

# **QuickPick System – UML Diagrams & OCL Constraints**

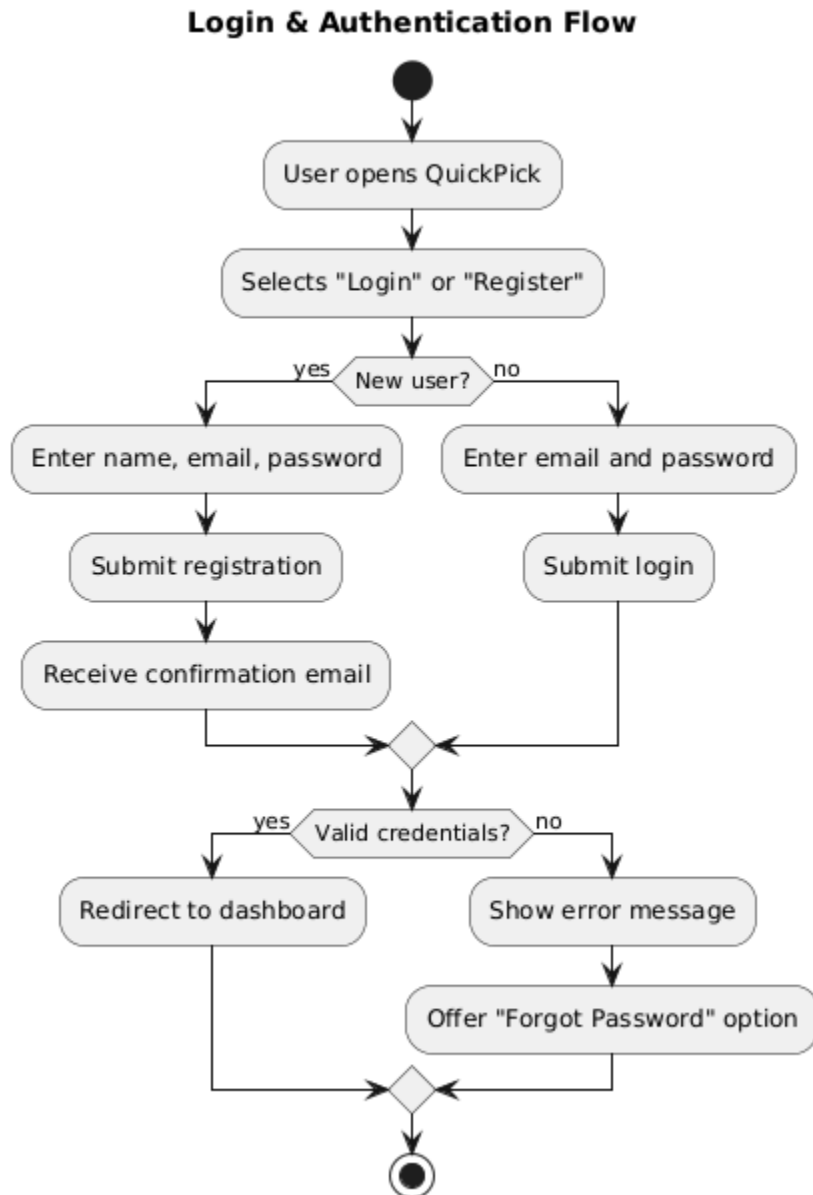
**UML 2**

**Niyati