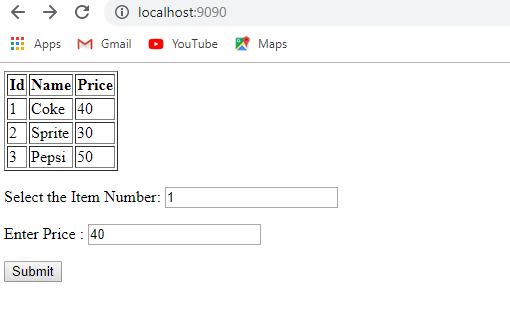
**Vending Machine Application**

**By Avula Venkata Ravi kumar**

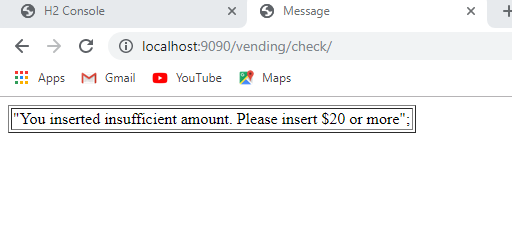
**On 24th December, 2019**

We have implemented basic user interface with **thymeleaf** using **spring boot** technology.

When we hit the url (<http://localhost:9090/> ) .We will get below screen.



After giving this input the selected item dispensed.



1. **Design Considerations:**

Scalability & Flexibility: You can change items, price being dispensed without any code changes.

**Technical Considerations:**

1. External webserver’s not required to run application, we have embedded servers with spring boot technology.
2. Abstraction of implementation layer (interfaces) to support changes in vending machines.
3. **Technical Architecture Design:**

The application architecture will divided into four categories: **presentation, application, domain, and infrastructure**.

**The presentation layer** contains **thymeleaf** templates responsible for presenting the UI to the end-user and sending the response back to the client (if we want to do operation other than GET).

* **The application layer** follows framework, contains business logic that is required by the application to meet its functional requirements. The application layer consisted of services orchestrating the domain objects to fulfill a use case scenario.
* **The domain layer**represents the underlying domain, mostly consisting of domain entities and, in some cases, services. Business rules, like invariants and algorithms, should all stay in this layer.
* **The infrastructure layer (also known as the persistence layer)**contains all the classes responsible for doing the technical stuff, like persisting the data in the database, like DAOs, repositories.

There are two important rules for a classical Layered Architecture to be correctly implemented:

1. All the dependencies go in one direction, from presentation to infrastructure. (Well, handling persistence and domain are a bit tricky because the infrastructure layer often saves domain objects directly, so it actually knows about the classes in the domain)
2. No logic related to one layer’s concern should be placed in another layer. For instance, no domain logic or database queries should be done in the UI.
3. **How to use the Vending Machine System:**

We are displaying list of items in vending machine from backend services and displays on UI with Cost and Item name and item id.

**Operating steps:**

1. Execute http://<server\_address>/h2-console, press connect and execute below insert script

**INSERT** **INTO** Item\_and\_Price (id,item\_name, item\_price) **VALUES**

(1,'Coke', '40'),

(2,'Sprite', '30'),

(3,'Pepsi', '50');

1. Display all item id, name, price from database. (http://localhost:9090)
2. Enter the Id and Price of item to be purchased in provided text boxes then click on submit button.
3. When deposited amount is greater than item price. We will populate one statement with item name and change.

“**Your Coke got dispensed with $10 returned**”

1. If user enter exact amount then item will be dispensed with following message.

“**Your Coke got dispensed**”

1. If user enter less amount then then item will be dispensed with following message.

“**You inserted insufficient amount. Please insert $20 or more**”