EAST TENNESSEE STATE
UNIVERSITY

CSCI 5400 – Software Production

Project Deliverable 2: Creating a Microservices Application

Deliverable Information

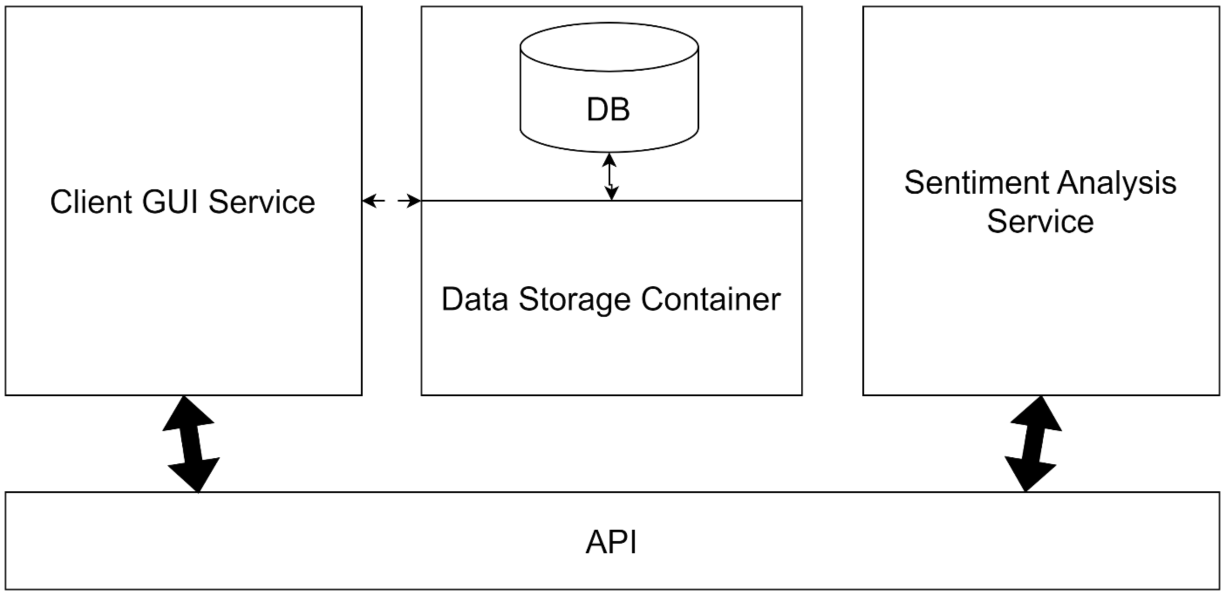
Date Due:	Monday, March 20, 2023 by 11:59pm
Terms:	**Group Submission**
Submission Details:	• Submit the following to the D2L Project Deliverable 2 Drop Box Folder: ○ Installation Instructions ○ Installation Materials (everything needed, except GitHub) ○ The Video Demonstration • Only one person on the team should submit the project.

Objectives

- Create a multi-container microservices application in Docker.
- Utilize containers and container images for development and deployment.

Application Description

Create an application that accepts a sentence or paragraph, estimates the sentiment, and displays the sentiment to the user. The system's architecture will include microservices housed within Docker containers, as follows.



Client GUI Service

The Client GUI Service is simply a web service that controls interaction with the software by a user. The service will do the following:

1. Allow a user to type a sentence into a text area.
2. Allow the user to press a button that says, "Classify Text."
3. Display the sentiment result to the user.
    a. The same action will auto-save this result to a database (the text and returned sentiment and score).
    b. Display a list of automatically-saved sentiments for the user to review.
4. Allow the user to delete text and their associated sentiments from the database.

Because this service requires a database and a sentiment analyzer to operate, it must communicate with the Data Storage Container and the Sentiment Analysis Service. The Client GUI Service may communicate with the Data Storage Container through *either* an API or directly by setting up a Docker Container Volume. The Client GUI Service ***must*** communicate with the Sentiment Analysis Service via an API.

Data Storage Container

The Data Storage Container holds the database of sentiment searches. The database will be simple, currently consisting of a single table. This table will contain an ID, a timestamp, the text that was searched, whether the sentiment is positive, negative, or neutral, and the percentage score.

Sentiment Analysis Service

The Sentiment Analysis Service should be similar to a Function-as-a-Service, although it will not die after responding to a request. Use a GET call through a REST API to pass a string of text to the service. The service should respond with whether the sentiment is positive, negative, or neutral, and provide a percentage score.

Specific Requirements

1. You must create a **Development Environment** for this project. The Development Environment will include: Containers for the software and a container for the database, and mechanisms to interact with an IDE (if used) and the version control system.
2. The Sentiment Analysis Service and the Client GUI Service **cannot be written using the same technology stack**. For example, if you use C# .Net on the Client GUI Service, you must not use .Net on the Sentiment Analysis Service. I recommend using Python for Sentiment Analysis, since Python's nltk (natural language toolkit) is available and fairly simple to use. If using Python, perhaps you would develop the API using FastAPI. Your database can be *any* relational database technology you wish, except SQLite.
    a. The code should be complete enough to run a simple application at the end of this project.
3. API calls must be done using REST.
4. You must include a GitHub repository for each service: the Client GUI Service, the Sentiment Analysis Service, and the Database creation.
5. You must provide written installation instructions to install your **development environment**, similar to those used in Project 0 and Project 1.

    a. Once installed, I must be able to run your development environment on my machine. I will install using your instructions, which means the testing of the environment and the instructions for building that environment should be complete and precise.
6. You must provide a video demonstration of the development environment in operation. This video should be **no longer than eight minutes long**, but should demonstrate the main points listed in the previous section. You should also show the Docker Containers running in Docker Desktop.

Deliverables

1. Submit complete instructions for installing your development environment.
2. Include all files (or git locations) necessary to install your system to my local computer.
3. Submit a video demonstrating the development environment in operation.
4. **Each individual** should complete a brief (half-page) narrative that describes their contributions to the team. This is an accountability step because the entire team will have access to read these. It should be uploaded **by each person** to the Project 2 Drop Box.
    a. NOTE: If you would like to refute someone's tasks (i.e., you believe they did not do what they claimed), you should contact me directly via email.

A grading rubric will be provided and is attached to the D2L dropbox.

While the grading rubric is intended to be a team grade, I reserve the right to adjust individual grades on this assignment downward based on feedback from team members or low contributions.