ID2: Peer Reviews/ Inspections

Peer Review Documentations

Code Inspection for Docker

Reviewers: Juan⇒ Reader, Inspectors, Jozua⇒ Inspectors , Ardalan⇒ Recorder, Moderator

Reviewed: Ralph(Author)

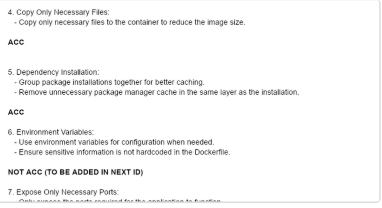
Pull Request Numbers: 78

Date of the Review: 3pm to 4pm Saturday (Feb 3rd)

Document for Inspection:

Check list for Dockerfile:

https://docs.google.com/document/d/1rk9CG1o8yMafcOXMjoVi8ZYTdJUKKABSRmA78vrxuLA/edit?usp=sharing



4. Copy Only Necessary Files:
   - Copy only necessary files to the container to reduce the image size.

ACC

5. Dependency Installation:
   - Group package installations together for better caching.
   - Remove unnecessary package manager cache in the same layer as the installation.

ACC

6. Environment Variables:
   - Use environment variables for configuration when needed.
   - Ensure sensitive information is not hardcoded in the Dockerfile.

NOT ACC (TO BE ADDED IN NEXT ID)

7. Expose Only Necessary Ports:

Zoom Meeting Link: https://usask-ca.zoom.us/j/93006837033?pwd=OVdlNzBIYWNsRHZLZ1NjbTRIS0lBUT09

Checklist for Dockerfile:

1. Base Image:

- The base image is from a reputable source.

- Uses the latest version of the base image for security updates.

ACC

2. Non-root User:

- Use a non-root user to run processes within the container.

- Avoid running processes as the root user for security reasons.

ACC

3. Minimize Layers:

- Minimize the number of layers in the Dockerfile for efficient image build and reduced image size.

N/A

4. Copy Only Necessary Files:

- Copy only necessary files to the container to reduce the image size.

ACC

5. Dependency Installation:

- Group package installations together for better caching.

- Remove unnecessary package manager cache in the same layer as the installation.

ACC

6. Environment Variables:

- Use environment variables for configuration when needed.

- Ensure sensitive information is not hardcoded in the Dockerfile.

NOT ACC (TO BE ADDED IN NEXT ID)

7. Expose Only Necessary Ports:

- Only expose the ports required for the application to function.

- Avoid exposing unnecessary ports for security reasons.

ACC

8. Clean-Up Commands:

- Use `RUN` commands to clean up temporary files and package manager caches.

- Combine cleanup commands to reduce the number of layers.

NOT ACC (TO BE ADDED)

9. Multi-Stage Builds:

Use multi-stage builds to separate build-time and runtime dependencies. This helps to reduce the final image size.

ACC

10. Entrypoint and CMD:

Use ENTRYPOINT in combination with CMD to provide default executable and arguments. This makes the container act like an executable.

ACC

Check list for Container Runtime:

11. Security Context:

- Use Docker's security features like `--security-opt` to enforce security policies.

- Review the `docker run` command for security-related options.

N/A (Yet)

12. Volume Mounts:

- Ensure sensitive data is not being exposed through volume mounts.

- Limit access to only necessary volumes.

ACC

13. Resource Limits:

- Set resource limits (CPU, memory) appropriately using the `--cpus` and `--memory` flags.

N/A(Yet)

14. Health Checks:

- Implement health checks to ensure the container's health is monitored.

NOT ACC (TO BE ADDED)

15. Network Configuration:

- Use Docker’s network features like --network to isolate the container's network. This can enhance security and performance.

ACC

16. Restart Policies:

- Set appropriate restart policies (--restart) to handle container failures.

ACC

Check list for Documentation:

17. README and Comments

- Ensure the Dockerfile includes clear comments explaining complex steps.

- Document any environment variables required for configuration.

ACC

18. Docker Ignore File:

- Document the use of .dockerignore file to exclude files that are not necessary for building the Docker image. This reduces build context size and prevents sensitive information from being included in the image.

ACC

Check list for General Best Practices:

19. Immutable Containers:

- Aim for immutable containers by avoiding changes in runtime.

ACC

20. Docker Compose:

- If using Docker Compose, ensure the configuration is well-structured and follows best practices.

ACC

21. Version Pinning:

- Pin versions for all dependencies and packages to ensure reproducibility.

ACC

22. Logs:

- Configure appropriate logging within the container for debugging and monitoring.

ACC

23. Single Responsibility Principle:

- Each container should have only one responsibility. This makes managing, updating, and troubleshooting easier.

ACC

24. Build Automation:

- Automate the build process using CI/CD pipelines. This ensures consistency and repeatability in the build process.

ACC

Meeting summary:

- **Went through READ ME**
- **Ralph explained and answered Questions about the Docker files and Properties and README**
- **Went through the Docker file & Properties**

Duration of Inspection: 1 hour

Defects Found:

- **In the README file under the Local Environment Set Up we need to have “Docker Initialized” before everything else**
- **No user interface for the backend (we might need to add one)**
- **Add explanation regard the cache in the README under the Container section**
- **Health Checks needs to be added**
- **Resource limits needs to be added**
- **Need to implement more functionalities for Security**
- **Need to add clean up commands**
- **Need to add Environmental Variables**

FileDropzone.tsx Review

Reviewers:

Cameron ⇒ Reader, Inspector,

Dan ⇒ Inspector ,

Eric ⇒ Recorder, Moderator

Reviewed: Marcus

Pull Request Numbers: #75

Date of the Review: 4pm to 5pm Friday (Feb 9th)

Document for Inspection: File upload page

https://docs.google.com/spreadsheets/d/1sOCsN_sHcQTDE3bypqXSnzTvue46FGIcMpcuq2I37nM/edit?usp=sharing

FileDropzoneControlProps

- make its own page if this is a common pattern we will use in future features

Coding Styles

- Use JSDoc for comments, they are similar to Javadocs

Use of assertions

- no use of assertions yet,
- possibly: assertions could be put at the ‘onDrop’, should be a check and assert the check worked
- possibly: Logging could be put at the ‘onDrop’

FileUploadPage

- no default selection for radio selections(FitBit, Apple)
- should we have a pop-up to select one of the 2 button once file selected?
- should one be defaulted?
- should it auto-select once a file is selected?

Testing

Coverage

- no functions to unit test
- playwright tests UI
- no further tests needed

Playwright - integration test

Jest - component unit test

Are tests documented??

- Yarn test, its all documented
- nothing more till we have features

Testing Dropzone

- mock data tests

Does the Dropzone auto change from xml/JSon when dropped?

Will it change if the wrong file is selected?

if user selects file, will system tell user that its the wrong file type?

- there is some file preprocessing before the file is dropped?
- what happens when a txt (file extensions other than xml/json) file is dropped?
- what if correct file extension but data is not formatted correctly?

- rejected files vs accepted files

Erics concerns:

- uploads size should be in MB not bytes
- upload buttons concerns
    - can you upload a mix of Fitbit/apple files
    - auto uploads once files are selected
    - cache should save uploaded files
    - select apple/Fitbit shows which files are uploaded in cache
    - whole dropzone is a button

PDF of Checklist:

https://prod-files-secure.s3.us-west-2.amazonaws.com/f946e05a-05ae-401d-a12f-3bc87066389d/c2280a79-ba86-4bb8-a402-025e6dd1cbc6/Team3_FileUploadPage_CodeReviewChecklist_-_Sheet1.pdf