PER SCHOLAS

SBA 318:

Express Server Application

Version 1.0, 07/25/23

Click here to open in a separate window.

Introduction

This assessment measures your understanding of Node and Express and your capability to implement their features in a practical manner. You have creative freedom in the topic, material, and purpose of the web application you will be developing, so have fun with it! However, remember to plan the scope of your project to the timeline you have been given.

This assessment has a total duration of **three (3) days**. This is a **take-home assessment**.

You have **three total days** (including weekends and holidays) to work on this assessment. This assessment will be due at **5:00pm** on the third day after it is assigned. Your instructor will provide you with at least four hours of class time to work on the assessment; during which time, you may discuss details of the project with your instructor, including the topic, scope, and implementation.

Objectives

- Create a server application with Node and Express.
- Create a RESTful API using Express.
- Create Express middleware.
- Use Express middleware.
- Use a template engine to render views with Express.
- Interact with a self-made API through HTML forms.

Submission

Submit the link to your completed assessment 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

You will create a small Node and Express server application. The topic and content of this application is entirely up to you; be creative!

Your work will be graded according to the technical requirements listed in the following section. Creativity and effort always work in your favor, so feel free to go beyond the scope of the listed requirements if you have the time.

Keep things simple. Like most projects you will encounter, you should finish the absolute minimum requirements *first*, and then add additional features and complexity if you have the time to do so. This will also help you understand what you can get done in a specific allotment of time if you were to be asked to do something similar in the future.

Once you have an idea in mind, briefly discuss it with your instructors to determine if it is appropriate for the amount of time you have been given.

Since topic and content are secondary to functionality for this assessment, we have included some resources below for free content that you can use to populate your application. Once you have gotten your functionality in place, you can return and fill in the content with something interesting.

Resources for free content:

- **Text:** Lipsum, a Lorem Ipsum text generator.
- **Images:** Pexels, a resource for stock photos (and other media).
- **GIFs:** Motion Elements, a resource for GIFs (and other media).

Requirements

The requirements listed here are **absolute minimums**. Ensure that your application meets these requirements before attempting to further expand your features.

Create your application locally, and initialize a local git repo. Make frequent commits to the repo. When your application is complete, **push your repo to GitHub and submit the link to the GitHub page** using the submission instructions at the top of this document.

Requirement	Weight
Create and use at least two pieces of custom middleware.	**5%**
Create and use error-handling middleware.	**5%**
Use at least three different data categories (e.g., users, posts, or comments).	**5%**
Utilize reasonable data structuring practices.	**5%**
Create GET routes for all data that should be exposed to the client.	**5%**
Create POST routes for data, as appropriate. At least one data category should allow for client creation via a POST request.	**5%**
Create PATCH or PUT routes for data, as appropriate. At least one data category should allow for client manipulation via a PATCH or PUT request.	**5%**
Create DELETE routes for data, as appropriate. At least one data category should allow for client deletion via a DELETE request.	**5%**
Include query parameters for data filtering, where appropriate. At least one data category should allow for additional filtering through the use of query parameters. **Note:** DO NOT use API keys; this makes it more difficult for instructors to grade finished projects efficiently.	**5%**
Utilize route parameters, where appropriate.	**5%**
Adhere to the guiding principles of REST.	**10%**
Create and render at least one view using a view template and template engine. This can be a custom template engine or a third-party engine. If you are stuck on how to approach this, think about ways you could render the current state of your API's data for easy viewing.	**8%**
Use simple CSS to style the rendered views. **Note**: This is not a test of design; it is a test of serving static files using Express. The CSS can be *very simple*.	**2%**
Include a form within a rendered view that allows for interaction with your RESTful API.	**3%**

Utilize reasonable code organization practices.	**5%**
Ensure that the program runs without errors (comment out things that do not work, and explain your blockers - you can still receive partial credit).	**10%**
Commit frequently to the git repository.	**5%**
Include a README file that contains a description of your application.	**2%**
Level of effort displayed in creativity, presentation, and user experience.	**5%**

Bonus Objectives

The objectives listed here are **not required**. Ensure that your application meets the requirements above before attempting to further expand your features.

These bonus objectives *cannot* increase your overall score above 100%. Successful completion of these objectives can, however, make up for lost points above. **Ensure your application works as outlined by the requirements above before attempting these objectives, time permitting.**

Include a practical usage of regular expressions within route paths. Note: A forced, arbitrary usage of this technique will not earn you any bonus credit. This must be **practical** and **sensible** in order to be considered for credit.	**+1%**	
Research and effectively use at least one third-party Node package for practical, sensible purposes. This cannot be a package that has been used in examples and lesson materials thus far. Step outside the box and be creative!	**+1%**	

Reflection (Optional)

Once you have completed your project, answer the following questions to help solidify your understanding of the process and its outcomes, as well as improve your ability to handle similar tasks in the future.

- *What could you have done differently during the planning stages of your project to make the execution easier?*

- *Were there any requirements that were difficult to implement? What do you think would make them easier to implement in future projects?*

- *What would you add to or change about your application if given more time?*

- *Use this space to make notes for your future self about anything that you think is important to remember about this process or that may aid you when attempting something similar.*