UDACITY

‹ Return to Classroom

Image Processing API

REVIEW

CODE REVIEW 9

HISTORY

Requires Changes

2 specifications require changes

Note: Please do not include node_modules in your submission or upload it to github. All the packages can be installed by running `npm install` itself.

Overall great effort in project implementation.

You need to make some changes to complete this project.

Don't worry, I have included feedback in code review as well as project review to help you with the changes.
I hope you are able to make the requisite changes.
If you face any issues in implementation, feel free to scout knowledge hub or drop a question there.

Highlighted changes and suggestions include:

- Arranging dependencies correctly.
- Configuring prettier script to run on TS files.

Feel free to include suggested changes as well.

Check out this link to know why we use ExpressJS for backend development.

Why use ExpressJS

Check out this link to understand how debugging works in VS code.
Debugging a Node JS application using VS Code -

I will look forward to your resubmission.
Good luck.
Have a good day :)

Do not forget to drop a feedback for this review.

Stay Udacious
UDACITY

Setup and Architecture

- **Source code is kept separate from compiled code.**
- **All tests should be contained in their own folder.**
- **Separate modules are created for any processing.**

✅ Source code is kept separate from compiled code.

- Source code should always be kept separate from the compiled code.
  It is a good way to create distinction between the functional code and other resources.

✅ All tests are contained in `tests` folder.

- Keeping tests in a separate folder makes it easy for any developer to access the code intuitively.
  The developer would know where to look exactly and add more test cases if necessary.

✅ Image processing is done in a separate module.

- Separating functionality is important to keep the code modular.
  It helps to debug the code in a much better way when you know where to look exactly for a particular function.

Additional reading:

1. Folder structure
2. Using custom middlewares

- **Package.json should contain both devDependencies, and dependencies.**
- **Scripts should be created for testing, linting/prettier, starting the server, and compiling TS.**
- **Build script should run without error.**

❌ supertest, jasmine and jasmine-spec-reporter must be a part of devDependencies.

- It is very important to keep dependencies separate from devDependencies.
  DevDependencies are used when you only need the packages for development.

  They are not used in testing like jasmine, supertest etc.
  In fact, you would not even need type definition support for any package since the production code would be build code that is written in javascript.

✅ Scripts are present for required functionalities.

- Creating scripts makes it easy for anyone to get to know your project better and work with it.
  As a reviewer, I can easily run and test your project by looking at your script and running the required one for functionality.

✅ Build script runs without error.

- Good job with build script!
  It helps to convert the code into javascript that the browser understands very well.
  Typescript is an awesome code formatter and helps catch compilation errors which will tell you where you are going wrong.

Suggested reading:

1. Dependencies-vs-devdependencies
2. Why typescript

Functionality

- **Start script should run without error**
- **Provided endpoint should open in the browser with status 200**

✅ Start script runs perfectly.

- There is no error when I run start script in the application.

✅ Provided endpoint works perfectly!

- The endpoint returns a response with a status of 200.
  Providing this information not only helps me as a reviewer to check the application but also any other developer working on the application to run the application easily.

Additional reading:
Why README?
Using dirname in node

- **Accessing the provided URL with image information should successfully resize an image and save it to disk on first access, then pull from disk on subsequent access attempts.**
- **An error message should be provided to the user when an image has failed to process or does not exist.**

✅ Accessing provided url resizes images. great job implementing caching as well.

- Image resizing works perfectly.
  The url accepts query parameters for file name, height and width and returns the desired response.
  The resized image is cached and served as a response.

✅ Good job with error handling.

- Error messages for common error messages scenarios are provided.
  1. Missing filename, height or width.
  2. Invalid input for filename e.g. `fjord123`.
  3. Invalid input for height or width e.g. `height=a`, `height=0` or `height=-1`.

Additional reading:
Error handling in application
app.use() vs app.get()

Code Quality

- **Test script runs and all tests created pass.**
- **There is at least 1 test per endpoint and at least one test for image processing.**

✅ Test script runs perfectly without any error in console.

- On running `npm run test`, test specs run as expected.
  All the specs run without fail and no error is produced.
  Well done!

✅ Test specs for image processing and endpoint testing are present.

- By image processing, the rubric is trying to convey that you need to directly import the function and perform testing.
  You have perfectly tested the function and endpoint!

Additional reading:
Jasmine reporter for testing
Testing using supertest

- **All code in the SRC folder should use the .ts filetype.**
- **Functions should include typed parameters and return types and not use the `any` type.**
- **Import and Export used for modules.**
- **Build script should successfully compile TS to JS.**

✅ All files in src have .ts file extension.

- Good job on checking this off the list.
  Typescript helps with checking type errors and compilation errors.

✅ Functions have typed parameters or return type.

- Great job including typed parameters and return types for all functions.

✅ `any` type is not used evidently. Good job including `noImplicitAny` in typescript configuration file.

- Use of `any` type defeats the purpose of using typescript and typed parameters.
  One configuration that would help in ensuring you do not use `any` type implicitly is using `noImplicitAny` in typescript configuration file.

✅ ES syntax is used for import and export of modules.

- Great job using Import and Export to use modules for their functionality in different files.

✅ Build script compiles to JS.

- You build script perfectly compiles to JS without throwing any errors.
  This would eventually be your production level code so it is important to have an error free code.

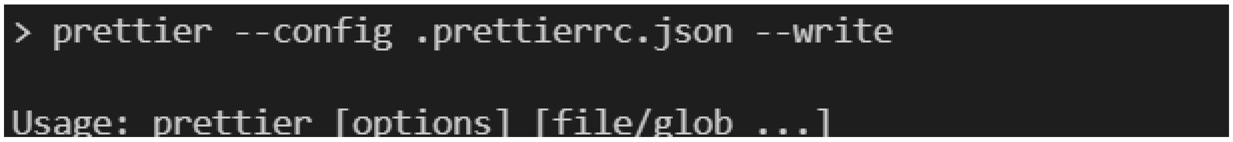
Additional reading:
ES6 syntax
Define return type
Use of async/await

Prettier and Lint scripts should run without producing any error messages.

❌ Lint scripts work perfectly without producing errors. You need to configure prettier script to run on files.

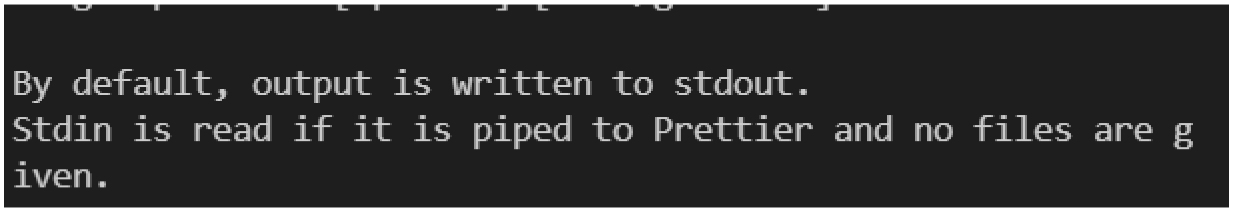
- Prettier and Lint should be configured to run only on TS files before they build to JS.
  You need not run it on JS or JSON files.
  Also, you can define your own configuration and rules for these formatters.
  Please refer the links below.

Additional reading:
Configure eslint and prettier



```
> prettier --config .prettierrc.json --write

Usage: prettier [options] [file/glob ...]
```



```
By default, output is written to stdout.
Stdin is read if it is piped to Prettier and no files are g
iven.
```

RESUBMIT

DOWNLOAD PROJECT

9 CODE REVIEW COMMENTS

Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)

RETURN TO PATH