**Challenge 3: Creating a Full Stack Application with Docker Compose**

1. Using the provided link, download and install Docker Desktop for your operating system.

<https://www.docker.com/products/docker-desktop/>

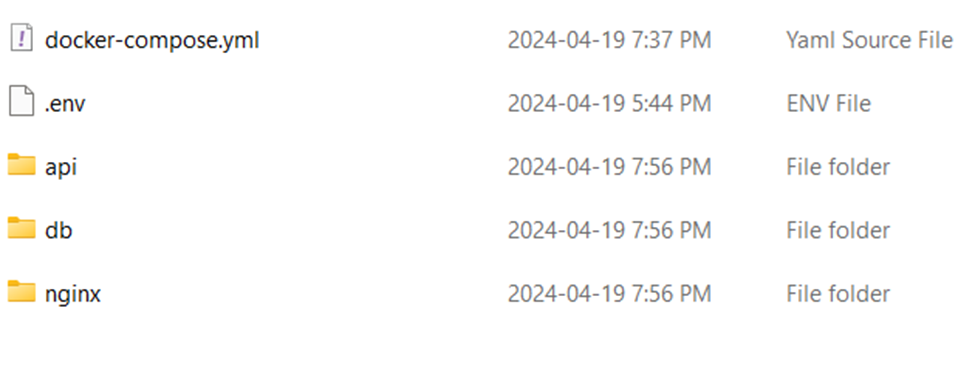
2. Follow the directions in the Docker installer to complete the installation using the default options.

3. Run docker -version in a command prompt to verify the installation.

4. Create a folder called "challenge 3" and three subfolders inside it: api, database, and nginx. These folders will contain the code and configuration files for each server.

5. In the "challenge 3" folder, build a docker-compose.yml and a.env file to store environment variables.

6. The folder structure should look like the image provided below:

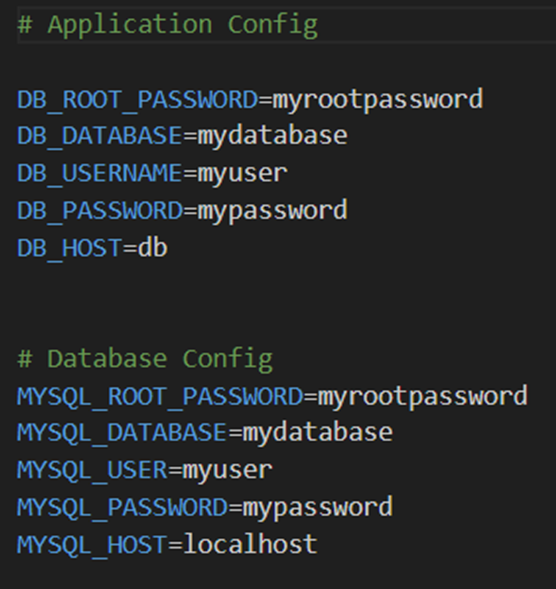


7. The "api" folder contains server.js, package.json, and a Dockerfile for the API server.

8. The "db" folder contains init.sql and a Dockerfile for the database server.

9. The "nginx" folder contains nginx.conf and a Dockerfile for the Nginx web server.

10. The.env file contains environment variables for both the application and database servers.



11. The docker-compose.yml file defines services for the API, database, and Nginx servers, allowing them to interact.

A screen shot of a computer

Description automatically generated

12. To start the services, run docker-compose up -d at the command prompt under the "challenge 3" directory.

13. Use docker-compose ps to ensure that all services are operational.

In your web browser, navigate to http://localhost:8080/api/books.

14. Go to http://localhost:8080/api/books/{id} to access individual book information.

15. Run docker-compose down to terminate and delete the services.

**Challenge 4: Scaling Up a Docker Service.**

1. Transfer all files and folders from the "challenge 3" directory to a new directory called "challenge 4."

2. Launch the services with Docker Compose instructions, then visit to http://localhost:8080/api/stats.

3. Take note of the constant hostname value, which indicates that the API server is only running one instance.

A screen shot of a computer

Description automatically generated

4. Modify the docker-compose.yml file to deploy three instances of the API server.

5. Include a "deploy" field with three replicas for the API server.

6. Remove the container's port mapping, as Docker manages many instances.

7. Launch docker-compose up -d from the "challenge 4" directory.

8. Use docker-compose ps to confirm that all services are running.

9. To view different hostname data, use a web browser and navigate to http://localhost:8080/api/stats.

10. After each refresh, one of the three hostname values is shown, demonstrating that scaling was effective.

11. Use docker-compose down to terminate all services and release ports.