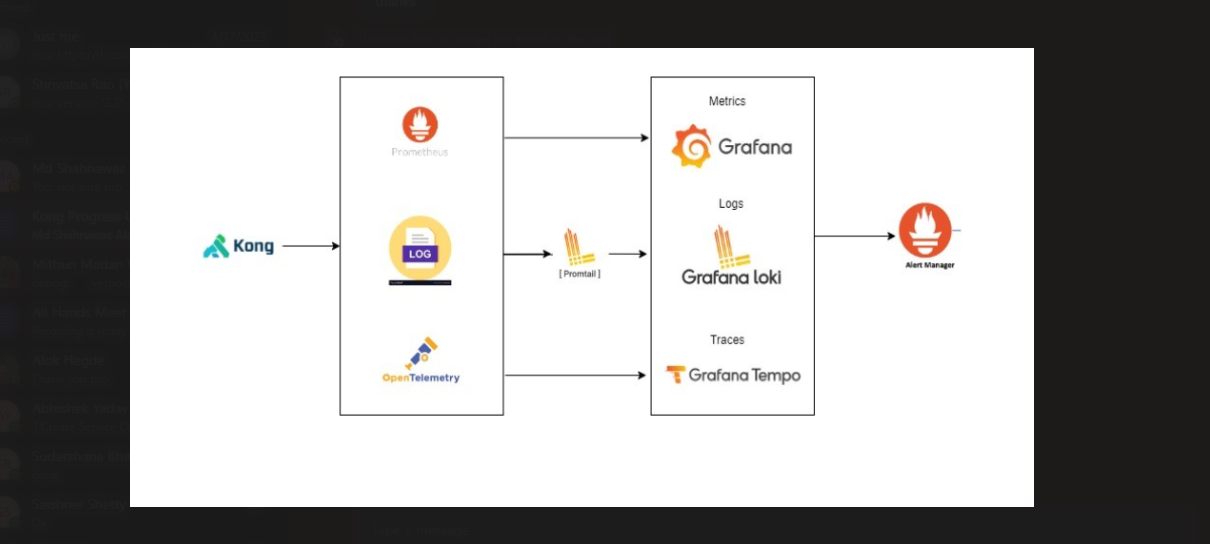
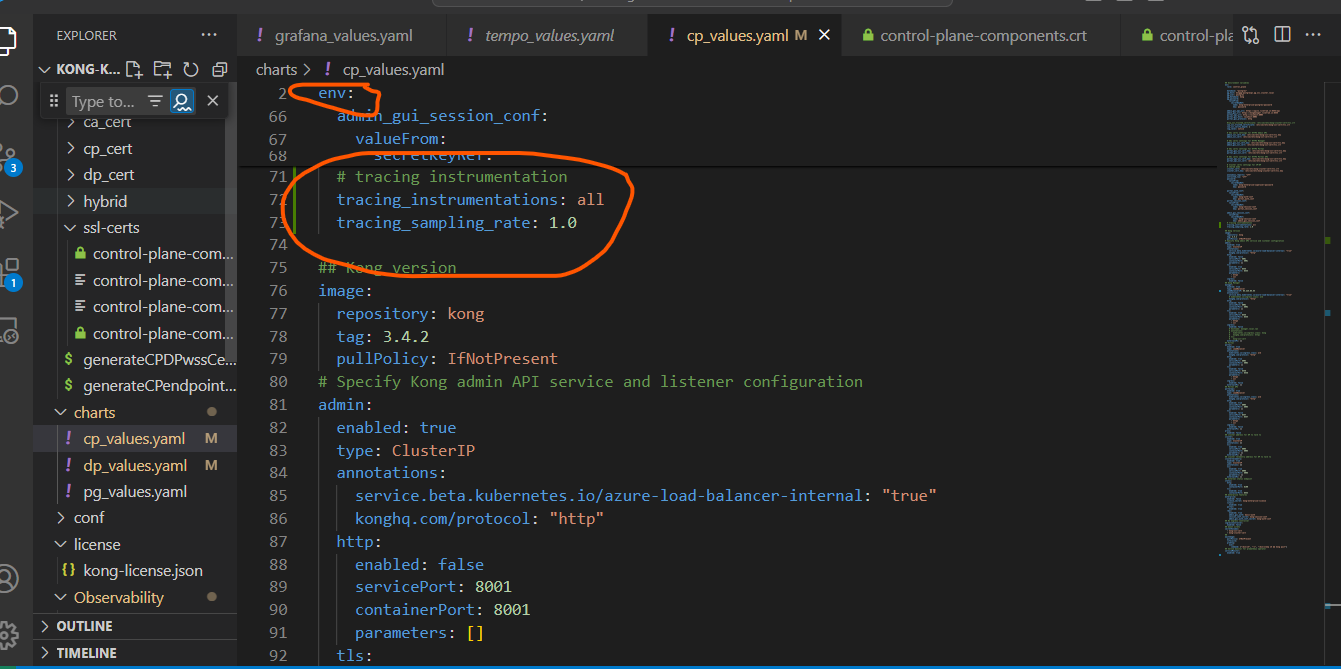
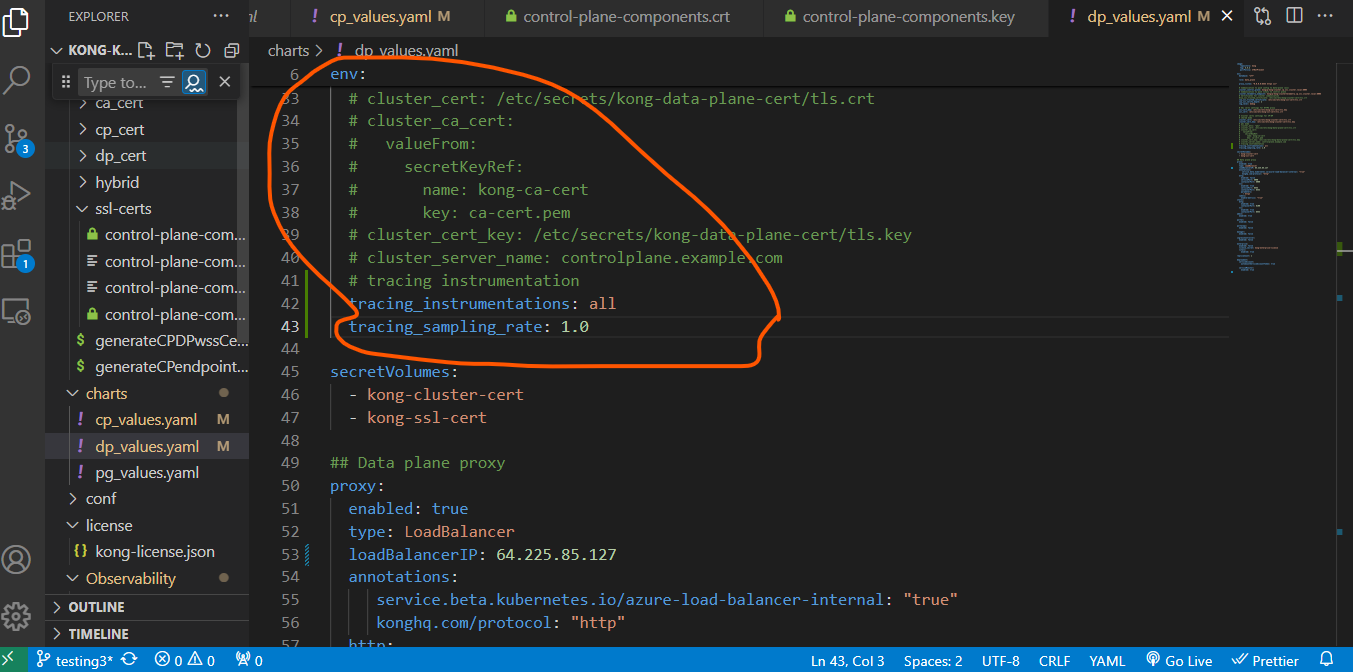
