



UNIVERSIDAD  
**NACIONAL**  
DE COLOMBIA

Ciclo III

## Desarrollo de Software



### Capa de Presentación: Introducción a Vue

13

**Jeisson Andrés Vergara Vargas**

Departamento de Ingeniería de Sistemas e Industrial

<http://colswu.unal.edu.co/~javergarav/>

[javergarav@unal.edu.co](mailto:javergarav@unal.edu.co)

2020

©

# Objetivo de Aprendizaje

**Identificar** la estructura y el funcionamiento general del framework Vue.

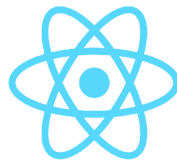
# Introducción

## Frameworks - Capa de Presentación

Existen **varias opciones** de **frameworks** para la construcción de **interfaces gráficas de usuario**:



Vue.js



React



SVELTE

# Vue

**Vue** es un **framework progresivo**, open source, de **JavaScript**, diseñado para la construcción de **interfaces de usuario** de manera **rápida y sencilla**.



# Instalación de Recursos

## Instalación de Node.js y NPM

**Vue** es un framework de **JavaScript**, por eso es necesario instalar **Node.js**, un **entorno de ejecución** de JavaScript.

También es necesario instalar **NPM**, que es el **administrador de paquetes** de **Node.js**.

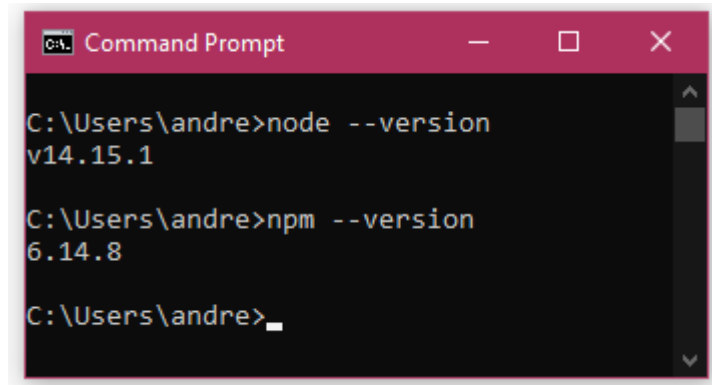
- **Windows:** descargar desde <https://nodejs.org/en/> el ejecutable y correrlo, seguir wizard.
- **Linux:** seguir la siguiente guía: <https://linuxize.com/post/how-to-install-node-js-on-ubuntu-18.04/>.

## Instalación de Node.js y NPM

Para **comprobar** si la instalación se realizó de manera **correcta**, ejecutar los siguientes **comandos** desde una **consola**:

```
node --version  
npm --version
```

Resultado esperado:



```
Command Prompt  
C:\Users\andre>node --version  
v14.15.1  
  
C:\Users\andre>npm --version  
6.14.8  
  
C:\Users\andre>
```



## Instalación del CLI-VUE

Para poder crear **proyectos en Vue**, es necesario instalar el **CLI** de **Vue**, este se instala usando **npm** y el siguiente **comando**:

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

Para **comprobar** si la fue **correcta** correr el siguiente comando:

```
vue --version
```

# Creación del Proyecto

## Creación del Proyecto

Para iniciar el **desarrollo del componente** se creará un **proyecto inicial** sobre el cual se podrá trabajar, para esto se usará el **cli de vue**.

Dirigirse a la **ubicación** donde se **creará el proyecto** y ejecutar el siguiente **comando**:

```
vue init webpack cajero_app
```

Esto iniciará una interfaz en la consola, con la construcción del proyecto.

## Creación del Proyecto

La **interfaz** solicitará unas **entradas** para configurar el proyecto:

