:

PARA SEGUIR LEYENDO: JS FRAMEWORKS

- Comparacíon de Frameworks JS: **https://tinyurl.com/comparacionLibsJS**
- **https://medium.com/dailyjs/the-deepest-reason-why-modern-javascript-frameworks-exist-933b86ebc445**
- **https://platzi.com/blog/stack-javascript-2020/**
- **https://developer.mozilla.org/es/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks**
- **https://codersera.com/blog/best-javascript-frameworks/**

VUE.JS

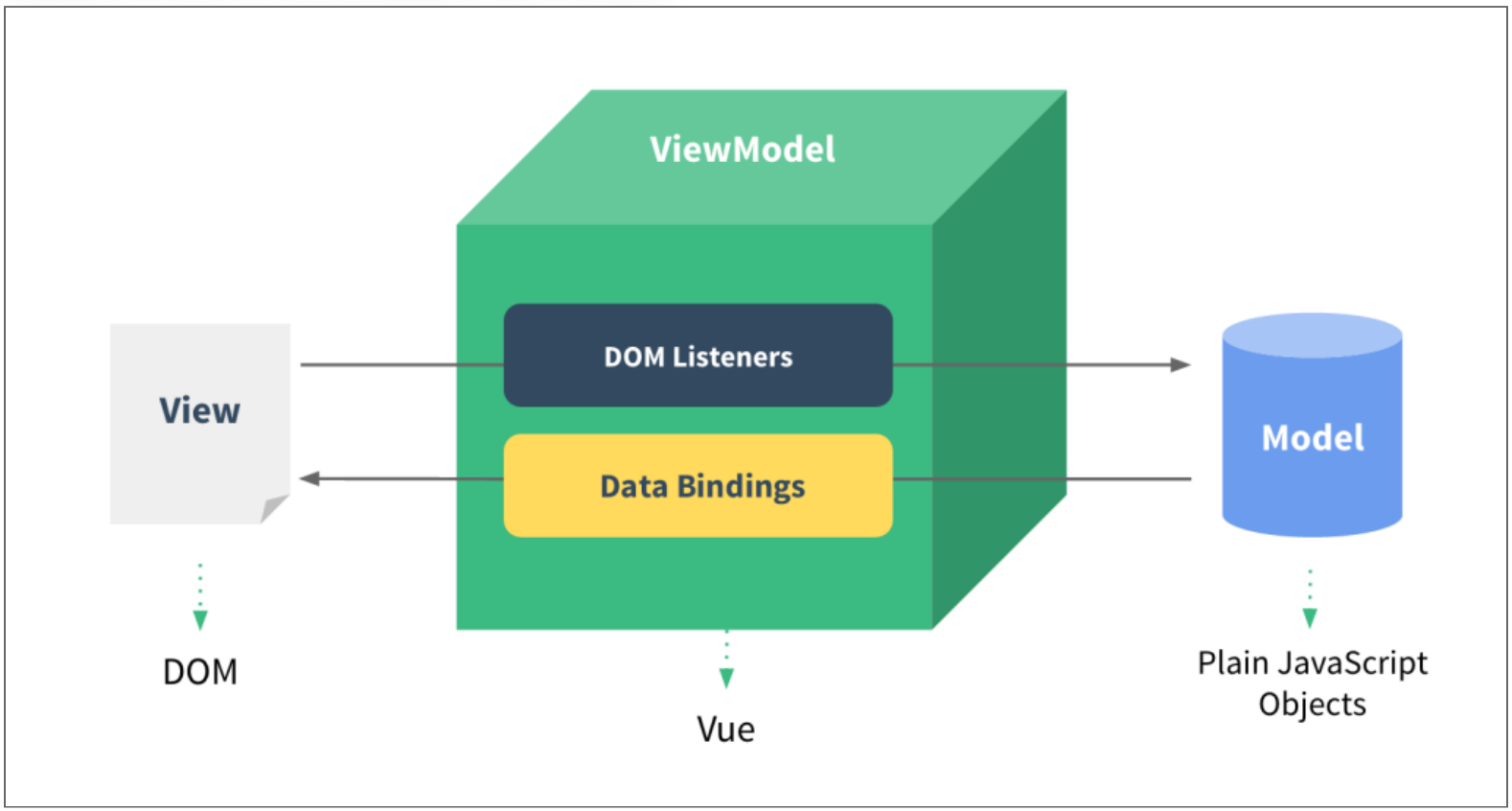
CARACTERÍSTICAS

- Vue.js es un framework de JavaScript para front-end.
- Facilidad de aprendizaje y uso con respecto a otros frameworks como **ReactJS**.
- Mejor rendimiento comparado con **AngularJS**.
- Vue.js es un framework **progresivo**. Tiene la facilidad para usarlo y adaptarlo a proyectos tanto grandes como pequeños.
- Ha tenido un grán crecimiento. Veamos su comunidad en **Github**.

