Several web frameworks exist that are similar to Express, each with its own set of features, strengths, and weaknesses. Here are a few popular ones:

1. **Koa:**
   * **Pros:**
     + Developed by the same team as Express, but with a focus on a more modern and streamlined design.
     + Uses async/await natively, making it more readable and expressive for handling asynchronous operations.
     + Smaller codebase compared to Express.
   * **Cons:**
     + Smaller community and fewer third-party middleware compared to Express.
2. **Hapi:**
   * **Pros:**
     + Designed for building large-scale applications with a configuration-driven approach.
     + Emphasis on configuration over code, making it easy to manage and extend.
     + Robust plugin system.
   * **Cons:**
     + Can be seen as more opinionated, which may not suit all developers' preferences.
3. **Fastify:**
   * **Pros:**
     + Built with a focus on speed and low overhead.
     + Designed to be highly modular and extensible.
     + Has automatic schema-based validation.
   * **Cons:**
     + Smaller community compared to more established frameworks like Express.
4. **Sails.js:**
   * **Pros:**
     + An MVC framework suitable for building enterprise-level, data-driven applications.
     + Built on top of Express and provides a set of conventions for building APIs and real-time features.
   * **Cons:**
     + Can be seen as heavyweight for smaller projects.
5. **NestJS:**
   * **Pros:**
     + Built with TypeScript and follows an Angular-inspired structure.
     + Strong dependency injection system.
     + Good for building scalable and maintainable server-side applications.
   * **Cons:**
     + Learning curve, especially for developers not familiar with Angular concepts.
6. **AdonisJS:**
   * **Pros:**
     + A full-featured MVC framework with a focus on developer ergonomics.
     + Comes with built-in authentication, ORM, and other tools.
   * **Cons:**
     + Smaller community compared to more established frameworks.

**Pros and Cons of Express:**

* **Pros:**
  + Large and active community with extensive third-party middleware.
  + Straightforward and minimalistic, making it easy to learn and get started.
  + Flexibility to structure applications in various ways.
* **Cons:**
  + Lacks certain features out of the box (e.g., built-in support for validation, dependency injection).
  + Asynchronous code can become nested and harder to manage in larger applications.
  + Not opinionated, which can lead to different project structures and practices among developers.

Ultimately, the choice of a framework depends on your project requirements, team preferences, and the specific features and trade-offs you are willing to accept. Each framework has its strengths, and the best choice often comes down to the specific needs of your application and your familiarity with the framework.