Introducing the Node.js-to-Angular Stack

Understanding the Basic Web Development Framework

The main components of any given web framework are

1. the user,
2. browser,
3. webserver,
4. and backend services.

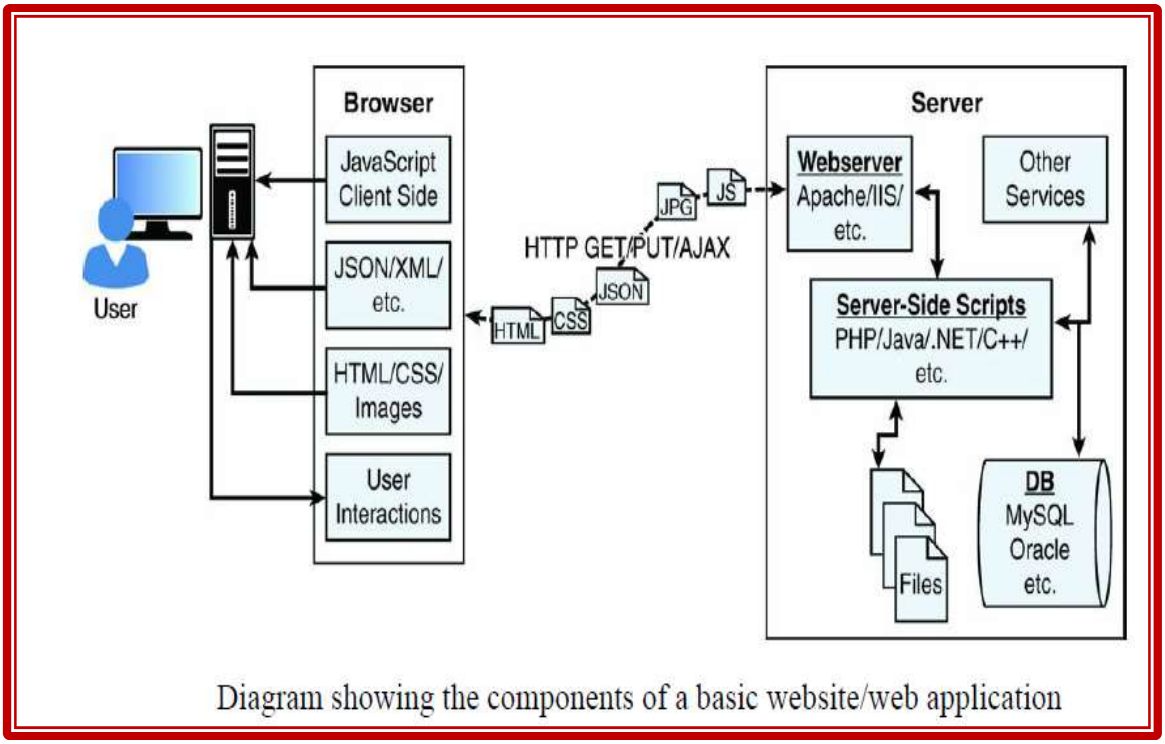


Diagram showing the components of a basic website/web application

User

- Users are a fundamental part of all websites; they are, after all, the reason websites exist in the first place.
- User expectations define the requirements for developing a good website, and these expectations have changed a lot over the years.
- The user role in a web framework is to sit on the visual output and interaction input of webpages.

- That is, users view the results of the web framework processing and then provide interactions using mouse clicks, keyboard input, and swipes and taps on mobile devices.

Browser

The browser plays three roles in the web framework.

- **First, it provides communication to and from the webserver.**
- **Second, it interprets the data from the server and renders it into** the view that the user actually sees.
- **Finally, the browser handles user interaction** through the keyboard, mouse, touchscreen, or other input device and takes the appropriate action.

Browser to Webserver Communication

- Browser-to-webserver communication consists of a series of requests using the
- HTTP and HTTPS protocols.
- Hypertext Transfer Protocol (HTTP) defines communication between the browser and the webserver.

The browser makes three main types of requests to the server:

1. **GET:** The GET request is typically used to retrieve data from the server, such as .html files, images, or JSON data.
2. **POST:** POST requests are used when sending data to the server, such as adding an item to a shopping cart or submitting a web form.
3. **AJAX:** Asynchronous JavaScript and XML (AJAX) is actually just a GET or POST request done directly by JavaScript running in the browser. Despite the name, an AJAX request can receive XML, JSON, or raw data in the response

Rendering the Browser View

- The browser reads data from the initial URL and then renders the HTML document to build a Document Object Model (DOM). The DOM is a tree structure object with the HTML document as the root. The structure of the tree basically matches the structure of the HTML document.

Webserver

- The webserver's main focus is handling requests from browsers.
- The webserver uses the HTTP headers as well as the URL to determine

What action to take. This is where things get different depending on the webserver, configuration, and technologies used.

