**SOFTWARE ENGINEERING**
**PROJECT 1c1**

# MVP: 10 Use Cases

## UC-MVP-1: Customer Places Simple Order

**Preconditions:** Customer has app installed; at least one restaurant available; customer has valid payment method.

**Main Flow:**

1. Customer opens app and confirms delivery address
2. System shows list of available restaurants with basic info (name, estimated delivery time)
3. Customer selects restaurant and views menu with prices
4. Customer adds items to cart with quantities
5. System calculates total (food + delivery fee + tax)
6. Customer proceeds to checkout and confirms order details
7. Customer selects payment method and places order
8. System processes payment and sends order to restaurant
9. System displays order confirmation with estimated delivery time

**Subflows:**

- 4a. Customer adds special instructions to items ("no onions", "extra sauce")
- 6a. Customer reviews and edits delivery address if needed
- 9a. System sends confirmation SMS/email with order number

**Alternative Flows:**

- 2a. No restaurants available → Display "No delivery available in your area"
- 8a. Payment fails → Show error, allow retry with same or different payment method
- 8b. Restaurant suddenly goes offline → Cancel order, refund immediately, suggest alternatives

---

## UC-MVP-2: Restaurant Accepts and Manages Order

**Preconditions:** Restaurant staff logged in; kitchen operational; order received from customer.

**Main Flow:**

1. System displays new order on restaurant dashboard with audio/visual alert
2. Staff reviews order details (items, quantities, special instructions, customer info)
3. Staff checks if all items are available and kitchen can fulfill
4. Staff accepts order and sets realistic preparation time
5. System notifies customer of acceptance and updated delivery estimate
6. Staff updates order status to "Preparing" when cooking starts
7. Staff marks order "Ready for Pickup" when complete

**Subflows:**

- 4a. Staff adjusts prep time based on current kitchen workload
- 6a. Staff can update customer with delays via system messages

**Alternative Flows:**

- 3a. Items unavailable → Staff rejects order, system immediately refunds customer and suggests alternatives
- 4a. Too busy/overwhelmed → Staff rejects order within 2 minutes, system handles customer notification
- 7a. Order sits ready too long → System alerts staff and notifies driver of potential delay

---

## UC-MVP-3: Driver Gets and Accepts Delivery

**Preconditions:** Driver logged in as available; has valid vehicle; order ready for delivery assignment.

**Main Flow:**

1. System assigns delivery to nearest available driver
2. Driver receives delivery request with basic info (pickup address, delivery address, estimated pay)
3. Driver has 60 seconds to review and accept or decline
4. Driver accepts delivery assignment
5. System provides restaurant contact info and basic pickup instructions
6. System notifies customer that driver is assigned
7. Driver navigates to restaurant using provided address

**Subflows:**

- 2a. Driver can see estimated time and distance for delivery
- 5a. System provides any special pickup instructions from restaurant

**Alternative Flows:**

- 3a. Driver declines → System immediately assigns to next available driver
- 3b. No response in 60 seconds → System auto-assigns to next driver
- 4a. No drivers available → System notifies customer of delay, offers pickup option or order cancellation

---

## UC-MVP-4: Driver Completes Pickup

**Preconditions:** Driver accepted delivery; arrived at restaurant; order marked ready by staff.

**Main Flow:**

1. Driver arrives at restaurant and locates pickup area/staff
2. Driver provides order number or customer name to staff
3. Staff confirms order details and hands over food bag
4. Driver quickly verifies bag contents match order summary
5. Driver marks order "Picked Up" in app
6. System notifies customer "Your order is on the way"
7. Driver begins delivery to customer address

**Subflows:**

- 4a. Driver takes quick photo of order bag for delivery record
- 7a. Driver uses phone's map app for navigation to customer

**Alternative Flows:**

- 3a. Order not ready → Driver waits up to 5 minutes, then contacts support if still not ready
- 4a. Bag contents clearly wrong → Driver alerts staff, waits for correction or contacts support
- 5a. App crashes → Driver calls/texts restaurant to confirm pickup completed

---

## UC-MVP-5: Customer Receives Order

**Preconditions:** Driver has order; customer available; driver arrived at delivery address.

**Main Flow:**

1. Driver arrives at customer's delivery address

2. Driver calls/texts customer "I'm here with your order"
3. Customer comes to door/meeting point within reasonable time
4. Driver verifies delivery with customer name or address
5. Driver hands order to customer
6. Customer confirms they received the complete order
7. Driver marks delivery "Complete" in app
8. System finalizes payment and sends receipt to customer