CARACTERÍSTICAS

- Es un framework "reactivo" que implementa **"two way data-binding"**: enlace de datos en dos direcciones (entre la vista y el modelo) de una manera muy eficiente y rápida.
- Se basa en el patrón **Model-View-Viewmodel**
- Vue.js, está más enfocado hacia la vista, y **puede ser implementado en el HTML de cualquier proyecto web** sin requerir cambios drásticos .
- Los navegadores modernos poseen la extensión **Vue.js devtools** que nos asiste en el desarrollo (la versión de vue3 aún está en beta).
- Vue.js soporta todos los browsers que sean compatibles con **ES5-compliant**. (Ver **Versiones JS**.)

MODEL-VIEW-VIEWMODEL EN VUEJS



INSTALACIÓN VUEJS:

- Descargando el js e incluyéndolo directamente en un tag `<script>`.
- Linkear directamente desde una **CDN** (Content Delivery Network).
  Vue 2:

```
<script src="https://unpkg.com/vue"></script>
```

  Vue 3:

```
<script src="https://unpkg.com/vue@next"></script>
```

INSTALACIÓN VUEJS:

- Intalar Vue CLI via **npm** (manejador de paquetes por defecto para Node.js):

```
$ npm install -g @vue/cli
```

o

```
yarn global add @vue/cli
```

Esta es la opción recomendada para para **proyectos más grandes**.

INCLUYENDOLO DE UNA CDN:

Veamos **hello_world**.

```
<!DOCTYPE html>
<html>
<head>
  <title>Mi primera aplicaci&oacute;n Vue</title>
  <script src="https://unpkg.com/vue@next"></script>
</head>
<body>
  <div id="app">
    <h2>{{ message }}</h2>
  </div>

  <script>
    const HelloVueApp = {
      data() {
        return {
          message: 'Hola Vue!!!'
        }
      }
    }
   vue_app = Vue.createApp(HelloVueApp).mount('#app')
  </script>
</body>
</html>
```

Los datos y DOM están ahora relacionados utilizando **{{}}**.
Modifiquemos **vue_app.message**.

DIRECTIVAS VUE

DIRECTIVAS VUE

- Son atributos específicos de Vue que comienzan con *v-*.
    - **v-text**, **v-once**, **v-html**.
    - **v-bind**, **v-model**.
    - Condicionales: **v-if**, **v-else**, **v-else-if**.
    - Bucles: **v-for**.
    - Eventos: **v-on**.
    - **v-show**.

DIRECTIVA V-BIND:

- La intepolación {{}} no funciona para atributos, se utiliza v-bind para relacionar atributos con datos de Vue.
- Veamos **v-bind**.

```
<div id="app">
  <span v-bind:title="message">
    Hover your mouse over me !!
  </span>
</div>
<script>
  const App = {
    data() {
      return {
        message: 'Fecha ' + new Date().toLocaleString()
      }
    }
  }
  vue_app = Vue.createApp(App).mount('#app')
</script>
```

DIRECTIVA CONDICIONAL V-IF:

- Veamos **v-if**.

```
<div id="app">
  <p>
  <span v-if="seen">Now you see me</span>
  <span v-else>Oh no 😢</span>
  </p>
</div>
<script>
  const App = {
    data() {
      return {
        seen: true
      }
    }
  }
  vue_app = Vue.createApp(App).mount('#app')
</script>
```

BUCLES V-FOR:

- Veamos **v-for**.

```
<div id="app">
  <ul>
    <li v-for="product in products">
      {{ product }}
    </li>
  </ul>
</div>

<script>
  const App = {
    data() {
      return {
        products: [
          'Harina',
          'Arroz',
          'Yerba'
        ]
      }
    }
  }
 vue_app = Vue.createApp(App).mount('#app')
</script>
```

- Modifiquemos **vue_app.products**, agregando: **vue_app.products.push("Manteca")** y eliminando: **vue_app.products.pop()**.

