PER SCHOLAS

ALAB 318.2.1:

Building an Express Application

Version 1.0, 07/07/23
Click here to open in a separate window.

Introduction

Now that you have foundational knowledge of Express tools and techniques, it is time to construct an application according to given requirements.

You are encouraged to use material developed during the lesson as a springboard for this application, but you should make significant contributions of your own. This lab is intended to give you time to practice what you learned, while also researching and implementing new concepts.

Objectives

- Create an Express application.
- Implement an Express template engine.
- Implement routes with route parameters.
- Render template views.
- Implement Express middleware.
- Serve static files.
- Send files for download.

Submission

Submit your completed lab using the **Start Assignment** button on the assignment page in Canvas.

Your submission should include:

- A link to the GitHub repository for your project.

Instructions

Throughout the assignment, commit frequently to your Git repository. Remember, establishing good Git practices is essential to becoming a successful developer; you should take the opportunity to use Git effectively whenever possible during your learning.

The steps outlined in each part describe functionality. Appearance is secondary in this assignment; do not worry about CSS or design unless you have adequate time to do so.

Part 1: Routes, Templates, and Views

Create a view engine that allows you to customize at least three portions of your views. Alternatively, research and use one of the following view engines:

- Pug
- Mustache
- EJS

Note that using a fully-featured view engine will allow you much more flexibility in creating views, and may reduce overall development time, but the research period may take more time upfront. Time management is another important skill in development, so perform a brief amount of research before deciding which path you would like to take.

Create at least two different view templates for your chosen view engine.

Within those views, include some type of navigation that allows you to swap between views. Since the default HTTP request method is "GET," this can be as simple as an anchor link that navigates to the other view's relative URL.

Include a basic form in at least one view that sends a POST request to a route within your application. For now, just log that data to the console within your route and send a simple "success" response. We will explore APIs in greater detail in a future lesson.

Within your routes, include at least one instance of a route parameter that modifies what is rendered to the client.

Part 2: Middleware

Within your application, create a piece of middleware. or for some practical purpose, use third-party middleware. This can be as simple as logging specific request data; however, it will benefit you to find a more interesting use case for practice purposes or to research third-party middleware.

Part 3: Exploring Response Options

Within one of your views, include an image that is stored locally in your project's file system. In order for this image to be served, you will need to use the Express static middleware, as shown in the lesson.

Build a "download" button that makes a request to your Express server on click. There are several ways to do this.

The response to this particular button should include the **res.download()** method, and allow the user to download the image you included within the view.

Research the response object's download method to see how it works, and implement the above in whatever way you think is appropriate. Being able to research well and sift through documentation is one of the most important skills for developers to have.

Remember, functionality takes precedence over appearance for this assignment.

Part 4: Open Exploration

As long as the functionality described in the parts above has been implemented, take any remaining time to explore other options available to you. Collaborate with your peers to brainstorm ideas, ask questions of your instructors, and see what is possible. Be creative!

This assignment is not graded, but the application should run properly before being submitted. After you are finished with your exploration period, comment out anything that is unfinished or prevents the application from running.

Part 5: Completion

Upload your project to a GitHub repository, and submit it according to the submission instructions at the beginning of this document.