



# Agustín Albarracín

## FULL STACK & REACT DEVELOPER.

4 años de carrera técnica en informática y 2 años de especialización en STACK MERN. Especialista de JavaScript, en entorno de React y Node.


+3 años trabajando como desarrollador web.


## DATOS PERSONALES


Edad: 27 años

Nacionalidad: Argentina

## CONTACTO

 agustin.albarracin@outlook.com

 +541130360442

 agustin-albarracin



## PERFIL PROFESIONAL

Soy un Full Stack con experiencia en el desarrollo de aplicaciones desde cero y como prestador de servicios para optimización de código, en empresas y con clientes en distintos rubros. Mi experiencia abarca desde, landing page, paginas web corporativas, blogs, e-commerce, dropshipping, dashboard, ERP, CRM, formularios y web de juegos online.

Me especializo en Node.js y React, cuento con experiencia en arquitectura de microservicios, bases de datos SQL / NoSQL y optimización de sistemas para el manejo de datos masivos y sensibles.

A lo largo de mi carrera, he trabajado en proyectos grupales tanto como en proyectos bajo el concepto de “end to end”.

### Algunas de mi actividades:

- Gestión y optimización de datos masivos en bases relacionales y no relacionales.
- Diseño y desarrollo de arquitecturas de APIs escalables y seguras.
- Implementación de pasarelas de pagos (débito, crédito).
- Estructuración de servidores aplicando principios SOLID.
- Desarrollo de FRONT-END siguiendo patrones de diseño y buenas prácticas.
- Optimización de código en sistemas financieros y transacción.
- Implementación de medidas de seguridad en consultas y páginas web.
- Automatización de procesos y CI/CD para optimizar despliegues.
- Colaboración en equipos ágiles bajo metodologías Scrum.



## EXPERIENCIA LABORAL

01/2024 -

Actual

### VIBRA Digital - Full Stack

Creación de aplicaciones y paneles administrativos para datos masivos.

El mismo cuenta con dashboard para las finanzas de la empresa, estadísticas de los empleados, acceso rápido a depósitos y retiros.

Vista y filtros para todos los empleados según su puesto, los cuales contiene botones de acciones, como gestionar cambio de contraseña, historial recientes de transacciones, edición de datos, eliminación del usuario.

Sección para filtrar movimientos de empresa, administrador, cajeros y usuarios.

Sección para administrar la vista interactiva de los usuarios.

Sección para administrar el proveedor de datos y aplicación según la Empresa.

### FRONT-END.

#### Biblioteca REACT.

Utilizamos todos los hooks, de rutas, estados, efectos, memoizado, referencias y contexto, desde v16 hasta v19.

Para los estilos se utilizo styled.components, y se utilizan conjuntos de librerías para cubrir la necesidad del cliente ya que una sola librería a veces no aporta las funciones necesaria para completar o solventar la solicitud del cliente, siempre dentro de la biblioteca de React.

### Librerías:

Para iconos: Librería de “react-icons” o diseños personalizados por agente de diseño gráfico.

Para notificaciones: “react-toastify” o se utiliza “sweetalert2” si se requiere de confirmaciones o una notificación interactiva.

Para formularios: “react-select” en caso de formularios complejos y personalizados.

Para fechas y horarios: “react-datepicker” para filtros de fechas o crear calendarios, junto a “moment” para formatear las fechas según zona horaria.

Para vista de cargas: “react-spinners” para visualizar la espera de la carga en caso de que se utilice.

Para modales: se utilizo “react-modals”.

Para los carrusel: “react-slick”

Para dashboard y finanzas: libreria de “echarts” interactiva.

Para solicitudes y comunicación con el servidor utilizamos: “axios”.

06/2021 -  
2024

## **BACK-END.** **Biblioteca de NODE.JS.**

El servidor lo dividimos en index, modales, rutas, controladores y helpers en caso de utilizarlos. Para la base de datos utilizamos MongoDB o MySQL si la asigna el host. Utilizamos Docker, para la gestión de caché y archivos de entornos para los datos privados.

### **Librerías:**

Para la gestión de Mongo: conexión a través de “moongose”.  
Para creación del server: “express”, “http”, y “cors” para la seguridad.  
Para carga de archivos: “multer” si se requiere de datos.  
Para encriptación: “bcryptjs” y tokens con “jsonwebtoken”.  
Para la comunicación continua con el front: “socket.io”, como para notificaciones en vivo.  
Para almacenamiento caché: “redis”, “redis-compose”.

## **Copycode Software Factory - Full Stack**

Creación de aplicaciones y páginas web, en la mismas me desempeñe en la creación de componentes reutilizables desde la biblioteca de React y manejar los contextos desde **Redux**.

### **FRONT-END.**

#### **Biblioteca REACT.**

Utilizamos todos los hooks, de rutas, estados, efectos, memoizado, referencias y contexto de v16.

Para los estilos se utilizo **SASS y Tailwind** y se utilizaron librerías según el equipo, como **Bootstrap, Material UI, Ant Design**.

### **Otras librerías:**

Para iconos: Librería de “react-icons” o diseños personalizados por agente de diseño gráfico..  
Para fechas y horarios: “react-datepicker” para filtros de fechas o crear calendarios, junto a “moment” para formatear las fechas según zona horaria.  
Para modales: se utilizo “react-modals”.  
Para dashboard y finanzas: librería de “echarts” interactiva.  
Para solicitudes y comunicación con el servidor utilizamos: “axios” y fetch.

### **BACK-END.**

#### **Biblioteca de NODE.JS.**

El servidor lo dividimos en index, modales, rutas, controladores y helpers en caso de utilizarlos. Para la base de datos utilizamos MongoDB o cualquier DB relacional.

### **Librerías:**

Para la gestión de Mongo: conexión a través de “moongose”.  
Para creación del server: “express”, “http”, y “cors” para la seguridad.  
Para encriptación: “bcryptjs” y tokens con “jsonwebtoken”.  
Para la comunicación continua con el front: “socket.io”, como para notificaciones en vivo.

## PROYECTOS

[github.com/Agus-Albarracin](https://github.com/Agus-Albarracin)



## EDUCACIÓN

**Escuela Técnica N°3 Arturo Arlt, informática.**  
2009 - 2015.

**HENRY Academy.**  
2022 - 2023.



## CONOCIMIENTOS

Pensamiento constructivo.  
Desarrollo de opciones para soluciones.  
Organización de equipos.  
Conocimiento de IA.

Gestión de clientes.  
Lectura de arquitectura para construcción de APIS.



## TECNOLOGÍAS

JavaScript.  
React.js.  
Redux.  
Node.js.  
Express.js.  
Vite.js.  
Next.js.  
REST API.  
PostgreSQL.  
MySQL.  
MongoDB.  
GIT.  
GITHUB.  
Thunder.  
Postman.

Metodologías Ágiles:  
Scrum.  
Trello.  
Jira.  
Kanban.

Arquitecturas:  
SPA.  
Client/Server.  
MERN tech Stack.  
Flux Architecture.  
MVC design pattern.

Principios:  
SOLID.  
Observation Principle.  
Dependency Injection Principle.