**High Level Design (HLD) for Selenium Lab**

1. **Introduction**

The high-level design provides an overview of the entire system in the Selenium Lab.

1. **System Architecture:** 
   1. **Modular Design**

* **Food.py:** The set of prices and items.
* **Cart.py:** Handles the cost information for items.
* **Constant\_values.py:** Contains the constants used in the program.
* **App.py:** Runs the program with the instance of Flask.

1. **Data Flow**
2. **Input Data Flow:**

* The app module receives information about the user: adding items to the cart or removing the items from the cart.
* The cart module receives data from the app module and calculates the total, subtotal, and discounts based on the selected items.

1. **Output Data Flow:**

* The app module generates order confirmation for the user.
* The cart module provides feedback to the user when items are added or removed from the cart.

**4. Database Design**

* **No Database Design.**

**5. User Interaction Interface**

* **Tester Module:** An example of a test script with PyTest and Selenium. The test script tests the application to see if it runs as expected.

**6. System Components and Their Interactions**

1. **Food Module:**

* References the items and prices in the shopping application.
* Interacts with the app module to retrieve price and item information.

2. **Cart Module:**

* Interacts with the app module to update and display the selected items in the user’s cart.
* Interacts with the food module to retrieve information such as name and price.

3. **Constant\_values Module:**

* The module is accessible to other modules to utilize the constant values when necessary.

4. **App Module:**

* Interacts with the constant\_values module to access the constant values such as prices.

7. **Security and Exception Handling**

* **Input Validation:** Protects the user’s sensitive information such as payment information.
* **Exception Handling Mechanism:** Built into the cart module to raise ValueErrors for applying discounts or the removal of items from the cart.

**8. Scalability and Maintenance**

* **Modular Design:** Components of the application are broken down into singular modules to perform specific functions. Modular Design promotes reusability of code!
* **Maintainability:** Modules can be maintained easily without disturbing the entire application.

**9. Conclusion**

The High-Level Design for the Selenium Lab provides a representation of the overall structure of the shopping application. The utilization of modular design makes maintenance of the system easier and modules can be modified with ease.