Tab 9

Pen & Pencil draft

Home kitchen cooked & ready to cook

Home kitchen cooked & ready to cook

target audience draft

household women

Working women

Hygiene conscious individuals

young entrepreneurs

Middle Class

Canteens/ Hospitals

Stake holders draft

.individuals with culinary skills

. couriers/ dispatch riders / delivery service providers

-packaging suppliers (for uniform standards)

.grocessorers

-value addition services- cold/soft drinks

-event management professionals/vendors

.Banks /Offices/ Commercial outlets

.Payment Partners, banks, online wallets

Determine Functions: draft

Commercial kitchen with hygienic quality home cooked food with traditional dishes in affordable prices with timely delivery and convenience by developing minimum viable product MVP.

Market Research: draft

1. Competitors: (what value they provide to consumers)

-Food Panda

-Uber Eat

1. Demography

-Target Audience

-Stakeholders

1. What are the favorite/expected features available in current apps

Application Development Process:

1. Define Roles:

-Manage : application development cycle (tasks/tracking/cost & time estimates)

-Understand: business Market /User requirements

-Desgin: UI/UX interfaces ( with workflow, transactions flow and input fields description)

-Develop and Plan: Application structure with Stakes of NEXTJS, SHADCN, FIGMA, Shopno)

-Test and Trial: the application at cross plateforms

1. Define Scope: draft

Business Model = Home Kitchen ⇒ Delivery Service Providers ⇒ Satisfied Customers

1. Application Features: draft
2. Customers:

-Easy Signup/Login

-Item Cart [Add/update/details]

-Search [Cuisines, Categories, Locations, Vendors]

-Order [Placement, Tracking, Cancellation, History/rep]

-Billing /Payments [Cash on Delivery, online banks & wallets, refunds]

1. Kitchens/individual culinary experts

-Singup/Login

-Update Menu items/listings [details with images, prices, and estimated times]

-Delivery Management

1. Delivery Service Providers

-Singup

-Notifications

-New order with Pickup/drop location addresses

-Push Messages

-Options to set availability

-Navigation features

-Delivery process summary

-Payment details

-Payments

1. Application Platform Administration:

-Customers /Vendors

-Orders

-Tracking

-Cancellations

-Billing

-Payments

-Reviews / Ratings

Business Model Canvas

Business Model Canvas Prep for: JAM2025 Prep by: Naimat Ullah Khan

| Key Partners  - Home cooks and culinary experts  - Delivery service providers (couriers/riders)  - Packaging suppliers  - Grocers and ingredient suppliers  - Payment partners (banks, online wallets)  - Cold/soft drink vendors  - Event management professionals  - Commercial outlets (banks, offices, hospitals) | Key Activities  - Develop an online platform for listing and ordering  - Quality control for home-cooked food  - Streamlined delivery management  - Customer feedback and reviews  - Partnership management with stakeholders  - Marketing and customer acquisition campaigns | Value Propositions  - Affordable, hygienic, and authentic home-cooked meals  - Support for household women to earn income  - Convenient food delivery for working-class individuals  - Focus on health-conscious and traditional dishes  - Transparent reviews and ratings  - Easy-to-use platform for customers and vendors | Customer Relationships  - Direct interaction via the app (chat, notifications)  - Reward programs for repeat customers  - Feedback-driven improvements  - Personalized offers for loyal users | Customer Segments  - Working-class professionals  - Health-conscious individuals  - Middle-class families  - Hospital canteens and small event catering  - Young entrepreneurs looking to sell homemade food |
| --- | --- | --- | --- | --- |
| Key Resources  - Culinary experts and kitchens  - Delivery fleet or logistics partners  - Packaging materials  - Technology stack (Next.js, ShadCN, Figma, Shipo)  - Marketing resources  - Payment processing systems | Channels  - Mobile app and web app  - Social media marketing (Instagram, Facebook, TikTok)  - Partnerships with local businesses and organizations  - Word-of-mouth referrals from satisfied customers |
| Cost Structure  - Platform development and maintenance  - Marketing and advertising expenses  - Packaging and branding costs  - Delivery service fees  - Quality assurance and training for cooks  - Payment gateway fees | | Revenue Streams  - Commission on every order (from kitchen partners)  - Delivery charges paid by customers  - Subscription plans for premium features (e.g., priority delivery)  - Advertising for related products/services (e.g., drinks, event vendors) | | |