**Subflows:**

- 2a. Driver sends photo of delivery location if contactless delivery requested
- 8a. System prompts customer to rate the order (1-5 stars)

**Alternative Flows:**

- 3a. Customer not available after 5 minutes → Driver attempts second contact, marks undelivered if no response
- 6a. Customer reports items missing → Driver notes issue, directs customer to report through app
- 7a. App failure → Driver uses backup method (call/text) to confirm delivery completion

---

## UC-MVP-6: Customer Reports Order Problem

**Preconditions:** Customer received order; issue with order (missing items, wrong food, quality problem).

**Main Flow:**

1. Customer opens app and finds their recent completed order
2. Customer taps "Report Issue" button
3. System shows simple issue categories (Missing Items, Wrong Order, Food Quality, Other)
4. Customer selects issue type and adds brief description
5. System reviews issue and order value to determine resolution
6. System offers immediate solution (full refund, partial refund, or credit)
7. Customer accepts offered resolution
8. System processes refund/credit and confirms with customer

**Subflows:**

- 4a. System requests photo for certain issue types (wrong items, damaged food)
- 8a. System logs issue details for restaurant feedback (without customer identity)

**Alternative Flows:**

- 5a. High-value order or complex issue → System escalates to human support with customer callback
- 7a. Customer rejects resolution → System escalates to human support for manual review
- 8a. Refund processing fails → System queues refund and confirms processing timeline with customer

---

## UC-MVP-7: Process Payment Securely

**Preconditions:** Customer completed order; selected valid payment method; payment processor available.

**Main Flow:**

1. System calculates final total (subtotal + delivery fee + tax + tip if added)
2. Customer reviews total and confirms payment
3. System securely sends payment request to payment processor (Stripe/Square)
4. Payment processor validates card and processes charge
5. System receives payment confirmation
6. System marks order as paid and triggers restaurant notification
7. System sends payment receipt to customer via email/SMS

**Subflows:**

- 1a. Customer adds optional tip amount during checkout
- 7a. System distributes payments (restaurant portion, driver portion, platform fee)

**Alternative Flows:**

- 4a. Card declined → System shows specific error, allows retry or different payment method
- 4b. Payment processor down → System queues payment for retry, notifies customer of delay
- 5a. Payment timeout → System cancels order, ensures no charge occurred

---

## UC-MVP-8: Cancel Order When Possible

**Preconditions:** Customer has active order; order status allows cancellation.

**Main Flow:**

1. Customer opens app and views their active order
2. Customer taps "Cancel Order" button
3. System checks current order status and cancellation policy
4. System shows cancellation terms (refund amount after any fees)
5. Customer confirms they want to cancel
6. System cancels order and notifies restaurant/driver immediately
7. System processes refund according to cancellation policy

**Subflows:**

- 4a. System shows timeline: "Full refund if cancelled within 5 minutes, partial refund after"
- 7a. System sends cancellation confirmation with refund details

**Alternative Flows:**

- 3a. Order already being prepared → System warns of limited refund, requires second confirmation
- 3b. Driver already assigned/picked up → System charges cancellation fee, processes partial refund
- 7a. Refund processing fails → System queues refund and provides customer with reference number

---

## UC-MVP-9: Send Critical Order Updates

**Preconditions:** Customer has active order; order status changed; customer has notification preferences set.

**Main Flow:**

1. Order status changes (confirmed, preparing, ready, picked up, delivered)
2. System determines if status change requires customer notification
3. System formats appropriate message for status update
4. System sends push notification to customer's phone
5. Customer receives notification and can tap to view order details
6. System logs notification as delivered

**Subflows:**

- 4a. If push notification fails, system falls back to SMS
- 5a. Notification includes estimated time for next status update

**Alternative Flows:**

- 4a. Customer has notifications disabled → System skips notification but updates order status in app
- 4b. Notification service down → System queues notification for when service resumes
- 6a. Critical status (delivered/cancelled) → System ensures notification is sent via backup method if primary fails

---

## UC-MVP-10: Handle Basic System Errors

**Preconditions:** System experiencing failures (server overload, database issues, payment problems); users trying to use platform.