Nagar                      202412051**

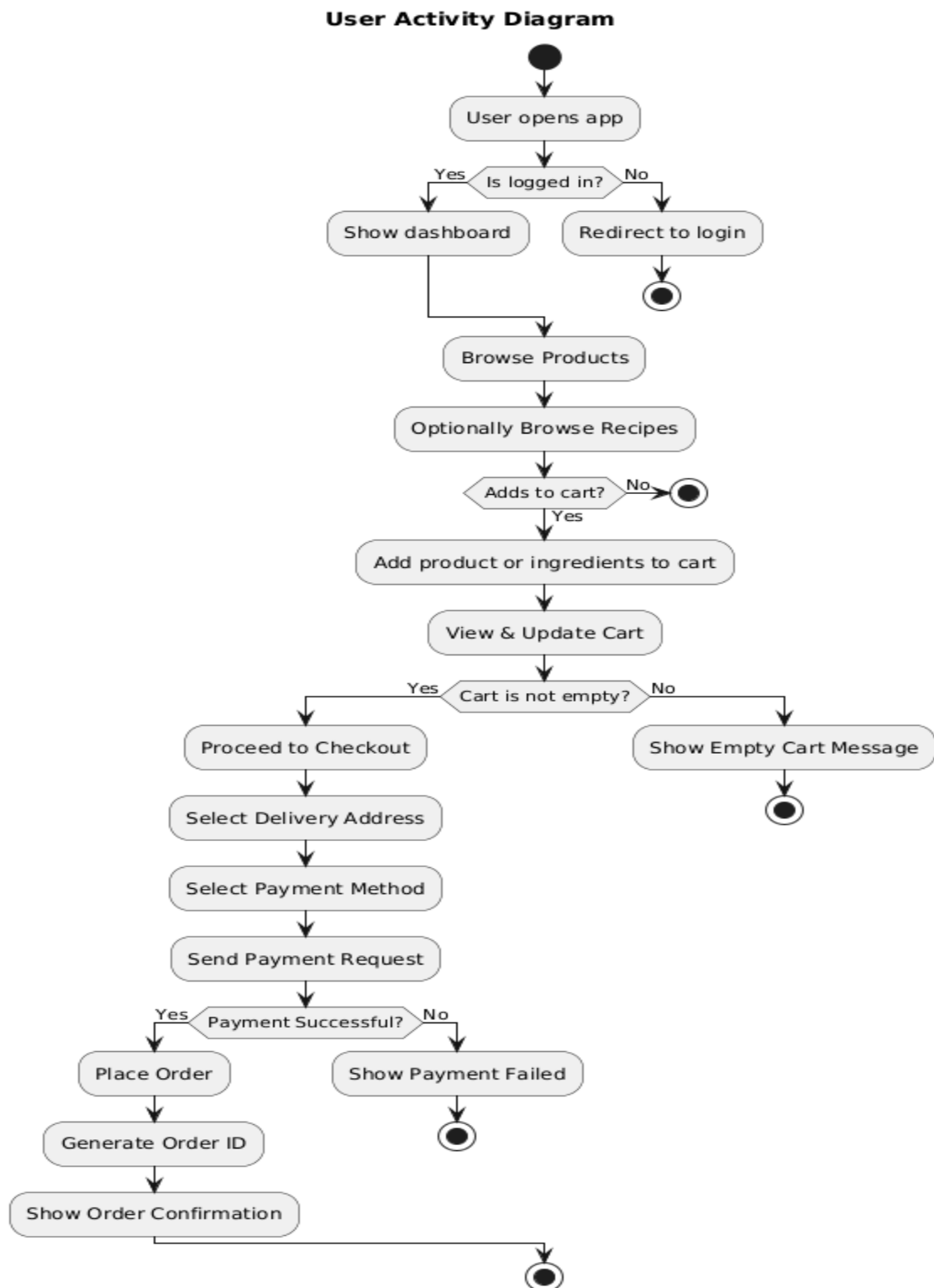
**Pankaj Mainani                      202412056**

