Deepseek-R1-7b Use Cases:

Use Case U1: Order Assignment upon Arrival

Preconditions:

- A new order request has been received with valid details (location, time window).
- Driver availability data is current.

Main Flow:

- System checks the 'Available Now' status of nearby drivers based on geo-location and task queue.
- Identifies eligible drivers within a pre-defined radius or travel-time threshold (e.g., <5 minutes).
- Selects one driver dynamically, considering ETA, historical performance, and current demand.
- Sends a push notification or app alert to the selected driver detailing order specifics [No Available Driver] [Driver Accepts Immediately].

Subflows:

- [Driver Accepts Immediately] Driver confirms acceptance instantly via app.
- [No Available Driver] Notify stakeholders and mark order as pending; optionally assign a driver from a nearby zone.

Use Case U2: Assigning Driver Based on Order Urgency/Priority

Preconditions:

- Order is marked urgent or high-priority (e.g., pre-order, time-sensitive item).
- Eligible drivers exist.

Main Flow:

- System identifies eligible drivers.
- Prioritizes assigning the highest-rated available driver according to urgency rules [Offer Bonus Incentive].
- Sends notification emphasizing urgency and any bonus details [Priority Override Conflicts].

Subflows:

- [Offer Bonus Incentive] Optionally incentivize driver for urgent delivery.

Alternative Flows:

- [Priority Override Conflicts] Diverting a driver mid-route triggers reassignment or renegotiation.

Use Case U3: Reassign Order after Last-Minute Cancellation

Preconditions:

- A previously assigned driver cancels shortly before pickup/delivery.
- Updated restaurant availability and customer location are known.

Main Flow:

- System detects the cancellation event.
- Updates 'Order Status' to 'Needs Reassignment'.
- Recalculates eligible drivers considering time sensitivity and potential overlaps.
- Assigns order to the next best driver [New Pickup Window Requested].

Alternative Flows:

- [New Pickup Window Requested] No drivers can take the order within the original window; system prompts for reschedule.

Use Case U4: Driver-Requested Route Confirmation during Assignment

Preconditions:

- An eligible driver is assigned an order.
- System has proposed optimal route using navigation data (traffic, roads).

Main Flow:

- System presents the proposed route to the driver.
- Driver accepts or requests an alternative route [Route Rejection].
- If requested, system gathers reasons and recalculates the route.
- Confirms final assigned route to driver.

Alternative Flows:

- [Route Rejection] Driver rejects proposed route; system recalculates or provides manager-assisted alternative.

Use Case U5: Driver Self-Scheduling - Selecting Orders based on ETOC

Preconditions:

- Self-scheduling feature is active (e.g., during rush hours).
- Drivers are logged into their apps with access to available orders.

Main Flow:

- Driver reviews available order requests in designated area.
- System calculates Estimated Time to Completion (ETOC) for each potential assignment.
- Driver selects order(s) based on schedule preference or ETOC targets [Compare Orders] [Overlapping Selection].

Subflows:

- [Compare Orders] App highlights differences in ETOC and potential earnings between orders.

Alternative Flows:

- [Overlapping Selection] Driver selects multiple orders exceeding capacity; system warns or prevents selection.

Use Case U6: Configure Commission Structures

Preconditions:

- Platform administrator is logged in with sufficient privileges.
- Operational model (Logistics or Aggregator) is defined.

Main Flow:

- Administrator navigates to "Pricing & Partnerships".
- Selects operational model for a restaurant group or area.
- System presents commission structure options:
    - Logistics: base commission + performance-based bonuses/fees.
    - Aggregator: standard percentages connecting orders from restaurants.
- Administrator adjusts parameters [Model Not Defined] [Commission Calculation Error].
- System confirms and saves configuration.

Alternative Flows:

- [Model Not Defined] Prompt administrator to select a model before proceeding.
- [Commission Calculation Error] Invalid parameters trigger error message for correction.

Use Case U7: Calculate Dynamic Pricing

Preconditions:

- Customer is viewing the app or searching for orders.
- Platform uses aggregator model with variable costs or logistics network.

Main Flow:

- System detects current demand (peak hours, high order density).
- Calculates base delivery fee based on demand and operational model [No Surge/Adjustment] [Invalid Model for Dynamic Pricing].
- If surge pricing is applicable, applies adjustment.
- Displays total price to customer [Surge Explanation].

Subflows:

- [Surge Explanation] Displays message explaining surge pricing.