Project Name: **carejo\_app**

Project Descripción: **una breve descripción**

Author: **su nombre**

Vue Build: **Runtime + Compiler**

Install vue-router? (Y/n): **n**

Use ESLint to lint your code? (Y/n): **n**

Set up unit test (Y/n): **n**

Setup e2e tests with Nightwatch? (Y/n): **n**

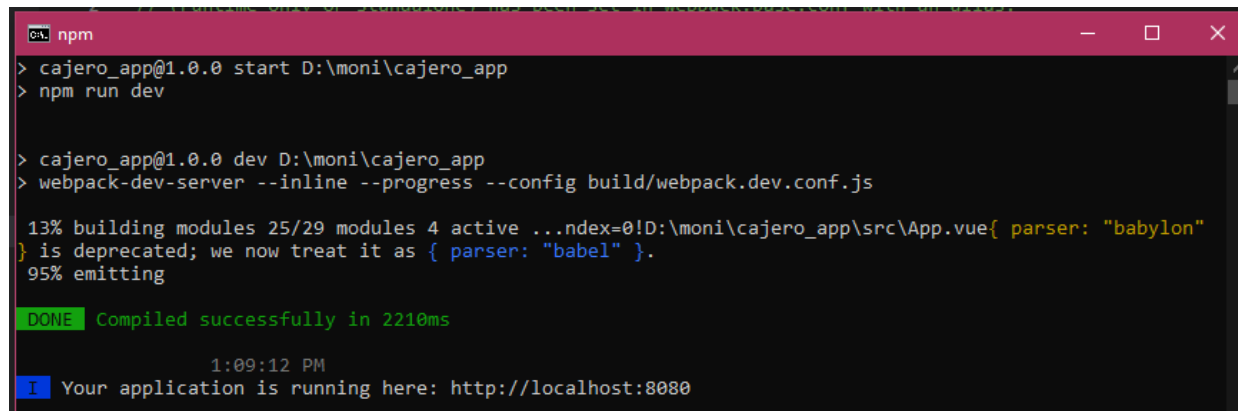
Should we run...: **Yes, use NPM**

## Probar el Proyecto

Para **comprobar** que el **proyecto** fue creado de manera **correcta**, ubicarse en la **raíz del proyecto** y ejecutar el siguiente **comando**:

`npm run start`

El resultado será:



```
cajero_app@1.0.0 start D:\moni\cajero_app
> npm run dev

cajero_app@1.0.0 dev D:\moni\cajero_app
> webpack-dev-server --inline --progress --config build/webpack.dev.conf.js

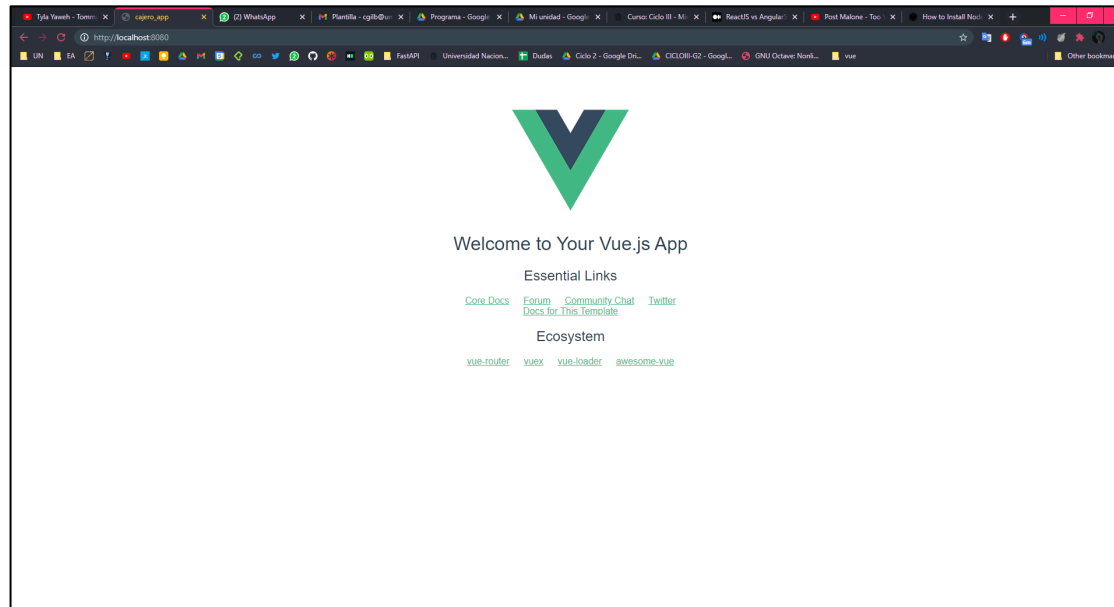
13% building modules 25/29 modules 4 active ...index=0!D:\moni\cajero_app\src\App.vue{ parser: "babel"
} is deprecated; we now treat it as { parser: "babel" }.
95% emitting

DONE Compiled successfully in 2210ms

1:09:12 PM
Your application is running here: http://localhost:8080
```