- Most out-of-the-box webservers, such as Apache and IIS, are made to serve static files such as .html, .css, and media files.
- ***A server-side program is really anything that can be executed by the webserver to perform the task the browser is requesting. These can be written in PHP, Python, C, C++, C#, Java, … the list goes on and on.***
- Webservers such as Apache and IIS provide mechanisms to include server-side scripts and then wire them up to specific URL locations requested by the browser.

Backend Services

- Backend services are services that run behind the webserver and provide data used to build responses to the browser. The most common type of backend service is a database that stores information.
- When a request comes in from the browser that requires information from the database or other backend service, the server-side script connects to the database, retrieves the information, formats it, and then sends it back to the browser.
- Conversely, when data comes in from a web request that needs to be stored in the database, the server-side script connects to the database and Updates the data.

Understanding the Node.js-to-Angular Stack Components:

- **Why MEAN Stack?**
  - ✓ Fast setup
  - ✓ Match the language on the frontend
  - ✓ Single thread request handling

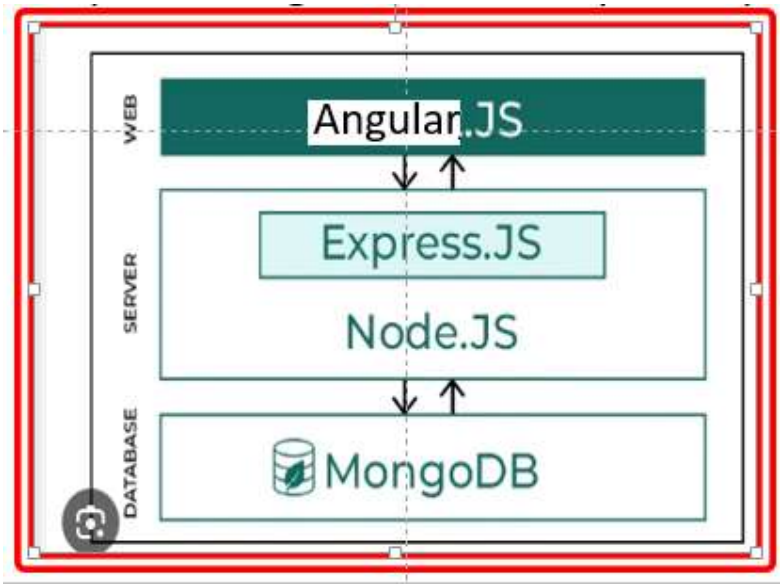
Node.js-to-Angular stack applications are different because the framework is different from more traditional JavaScript web application programming.

- JavaScript frontend and backend web application setup. It uses:
- MongoDBNoSql database
- ExpressWeb application framework
- AngularFrontend databinding framework
- NodeServer-side JavaScript environment

What is MEAN Stack?

The MEAN stack is a JavaScript-based framework for developing web applications. MEAN is named after MongoDB, Express, Angular, and Node, the four key technologies that make up the layers of the stack

- MEAN Stack is one of the most popular Technology Stack. It is used to develop a Full Stack Web Application. Although it is a Stack of different technologies, all of these are based on JavaScript language.
- **MEAN Stands** for:
    - **M** – MongoDB
    - **E** – Express
    - **A** – Angular
    - **N** – Node.js



- This stack leads to faster development as well as the deployment of the Web Application. Angular is Frontend Development Framework whereas Node.js, Express, and MongoDB are used for Backend development

Node.js-to-Angular stack Componenets:

The Node.js-to-Angular stack comprised of four Componenets.They are

1. MongoDB,
2. Express,
3. Angular,
4. and Node.js.

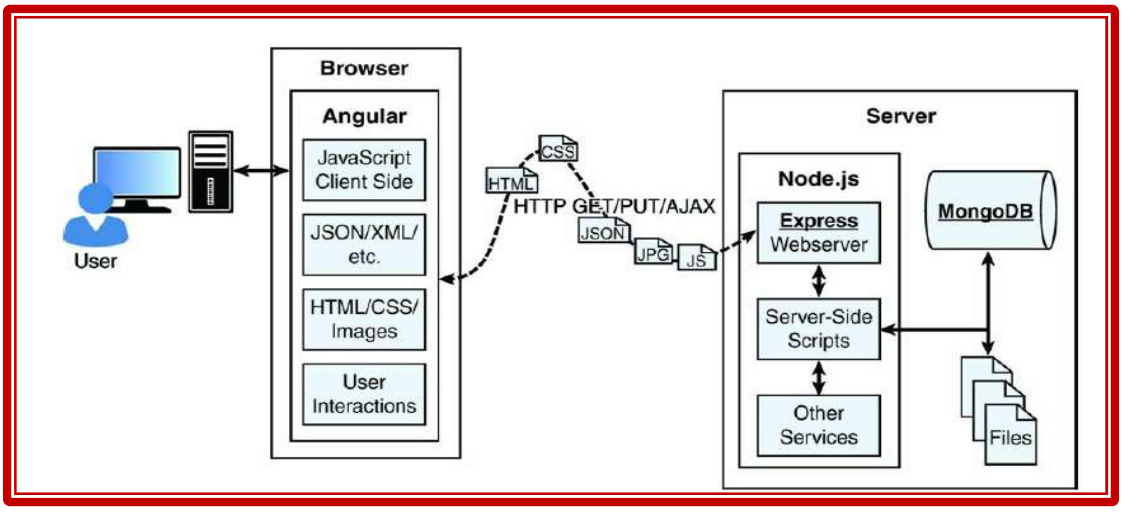


Fig. Node.js-to-Angular Stack web paradigm

1.Node.js

- Node.js is a development framework based on Google's V8 JavaScript VM Engine. Therefore, Node.js code is written in JavaScript and then compiled into machine code by V8 to be executed.
- Many of your backend services can be written in Node.js, as can the server-side scripts and any supporting web application functionality.
- In the Node.js-to-Angular stack, Node.js provides the fundamental platform for development. The backend services and server-side scripts are all written in Node.js. MongoDB provides the data store for the website but is accessed via a MongoDB driver Node.js module. The webserver is defined by Express, which is also a Node.js module.

The following are valuable features of Node.js:

1. **JavaScript end-to-end:** One of the biggest advantages to Node.js is that it allows you to write both server- and client-side scripts in JavaScript.
2. **Event-driven scalability:** Node.js applies a different logic to handling web requests. Rather than having multiple threads waiting to process web requests, they are processed on the same thread using a basic event model.
3. **Extensibility:** Node.js has a great following and an active development community. New modules to extend Node.js functionality are being developed all the time. Also it is simple to install and include new modules in Node.js, making it easy to extend a Node.js project to include new functionality in minutes.
4. **Time:** Let's face it, **time is valuable**. Node.js is super easy to set up and develop in. In only a few minutes, you can install Node.js and have a working webserver.

2.MongoDB

- ***MongoDB is an agile and scalable NoSQL*** database.
- It is based on the ***NoSQL document store model***, meaning that data is stored in the database as **a form of JSON objects** rather than the traditional columns and rows of a relational database.
- ***MongoDB provides great website backend storage for high traffic websites*** that need to store data such as user comments, blogs, or other items because it is fast, scalable, and easy to implement
- ***MongoDB driver*** library to access MongoDB **from Node.js.**

The following are valuable features of MongoDB:

1. **Document orientation:** Because MongoDB is document-oriented, the data is stored in the database in a format close to what you will be dealing with in both server-side and client-side scripts. This eliminates the need to transfer data from rows to objects and back.
2. **High performance:** MongoDB is one of the highest performing databases available. Especially today when more and more people interact with websites, it is important to have a backend that can support heavy traffic.
3. **High availability:** MongoDB's replication model makes it easy to maintain scalability while keeping high performance.
4. **High scalability:** MongoDB's structure makes it easy to scale horizontally by sharing the data across multiple servers.
5. **No SQL injection:** MongoDB is not susceptible to SQL injection (putting SQL statements in web forms or other input from the browser that compromises the DB security) because objects are stored as objects, not using SQL strings.

3.Express

- The Express module acts as the webserver in the Node.js-to-Angular stack.
- The fact that it is running in Node.js makes it easy to configure, implement, and control.
- the Express module provides the ability to easily set up destination routes (URLs) for users to connect to.
- It also provides great functionality on working with the HTTP request and response objects, including things like cookies and HTTP headers.

- The Express module extends Node.js to provide several key components for handling web requests.

The following are valuable features of Express:

1. **Route management:** Express makes it easy to define routes (URL endpoints) that tie directly to Node.js script functionality on the server.
2. **Error handling:** Express provides built-in error handling for documents not found and other errors.
3. **Easy integration:** An Express server can easily be implemented behind an existing reverse proxy system such as Nginx or Varnish. This allows it to be easily integrated into your existing secured system.
4. **Cookies:** Express provides easy cookie management.
5. **Session and cache management:** Express also enables session management and cache management.

4.Angular

- ✓ Angular is a perfect client-side framework for most web applications because it provides a very clean and structured approach. With a clean, structured front end,
- ✓ Angular is a client-side **framework developed by Google.**
- ✓ Angular provides all the functionality needed to handle user input in the browser, manipulate data on the client side, and control how elements are displayed in the browser view.
- ✓ It is written ***using TypeScript.***

Here are some of the benefits of Angular:

1. **Data binding:** Angular has a clean method to bind data to HTML elements using its powerful scope mechanism.
2. **Extensibility:** The Angular architecture allows you to easily extend almost every aspect of the language to provide your own custom implementations.
3. **Clean:** Angular forces you to write clean, logical code.
4. **Reusable code:** The combination of extensibility and clean code makes it easy to write reusable code in Angular. In fact, the language often forces you to do so when creating custom services.
5. **Support:** Google is investing a lot into this project, which gives it an advantage over other similar initiatives.

6. **Compatibility:** Angular is based on TypeScript, which makes it easier to begin Integrating Angular into your environment and to reuse pieces of your existing code within the structure of the Angular framework.