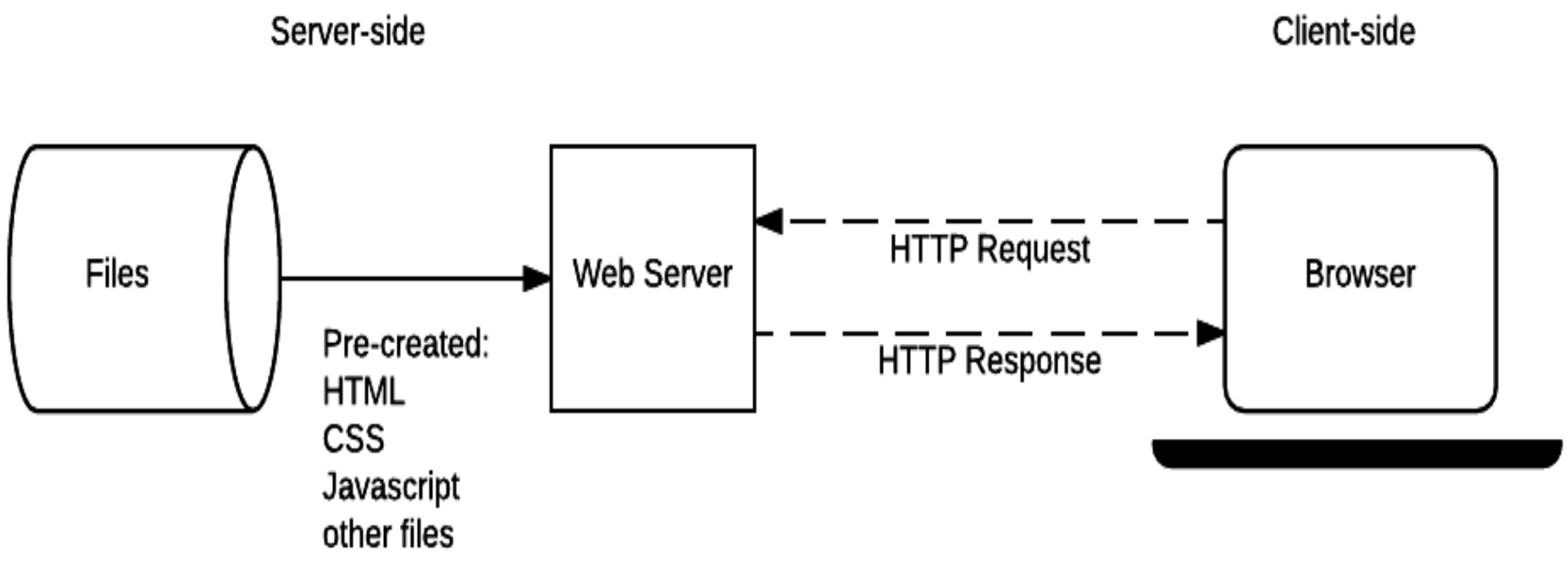
Back End Scripting

What is Back End Scripting?

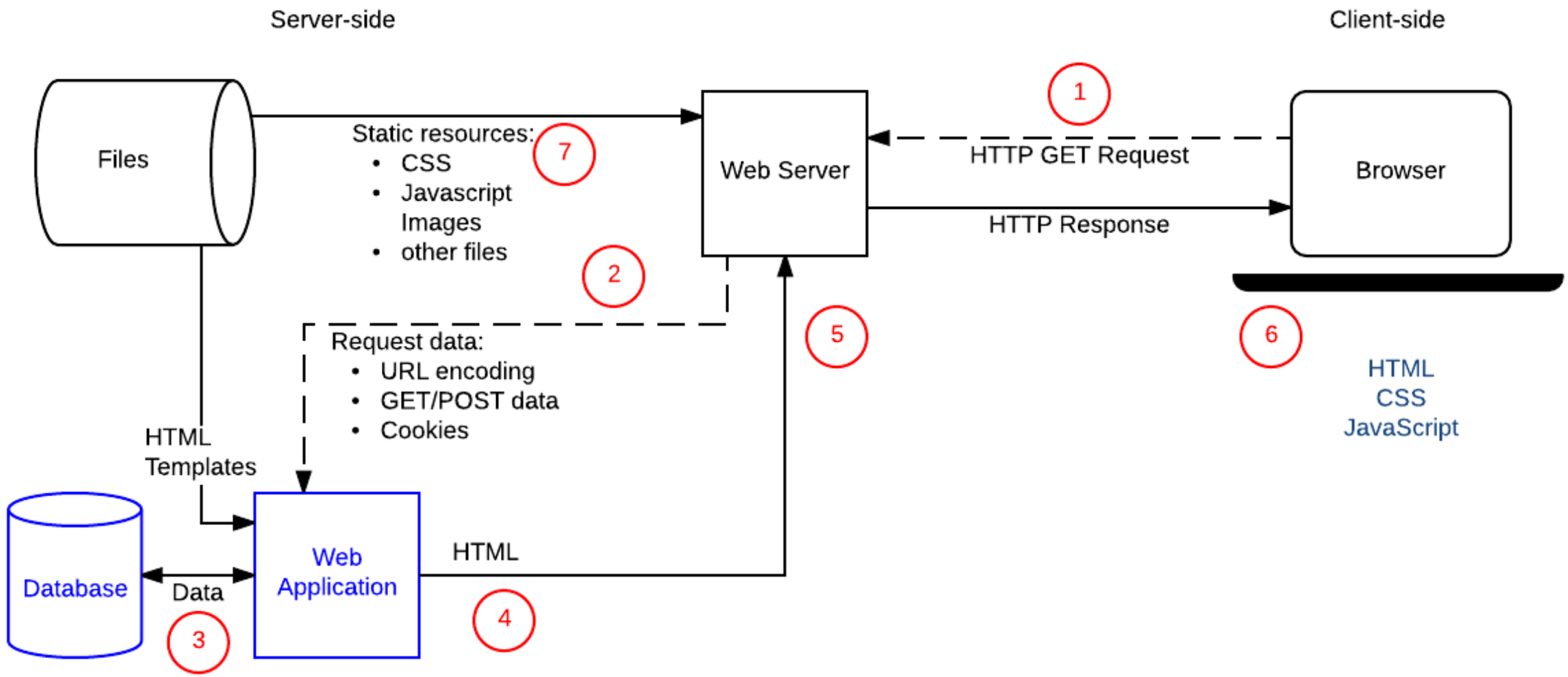
- Most large-scale websites use back end code to dynamically display different data when needed, generally pulled out of a database stored on a server and sent to the front end (client) to be displayed via some code (e.g. HTML and JavaScript).