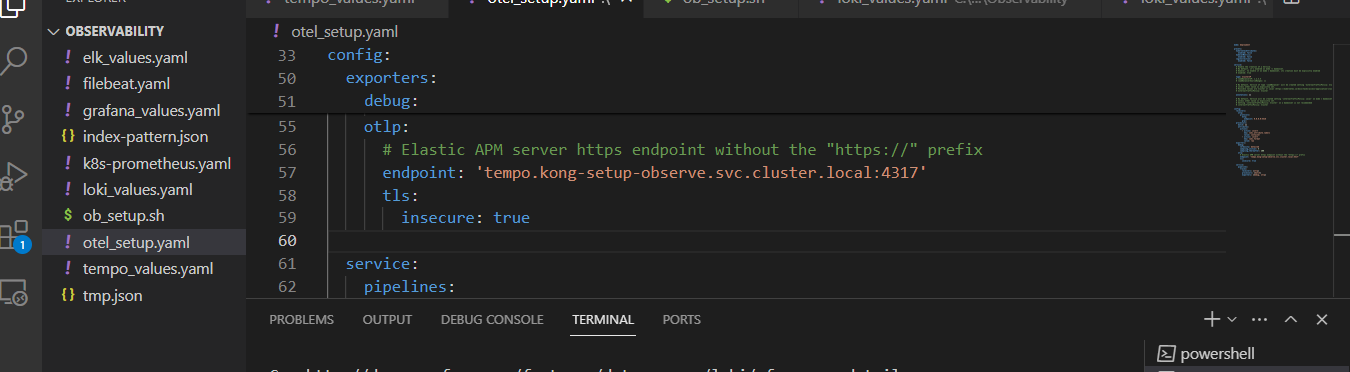
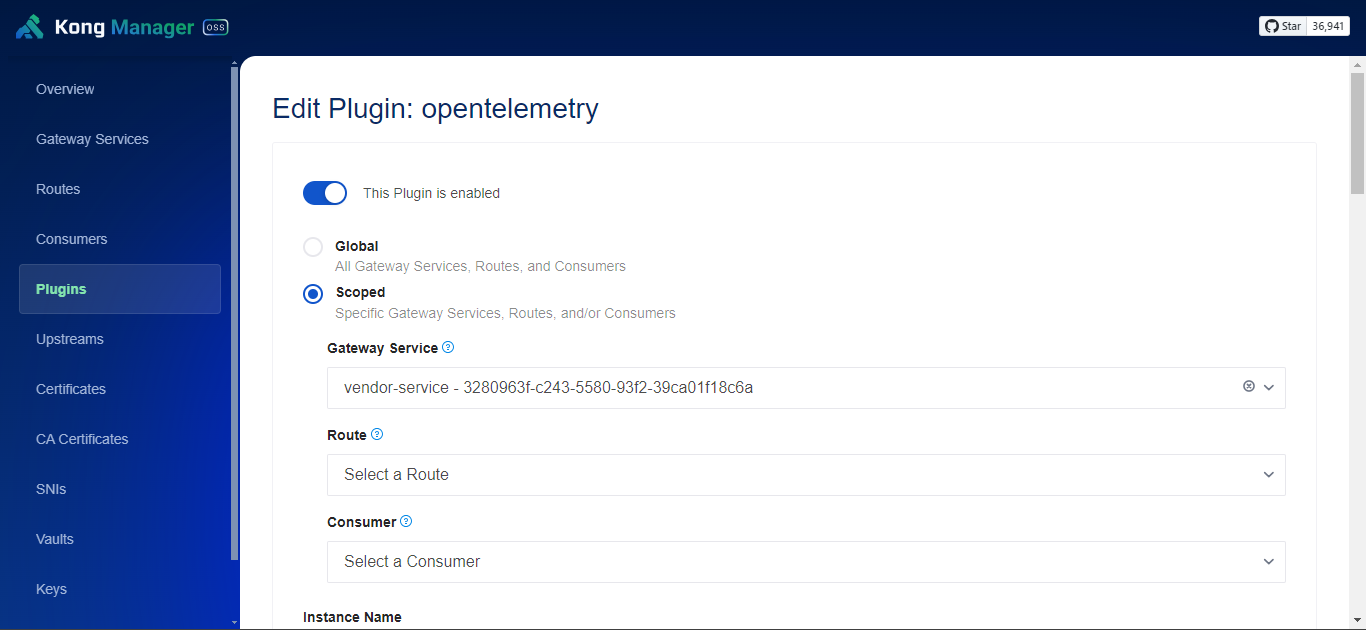