MÉTODOS Y EVENTOS V-ON:

- La directiva **v-on** nos permite actuar cuando se produzca algún **evento DOM**.
- Dentro de la sección **methods** ponemos el método que se va a disparar cuando el evento se produzca.
- Veamos **v-on**.

```
<div id="app">
  <p>{{ message }}</p>
  <button v-on:click="reverseMessage">Reverse
Message</button>
</div>

<script>
  const App = {
    data() {
      return {
        message: 'Bienvenidos a Proyecto de Desarrollo!!'
      }
    },
    methods: {
      reverseMessage: function () {
        this.message =
this.message.split('').reverse().join('')
      }
    }
```

```
    }
    vue_app = Vue.createApp(App).mount('#app')
</script>
```

EVENTOS V-ON:

- Incluso es posible "colgarse" de múltiples eventos:

```
<div v-on="
  click   : onClick,
  keyup   : onKeyup,
  keydown : onKeydown
">
</div>
```

- Notar que se modifica el estado de nuestra applicación sin tocar el DOM, todo eso lo hace Vue.
- El código queda simplificado y enfocado en la lógica de lo que hay que resolver.

DIRECTIVA V-MODEL:

- Hace la relación bidireccional entre un input y los datos de la aplicación Vue.
- Veamos **v-model** y **v-models**.

```
<div id="two-way-binding">
  <p>{{ message }}</p>
  <input v-model="message">
</div>

<script>
  const App = {
    data() {
      return {
        message: 'Bienvenidos a Proyecto!!'
      }
    }
  }
 vue_app = Vue.createApp(App).mount('#two-way-binding')
</script>
```

PROPIEDADES COMPUTADAS:

- Veamos **propiedades-computadas**.
- Nos evita poner demasiada lógica en la visualización.

```
<div id="app">
  <ul>
    <li v-for="product in products">
      {{ product }}
    </li>
  </ul>
  Cantidad de elementos: {{ countProducts }}
</div>
<script>
  const App = {
    data() {
      return {
        products: [
          'Harina',
          'Arroz',
          'Yerba'
        ]
      }
    },
    computed: {
      // a computed getter
      countProducts: function () {
        return this.products.length
      }
```

```
      }
    }
  vue_app = Vue.createApp(App).mount('#app')
</script>
```

WATCHERS:

- Veamos **watcher**.
- Para reaccionar cuando un dato cambia.

```
<div id="app">
  <ul>
    <li v-for="product in products">
      {{ product }}
    </li>
  </ul>
  Cantidad de elementos: {{ countProducts }}
</div>
<script>
  const App = {
    data() {
      return {
        products: [
          'Harina',
          'Arroz',
          'Yerba'
        ],
        countProducts: 3
      }
    },
    watch: {
      products: {
        handler () {
          this.countProducts=this.products.length
```

```
            },
            deep: true
        }
      }
    }
    vue_app = Vue.createApp(App).mount('#app')
</script>
```

CONSUMIENDO UNA API CON VUEJS

EJEMPLO BÁSICO CONSULTANDO UNA API CON FETCH

- Veamos **lista-api**
- En este caso utilizamos el hook **created** dentro del ciclo de vida de la instancia Vue.

```
  <div id="app">
    Datos de: https://jsonplaceholder.typicode.com/
    <ul>
      <li v-for="post in posts">
        {{ post.id }} - <b>{{ post.title }}:</b> {{
post.body }}
      </li>
    </ul>
  </div>

  <script>
    const App = {
      data() {
        return {
          posts: []
        }
      },
      created() {

fetch('https://jsonplaceholder.typicode.com/posts')
          .then(response => response.json())
          .then(json =>{
```

```
            this.posts = json
          })
      }
    }
    vue_app = Vue.createApp(App).mount('#app')
  </script>
```

UTILIZANDO EL CLIENTE HTTP AXIOS

- Realiza los **XMLHttpRequests** del cliente (browser).
- Realiza las peticiones HTTP en node.js (servidor).
- Soporta API de **Promesas de JS**.
- Intercepta y transforma los datos de requerimientos y respuestas.
- Transforma automáticamente a JSON.
- Soporte del lado del cliente para protección contra **CSRF**.

EJEMPLO BÁSICO CONSULTANDO UNA API CON AXIOS

- Veamos **lista-api-axios**.
- Ahora, abramos directamente el archivo local en el navegador.

