Class & Subclass Overview with Best Practices – Eventsbridge

# 📦 Class Structure Overview

🔹 Abstract Class

* 1. User (Abstract Base Class)

🔹 Subclasses (Inherit from User)

* 2. Customer
* 3. Vendor
* 4. Admin

🔹 Other Entity Classes

* 5. Service
* 6. Location
* 7. Booking
* 8. ServiceMedia

✅ Total Classes: 8

# 🛠️ Best Practices for Class Design (in a Microservices System)

## ✅ 1. Separation of Concerns

You're cleanly separating concerns:  
- Customer, Vendor, and Admin have clearly defined responsibilities.  
- Service, Booking, etc. are business entities — good separation!

✅ Best Practice: Keep domain logic encapsulated per class or service.

## ✅ 2. Inheritance for User Types

Using an abstract User class with subclasses for Customer, Vendor, and Admin is a solid design.

🔸 Alternative (For microservices): Instead of strict OOP inheritance, consider representing user types with a 'role' field if you’re going full microservices — keeps services loosely coupled.

## ✅ 3. Entity Normalization

Entities like Service, Location, and Booking are independent — great for reusability and DB normalization.

# 📦 Suggested Microservice Mapping

Here’s how you could split services for scalability:

|  |  |
| --- | --- |
| Microservice | Handles Classes |
| User Service | User, Customer, Vendor, Admin |
| Service Catalog | Service, Location, ServiceMedia |
| Booking Service | Booking, with booking logic |
| Payment Service | Payment handling (not shown but implied) |
| Review Service | Store and manage reviews (extendable) |
| Admin Panel API | Restricted APIs for Admin management |