**Main Flow:**

1. System monitoring detects service disruption or user reports error
2. System activates basic fallback procedures for affected components
3. System displays clear, helpful error messages to users
4. System logs error details with timestamp and user context
5. System attempts automatic restart/recovery of failed services
6. System restores normal operation when technical issue resolved
7. System confirms all user data and transactions are intact

**Subflows:**

- 2a. System queues failed orders/payments for retry when service restored
- 3a. Error messages include estimated restoration time when known

**Alternative Flows:**

- 5a. Auto-recovery fails → System alerts technical support team immediately
- 5b. Extended outage → System displays maintenance page with updates every 30 minutes
- 7a. Data integrity issues → System prevents new transactions until validation complete

---

## Critical Dependencies - If Any Fails, Platform Dies

1. **UC-MVP-1** - No customer orders = no business
2. **UC-MVP-2** - Restaurants can't fulfill = no food delivered
3. **UC-MVP-3** - No driver assignment = food stays at restaurant
4. **UC-MVP-4** - Failed pickup = hungry customers, wasted food

5. **UC-MVP-5** - Failed delivery = customer never gets food
6. **UC-MVP-6** - No issue resolution = angry customers leave forever
7. **UC-MVP-7** - Payment failures = no revenue, legal issues
8. **UC-MVP-8** - Can't cancel orders = trapped customers, chargebacks
9. **UC-MVP-9** - No order updates = anxious customers calling constantly
10. **UC-MVP-10** - No error handling = system crashes kill everything

**MVP Success Metric:** Can 100 orders flow through this system in one day without any manual intervention or major customer complaints?

Everything else is a luxury until you hit this baseline reliability.

# Reflection Document

## How did we decide what NOT to do?

We used two tests to decide what to cut:

1. **Make-or-Break Test** – If this fails, does the platform die?

2. **Core Value Test** – Does this help deliver a single meal?

From 30 use cases, we kept 10 on the critical path: order → prepare → deliver → receive & pay.

**What Got Cut:**

- **Customer:** group orders, promotions, health filters, social/loyalty features → nice but not core.

- **Restaurant:** POS integration, inventory alerts, analytics, onboarding wizards → manual entry & personal onboarding suffice.

- **Driver:** route optimization, scheduling, dashboards, advanced dispatch → basic assignment + GPS works.

- **Admin:** forecasting, monitoring, governance, market strategies → need data later; first prove basics.

## What negative impacts or disappointments this MVP could have for your stakeholders

**Customers Will Be Frustrated By:** lack of personalization (no favorites, recommendations, or rewards), limited functionality (no post-order changes or group ordering), weak search and filters (no diet, price, or rating options), manual support through calls/emails instead of instant chat, and a bare-bones interface with minimal notifications and no real-time tracking.

**Restaurants Will Struggle With:** extra work from manually entering online orders (no POS link), no access to business insights like sales trends or popular items, missing tools for inventory, promotions, or alerts, limited ability to contact customers about delays, and slower manual onboarding instead of self-service.

**Drivers Will Miss:** efficiency tools such as route optimization and multi-delivery batching, flexible scheduling (only basic availability toggle), detailed earnings breakdowns or performance tracking, and stronger dispatch support beyond simple assignment.

**Platform Will Face:** heavy reliance on manual customer service and monitoring, limited scalability without forecasting or dynamic pricing, reactive problem-solving instead of proactive systems, and a basic revenue model with only commissions.

## What changes you made (and why) to the MVP to appease at least some of the stakeholders

**Strategic Additions to the MVP**

**For Customers:**

- **Problem Reporting (UC-MVP-6):** Ensures customers can report issues; prevents negative reviews from one bad experience.
- **Order Cancellation (UC-MVP-8):** Reduces frustration, chargebacks, and brand damage.
- **Order Notifications (UC-MVP-9):** Basic status updates reduce “where’s my food?” calls.

**For Restaurants:**

- **Enhanced Order Management (UC-MVP-2):** Reject orders, report delays, and maintain control.

- **Flexible Timing:** Set realistic prep times based on workload.
- **Order Verification:** Built-in checks to avoid costly mistakes.

**For Drivers:**

- **Fair Assignment (UC-MVP-3):** First-available rule avoids favoritism.
- **Clear Protocols:** Defined pickup/delivery steps reduce errors.
- **Upfront Pay Info:** Drivers see estimated earnings before accepting.

**For Platform Success:**

- **Error Handling (UC-MVP-10):** Prevents minor issues from escalating into disasters.
- **Secure Payments (UC-MVP-7):** Essential for compliance and revenue protection.
- **Comprehensive Logging:** Tracks every action for debugging and dispute resolution.

**Approach:** Instead of half-building many weak features, we fully built the essentials. For example, a reliable manual order system was prioritized over a buggy POS integration.

**Result:** A simple, functional platform that delivers food reliably today, with room to add advanced features tomorrow.

## Prompt History

https://g.co/gemini/share/7de211fd041f

https://claude.ai/share/21330d7b-a042-4dee-9dae-f9660a8601e6