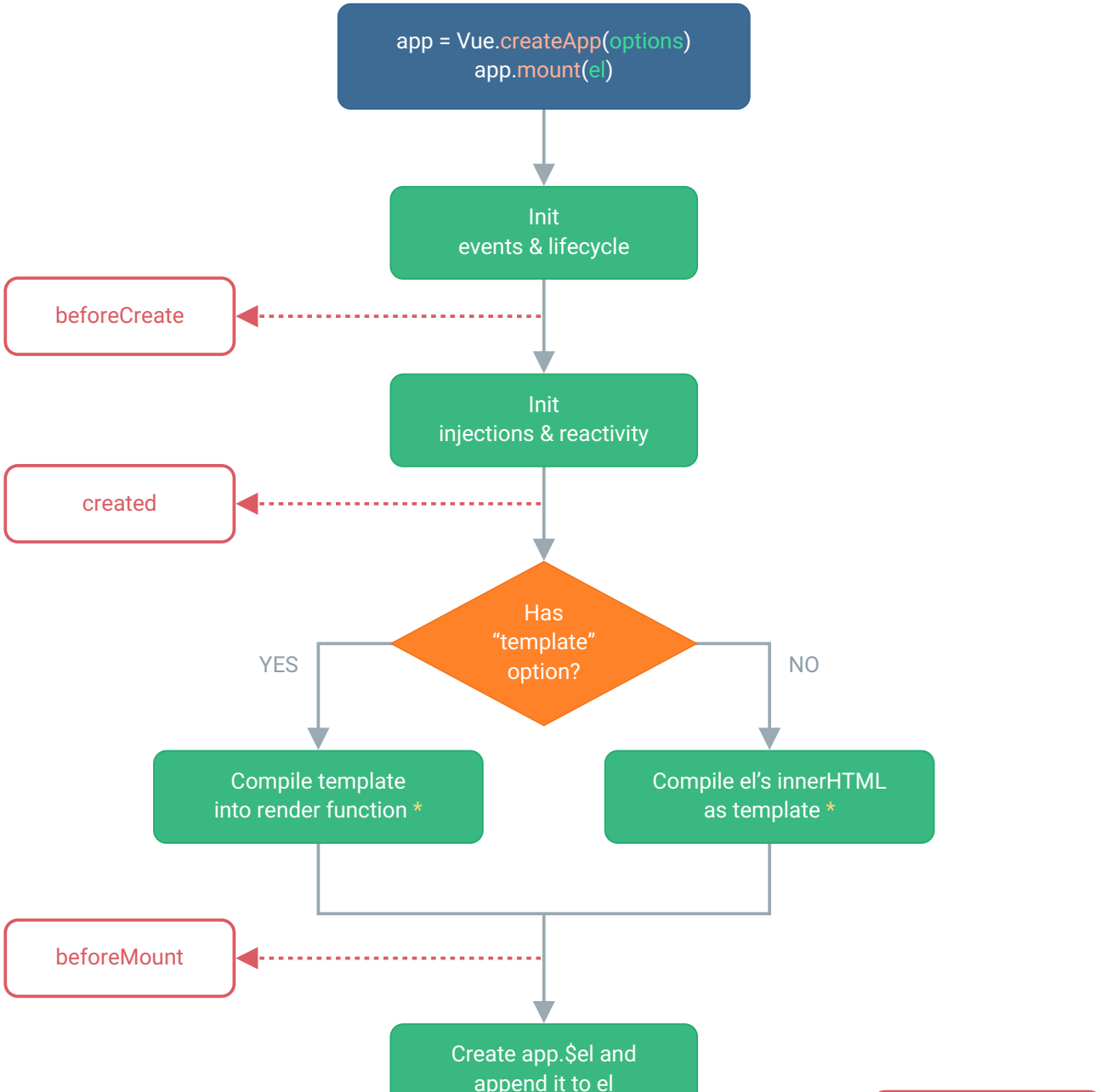
```
<!DOCTYPE html>
<html>
<head>
  <title>Provincias Argentinas</title>
  <meta charset="UTF-8" />
  <script src="https://unpkg.com/vue@next"></script>
  <script src="https://unpkg.com/axios/dist/axios.min.js">
</script>
</head>
<body>
  <div id="app">
    <div v-if="provincias && provincias.length">
      <h1>Provincias Argentinas:</h1>
      <ul>
        <li v-for="p in provincias">
          {{ p.id }} - {{ p.nombre }}
        </li>
      </ul>
    </div>
    <div v-if="errors && errors.length">
      <h1>Errores:</h1>
      <ul>
        <li v-for="error of errors">
          {{error.message}}
```

