**Food Trucks Project**

**Approach:**

1. Implemented using Java/Spring boot framework to display the list of food trucks from SF Gov REST API.
2. Have implemented the command line approach and as a web application
3. README file has all instructions to run from command line vs web application

**Web Application Architecture:**

**Client Controller Handler Helper service**

**Client:**

Either mobile or any PWA app can trigger the http rest call to Food Trucks project using the format: protocol://domain/sfgovFoodTrucks/getFoodTrucks?limit={}&offset{}

Request params limit and offset helps to have pagination feature on UI.

**Controller:**

Request mapping acts as a dispatcher for the client requests navigate the requests from client to appropriate implementation method based on mapping (here it is /getFoodTrucks) that accepts limit and offset as request params

**Handler:**

This is where the client request is transformed that need to make a REST call to SF gov service and response is processed from SF gov service to assemble to dto based on client requirement.

Construct the common response object kind of wrapper that wraps the core data needed for client and have all base params used to process by client.

**Helper:**

Implement the actual REST API call with all header info needed for SF gov service and end point.

Handle the exception from the remote service and create application specific exception that should be commonly handled across the application

Return the results back to handler for better polishing the response based on client needs.

With the increase number of concurrent users, have it implemented with session pooling (with tokens) which helps to reuse the session across different users from available pool. Once the session is completed the task, place the session back from busy pool to available pool. This way the session is always in your app control and protecting the remote site from over loading.

**Circuit Breaker**

We can implement circuit breaker which helps to save resources when the remote API is not responding. We can have a threshold limit to half open a circuit to check the health state and continue accessing the remote API if the health state is responded OK.

**Swagger**

Swagger implementation helps to visualize the complete API with sample request/response that can used for API consumption by third party.

**Health check**

Actuator can be added to application to incorporate monitoring of the service for health status. This helps to trigger the alerts to stake holders if app health is down.