Technical Design and Development Roadmap

Persona Story:

In this application, my assumption is that it is a pizza ordering system for which we will develop the backend service.

From a user's perspective, the process is as follows:

- The customer can view all available pizzas on the menu.
- They can add pizzas to their order and have the option to customize them by selecting toppings (veg or non-veg, based on the pizza type).
- The system will display extra options such as additional cheese.
- It will also show side items like drinks or desserts.
- The customer can order multiple quantities of any item.
- Once all items are selected, the system processes the order and moves it forward.

System Components:

Controller Layer (API Endpoints):

Pizza Controller: Handles customer requests for pizza listing along with customization options I.e. toppings, extras,variant etc.

Order Controller: It handle customer order placement if order passed from rule engine validation business rule .

Service Layer:

Pizza Service: This is business logic layer for Pizza endpoint it contains business logic and forming final responses of customer request related to pizza listing and customization option response.

Order Service: It handles business logic applicable while saving and processing order and form response for order placement request by customer.

Order Validation : This component handles the execution of business rules as defined by the rule engine. Each type of action—Exclusion, Anyone, and Free—has a dedicated method, ensuring clear separation of logic. These action functions are implemented within this component based on various criteria such as Topping, Crust, and their applicability to different pizza variants. All rules are fully configurable and managed within the Rule_Engine table in the database, which is explained in the following section.

Rule Engine:

Model: rule_engines

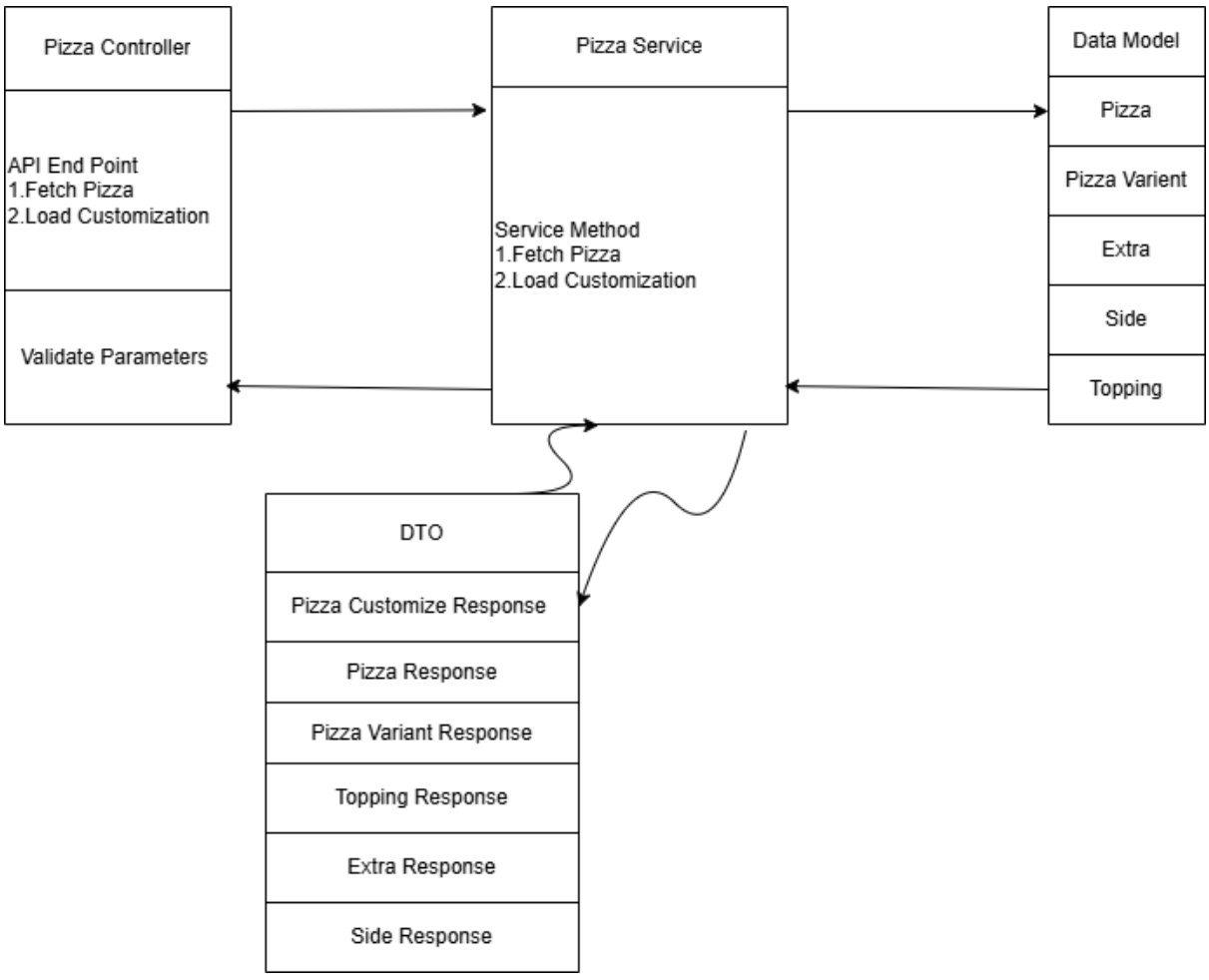
Attribute Name	Data Type	Specification
Category	Integer	It have value 0 and 1, 0 means veg and 1 means non veg.
Rule Category	String	This column contains criteria values such as "Topping", "Crust" on which this rule will get applied.
Type Of Rule	String	This column basically contains action values I.e. Exclusion, Anyone, Free.
Value	String	This column holds the data on which criteria and actions are applied. It contains comma-separated values. For example, in the case of a veg pizza, non-veg toppings are not allowed. This is considered an exclusion rule, and the column will contain the IDs of non-veg toppings, such as "5,6,7".
Variant size	String	This column basically contains comma separated size of pizza such as Regular, Medium and Large. Based on the value of size rule get executed if applicable.
Count Value	Integer	This column contains numerical value and it is applicable if it have value greater than 0 and also for action Free.
Error Message Local Key	String	This column contains local key for messages if rule get failed and messages are configured on local yml file of project.

