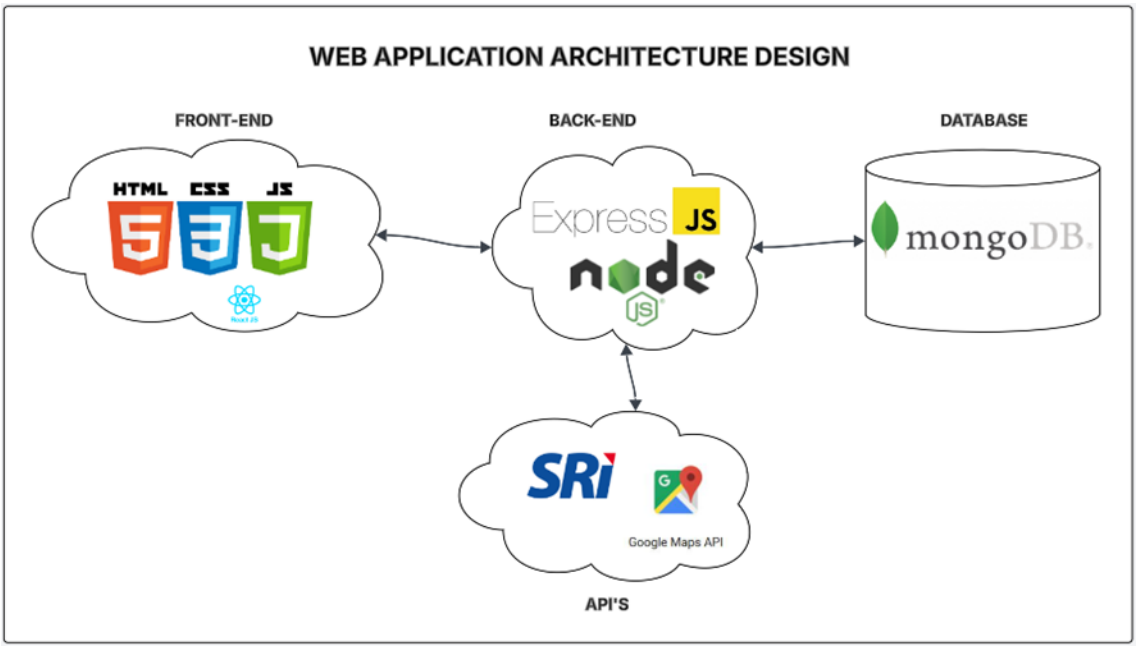
TEAM 5

- Lainez Ricardo
- Llumiquinga ariel
- Santi Jeancarlo

WEB APPLICATION ARCHITECTURE DESIGN



This project is developed using a modern web application architecture, decoupling the frontend (client) from the backend (server). The core technology is based on the MERN stack (MongoDB, Express, React, Node), substituting React with "vanilla" JavaScript for DOM manipulation.

1. Frontend (Presentation Layer)

The frontend is the user interface (UI) that the client interacts with directly in their web browser. It is responsible for the visual presentation, interactivity, and collection of user data.

- **HTML5:** Used as the standard markup language to define the **semantic structure** and content for all application views (e.g., menu, registration, and contact pages).
- **CSS3:** Employed for **visual design and presentation**. This includes the color palette, typography, layout (via Flexbox or Grid), and the implementation of **responsive design** to adapt to both mobile and desktop devices.
- **JavaScript:** This is the client-side programming language. Its function is to manage page **interactivity**, validate forms (client-side validation), manipulate the DOM (showing/hiding modals, updating content), and, fundamentally, to act as the HTTP client to communicate with the backend via the fetch API to send and receive data (e.g., submitting a registration form).

2. Backend (Logic and Server Layer)

The backend is the logical core of the application. It runs on a server and is the only component with direct access to the database and external services, ensuring data security and integrity.

- **Node.js:** Selected as the **server-side runtime environment**. Its primary advantage is the use of the JavaScript language, which allows for unified development in a single language (isomorphism). Its asynchronous, non-blocking I/O (Input/Output) model makes it highly efficient for handling multiple concurrent connections, such as user requests.
- **Express.js (Framework):** Built on top of Node.js, the Express framework is used to simplify the creation of the **RESTful API**. Express manages routing (defining endpoints like /api/register), handles HTTP requests (GET, POST, PUT, DELETE), and administers *middlewares* (such as cors for security or bcrypt for password encryption).

3. Database (Persistence Layer)

The database is the system responsible for storing, organizing, and retrieving all persistent application information, such as user profiles, reservations, and the menu.

- **MongoDB (Atlas):** MongoDB has been chosen as the database management system (DBMS). It is a document-oriented **NoSQL** database. It stores data in a JSON-like format (BSON), which allows for **great schema flexibility** and a native, seamless integration with JavaScript/Node.js. The **MongoDB Atlas** cloud service is used to simplify database deployment, security, and scalability.

4. Third-Party Services (External APIs)

To extend the system's functionality without reinventing complex components, the backend will integrate with external service APIs.

- **Google Maps API:** This will be consumed to provide geolocation functionalities. Its main use will be to display interactive maps in the "Locations" section, allowing users to visualize the restaurant's location and get directions.
- **SRI API:** This integration is critical for business logic. The backend will communicate with the web services of the SRI (Ecuador's Internal Revenue Service) for **electronic invoicing generation**, ensuring tax compliance for transactions or reservations generated through the platform.