Overview of the System:

The given code creates a simple Flask web application that generates random dice rolls ranging from 1 to 10. It uses OpenTelemetry to trace and measure the performance of the application. By importing the OpenTelemetry “tracing” and “metrics” library, as well as the Flask web framework and Python’s built-in “random” library, a “roll counter” object is created as a counter metric that tracks the number of times each possible value is rolled. The Flask web application is created with a single route called “`/rolldice`”, is accessed calls the “`roll_dice`” function which returns a result from the random dice roll (between 1 to 10). This is reflected on the page when the user enters the prompted web address. The “`do_roll`” function creates a new span in the OpenTelemetry trace using the “`tracer.start_as_current_span`” function. It sets an attribute on this span called “`roll.value`” with the result of the dice roll.

Description of Instrumentation:

To elaborate on the instrumentation details, the program is instrumented by creating a “tracer” and “meter” object using the OpenTelemetry API, as well as a metric for tracking the number of times each possible value is rolled. Instrumentation data is produced and outputted to a text file containing both telemetry data on traces and metrics. When the user navigates to http://127.0.0.1:5000/rolldice, the “`role_die`” function is called which in turns generate the random number. During this time, “`do_roll`” is executed, and tracing is used to create a new span for each roll (such as when someone visits the link for the first time, or refreshes the page), and an attribute “`roll.value`” is added to the span to capture the result. In the GitHub link that is provided in the initial PDF submission, includes separated text files for both traces and metrics data for when I had performed a test run of the program.

Description of Telemetry Data Visualization:

Unfortunately, I found that there was difficult implementing a backend service to visualize the telemetry data. Telemetry data was found when running the program, however implementing the OpenTelemetry Collector was where I became stuck. The given text file(s) in the GitHub README.md should include the found telemetry information.