Alternative Flows:

- [No Surge/Adjustment] Low demand; base fee displayed.
- [Invalid Model for Dynamic Pricing] If model doesn't support dynamic pricing, show standard price.

Use Case U8: Manage Customer Loyalty Points

Preconditions:

- Customer is logged into their account.
- Active loyalty points program exists.

Main Flow:

- System tracks completed orders and user actions for point accrual.
- Customer accesses loyalty points balance [View Point History].
- Customer redeems points for discounts, cashback, free delivery, or other rewards [Redemption Failure] [Insufficient Points].

Subflows:

- [View Point History] Customer sees log of point awards and redemptions.

Alternative Flows:

- [Insufficient Points] Customer lacks points to redeem.
- [Redemption Failure] System logs error and informs customer.

Use Case U9: Process Payment Including Delivery Fee & Convenience Charge

Preconditions:

- Customer has selected items from a restaurant menu.
- Cart contains valid order items.

Main Flow:

- System presents order summary.
- Adds delivery fees based on location/rider availability [Select Delivery Address].
- Adds convenience charges or optional rider premium.
- Customer selects payment method [Payment Method Selection].
- Customer confirms order and payment details.
- Payment gateway processes the transaction [Insufficient Funds/Balance] [Payment Method Unavailable/Declined].

Subflows:

- [Select Delivery Address] Choosing address may recalculate delivery fees.
- [Payment Method Selection] Choosing method from saved cards/bank accounts/wallets.

Alternative Flows:

- [Insufficient Funds/Balance] Payment fails due to low balance.
- [Payment Method Unavailable/Declined] Method is declined; user prompted to select/add another.

Use Case U10: Apply Order-Based Discounts (Coupons)

Preconditions:

- Customer has a valid coupon code or is eligible for a promotion.
- System is configured with active discounts.

Main Flow:

- Customer enters coupon code manually or selects automatic discount [Checkout with Manual Coupon] [Automatic Discount Application].

- System validates code against active offers.
- If valid:
    - Displays discount amount or applies automatically.
- Recalculates final price based on applied discount [Redemption Failure].

Subflows:

- [Checkout with Manual Coupon] Step-by-step process for entering/applying code.
- [Automatic Discount Application] System applies discount based on criteria automatically.

Alternative Flows:

- [Redemption Failure] Internal error during coupon application; user notified.

Use Case U11: Browse Restaurants by Cuisine

Preconditions:

- User is logged into the Food Delivery App.
- Restaurant directory has various cuisines available for the user’s location.
- Internet connection is active.

Main Flow:

1. Customer opens the main menu of the app.
2. Customer selects "Restaurants" from menu options [Search Functionality] [Alternative Cuisine Filter].
3. System displays available restaurants categorized by cuisine [No Restaurants Found].
4. Customer scrolls through the list and taps a restaurant.
5. System presents the restaurant's menu page [Order Confirmation].
6. Customer reviews menu items and prices.

Subflows:

- [Alternative Cuisine Filter] Customer may filter restaurants by delivery time, minimum order amount, or other criteria.
- [Search Functionality] Customer may search directly by cuisine or restaurant name.

Alternative Flows:

- [No Restaurants Found] No restaurants match the criteria; alert suggests alternative searches.
- [Order Confirmation] Selecting items leads to order summary and checkout.

Use Case U12: Add Item(s) with Customization

Preconditions:

- Customer is viewing a restaurant menu.
- Customer has selected a specific dish.
- App allows item customization.

Main Flow:

1. Customer taps on a desired item.
2. System presents an interactive page showing base price and customization options (spice level, toppings, sides, etc.).
3. Customer adjusts customization options [Add Note].
4. Customer selects quantity [Insufficient Stock] [Sides/Drinks Selection].
5. System updates cart to reflect total items and cost.

Subflows:

- [Sides/Drinks Selection] Customer may add drinks or sides simultaneously.
- [Add Note] Customer may add special instructions (e.g., "Extra napkins").

Alternative Flows:

- [Insufficient Stock] Requested quantity exceeds stock; system prevents addition.

Use Case U13: Checkout with Loyalty Discount

Preconditions:

- Customer has items in cart.
- Customer is logged in.
- Customer belongs to a loyalty program with points accumulated.

Main Flow:

1. Customer taps "Checkout" from order summary.
2. System displays delivery/pickup options.
3. Customer selects "Delivery".
4. System shows loyalty points balance and available discounts [View Discount Offers].
5. Customer applies a discount using points [Modify Order][Insufficient Points].
6. System calculates total cost including taxes and delivery fee.
7. Checkout continues to payment selection.