## Activity Diagrams

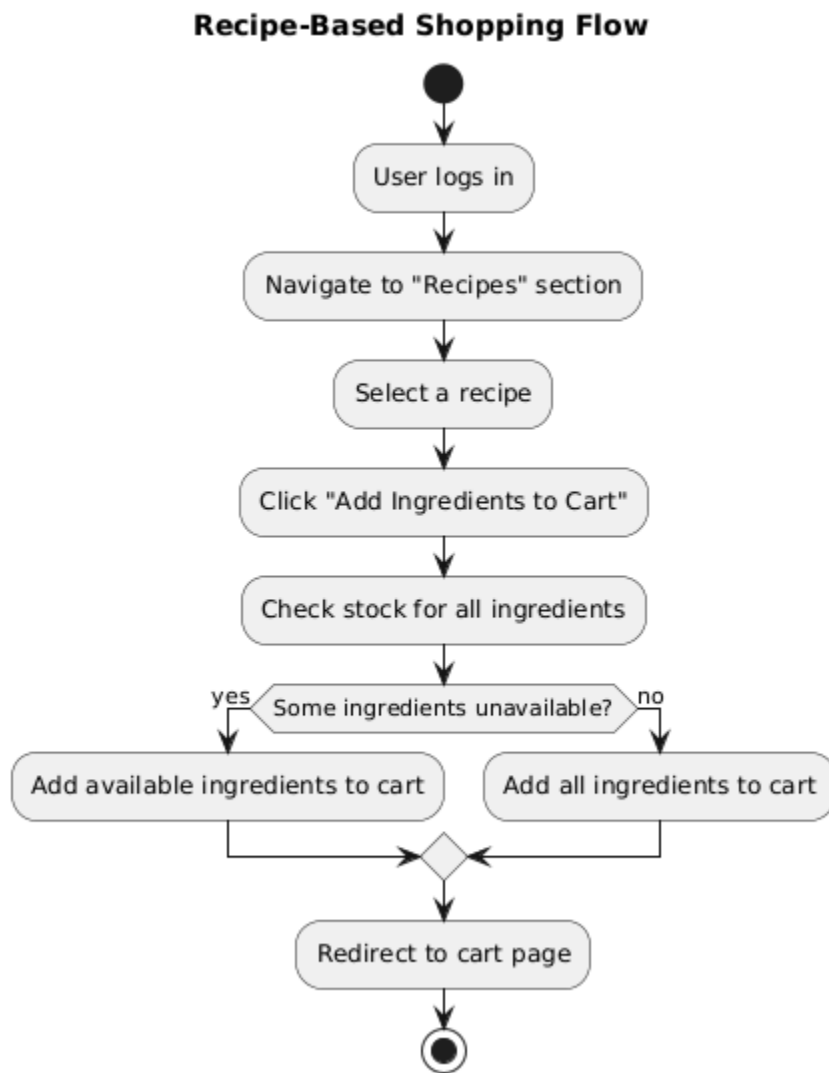
### Activity Diagram: Login & Authentication Flow



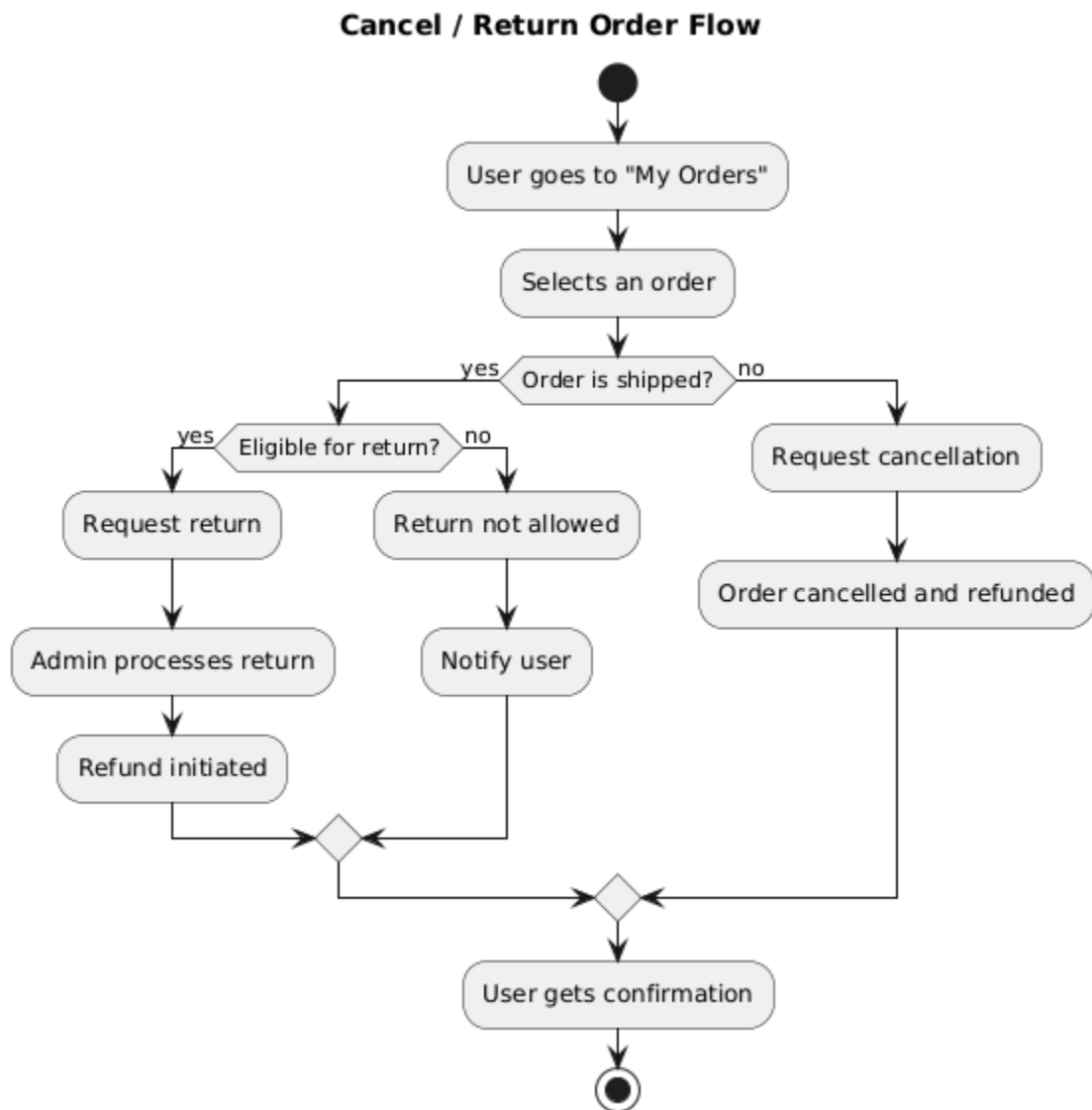
## Activity Diagram: User Order Placement Flow