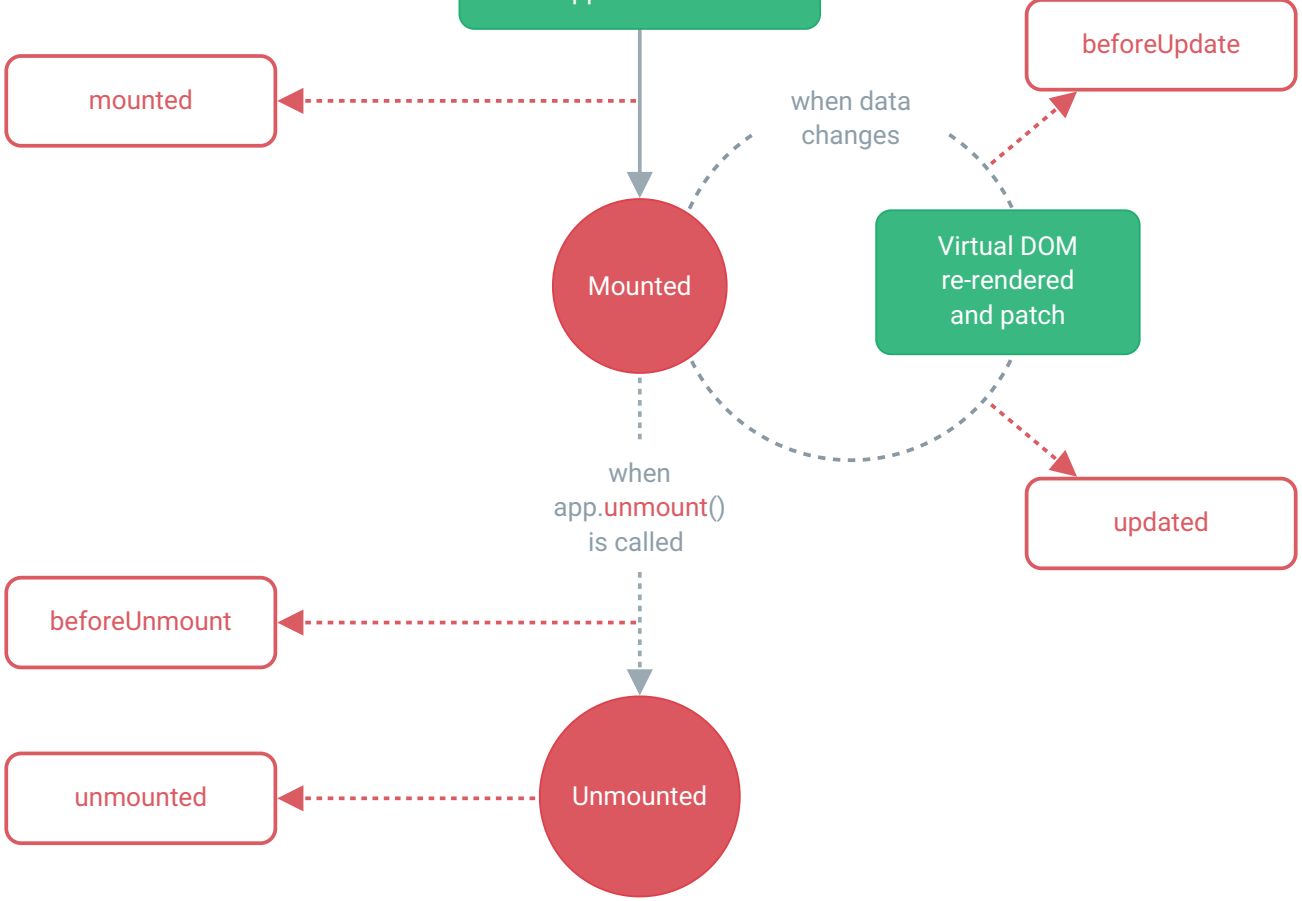
```
        </li>
      </ul>
    </div>
  Fuente: <a href="https://datos.gob.ar/"
target="blank">https://datos.gob.ar/</a>
  </div>
  <script>
    const App = {
      data() {
        return {
          provincias: [],
          errors: []
        }
      },
      created() {

axios.get('https://apis.datos.gob.ar/georef/api/provincias
')
              .then(response => {
              // JSON responses are automatically parsed.
              this.provincias = response.data.provincias
              })
              .catch(e => {
                this.errors.push(e)
              })
      }
    }
    vue_app = Vue.createApp(App).mount('#app')
  </script>
</body>
</html>
```

CICLO DE VIDA DE UNA INSTANCIA VUE:

Diagrama


app = Vue.createApp(options)
app.mount(el)
Init
events & lifecycle
beforeCreate
Init
injections & reactivity
created
Has
“template”
option?
YES
NO
Compile template
into render function *
Compile el’s innerHTML
as template *
beforeMount
Create app.$el and
append it to el



* Template compilation is performed ahead-of-time if using a build step, e.g., with single-file components.

REFERENCIAS VUE

- Guia Oficial Vue3 (inglés): **https://v3.vuejs.org/guide/introduction.html**
- Vue Mastery's Intro to Vue 3: **https://www.vuemastery.com/courses/intro-to-vue-3/intro-to-vue3/**

¿DUDAS?

SEGUIMOS CON VUEJS LA PRÓXIMA SEMANA...