# Probar el Proyecto

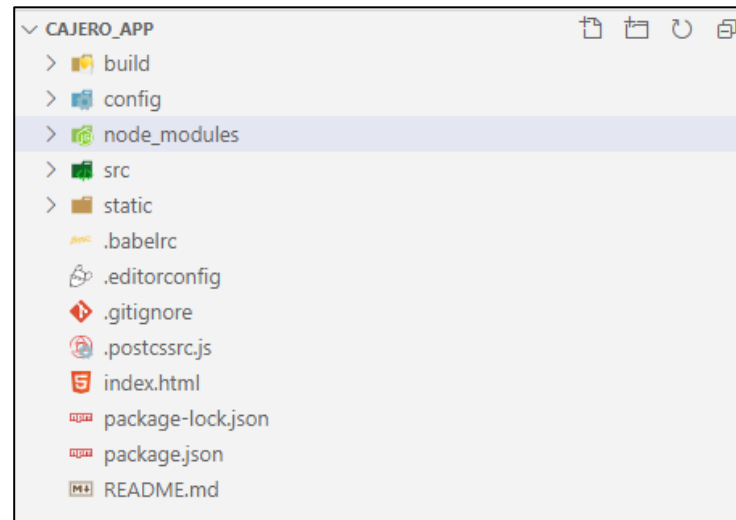
Dirigirse a la URI <http://localhost:8080>:



# Estructura del Proyecto

## Estructura del Proyecto

La **estructura del proyecto** será la siguiente:

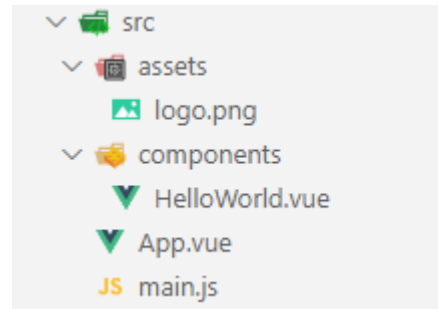


No es necesario **entender** todo el código, lo importante estará en **src**:



## Estructura del Proyecto

Dentro de **src**:



- **assets**: archivos planos
- **componets**: se ubican todos los componentes de la app
- **App.vue** el componente inicial de la app
- **main.js** el archivo principal de la app

## Archivo **main.js**

Este archivo **crea la app** e integra el componente **principal** App.vue:

```
import Vue from 'vue'
import App from './App'

Vue.config.productionTip = false

new Vue({
  el: '#app',
  components: { App },
  template: '<App/>'
})
```

## Eliminar Código Auto-Generado

Cuando se crea el **proyecto Vue**, se crean algunos **ejemplos de código**, estos se deben eliminar:

- Eliminar el archivo **HelloWorld.vue**
- Editar el archivo **App.vue**

## Editar App.vue

El archivo App.vue tendrá únicamente lo siguiente:

```
<template>  
  <div id="app"> </div>  
</template>
```

```
<script>  
export default {  
  name: 'App',  
}  
</script>
```

```
<style>  
</style>
```

# Conceptos Básicos de Vue

# Componente

Los **componentes** son elementos reutilizables, que representan una **pieza de código** que lleva acabo una **funcionalidad**, un componente tiene 3 partes principales:

Template

<HTML>

Script

<JavaScript>

Style

<CSS>

Cada una de las partes del código cumple una función.

