# Functional Requirements

## 1) Startup & Inventory Loading

- UI requests GET /inventory from InventoryService.
- Response includes a list of items with { sku, name, price, inStock }.
- UI displays all items; OOS items are visually marked and disabled for selection.

## 2) Product Selection

- User can only select an in-stock item.
- After selection:
  - UI prompts for payment method (cash or card).
  - Starts a 10-second selection timer.

## 3) Selection Timeout

- If no payment method is chosen within 10 seconds:
  - Selection timer expires.
  - UI resets to default state.

# Payment Processing

## 1) Cash Payment

1. Initiate Payment
   - UI sends POST /payments { amount, method: "cash" } to PaymentService.
   - Response contains { status: "received", paid: <amountInserted> }.
2. User Cancellation
   - Before proceeding to change check, user can cancel:
     - UI sends POST /refund to PaymentService.
     - UI displays: “Payment canceled — refund issued”.
     - System resets to default state.
3. Change Availability Check
   - UI calls GET /change-availability?price=<itemPrice>&paid=<amountInserted> to ChangeDispenser.
   - If hasChange: true → proceed to dispense.

- If hasChange: false:
    - UI sends POST /refund to PaymentService.
    - UI displays: “Unable to provide change — payment refunded”.
    - Reset to default state.

## 2) Card Payment

1. Initiate Payment
    - UI sends POST /payments { amount, method: "card" } to PaymentService.
2. User Cancellation
    - During or after authorization, user may cancel:
        - UI cancels payment (via POST /void or equivalent).
        - UI displays: “Payment canceled — returning to default state”.
        - Reset to default state.
3. Authorization Result
    - If authorized → proceed to dispense.
    - If declined:
        - Response contains reason (e.g., insufficient_funds, network_error, authentication_failed).
        - UI shows decline reason and offers Retry or Cancel.
4. Decline Handling
    - Retry:
        - User triggers a new payment request.
        - If successful → proceed; if declined again → repeat decline handling.
    - Cancel:
        - UI resets to default state.
    - No action for 5 seconds:
        - Inactivity timer expires.
        - UI resets to default state.

# Dispensing

- On successful payment (cash or card):
    - UI sends POST /dispense { sku } to DispenserController.
    - If { status: "ok" } → UI displays: “Please take your drink”.

## Exception Handling

| Condition | Handling |
|---|---|
| **Jam / hardware error** | Dispenser returns { status: "error", code: "JAM" } → Refund payment, show apology, reset. |
| **No change available** | (Cash only) Refund and inform the user before dispensing. |
| **Payment declined (card)** | Show reason, allow retry/cancel, auto-reset after 5s inactivity. |
| **Selection inactivity** | Auto-reset after 10s without payment method choice. |
| **User-initiated cancel** | Refund or void payment (depending on method), return to default state. |