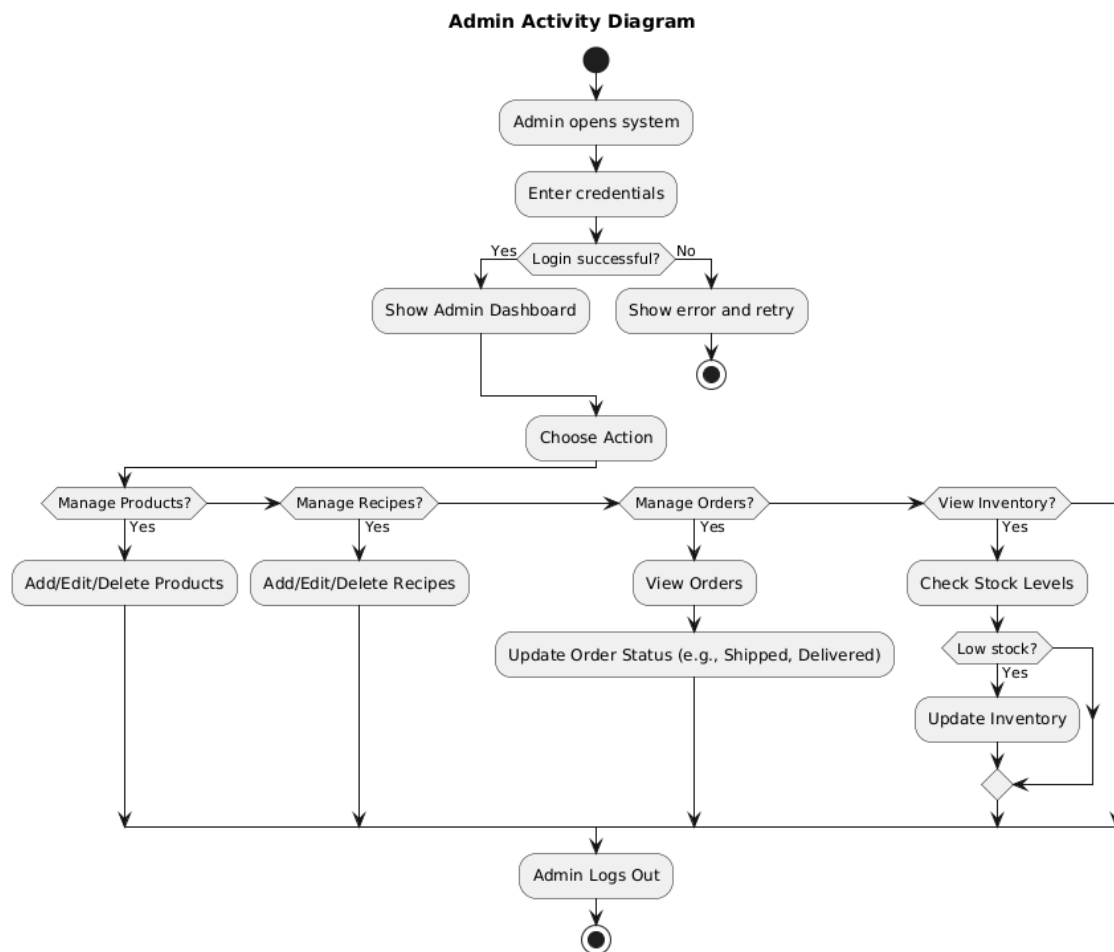
## Activity Diagram: Recipe-Based Shopping Flow