Subflows:

- [View Discount Offers] Customer may view available discounts before selecting one.

Alternative Flows:

- [Insufficient Points] Customer lacks enough points to apply selected discount.
- [Modify Order] Customer modifies selections before finalizing checkout.

Use Case U14: Select Delivery/Pickup Address and Time

Preconditions:

- Customer has items in cart.
- Customer is logged in.
- App knows user's delivery area or requires address entry.

Main Flow:

1. Customer proceeds to checkout.
2. System displays delivery/pickup method options.
3. Customer selects "Delivery".
4. System prompts for:
    - Saved address (if available) [Address Not Saved]
    - Delivery time slot (e.g., ASAP or specific window)
5. Customer confirms selected address and time slot [Schedule Delivery] [Time Window Unavailable].

Subflows:

- [Schedule Delivery] Customer may refine selection of available slots.

Alternative Flows:

- [Address Not Saved] System prompts for a valid address if none exists.
- [Time Window Unavailable] If chosen slot is unavailable, system suggests alternatives.

Use Case U15: Handle Payment Failure and Retry

Preconditions:

- Customer is attempting payment during checkout.
- Delivery/pickup method and address/time have been selected.
- Payment attempt via Card X has been made.

Main Flow:

1. Customer proceeds through checkout (delivery, address, time).
2. System displays payment screen [Alternative Payment Method].
3. Customer taps "Pay Now".
4. System attempts to process payment via Card X [Payment Declined].

Subflows:

- [Alternative Payment Method] Customer can select another saved method or add new.

Alternative Flows:

- [Payment Declined] Payment fails (e.g., "Card declined"); system prompts retry or method change.

Use Case UC16: Restaurant Menu Management

Preconditions:

- Restaurant has an active profile and API integration with a delivery platform (or uses its own internal ordering system).
- Designated staff member (owner/manager) has administrative access.

Main Flow:

1. Staff logs into the admin dashboard or internal ordering system.
2. Staff navigates to the "Menu Management" section [Unauthorized Access Attempted].
3. Staff views, adds, edits, or deactivates menu items (name, price, description, image) [Update Item Availability] [Add Promotional Offer].
4. System saves changes in real-time to connected platforms [Error Saving Changes].
5. Staff logs out.

Subflows:

- [Update Item Availability] Mark items temporarily unavailable or update quantities.
- [Add Promotional Offer] Define special offers for online delivery orders.

Alternative Flows:

- [Unauthorized Access Attempted] Unauthenticated user cannot access menu management.

- [Error Saving Changes] System fails to save updates; allows retry.

Use Case UC17: Order Receiving & Preparation Initiation

Preconditions:

- Restaurant has active API integration with delivery platforms.
- Orders are routed based on availability and location.

Main Flow:

1. Customer places an order via third-party app or restaurant's website/app.
2. Restaurant system receives the order and assigns it to kitchen staff (first-come-first-serve) [Order Routing Logic].
3. Kitchen staff confirms readiness of items [Order Conflict].
4. Payment is processed securely by platform or restaurant system.

Subflows:

- [Order Routing Logic] System may prioritize orders based on commission rates or delivery time.

Alternative Flows:

- [Order Conflict] If ingredients are unavailable, system notifies the delivery platform.

Use Case UC18: Managing Inventory Levels

Preconditions:

- Restaurant POS or internal database is integrated with platform back-end.
- Staff has access to inventory management tools.

Main Flow:

1. Staff opens inventory overview in admin dashboard or ordering system.
2. System displays current stock, highlighting low items (e.g., "Low stock: Mushroom Risotto - 5 units left") [Automatic Inventory Depletion].
3. Staff updates inventory manually.
4. System uses inventory data to update menus or flag order issues.

Alternative Flows:

- [Automatic Inventory Depletion] System estimates remaining days based on order volume and current stock.

Use Case UC19: Driver Management & Assignment

Preconditions:

- Restaurant has enabled driver management.
- Active pool of drivers is logged into the system/platform.

Main Flow (Third-Party Perspective):

1. Admin receives order update requiring preparation and driver allocation.
2. System checks nearby active drivers or in-house staff [No Available Drivers] [Driver Declines Assignment] [Driver Profile View] [Set Max Delivery Radius].
3. System presents list of suitable couriers for assignment.

Subflows:

- [Driver Profile View] Displays availability, ratings, and delivery history.
- [Set Max Delivery Radius] Configures max distance for driver assignment.

Alternative Flows:

- [No Available Drivers] System alerts admin if no suitable drivers/staff are available.
- [Driver Declines Assignment] System rechecks availability and assigns another driver automatically.