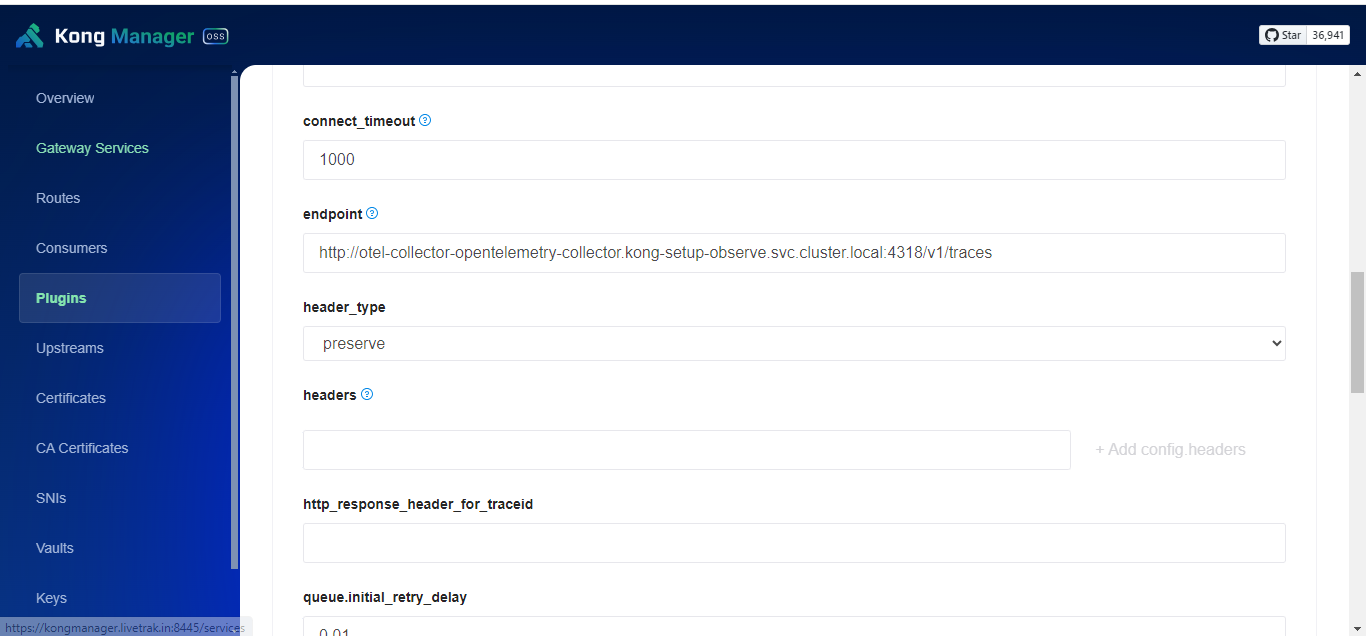
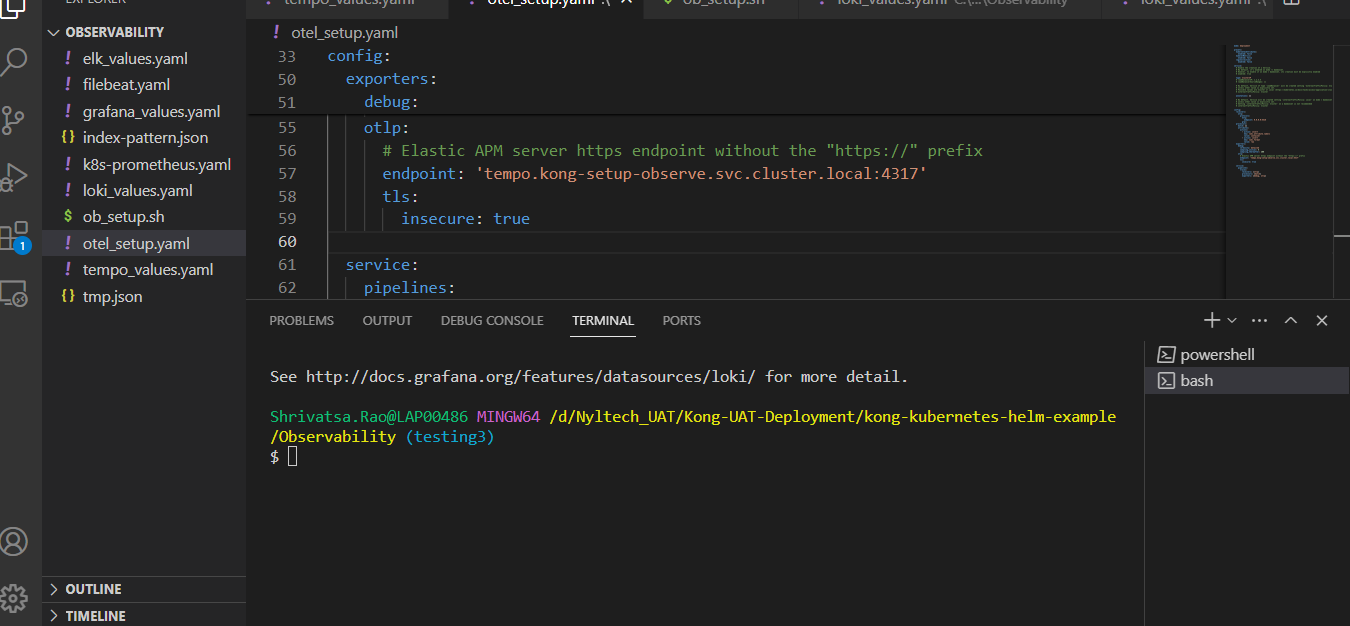
Tempo

