Proj 1c1
Minimum Viable Product (MVP) Proposal

Aniruddh Sanjeev Bhagwat Hardik Majhetia Pranshav Gajjar
Ishan Patel

10 New Use Cases for the MVP Design

The original document lacks some fundamental use cases needed to create a functional product. These 10 use cases define the core MVP experience.

UC-MVP-1: User Registration and Profile Management

- **Preconditions:** A user (Customer, Restaurant, or Rider) has downloaded the app or accessed the website.
- **Main Flow:**
  - The user selects their role (Customer, Restaurant, or Rider).
  - The user provides basic information (name, email, password).
  - The system creates a basic profile.
  - The user can later edit their profile information.
- **Subflows:**
  - [**Edit Profile**]:

    Users can update their name, contact info, and password. Customers can add delivery addresses.
- **Alternative/Error Flows:**
  - If the email is already in use, the system prompts the user to log in or use a different email.

UC-MVP-2: Restaurant Menu Creation and Management

- **Preconditions:** A user has a verified Restaurant account.
- **Main Flow:**
  - The Restaurant owner navigates to the menu management section.
  - The owner adds a new menu item, providing a name, description, and price.
  - The owner uploads a photo for the item.
  - The owner saves the item, making it visible to customers.
- **Subflows:**
  - [**Add Item Options**]:

    The owner can add customization options (e.g., size, toppings) with corresponding price adjustments.

    [**Mark as Unavailable**]:

    The owner can temporarily hide an item that is out of stock.
- **Alternative/Error Flows:**
  - If required fields (name, price) are missing, the system prevents the item from being saved and highlights the missing information.

UC-MVP-3: Customer Browses Menus and Creates Order

- **Preconditions:** A customer is logged into their account.
- **Main Flow:**
  - The customer searches for restaurants by name or cuisine type.
  - The customer selects a restaurant and views its menu.
  - The customer adds items to their cart.
  - The customer proceeds to checkout.
- **Subflows:**
  - **[Customize Item]:**

    The customer selects from the options provided for a menu item (e.g., ”extra cheese”).
- **Alternative/Error Flows:**
  - If a restaurant is closed, its menu is grayed out and items cannot be added to the cart.

UC-MVP-4: Process Customer Payment

- **Preconditions:** A customer has items in their cart and has proceeded to checkout.
- **Main Flow:**
  - The system displays the order summary, including subtotal, taxes, and fees.
  - The customer enters their credit card information.
  - The system sends the payment information to a third-party payment gateway.
  - Upon successful payment confirmation, the order is finalized.
- **Subflows:**
  - **[Save Payment Method]:**

    The customer can choose to save their card details for future orders.
- **Alternative/Error Flows:**
  - If the payment is declined, the system displays an error message and prompts the customer to use a different card or check their information.

UC-MVP-5: Restaurant Accepts and Prepares Order

- **Preconditions:** A customer has successfully placed and paid for an order.
- **Main Flow:**
  - The system sends the new order to the restaurant’s terminal/app.
  - The restaurant staff reviews the order and confirms they can fulfill it.
  - The system updates the order status to ”Preparing.”
  - Once the food is packaged, the restaurant marks the order as ”Ready for Pickup.”
- **Subflows:**
  - **[Review Order Details]:**

    Staff can see all items, special instructions, and allergen information.
- **Alternative/Error Flows:**
  - If the restaurant cannot fulfill an order (e.g., an item is suddenly out of stock), they can reject the order. The system then automatically notifies the customer and processes a full refund.

UC-MVP-6: System Assigns Rider to Order

- **Preconditions:** A restaurant has marked an order as "Ready for Pickup."
- **Main Flow:**
  - The system broadcasts the delivery job to nearby, available riders.
  - A rider accepts the job.
  - The system assigns the order to that rider and provides them with restaurant and customer details.
- **Subflows:**
  - [**View Job Details**]:

    Before accepting, a rider can see the restaurant, the customer's general location, and their potential earnings.
- **Alternative/Error Flows:**
  - If no riders are available within a certain timeframe, the system may alert customer support for manual intervention or notify the customer of a potential delay.

UC-MVP-7: Real-Time Order Tracking

- **Preconditions:** An order has been assigned to a rider.
- **Main Flow:**
  - The customer's app displays the current status of the order (e.g., Preparing, In Transit, Delivered).
  - Once the rider picks up the order, the app displays the rider's location on a map in near real-time.
  - The customer receives notifications at key stages (e.g., "Your rider is on the way").
- **Subflows:**
  - [**Contact Rider**]:

    The customer can call or message the rider through a masked number in the app.
- **Alternative/Error Flows:**
  - If the rider's GPS signal is lost, the map displays their last known location and a "Signal Lost" message.

UC-MVP-8: Rate and Review Order

- **Preconditions:** An order has been marked as "Complete."
- **Main Flow:**
  - The system prompts the customer to rate their experience with the food and the delivery.
  - The customer provides a star rating (1-5) for both the restaurant and the rider.
  - The customer can optionally leave a written review.
  - The system saves the feedback.
- **Subflows:**
  - [**Review an Old Order**]:

    Customers can find past orders in their history and add or edit a review within a specific timeframe (e.g., 7 days).
- **Alternative/Error Flows:**
  - The customer can choose to dismiss the rating prompt and not leave a review.