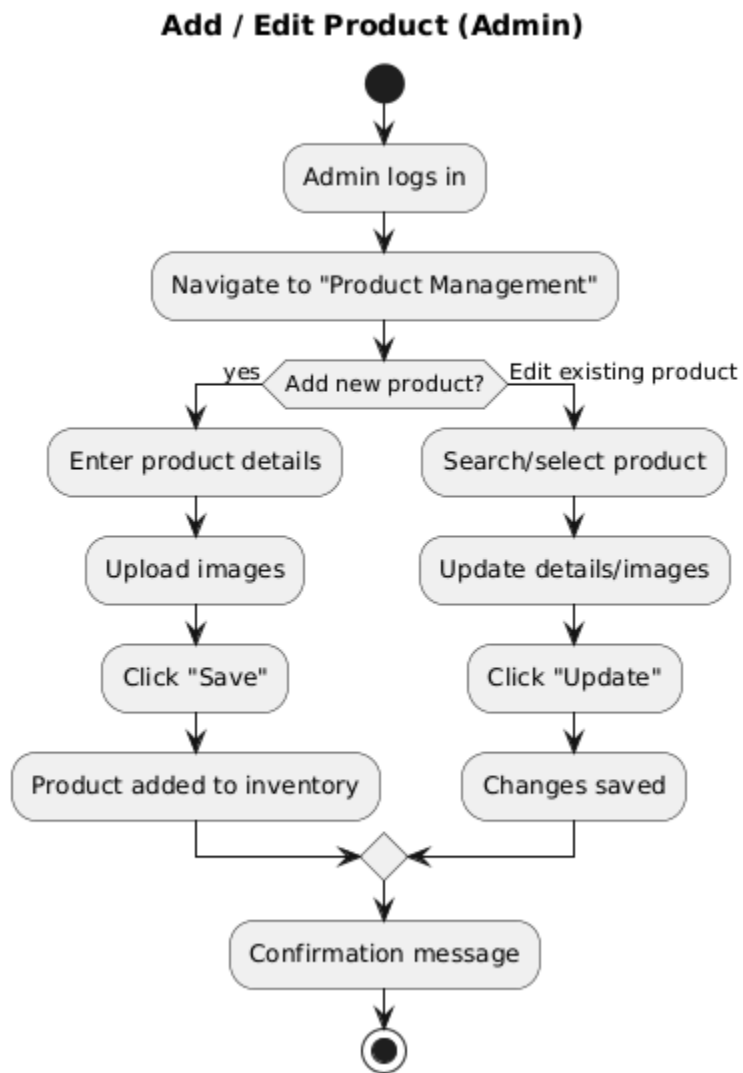
## Activity Diagram: Cancel/Return Order Flow