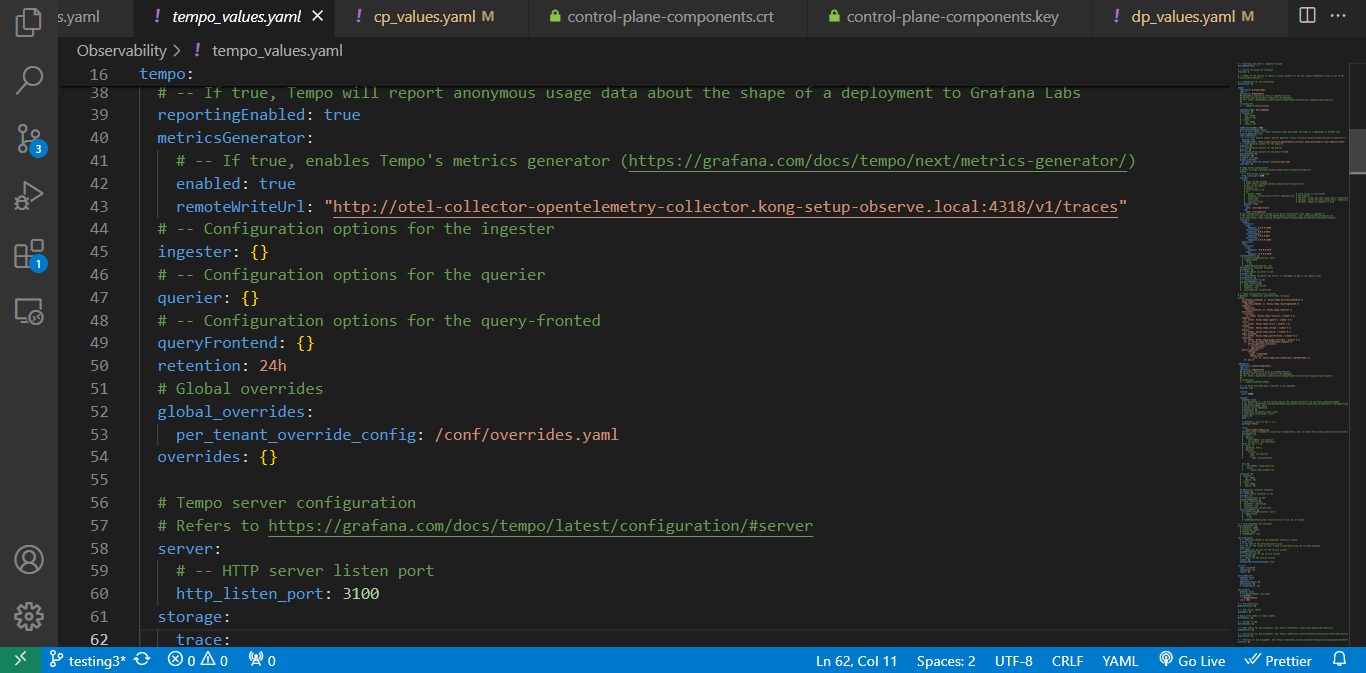


* Tempo mainly we are doing for to get the trace data
* For trace data here kong sends the data to otel and from otel it goes to tempo and it will be displayed in the browser.