UC-MVP-9: Rider Onboarding and Verification

- **Preconditions:** A user has registered for a Rider account.
- **Main Flow:**
  - The system prompts the rider to complete their profile by uploading required documents (e.g., driver's license, vehicle registration).
  - The rider submits their tax information (UC-17).
  - A platform employee manually reviews the documents for validity.
  - Upon approval, the system activates the rider's account, allowing them to receive delivery jobs.
- **Subflows:**
  - **[Background Check]:**

    The system integrates with a third-party service to initiate a criminal background check after the rider gives consent.
- **Alternative/Error Flows:**
  - If documents are expired or invalid, the system notifies the rider and keeps the account pending until valid documents are provided. The account is rejected if the background check fails.

UC-MVP-10: Manage Payouts to Restaurants and Riders

- **Preconditions:** A Restaurant or Rider has completed orders and has valid tax/banking information on file.
- **Main Flow:**
  - The system calculates the net earnings for the user over a defined period (e.g., weekly).
  - The calculation subtracts platform commissions and fees from the total sales/delivery fees.
  - The system initiates a direct deposit to the user's linked bank account.
  - The system generates a basic earnings statement for the user.
- **Subflows:**
  - **[View Earnings Dashboard]:**

    Users can see a summary of their earnings, a list of transactions, and upcoming payout dates.
- **Alternative/Error Flows:**
  - If a bank transfer fails, the system puts the payout on hold and notifies the user to verify their banking information.

Strategic Decisions and Stakeholder Impact

How did you decide what NOT to do?

The decision to exclude features was driven by the core principles of an MVP: focus, speed, and learning. I prioritized the single most critical path to value: a customer ordering, a restaurant cooking, and a rider delivering. Anything not directly on this "happy path" or required for basic legal/safety operation was deferred. This strategy reduces initial engineering complexity, minimizes operational overhead (e.g., no need for a large compliance team at launch), and allows the business to get to market faster to start learning from real users before investing in more advanced, and potentially unused, features.

What negative impacts or disappointments could this MVP have for the stakeholders?

- **Customers:** May be disappointed by the lack of features they see on competing apps, such as detailed nutritional information, charity donations, or special meal programs. The absence of a visible, robust health compliance badge system (UC-3) might make some users hesitant to trust new restaurants on the platform.
- **Restaurants:** Might feel the platform is too basic. They lack advanced promotional tools (UC-6), sophisticated menu management (e.g., ingredient-level disclaimers from UC-28), or automated support for complex tax situations, which could create administrative burdens for them.
- **Riders:** May feel anxious about the lack of automated tax reporting (UC-12), as this places the burden of tracking and filing squarely on them. They also miss out on route optimizations for perishable goods (UC-27) that could affect their efficiency and customer ratings.
- **The Business:** By deferring robust compliance and audit features (UC-3, UC-21), the business assumes more risk. A single major safety incident or a failure to meet a niche regulatory requirement could have severe reputational and legal consequences.

What changes did you make (and why) to the MVP to appease at least some of the stakeholders?

To mitigate the most significant risks and disappointments, I would make three strategic additions to the MVP, creating a slightly more robust "MVP 1.5":

1. **Add a simplified UC-3: Manage Restaurant Health and Safety Compliance.** Why: Trust is paramount. Even without a full audit team, requiring restaurants to upload a valid food safety certificate during onboarding is a low-cost way to establish a minimum safety standard. This appeases customers by showing a commitment to safety and helps the business mitigate some risk.
2. **Add UC-5: Manage Nutritional Information Display.** Why: This feature provides immediate, tangible value to a large segment of customers and is becoming a legal requirement in many jurisdictions. Adding a simple "calories" field to the menu management tool (UC-MVP-2) is a relatively small engineering task that significantly enhances the user experience.
3. **Add UC-12: Report Earnings to Tax Authorities (Rider).** Why: This is a major point of friction and anxiety for gig workers. Automating the generation of tax forms (like the 1099-NEC) is a powerful retention tool for riders. It also ensures the business meets its legal reporting obligations accurately and on time, which is far safer than relying on a manual year-end process.

Part 3: Prompt History

We tried out three mail LLMs: the Gemini-2.5-Pro, DeepSeek, and ChatGPT5. The first two were used extensively for our *1b1* submission, along with a few representative prompt histories, which yielded good results and provided auxiliary use cases, as mentioned below:

- DeepSeek output: `https://docs.google.com/document/d/1uP_y_yCkt9EIGGaFgmE5ZGRYp-2RWOhjfIgPkuMSlfc/` `edit?usp=sharing`
- DeepSeek prompt examples: `https://docs.google.com/document/d/1dQ0CsVFdm08s0hLrIU7CdnodvxWOXihW4mx` `edit?usp=sharing`
- Gemini Pro 2.5: `https://docs.google.com/document/d/10BaFoPejinAix3NrdD8pOV1csn7OAhX5pUfF9NKPlkM/` `edit?usp=sharing`
- ChatGPT: `https://docs.google.com/document/d/1rjZVYys0EroeFJu8c1OByOm5th4Ob4TeM5y29Y6NqyE/` `edit?usp=sharing`