## Activity Diagram: Admin Dashboard Management



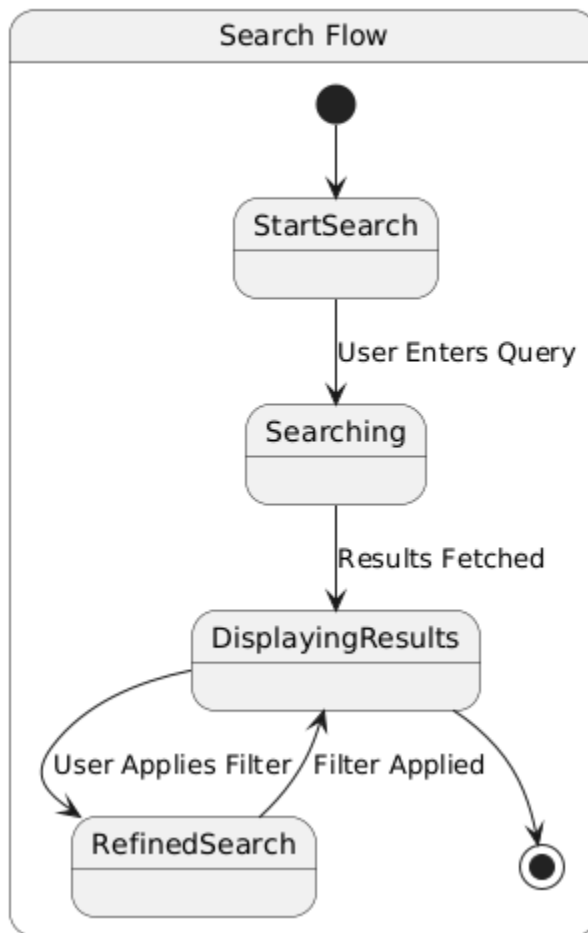
## Activity Diagram: Admin Add/Update Product Flow



## State Diagrams

### State Diagram: Search Feature States

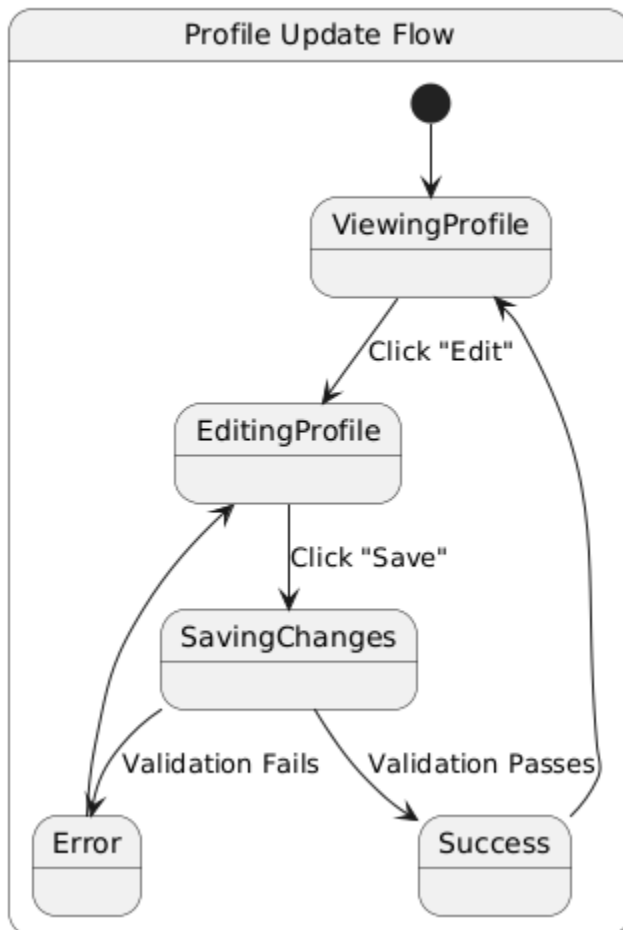
#### Product Search Flow - QuickPick System