Static Site

- When a user wants to navigate to a page, the browser sends an HTTP "GET" request specifying its URL.
- The server retrieves the requested document from its file system and returns an HTTP response containing the document (along with an HTTP status code).



Dynamic Sites



Dynamic Sites

1. The web browser creates an HTTP GET request to the server using the base URL for the resource and encodes any data. A GET request is used because the request is only fetching data (not modifying data).

2. The *Web Server* detects that the request is "dynamic" and forwards it to the *Web Application* for processing.

3. The *Web Application* identifies the *intention* of the request. The *Web Application* then gets the required information from the database.

Dynamic Sites

4. The *Web Application* dynamically creates an HTML page by putting the data (from the *Database*) into placeholders inside an HTML template.

5. The *Web Application* returns the generated HTML to the web browser (via the *Web Server*), along with an HTTP status code of 200 ("success"). If anything prevents the HTML from being returned then the *Web Application* will return another code — for example "404" to indicate that the team does not exist.

6. The Web Browser will then start to process the returned HTML, sending separate requests to get any other CSS or JavaScript files that it references (see step 7).

7. The Web Server loads static files from the file system and returns them to the browser directly.

Back End Scripting

- Back End code can be written in any number of programming languages — examples of popular server-side web languages include PHP, Python, Ruby, C# and JavaScript. The back end code has full access to the server operating system and the developer can choose what programming language (and specific version) they wish to use.

- Developers typically write their code using **web frameworks**. Web frameworks are collections of functions, objects, rules and other code constructs designed to solve common problems, speed up development, and simplify the different types of tasks faced in a particular domain.

Back End Scripting

- Again, while both front and back end code use frameworks, the domains are very different, and hence so are the frameworks. Front end web frameworks simplify layout and presentation tasks while back end web frameworks provide a lot of “common” web server functionality that you might otherwise have to implement yourself (e.g. support for sessions, support for users and authentication, easy database access, templating libraries, etc.).

Back End Web Frameworks

- Back end web frameworks (web application frameworks) are software frameworks that make it easier to write, maintain and scale web applications. They provide tools and libraries that simplify common web development tasks, including:
  - Work directly with HTTP requests and responses,
  - Route requests to the appropriate handler,
  - Make it easy to access data in the request,
  - Abstract and simplify database access, and
  - Rendering data.

Express

- **Express** is a fast, unopinionated (trusts the developer to make the right decisions and puts more control in their hands), flexible and minimalist web framework for Node.js It provides a robust set of features for web and mobile applications and delivers useful HTTP utility methods and middleware.

- Express is extremely popular, partially because it eases the migration of front end JavaScript web programmers into back end development, and partially because it is resource-efficient (the underlying node environment uses lightweight multitasking within a thread rather than spawning separate processes for every new web request).

Node.js

- **Node.js** is an open source development platform for executing JavaScript code on the back end (server-side). Node.js is useful for developing applications that require a persistent connection from the browser to the server and is often used for real-time applications such as chat, news feeds and web push notifications.

- Node.js is a platform built on Google Chrome's JavaScript Engine (V8 Engine) for easily building fast and scalable network applications.

Node.js

- Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

- Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.