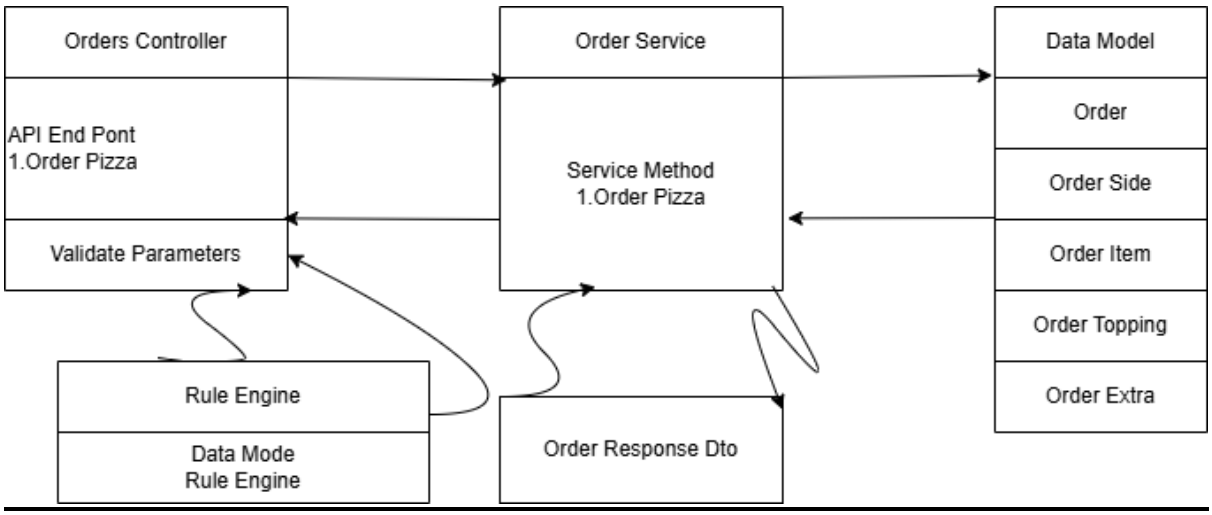
Data Transfer Objects (DTOs):

- **PizzaResponseDTO**: Structure for returning pizza details.
- **PizzaVariantResponseDTO**: Structure for returning available pizza variants details.
- **SideResponseDTO**: Structure for returning side details.
- **ToppingResponseDTO**: Structure for returning topping details.
- **PizzaCustomizeResponseDTO**: Structure for returning available pizza variant, topping, side details.
- **OrderResponseDTO:** Structure for returning order place successfully or not .

