**What is Express.js?**

Express.js is a popular, fast, and minimalist web application framework for Node.js. It simplifies the process of building web applications and APIs by providing a robust set of features and tools. Express is designed to be minimal, flexible, and unopinionated, allowing developers to build scalable, dynamic, and high-performance web applications with ease.

Express.js is part of the **MEAN** stack (MongoDB, Express, Angular, Node.js) and **MERN** stack (MongoDB, Express, React, Node.js) and is frequently used to build RESTful APIs, real-time applications, and server-side rendered websites.

**Necessity of Express.js in Web Development**

Web development today demands high performance, scalability, and flexibility. While **Node.js** provides the runtime environment to run JavaScript on the server-side, it lacks the high-level abstractions and utilities that can help developers build web applications quickly. This is where Express.js comes in.

1. **Simplifies HTTP Requests Handling**: Express.js abstracts away the complexities of handling HTTP requests, routing, and middleware. It provides an intuitive API for managing HTTP methods like GET, POST, PUT, DELETE, etc.
2. **Middleware Integration**: Express.js makes it easier to add middleware functionality to the server. Middleware is used to handle things like authentication, logging, error handling, and request processing in a clean, modular way.
3. **Routing**: Express provides an easy and efficient way to define routes for various HTTP requests. This allows developers to organize their application routes logically and handle them dynamically.
4. **Error Handling**: Express includes built-in mechanisms to handle errors at a global or per-route level, which simplifies debugging and error tracking in production environments.
5. **Template Engine Support**: Express allows the use of various template engines like EJS, Pug, and Handlebars for rendering dynamic HTML content on the server side.
6. **Integration with Databases**: Express works seamlessly with databases (both relational and NoSQL), making it easy to fetch data from, store data in, and manipulate data within databases.
7. **Lightweight**: Express itself is minimal, and developers can choose the exact set of features they need by adding packages and libraries, thus preventing unnecessary bloat.

**Core Features of Express.js**

* **Routing**: Handles routes for HTTP requests (GET, POST, etc.), allowing developers to structure their applications effectively.
* **Middleware**: Functions that execute during the lifecycle of a request to modify the request, perform actions, or terminate the request-response cycle.
* **Template Engines**: Support for engines like EJS, Pug, Handlebars, which allow rendering dynamic HTML from templates.
* **Serving Static Files**: Express simplifies serving static files like images, CSS, and JavaScript to clients.
* **Error Handling**: Built-in error-handling middleware simplifies error management.
* **Request & Response Handling**: Express provides methods to manage request data (query parameters, body, headers) and send responses in multiple formats (JSON, HTML, plain text).

**Pros of Express.js**

1. **Minimal and Unopinionated**: Express is very lightweight and allows developers to structure their apps the way they want. It doesn’t enforce any particular structure or libraries to use.
2. **Great Flexibility**: Express is highly flexible and can be used to build a wide range of applications, from simple APIs to complex server-side rendered applications.
3. **Large Ecosystem**: Being built on top of Node.js, Express benefits from the vast npm (Node Package Manager) ecosystem, which provides many libraries and modules that can be easily integrated.
4. **Middleware Support**: Express’s middleware functionality enables the inclusion of code at specific points during the request-response cycle, allowing for advanced request handling, authentication, logging, and more.
5. **Fast Development Speed**: Express simplifies a lot of the web development process. Its concise and easy-to-understand API lets developers build applications quickly and with fewer lines of code.
6. **Scalable**: Express is well-suited for building scalable applications that handle a large number of concurrent users. Its non-blocking, event-driven architecture ensures that it can manage many requests simultaneously without significant performance degradation.
7. **Community Support**: Express has been around since 2010 and has a massive community of developers and contributors. There’s extensive documentation, a wide variety of tutorials, and a large number of plugins and extensions.
8. **Asynchronous**: Built on Node.js, Express inherits Node’s non-blocking, asynchronous capabilities, which improves the performance and responsiveness of applications.

**Cons of Express.js**

1. **Minimal Features**: While Express is minimal, it sometimes requires more configuration for features that might be built-in in other frameworks. For example, built-in support for input validation or session handling is not available, so developers need to use external packages.
2. **No Built-In ORM**: Unlike frameworks like Django (Python) or Ruby on Rails, Express does not include a built-in ORM (Object-Relational Mapping) for database management. While this can be a pro for developers who want flexibility, it also means additional work is needed to integrate a database.
3. **Callback Hell**: Since Express is built on Node.js, it’s still prone to the famous "callback hell" problem in case developers don’t use promises or async/await to handle asynchronous operations.
4. **Limited Built-In Functionality**: Express does not include all the built-in features some other full-fledged frameworks have (e.g., user authentication, session management). This means developers often have to use additional packages or custom code to achieve certain functionalities.
5. **Performance Issues with Heavy CPU Operations**: Node.js and Express, being single-threaded, can encounter performance bottlenecks when dealing with CPU-heavy operations, as it may block the event loop.

**Industrial Use of Express.js**

1. **Building RESTful APIs**: Express.js is extensively used for building REST APIs, which are lightweight and highly scalable. These APIs can serve data to other services, mobile apps, and web clients. Its ability to handle JSON data natively makes it perfect for this purpose.
2. **Single Page Applications (SPAs)**: Express is often used as the backend of SPAs, where the frontend is a JavaScript framework (e.g., React, Angular, Vue.js) and the backend is built with Express. Express serves API endpoints that provide data to the frontend.
3. **Real-Time Web Applications**: With its non-blocking architecture, Express is an excellent choice for real-time web applications, such as chat applications, live notifications, and collaborative tools, where many users interact simultaneously.
4. **Server-Side Rendering (SSR)**: Express can be used to render dynamic content on the server before sending it to the client. This approach improves SEO and provides a faster time-to-first-byte compared to client-side rendering. It works well with template engines like EJS, Pug, and Handlebars.
5. **Microservices**: Express.js is a lightweight framework, making it a good choice for building microservices. Each service can handle a specific functionality, and Express provides flexibility to deploy them independently.
6. **MERN/MEAN Stack**: Express is an essential part of the **MERN** and **MEAN** stacks (MongoDB, Express, React/Angular, Node.js), making it widely used in full-stack JavaScript applications for both web and mobile platforms.
7. **E-commerce Platforms**: Express is used in the backend of e-commerce websites, where it handles things like product listings, order processing, and user management.
8. **Content Management Systems (CMS)**: Some content management systems, such as those built for blogs or enterprise systems, leverage Express for handling HTTP requests, serving static content, and interacting with databases.
9. **Server-Side Applications**: Express is also used for building server-side applications that need to interact with databases, send emails, process background jobs, etc.

**Conclusion**

Express.js is a versatile, fast, and flexible web framework for Node.js that simplifies the process of building web applications and APIs. Its minimalistic design gives developers the freedom to structure their apps as they see fit, while also allowing for easy scalability and integration with databases and third-party services.

Express's popularity in the industry can be attributed to its simplicity, performance, and the wide support it enjoys from the open-source community. While it may not provide all the features out of the box (like some other full-fledged frameworks), it offers developers the freedom to customize and extend the framework with third-party packages.

Overall, Express.js is an essential tool for building modern web applications, especially for projects that require high performance, scalability, and flexibility. It has widespread industrial use, from building APIs to powering real-time applications and microservices.