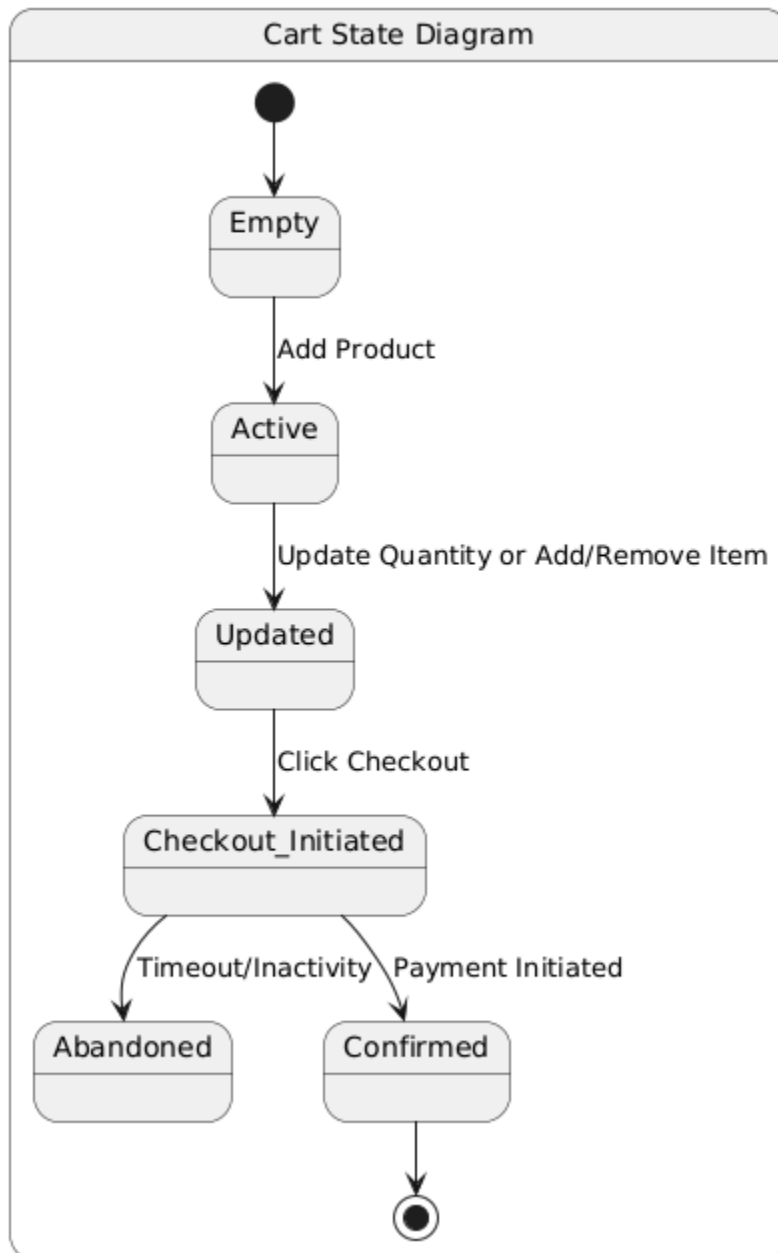


## State Diagram: User Profile Lifecycle

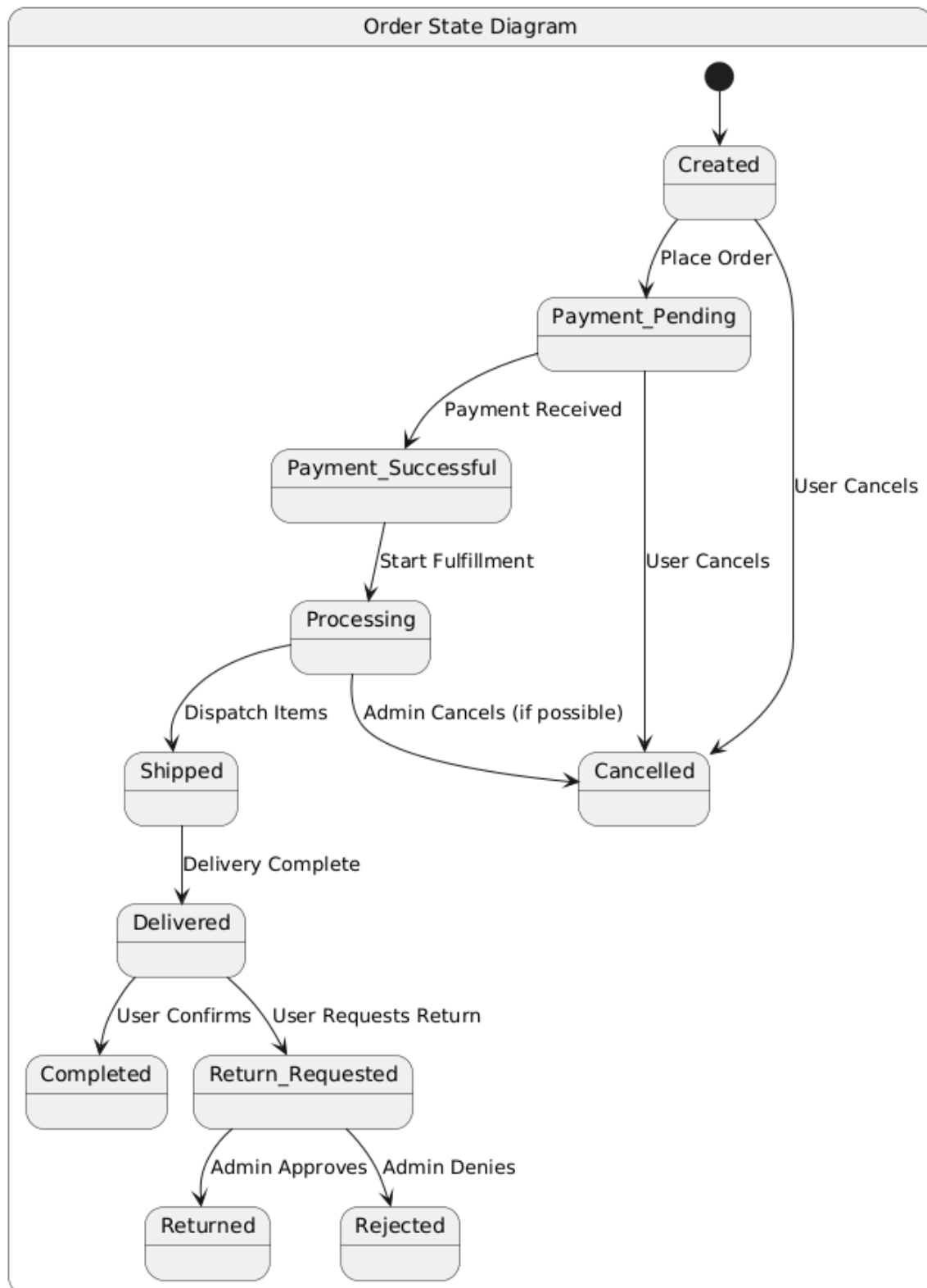
### Profile Update Flow - QuickPick System



## State Diagram: Shopping Cart States



## State Diagram: Order Status Lifecycle



## Object Constraint Language (OCL) Rules for QuickPick System

### Unique Email for Registration

```
context User
inv UniqueEmail:
    User.allInstances()->isUnique(u | u.email)
```

### Valid Password Length

```
context User
inv PasswordLength:
    self.password.size() >= 8
```

### Address Required for Checkout

```
context User
inv MustHaveAddress:
    self.addresses->size() > 0
```

### Phone Number Must Be 10 Digits

```
context User
inv PhoneNumberFormat:
    self.phone.matches('[0-9]{10}')
```

### Cannot Register if Already Logged In

```
context User
inv NotAlreadyLoggedIn:
    not self.isLoggedIn
```

### Product Price Must Be Positive

```
context Product
inv PricePositive:
    self.price > 0
```

### Product Stock Cannot Be Negative

```
context Product
inv NonNegativeStock:
    self.stockQuantity >= 0
```

### Product Must Belong to a Category

```
context Product
inv HasCategory:
    not self.category.oclIsUndefined()
```

### Recipe Must Have at Least One Ingredient

```
context Recipe
inv HasAtLeastOneIngredient:
    self.ingredients->size() > 0
```

### Order Total Must Match Sum of Items

```
context Order
inv ValidTotalAmount:
    self.totalAmount = self.items->collect(p | p.price *
p.quantity)->sum()
```

### Order Must Belong to a User

```
context Order
inv BelongsToUser:
    not self.user.oclIsUndefined()
```

### Can Only Cancel if Not Delivered

```
context Order
inv CancelConstraint:
    self.status <> 'Delivered'
```

### **Payment Must Be Linked to an Order**

```
context Payment
inv LinkedToOrder:
    not self.order.oclIsUndefined()
```

### **Payment Amount Must Match Order Total**

```
context Payment
inv AmountMatchesOrder:
    self.amount = self.order.totalAmount
```

### **Payment Method Must Be Valid**

```
context Payment
inv ValidMethod:
    self.method = 'UPI' or self.method = 'CreditCard' or
self.method = 'COD'
```

### **Only Admin Can Add/Edit/Remove Products**

```
context Admin
inv CanModifyProduct:
    self.role = 'admin'
```