Use Case UC20: Handling Customer Feedback & Ratings

Preconditions:

- Platform allows post-delivery customer feedback/ratings.
- Order status confirmed as delivered.

Main Flow:

1. Customer completes order delivery via app/website.
2. Platform prompts customer for rating shortly after delivery.
3. Restaurant staff accesses aggregated reviews and ratings [View Negative Feedback].
4. Staff views specific comments linked to recent orders or specific couriers/days [Low Review Count] [Rating Discrepancy].

Subflows:

- [View Negative Feedback] Filter reviews by rating score to identify issues.

Alternative Flows:

- [Low Review Count] Few reviews (<10); confidence bar shows "N/A" or low percentage.
- [Rating Discrepancy] Customer rating differs from internal quality score; system flags for manual review.

Use Case U21: Report Unexpected Order Delay

Preconditions:

- User is logged into their account on the Houston Food Delivery app.
- An order has been placed and scheduled for delivery within the last 2 hours.
- Estimated arrival time (ETA) is significantly exceeded with no driver update.

Main Flow:

1. User accesses "Active Orders" or "Order History" and specifies the order ID if prompted.
2. System displays the current order status and provides support options [Contact Support] [Track Driver Location] [No Active Order with ETA].
3. User selects "Report Delay" or contacts support via in-app chat, call, or email.
4. Customer support contacts the restaurant's dispatch system or driver network [Restaurant Dispatch System Integration].

Subflows:

- [Contact Support] Options: chat, phone, email.
- [Track Driver Location] Support may access driver location if feature enabled.
- [Restaurant Dispatch System Integration] Real-time order/driver info accessed via POS/booking system.

Alternative Flows:

- [No Active Order with ETA] If no active order is found or ETA was initially delayed, the system offers general troubleshooting (internet check, app restart) before allowing support contact.

Use Case U22: Request Correction/Update of Address

Preconditions:

- User is logged in.
- Order has been placed or is being prepared.
- User identifies incorrect, incomplete, or poorly formatted address.

Main Flow:

1. User navigates to active order details.
2. System displays "Edit Order" button [Address Already Sent Directly to Restaurant].
3. User edits the delivery address in a form [Use Map Feature].

Subflows:

- [Use Map Feature] Map interface allows geographic address correction.

Alternative Flows:

- [Address Already Sent Directly to Restaurant] Suggest contacting the restaurant directly.

Use Case U23: Order Status Shows Preparing but Driver Assigned

Preconditions:

- User is logged in.
- Order status shows 'Preparing'.
- User notices a driver assigned prematurely.

Main Flow:

1. User views active order details [View Order Timeline].
2. System displays 'Preparing' status and allows viewing driver info.
3. User selects "Report Discrepancy" or contacts support [Driver Assigned is Incorrect].

Subflows:

- [View Order Timeline] Shows all preparation and dispatch stages.

Alternative Flows:

- [Driver Assigned is Incorrect] Support verifies driver assignment.

Use Case U24: Order Item Incorrect

Preconditions:

- User is logged in.
- Order delivered or partially delivered.
- Received items do not match order.

Main Flow:

1. User accesses "My Orders" and selects the relevant order.
2. System displays "Report Issue" / "Contact Support".
3. User submits form or contacts support to report the incorrect item [Upload Evidence] [No Photos Uploaded].

Subflows:

- [Upload Evidence] Attach photos of discrepancy.

Alternative Flows:

- [No Photos Uploaded] Flag missing evidence; support follows up via email/SMS.

Use Case U25: Unable to Place an Order - Payment Failure During Checkout

Preconditions:

- User logged in.
- User attempts order payment.
- Payment error occurs (e.g., declined, invalid card, network issue).

Main Flow:

1. User proceeds through checkout to "Review Order & Pay."
2. Payment fails at final confirmation.
3. System displays error message and provides support or alternative payment options [Alternative Payment Method Prompt].

Subflows:

- [Alternative Payment Method Prompt] User can select a different saved card/method.

Use Case UC26: Manage Account Information

Preconditions:

- Customer is logged into their account.
- Customer has an existing profile in the system.

Main Flow:

1. Customer navigates to "Account Settings" or "My Profile" from the main menu.
2. System displays current profile information (name, email, phone number).
3. Customer selects "Edit Profile" to modify a detail [Change Security Questions].
4. Customer enters new value in the input field [Incorrect Format].
5. System validates data format and uniqueness.
6. Customer confirms changes if security details are updated.
7. System updates profile information securely and notifies the customer.