Business Model Canvas: Homecooked

Key Partners

- Home cooks and culinary experts

- Delivery service providers (couriers/riders)

- Packaging suppliers

- Grocers and ingredient suppliers

- Payment partners (banks, online wallets)

- Cold/soft drink vendors

- Event management professionals

- Commercial outlets (banks, offices, hospitals)

Key Activities

- Develop an online platform for listing and ordering

- Quality control for home-cooked food

- Streamlined delivery management

- Customer feedback and reviews

- Partnership management with stakeholders

- Marketing and customer acquisition campaigns

Key Resources

- Culinary experts and kitchens

- Delivery fleet or logistics partners

- Packaging materials

- Technology stack (Next.js, ShadCN, Figma, Shipo)

- Marketing resources

- Payment processing systems

Value Propositions

- Affordable, hygienic, and authentic home-cooked meals

- Support for household women to earn income

- Convenient food delivery for working-class individuals

- Focus on health-conscious and traditional dishes

- Transparent reviews and ratings

- Easy-to-use platform for customers and vendors

Customer Relationships

- Direct interaction via the app (chat, notifications)

- Reward programs for repeat customers

- Feedback-driven improvements

- Personalized offers for loyal users

Channels

- Mobile app and web app

- Social media marketing (Instagram, Facebook, TikTok)

- Partnerships with local businesses and organizations

- Word-of-mouth referrals from satisfied customers

Customer Segments

- Working-class professionals

- Health-conscious individuals

- Middle-class families

- Hospital canteens and small event catering

- Young entrepreneurs looking to sell homemade food

Cost Structure

- Platform development and maintenance

- Marketing and advertising expenses

- Packaging and branding costs

- Delivery service fees

- Quality assurance and training for cooks

- Payment gateway fees

Revenue Streams

- Commission on every order (from kitchen partners)

- Delivery charges paid by customers

- Subscription plans for premium features (e.g., priority delivery)