Authentication and Authorization :

- All APIs are secured using the OAuth2 authentication framework.
- We have implemented API security using the Doorkeeper gem.
- Authentication is based on the client ID and client secret method, as per our requirements. Each terminal is assigned a unique application with a client ID and secret for authentication.
- The token generation endpoint is included in the Postman collection.
- When running the seed data command, the client ID and client secret are printed in the console. These credentials can be used for authentication and authorization. Alternatively, they can be retrieved from the Doorkeeper applications table.





Postman request:

URL	Method Type	Request Body	Response

/oauth/token	POST	grant_type: client_credentials
		
		client_id:
		
		client_secret:
			{
			"access_token": "2aFmI6pyqy0nKJBqAhjQ8WwuHjSKvU5I7u9pWdBkULc",
			"token_type": "Bearer",
			"expires_in": 7200,
			"created_at": 1739120810
			}

/pizzaapi/fetchPizza	GET		[{ "id": 1, "pizza_name": "Deluxe Veggie", "category": 0 }, { "id": 2, "pizza_name": "Cheese and Corn", "category": 0 }, { "id": 3, "pizza_name": "Paneer Tikka", "category": 0 },

			{
			"id": 4,
			"pizza_name": "Non-Veg Supreme",
			"category": 1
			},
			{
			"id": 5,
			"pizza_name": "Chicken Tikka",
			"category": 1
			},
			{
			"id": 6,
			"pizza_name": "Pepper Barbecue Chicken",
			"category": 1
			}
			]

/pizzaapi/loadCustomization	POST	{"categoryid":0,
		"pizza_id":
		1
		}
			{
			"sides": [
			{
			"id": 1,
			"side_name": "Cold Drink",
			"price": "55.0"
			},
			{
			"id": 2,
			"side_name": "Mousse Cake",
			"price": "90.0"
			}
			],
			"extras": [
			{
			"id": 1,
			"extra_name": "Extra Cheese",
			"price": "35.0"
			},
			{
			"id": 2,
			"extra_name": "Cheese",
			"price": "32.0"
			},
			{
			"id": 3,
			"extra_name": "Mayonise",
			"price": "34.0"
			}
			],
			"toppings": [
			{
			"id": 1,

			"topping_name": "Black Olive",
			"price": "20.0"
			},
			{
			"id": 2,
			"topping_name": "Capsicum",
			"price": "25.0"
			},
			{
			"id": 3,
			"topping_name": "Paneer",
			"price": "35.0"
			},
			{
			"id": 4,
			"topping_name": "Mushroom",
			"price": "30.0"
			},
			{
			"id": 5,
			"topping_name": "Fresh Tomato",
			"price": "10.0"
			}
			],
			"variant": [
			{
			"variant_id": 1,
			"variant": "Regular",
			"price": "150.0"
			},
			{
			"variant_id": 2,
			"variant": "Medium",
			"price": "200.0"
			},
			{
			"variant_id": 3,
			"variant": "Large",
			"price": "325.0"
			}
			]
			}

/pizzaapi/loadCustomization	POST	{
		"order": {
		"customer_name":"Samaanta",
		"order_items": [
			{
			"message": "Order Successfully Placed",
			"status": 1
			}

		{"pizza_id": 1, "variant_id": 1, "crust_id": 2, "category": 0, "price": 150, "toppings": [{"topping_id": 1, "price": 25 }], "extras":[{ "extra_id": 1, "price":32}]}], "sides": [{ "side_id": 1, "price": 55 },{ "side_id": 2, "price": 90 }]}}	

Project Setup:

- Assumption that rails 7x and ruby3x is installed.
- Clone the repository using:git clone <https://github.com/pravat12345/Pizza.git>.
- Install dependencies:bundle install Rails db:migrate.
- Run database migrations:rails db:migrate.
- Seed the database:rails db:seed.
- Running Unit Tests.
- Prepare the test database:rails db:test:prepare.
- Execute tests:rails test.
- Running the Project
- Start the Rails server:rails s

- Use the Postman collection provided in the original email. Authenticate using your client ID and secret, which were generated during rails db:seed or retrieve them directly from the database.