Subflows:

- [Change Security Questions] Modify account recovery questions.

Alternative Flows:

- [Incorrect Format] System displays an error for invalid input, returns focus to the relevant field for correction.

Use Case UC27: Manage Payment Methods

Preconditions:

- Customer is logged in and has viewed payment methods section.
- Customer has at least one saved or available payment method.

Main Flow:

1. Customer navigates to "Payment Methods" or "Billing".
2. System displays all stored payment methods (credit/debit cards, digital wallets).
3. Customer selects "Add New Payment Method".
4. System presents form to enter card number, expiry date, CVV, cardholder name.
5. Customer completes all required fields [Incorrect Format].
6. System validates fields, checks card validity, and securely stores the method.
7. If successful, system confirms addition/update and notifies the customer.

Alternative Flows:

- [Incorrect Format] System flags invalid input (e.g., incorrect card number length).

Use Case UC28: View and Access Order History

Preconditions:

- Customer is logged in.
- System has recorded past orders in the cloud database.

Main Flow:

1. Customer navigates to "Order History" or "My Orders".
2. System displays a list of previous orders by date (newest first) [View Full Details].
3. Customer selects an order to view detailed information:
    - Date/time of order
    - Delivery address (optionally blurred)
    - Ordered items with descriptions, quantities, and prices [View Loyalty Redemption Details]
    - Subtotal, delivery fee, total charged
    - Order status (Placed, Preparing, On the way, Delivered, Cancelled)
4. Customer can interact further with order info [No Orders Found].

Subflows:

- [View Full Details] Detailed breakdown of items and charges.
- [View Loyalty Redemption Details] Shows discounts applied via loyalty points.

Alternative Flows:

- [No Orders Found] System displays "No order history available" if none exist.

Use Case UC29: Set Dietary Preferences

Preconditions:

- Customer is logged in.
- System supports dietary preference management.

Main Flow:

1. Customer navigates to "Dietary Preferences" or "My Profile > Food Allergies/Diets".
2. System presents interface to specify dietary requirements (vegetarian, vegan, gluten-free, allergies) [Conflict Resolution Needed].
3. Customer selects one or more predefined options or provides further details if required.
4. System saves preferences securely in the customer profile.

Alternative Flows:

- [Conflict Resolution Needed] If ordering, system flags conflicts between dietary preferences and menu items.

Use Case UC30: Update Profile Security Settings

Preconditions:

- Customer is logged in.
- Customer has a profile with security settings configured.

Main Flow:

1. Customer navigates to "Security Settings" within Account Settings.
2. System displays current options (password, 2FA, security questions).
3. Customer selects an item to update [Update 2FA] [Change Password] [Change Security Questions].
4. Customer enters new values [Incorrect Format].
5. System validates inputs and ensures security policies (e.g., password strength).
6. Customer confirms changes.
7. System updates settings securely and notifies customer.

Subflows:

- [Change Password] Update password according to policy.
- [Update 2FA] Enable or disable two-factor authentication.
- [Change Security Questions] Modify recovery questions.

Alternative Flows:

- [Incorrect Format] System highlights invalid input and requests correction.

Reflection Between DeepSeek R1-7b and Gemma 3 4b:

We decided to put DeepSeek R1-7b and Gemma 3 4b to the test to see how well they could handle building structured use cases from our instructions and reference materials. Both models were given a framework and a list of sources to guide them. What we noticed was that Gemma 3 4b often missed some of the details and usually relied on just one reference per case. DeepSeek R1-7b, on the other hand, was much more thorough, it pulled from multiple sources and kept the use cases consistent and accurate.

We also looked at how zero-shot prompting compared to using carefully structured prompts. With zero-shot, both models tended to be a bit too generic. They didn’t always include all the flows and subflows, mostly because they didn’t have examples or enough context to go on. DeepSeek handled zero-shot better than Gemma, but even it needed some editing. When we provided clear, detailed prompts with examples, the results were way stronger. DeepSeek produced almost ready-to-use use cases, and even Gemma’s outputs were closer to what we wanted, though still needing a bit of cleanup.

Overall, it really showed that giving models context and examples upfront makes a huge difference. Without that, even a powerful model can miss important details.

Total Cost of LLM Usage:

Total Cost: $0. We hosted the LLMs on our own machines using Ollama's framework. This was very convenient, as our hardware could handle the models without affecting performance. It also saved us money since we didn't need to pay for any API keys, which would have been costly and subject to usage limits.