- Advertising for related products/services (e.g., drinks, event vendors

Workflow draft 👍

Workflow draft 👍

1, LOGIN /SIGNUP

1.1 Provide Email, contact and address

1.2 confirm registration /login credentials

1.3 Access listing based on customer details

1.4 Open Cart

2.CART Management

2.1 Search Items in listings

2.2 Select items with details of quantities to add in Cart

2.3 Repeat add to cart until final order

2.4 Confirm final cart and Proceed to Billing

3. BILLING

3.1 Choose billing options (COD, online banking, wallets)

3.2 Connect with payment module

3.3 proceed to order summary upon successful payment

3.4 repeat in case of failed transaction

4. ORDER

4.1 Order details sent to kitchen

4.2 Renegotiation with customer for any delays in order fulfilment

4.3 Kitchen confirmation with pickup time and location

4.4 Send confirm time to pickup to delivery service provider

5.Delivery Service

5.1 Confirm availability for the order

5.2 Share details of the rider with kitchen and the customer

5.3 Pick up order from kitchen

5.4 Provide tracking of the dispatch pick and drop

5.5 Confirm customer arrival of the delivery

5.6 Receive Payment / confirmation

5.7 Handover package to the customer

5.8 Proceed to order completion

6. ORDER Closer

6.1 Payment confirmation

6.2 Customer Review taken

6.3 Order closed and saved in History

Workflow & Transaction flow

**Workflow Overview**

1. Customer Workflow

1. Signup/Login: Email, contact, address verification.

2. Search Listings: Filter by cuisine, category, location.

3. Add to Cart: Add items with quantity and price details.

4. Order Placement: Confirm order, choose billing method.

5. Payment Processing: COD, online wallets, bank transfers.

6. Order Tracking: Real-time updates for preparation and delivery.

7. Review/Rating: Post-delivery feedback and rating.

2. Vendor Workflow (Kitchen)

1. Signup/Login: Basic info, menu creation, availability.

2. Menu Updates: Manage item details (prices, availability).

3. Order Management: Accept/reject orders, negotiate delays.

4. Handover: Pack orders for delivery with proper labeling.

3. Delivery Workflow

1. Rider Signup: Availability, location settings.

2. Order Pickup: Details of kitchen, estimated delivery time.

3. Delivery Tracking: Real-time location updates for customers.

4. Payment Collection: COD or delivery confirmation.

5. Completion: Confirm drop-off, close the order.

**Customer Workflow**

1. Signup/Login

- Action by Customer:

- Enter name, email, phone number, and address.

- Create a password or sign in via Google/Facebook.

- System Response:

- Validate input fields.

- Send email or SMS verification for registration.

- Allow login and redirect to the home page.

- Input Fields:

- Name, email, phone number, address, password.

2. Browse/Search Listings

- Action by Customer:

- Search for dishes, cuisines, or vendors using filters.

- Apply filters (e.g., dietary preferences, price range, location).

- View details of dishes (e.g., name, price, estimated delivery time, ingredients, reviews).

- System Response:

- Fetch and display search results from the database.

- Dynamically update UI based on filters.

- Input Fields:

- Search bar, cuisine/category dropdown, location.

3. Add to Cart

- Action by Customer:

- Select a dish and specify the quantity.

- View dish details (ingredients, price per item, delivery time).

- Add selected items to the cart.

- System Response:

- Update cart with selected items and quantities.

- Display total price and estimated delivery time.

- Input Fields:

- Dish ID, quantity.

4. View Cart and Confirm Order

- Action by Customer:

- Open the cart to review selected items.

- Modify quantities or remove items if needed.

- Confirm the final selection and proceed to billing.

- System Response:

- Recalculate total price and update delivery estimate.

- Input Fields:

- Cart items, modifications (add/remove/update).

5. Billing/Payment

- Action by Customer:

- Select a payment method (Cash on Delivery, online banking, or wallets).

- Enter payment details if using online methods.

- Confirm payment and proceed to place the order.

- System Response:

- Validate payment.

- Generate a payment receipt and update order status.

- Handle failed transactions with retry options.

- Input Fields:

- Payment method, card/wallet details (if applicable).

6. Order Placement

- Action by Customer:

- Review the final order summary (items, total cost, estimated delivery time).

- Confirm order placement.

- System Response:

- Save the order in the database.

- Notify the kitchen partner of the new order.

- Display the order tracking page.

- Input Fields:

- Order confirmation.

7. Order Tracking

- Action by Customer:

- Monitor the order's preparation and delivery in real-time.

- Receive notifications for each stage (order accepted, in preparation, out for delivery, delivered).

- System Response:

- Update order status dynamically.

- Provide estimated delivery time and rider location.

- Input Fields:

- None (real-time updates).

8. Delivery Completion

- Action by Customer:

- Confirm receipt of the order upon delivery.

- Check for issues (e.g., incorrect items, missing items).

- System Response:

- Close the order as delivered.

- Initiate payment confirmation (for COD).

- Input Fields:

- Delivery confirmation (button or prompt).

9. Post-Delivery Review

- Action by Customer:

- Rate the order experience (1-5 stars).

- Provide feedback on food quality, delivery time, and service.

- System Response:

- Save review data.

- Display ratings for the respective vendor.

- Input Fields:

- Rating scale, text feedback.

Customer Workflow Diagram

1. Login/Signup → 2. Search Listings → 3. Add to Cart → 4. Confirm Order →

5. Billing/Payment → 6. Order Placement → 7. Order Tracking → 8. Delivery Completion → 9. Post-Delivery Review

**Vendor Workflow (Kitchen)**

1. Registration & Login

1. Signup :

- Vendors provide basic information:

- Name, contact details, address.