## Script

El **principal** bloque de código de un componente es el **Script**, este también está dividido en varias partes, las 2 principales son:



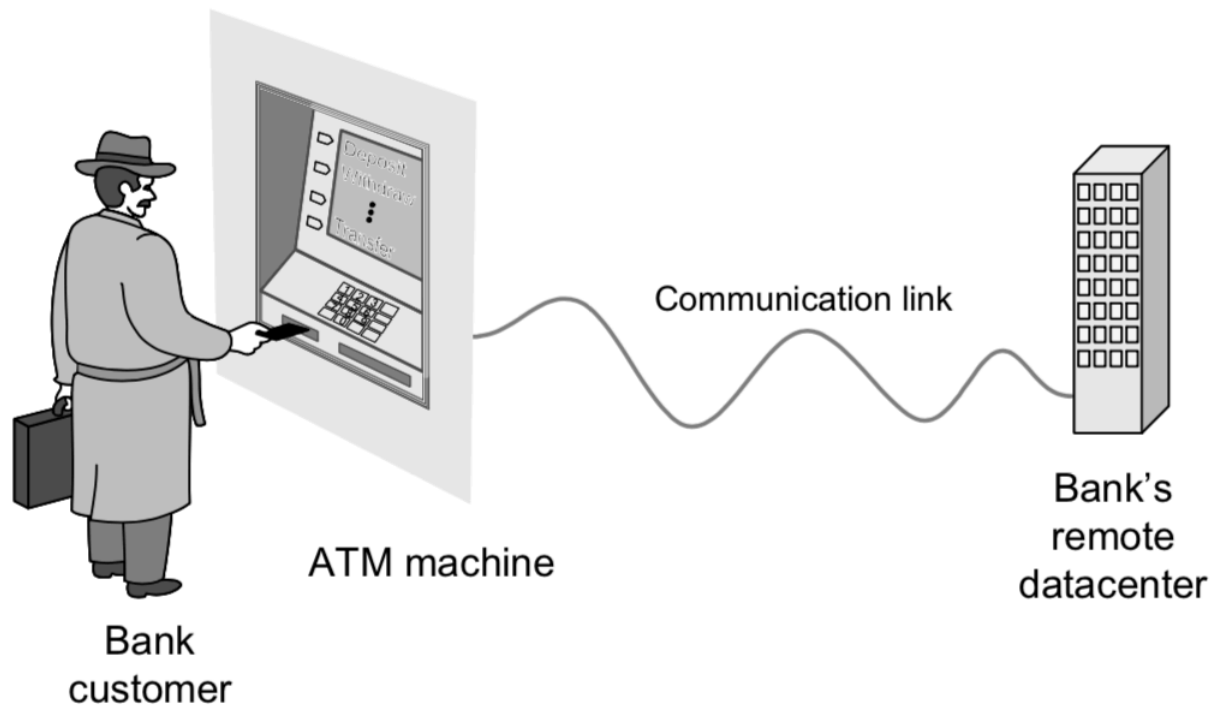
Data

Métodos

# Ejemplo



## Software para un «ATM»



## Ejemplo: Interfaz Principal

En esta **sección** se creara la **app** únicamente con el **componente padre**, en este estará la estructura global de la interfaz, que será:

- **header** con algunas opciones (sesión iniciada)
- **sección principal** en la que luego se añadirán componentes.
- **footer**

Se tendrán unas opciones solo disponibles cuando el usuario haya **iniciado sesión**.

## Ejemplo: Interfaz Principal



## Ejemplo: Usuario Precargado

De manera **provisional** , existirá un **usuario precargado** que simulará que ya se ha **iniciado sesión**. Lo anterior:

- **Evitará** que inicialmente se dediquen esfuerzos innecesarios al **inicio de sesión**.
- **Facilitará** implementar algunas funcionalidades.

El usuario estará **precargado** en **localStorage**:

# Implementación

## Implementación

Para **implementar** lo anterior se **editaré** únicamente el archivo **App.vue**, a continuación se muestra el código de cada una de las tres partes.

## Implementación: Template

El **código** que debe ir dentro de *template* será, **parte 1**:

```
<template>
  <div id="app">
    <div class="header">

      <h1>Banco UN</h1>
      <nav>
        <button v-on:click="init" v-if="is_auth" > Inicio </button>
        <button v-on:click="getBalance" v-if="is_auth" > Saldo </button>
        <button v-if="is_auth" > Transacción </button>
        <button v-if="is_auth" > Cerrar Sesión</button>
      </nav>
    </div>
```

## Implementación: Template

El **código** que debe ir dentro de *template* será, **parte 2**:

```
<div class="main-component">  
  
</div>  
  
<div class="footer">  
  <h2>Misión TIC 2022</h2>  
</div>  
  
</div>  
</template>
```