* Here below is setup details
* In CP and DP we need to include some details inside the **env section,** only then it works well
* 
* 
* Mainly we need to include in this CP and DP
* Then we need to do otel setup
* 
* Exporters: Defines exporters for sending data to external systems. It includes a debug exporter for debugging purposes and an OTLP exporter configured to send traces to a Tempo service (tempo.kong-setup-observe.svc.cluster.local:4317).
* OpenTelemetry Collector Configuration: You've configured the OpenTelemetry Collector to listen for incoming trace data on http://otel-collector-opentelemetry-collector.kong-setup-observe.svc.cluster.local:4318/v1/traces.
* Sending Traces: When requests are made to your Kong services, the OpenTelemetry plugin collects trace data for those requests. This trace data includes information such as the timing and context of the request.
* Forwarding Traces to OpenTelemetry Collector: The OpenTelemetry plugin in Kong sends this trace data to the specified endpoint of the OpenTelemetry Collector (http://otel-collector-opentelemetry-collector.kong-setup-observe.svc.cluster.local:4318/v1/traces).
* Processing by OpenTelemetry Collector: The OpenTelemetry Collector receives the trace data and processes it according to its configuration. This may include filtering, sampling, and transformation of the trace data.
* Exporting to Tempo: From your OpenTelemetry Collector configuration, it seems that traces are then exported to Tempo, which is an open-source distributed tracing backend. The trace data is sent to Tempo for storage, analysis, and visualization.
* We need to install otel plugin and we need to mention the endpoint in opentelemerty plugin where kong trace data will be sent
* 
* Then we need to go for endpoint setup
* 
* So kong trace will be sent to the otel endpoint
* 
* So, when the OpenTelemetry Collector processes trace data, it will send that data to 'tempo.kong-setup-observe.svc.cluster.local' on port 4317, which is where the Tempo service is expected to be running. This data will then be received and processed by Tempo for storage, analysis, and visualization.

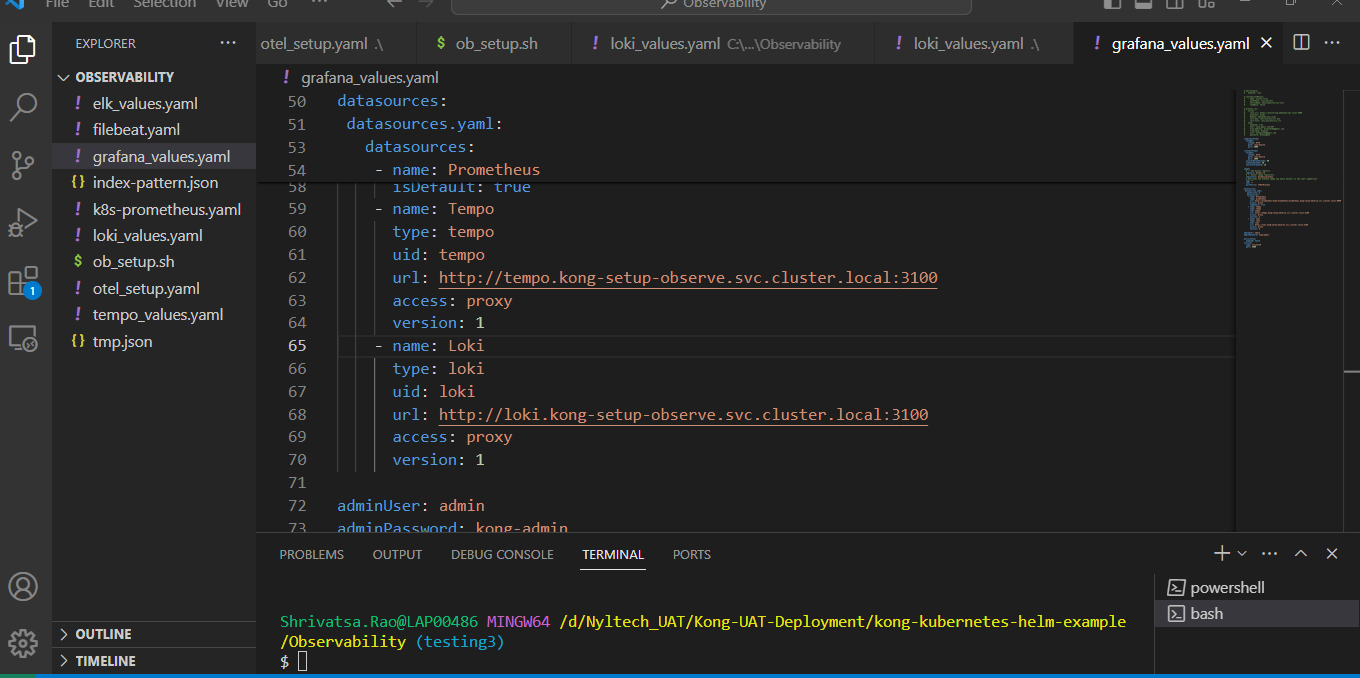
Endpoint: 'tempo.kong-setup-observe.svc.cluster.local:4317'

* tempo.kong-setup-observe.svc.cluster.local: This represents the Kubernetes Service DNS name for the Tempo service within your cluster. The svc.cluster.local domain is a standard domain used by Kubernetes for internal service discovery.