- Kitchen name and logo (optional).

- Necessary licenses or certifications (if required).

- Upload profile picture and description of their kitchen (optional).

2. Login :

- Vendors can log in using their email and password.

2. Profile Setup

1. Kitchen Profile Details :

- Add kitchen details (address, operational hours, cuisine specialty).

- Configure delivery radius (in km/miles).

- Set up bank account or payment wallet for payouts.

2. Menu Creation :

- Add dishes to the menu with:

- Dish name, description, category (e.g., Main Course, Dessert).

- Price (editable).

- Images of dishes.

- Estimated preparation time.

- Set availability for each item (toggle on/off).

3. Order Management

1. Receive New Orders :

- Notifications for new orders with details:

- Dish name, quantity, special instructions (if any).

- Expected delivery or pickup time.

- Vendors can choose to:

- Accept the order (default).

- Reject the order (if unavailable).

- Notify the customer in case of delays.

2. Order Preparation :

- Begin preparing accepted orders.

- Update order status: "Preparing" → "Ready for Pickup."

3. Packaging :

- Package orders with:

- Proper labeling (customer name, order ID, dish details).

- Sealed packaging to ensure hygiene.

4. Communication with Delivery Service

1. Confirm Pickup Time :

- Notify the delivery service of the ready order.

- Share pickup location details.

2. Handover to Delivery Rider :

- Handover the order with confirmation via the app.

- Update status to "Order Picked."

5. Financials & Payments

1. Earnings Dashboard :

- View total earnings, pending payouts, and completed transactions.

2. Payouts :

- Weekly or monthly payouts directly to the vendor’s registered account.

- Automatic reconciliation for platform commissions.

6. Feedback & Reviews

1. Customer Feedback :

- View ratings and reviews for completed orders.

- Respond to customer reviews (optional).

2. Performance Dashboard :

- Access metrics such as:

- Total orders completed.

- Average preparation time.

- Customer satisfaction score.

7. Notifications

- Vendors receive notifications for:

- New orders.

- Order status changes.

- Customer queries (if any).

- Payments or payout updates.

Vendor Workflow Diagram

1. Vendor Signs Up → 2. Profile Setup → 3. Adds Menu Items → 4. Receives Order Notification → 5. Accepts/Rejects Order → 6. Prepares Order → 7. Notifies Delivery Partner → 8. Hands Over Order → 9. Tracks Order Status → 10. Receives Payment → 11. Reviews Customer Feedback.

**Delivery Service Workflow**

Step 1: Signup/Login

- Input Fields:

- Name

- Contact number

- Email address

- Vehicle details (e.g., bike, car, etc.)

- Service availability (toggle online/offline)

- Process:

- Delivery riders create an account or log in.

- Update availability status (Online/Offline).

Step 2: Receive Order Notification

- Trigger:

- New order placed by a customer.

- Actions:

- Rider receives a push notification with the following details:

- Pickup location (kitchen address)

- Drop-off location (customer address)

- Estimated distance/time

- Payment method (COD/Online)

- Option:

- Rider accepts or rejects the order.

Step 3: Pickup

- Actions:

- Navigate to the pickup location using an in-app map.

- Mark "Arrived at Pickup" once at the kitchen.

- Collect the order package.

- Confirm pickup by scanning a QR code or marking it in the app.

Step 4: Delivery

- Actions:

- Navigate to the drop-off location using in-app navigation.

- Notify the customer upon arrival through in-app notification or call.

- Hand over the package to the customer.

- For COD orders :

- Collect payment.

- Update payment received in the app.

Step 5: Confirm Completion

- Actions:

- Mark the order as "Delivered" in the app.

- Provide a brief delivery summary, such as:

- Time taken

- Delivery success or failure reason (if applicable)

- Post-Delivery:

- A push notification is sent to the customer for feedback.