## Implementación: Script

El **código** que debe ir dentro de *script* será, **parte 1**:

```
<script>
export default {
  name: 'App',

  components: {},

  data: function(){
    return {
      is_auth: localStorage.getItem('isAuth') || false
    }
  },
}
```

## Implementación: Script

El **código** que debe ir dentro de `script` será, **parte 2**:

```
methods: {  
  
  },  
  
  beforeCreate: function(){  
    localStorage.setItem('current_username', 'camilo24')  
    localStorage.setItem('isAuth', true)  
  }  
}  
</script>
```

## Implementación: Style

El **código** que debe ir dentro de `style` será, **parte 1**:

```
<style>
  body{
    margin: 0 0 0 0;
  }
  .header{
    margin: 0%;
    padding: 0;
    width: 100%;
    height: 10vh;
    min-height: 100px;
    background-color: #283747 ;
    color: #E5E7E9 ;
    display: flex;
    justify-content: space-between;
    align-items: center;
  }
```

## Implementación: Style

El **código** que debe ir dentro de *style* será, **parte 2**:

```
.header h1{  
  width: 20%;  
  text-align: center;  
}  
  
.header nav {  
  height: 100%;  
  width: 45%;  
  
  display: flex;  
  justify-content: space-around;  
  align-items: center;  
  
  font-size: 20px;  
}
```

## Implementación: Style

El **código** que debe ir dentro de `style` será, **parte 3**:

```
.header nav button{  
  color: #E5E7E9;  
  background: #283747;  
  border: 1px solid #E5E7E9;  
  
  border-radius: 5px;  
  padding: 10px 20px;  
}  
  
.header nav button:hover{  
  color: #283747;  
  background: #E5E7E9;  
  border: 1px solid #E5E7E9;  
}
```

## Implementación: Style

El **código** que debe ir dentro de *style* será, **parte 4**:

```
.main-component{  
  height: 75vh;  
  margin: 0%;  
  padding: 0%;  
  background: #FD FE FE ;  
}  
  
.footer{  
  margin: 0;  
  padding: 0;  
  width: 100%;  
  height: 10vh;  
  min-height: 100px;  
  background-color: #283747;  
  color: #E5E7E9;  
}
```

## Implementación: Style

El **código** que debe ir dentro de `style` será, **parte 5**:

```
.footer h2{  
  width: 100%;  
  height: 100%;  
  
  display: flex;  
  justify-content: center;  
  align-items: center;  
}
```

```
</style>
```

# Ejecución

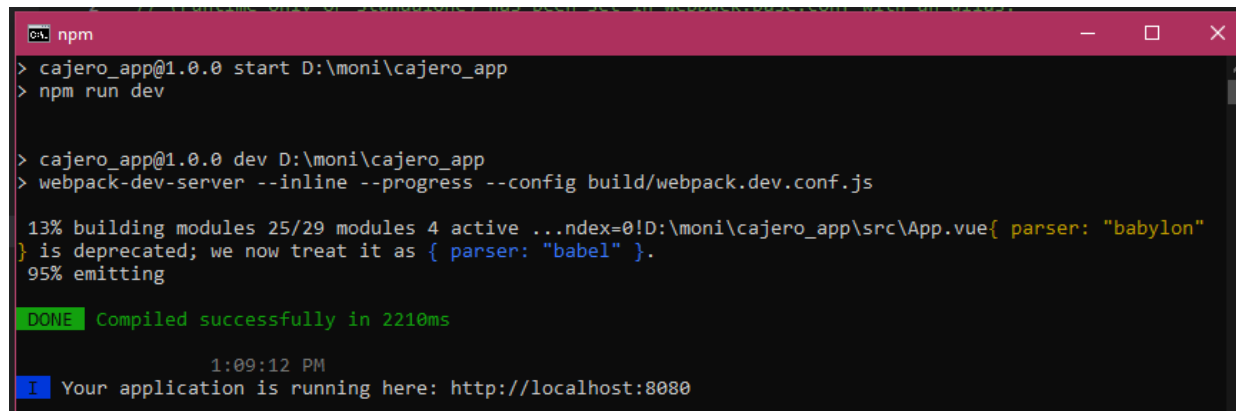


## Ejecución

Ubicarse desde una **consola** en la raíz del **proyecto** y ejecutar el **comando**:

`npm run start`

El resultado será:



```
npm
> cajero_app@1.0.0 start D:\moni\cajero_app
> npm run dev

> cajero_app@1.0.0 dev D:\moni\cajero_app
> webpack-dev-server --inline --progress --config build/webpack.dev.conf.js

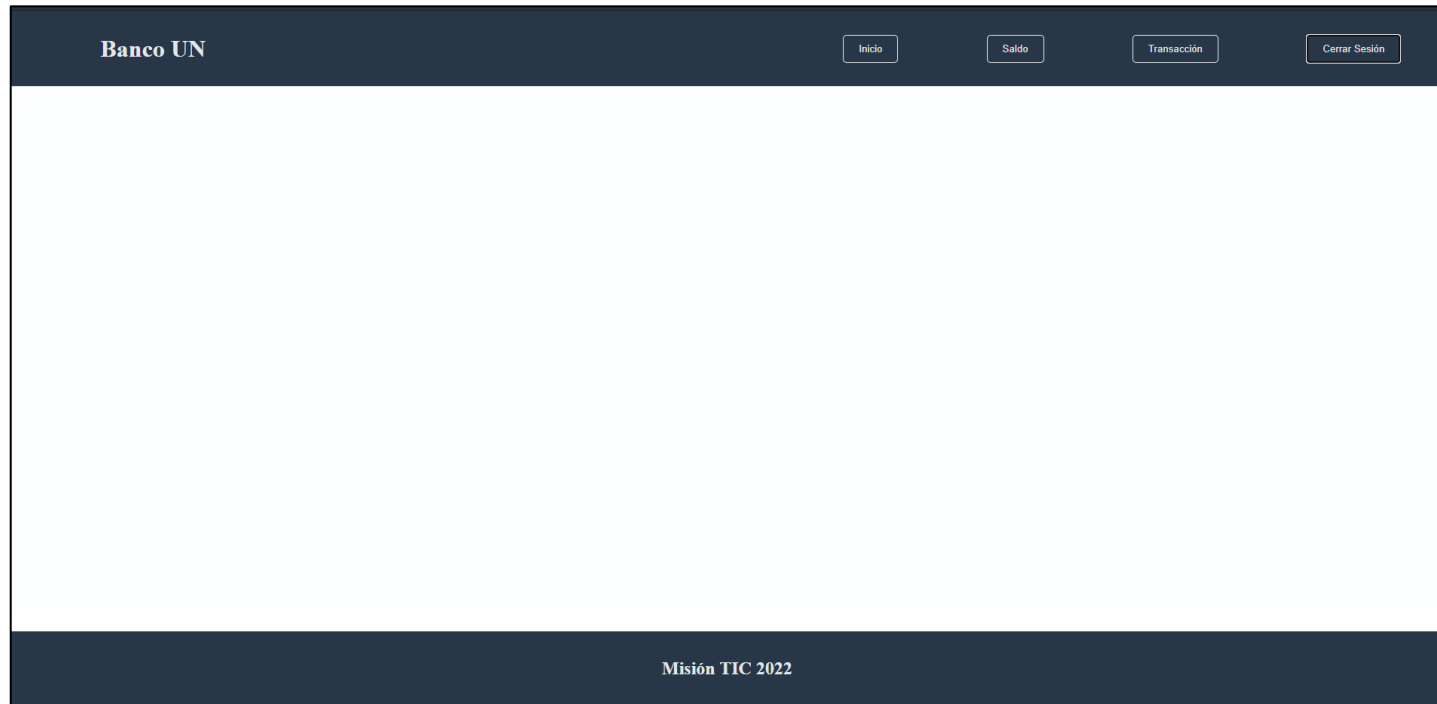
13% building modules 25/29 modules 4 active ...index=0!D:\moni\cajero_app\src\App.vue{ parser: "babel"
} is deprecated; we now treat it as { parser: "babel" }.
95% emitting

DONE Compiled successfully in 2210ms

1:09:12 PM
Your application is running here: http://localhost:8080
```

# Resultado

Dirigirse a <http://localhost:8080>:



## Referencias

- [Vue.js] (2020). Retrieved 6 December 2020, from <https://es.vuejs.org/>