* 4317: This is the port number on which the Tempo service is listening to receive OTLP (OpenTelemetry Protocol) trace data. OTLP is a standard protocol for exporting trace data in OpenTelemetry.
* Kong Trace Data: Initially, trace data is collected by Kong services using the OpenTelemetry plugin. This plugin instruments Kong to capture trace data for requests it processes.
* Sending Trace Data to OpenTelemetry Collector (OTel): The OpenTelemetry plugin in Kong is configured to send trace data to an OpenTelemetry Collector instance. This is specified in the remoteWriteUrl parameter of the OpenTelemetry plugin configuration. The collector instance is running at "http://otel-collector-opentelemetry-collector.kong-setup-observe.local:4318/v1/traces".
* Processing Trace Data in OTel: The OpenTelemetry Collector receives the trace data from Kong and processes it according to its configuration. This may include aggregation, sampling, or transformation of the trace data.
* Sending Processed Trace Data to Tempo: After processing, the OpenTelemetry Collector is configured to send the trace data to Tempo for storage, analysis, and visualization. This is typically done using Tempo's ingest endpoint, where trace data is sent for storage and querying.
* Generating Metrics in Tempo: In addition to storing trace data, Tempo is configured to generate additional metrics based on the incoming trace data. This is enabled by setting metricsGenerator.enabled: true in the Tempo configuration. The generated metrics may include statistics such as request latency, error rates, and throughput.
* Sending Metrics Data to OTel: Finally, Tempo's metrics generator is configured to send the generated metrics data back to an OpenTelemetry Collector instance. This is specified in the remoteWriteUrl parameter of the Tempo configuration. The collector instance is running at "http://otel-collector-opentelemetry-collector.kong-setup-observe.local:4318/v1/traces".

In summary, trace data from Kong is initially sent to an OpenTelemetry Collector, processed, and then forwarded to Tempo for storage and analysis. Additionally, Tempo generates metrics based on the incoming trace data, which are then sent back to an OpenTelemetry Collector for further processing or integration with other observability tools.

* This is about the otel setup.
* 
* This endpoint will be provided in tempo setup.
* remoteWriteUrl: This parameter defines the URL endpoint where Tempo should send its generated metrics data. In this case, the URL is set to "http://otel-collector-opentelemetry-collector.kong-setup-observe.local:4318/v1/traces".
  + The URL consists of the hostname otel-collector-opentelemetry-collector.kong-setup-observe.local and the port 4318, followed by the path /v1/traces.
  + This URL indicates that the generated metrics data should be sent to an OpenTelemetry Collector instance running within your Kubernetes cluster, specifically to the endpoint /v1/traces.
  + The OpenTelemetry Collector typically exposes endpoints to receive various types of telemetry data, including traces and metrics. In this case, the /v1/traces endpoint is likely configured to accept incoming metrics data in a specific format (e.g., OTLP format).

In summary, the remoteWriteUrl parameter instructs Tempo's metrics generator to send its generated metrics data to the specified endpoint, which is likely an OpenTelemetry Collector instance configured to receive metrics data for further processing or storage.

* For tempo we need to include the url as mentioned below
* 
* We need to include url in grafana then it will work well.