Step 6: Payment Settlement

- Actions:

- COD payments:

- Rider deposits collected cash at a designated point or via mobile wallet.

- Online payments:

- System automatically updates the rider’s earnings.

- Earnings Dashboard:

- Riders can view their earnings for each order, along with total

weekly/monthly payments.

System Flow Overview

1. Order Assignment:

- The system checks available riders near the pickup location.

- Assigns the order to the nearest available rider.

2. Notifications:

- Riders receive notifications for:

- New orders

- Customer location changes

- Delivery confirmations

3. Tracking:

- Customers and kitchens can track the rider’s real-time location during pickup and delivery.

4. Dispute Handling:

- If a delivery fails, riders can report the issue (e.g., customer unavailable or wrong address).

Input Fields

**Input Fields**

1.Customer

- Name, email, phone number, address

- Search: Cuisine type, location, keywords

- Order details: Item, quantity, preferences

- Payment details: Card info, wallet, COD preference

2. Kitchen/Vendor

- Profile: [Name, address, kitchen license (if needed)]

- Menu details: [Dish name, description, price, image]

- Order management: [Accept/reject, availability updates]

3. Delivery Rider

- Name, contact, vehicle details

- Availability toggle

- Notifications: New orders, navigation, delivery summary

**Express.js Implementation:**

| Express.js Routes for Workflow Implementation | const express = require('express');  const router = express.Router(); |
| --- | --- |
| Endpoint: /customers (GET) | router.get('/customers', async (req, res) => {  try {  const customers = await Customer.find(); res.json(customers);  } catch (error) {  res.status(500).json({ error: 'Failed to fetch customers' });  }  }); |
| Endpoint: /vendors (GET) | router.get('/vendors', async (req, res) => {  try {  const vendors = await Vendor.find(); ]  res.json(vendors);  } catch (error) {  res.status(500).json({ error: 'Failed to fetch vendors' });  }  }); |
| Endpoint: /express-delivery-status (GET) | router.get('/express-delivery-status', async (req, res) => {  try {  const { orderId } = req.query;  const order = await Order.findOne({ id: orderId });  if (!order) return res.status(404).json({ error: 'Order not found' });  res.json({ orderId: order.id, status: order.status });  } catch (error) {  res.status(500).json({ error: 'Failed to fetch delivery status' });  }  });  module.exports = router; |
| Endpoint: /delivery-riders (GET) | router.get('/delivery-riders', async (req, res) => {  try {  const riders = await Rider.find(); res.json(riders);  } catch (error) {  res.status(500).json({ error: 'Failed to fetch delivery riders' });  }  }); |

Technology Stack

**Technology Stack**

Frontend:

- Next.js: Building dynamic and SEO-friendly pages.

- ShadCN: UI components with accessibility and consistency.

- Figma: Designing intuitive user interfaces.

Backend:

- Shopno: Simplified backend structure for commerce apps.

- Prisma ORM: Database schema management for Next.js.

Database Schema

1. Users

- `id`: Unique identifier

- `name`, `email`, `phone`, `address`

- `role`: Customer/Kitchen/Delivery

- `createdAt`, `updatedAt`

2. Orders

- `id`, `userId`, `kitchenId`, `deliveryId`

- `status`: Pending/In Progress/Delivered

- `items`: Array of dish IDs

- `totalPrice`, `paymentStatus`

- `createdAt`, `updatedAt`

3. Menu

- `id`, `kitchenId`

- `dishName`, `description`, `price`, `imageURL`

- `availability`: Boolean

4. Delivery

- `id`, `orderId`

- `riderId`, `pickupLocation`, `dropLocation`

- `status`: Assigned/In Transit/Delivered

- `createdAt`, updatedAt

Data Schema for Delivery Service

1. Rider Table

- `id`: Unique identifier

- `name`, `phone`, `vehicleType`

- `status`: Online/Offline

- `totalDeliveries`, `earnings`

2. Delivery Table

- `id`: Unique identifier

- `orderId`: Associated order ID

- `riderId`: Assigned rider

- `pickupLocation`, `dropLocation`

- `status`: Pending/In Transit/Delivered/Failed

- `paymentStatus`: COD/Online

- `createdAt`, `updatedAt`

3. Notifications Table

- `id`: Unique identifier

- `riderId`: Notification recipient

- `message`: Content of the notification

- `type`: Order/New Update/Reminder

- `readStatus`: Read/Unread

- `createdAt`

Sanity Schema:

Sanity Schema:

| // Sanity Schema for Customers  default export {  name: 'customer',  type: 'document',  fields: [  { name: 'name', type: 'string', title: 'Customer Name' },  { name: 'email', type: 'string', title: 'Email' },  { name: 'phone', type: 'string', title: 'Phone Number' },  { name: 'address', type: 'string', title: 'Address' },  { name: 'createdAt', type: 'datetime', title: 'Created At' },  { name: 'updatedAt', type: 'datetime', title: 'Updated At' }  ]  }; |
| --- |
| // Sanity Schema for Vendors  default export {  name: 'vendor',  type: 'document',  fields: [  { name: 'name', type: 'string', title: 'Vendor Name' },  { name: 'email', type: 'string', title: 'Email' },  { name: 'phone', type: 'string', title: 'Phone Number' },  { name: 'kitchenAddress', type: 'string', title: 'Kitchen Address' },  { name: 'menu', type: 'array', of: [{ type: 'reference', to: [{ type: 'product' }] }], title: 'Menu' },  { name: 'createdAt', type: 'datetime', title: 'Created At' },  { name: 'updatedAt', type: 'datetime', title: 'Updated At' }  ]  }; |
| // Sanity Schema for Delivery Riders  default export {  name: 'rider',  type: 'document',  fields: [  { name: 'name', type: 'string', title: 'Rider Name' },  { name: 'phone', type: 'string', title: 'Phone Number' },  { name: 'vehicleType', type: 'string', title: 'Vehicle Type' },  { name: 'status', type: 'string', title: 'Availability Status', options: { list: ['Online', 'Offline'] } },  { name: 'createdAt', type: 'datetime', title: 'Created At' },  { name: 'updatedAt', type: 'datetime', title: 'Updated At' }  ]  }; |

API Endpoints

API Endpoints

| Endpoint | Method | Purpose | Response code |
| --- | --- | --- | --- |
| /customers | GET | Fetches all customer details | { "id": 1, "name": "John Doe", "email":"john@ex.com" } |
| /customer/:id | GET | Fetches details of a specific customer | "id": 1, "name": "John Doe", "email":"john@ex.com" } |
| /customers/create | POST | Creates a new customer | { "success": true, "customerId": 1 } |
| /vendors | GET | Fetches all vendor details | { "id": 1, "name": "Vendor A", "menu": [ ... ] } |
| /vendor/:id | GET | Fetches details of a specific vendor | { "id": 1, "name": "Vendor A", "menu": [ ... ] } |
| /vendors/create | POST | Creates a new vendor | { "success": true, "vendorId": 1 } |
| /delivery-riders | GET | Fetches all delivery rider details | { "id": 1, "name": "Rider A", "status": "Online" } |
| /delivery-rider/:id | GET | Fetches details of a specific delivery rider | "id": 1, "name": "Rider A", "status": "Online" } |
| /delivery-riders/create | POST | Creates a new delivery rider | { "success": true, "riderId": 1 } |
| /orders | GET | Fetches all orders | [ { "id": 1, "status": "In Progress" } ] |
| /order/:id | GET | Fetches details of a specific order | { "id": 1, "status": "In Progress" } |
| /orders/create | POST | Creates a new order | { "success": true, "orderId": 123 } |
| /express-delivery-status | GET | Fetches real-time tracking of an order | { "orderId": 123, "status": "In Transit" } |