Project Roadmap

Restaurant OS - Phase 1 MVP

Version: 1.0
Last Updated: December 2024
Team Size: 5 (AI-Assisted Development)
Target: MVP Launch in 6-7 Months
Budget: Bootstrapped (₹15,000/month operational)

Table of Contents

---

1. Team Structure

1.1 Role Assignments

Person	Primary Role	Secondary Responsibilities
Person A	Product Lead + UI/UX	User testing, docs writing, customer research
Person B	Backend Lead	API design, database, deployment
Person C	Admin Frontend Lead	Admin dashboard, state management
Person D	Customer Frontend Lead	QR ordering app, marketing site
Person E	Full-Stack Support	Bug fixes, testing, DevOps, integration

1.2 Working Agreements

Daily Standup (15 min):

- What did you complete yesterday?
- What will you work on today?
- Any blockers?

Code Reviews:

- All PRs require 1 approval
- Use AI to help review code quality
- Check multi-tenancy isolation

Communication:

- Slack/WhatsApp for quick questions
- Daily standup for sync
- Weekly planning on Sundays

AI Usage:

- Document all prompts used
- Share successful patterns
- Help each other with debugging

---

2. Timeline Overview

2.1 High-Level Milestones

```
Month 1-2: Foundation + Core Features
├─ Sprint 1: Setup & Auth (2 weeks)
├─ Sprint 2: Menu Management (2 weeks)
└─ Sprint 3: Tables & QR (2 weeks)

Month 3-4: Customer Experience
├─ Sprint 4: Customer Ordering (2 weeks)
├─ Sprint 5: Orders Dashboard (2 weeks)
└─ Sprint 6: Real-time + Polish (2 weeks)

Month 5: Testing & Marketing
├─ Sprint 7: Testing & Bug Fixes (2 weeks)
└─ Sprint 8: Marketing Site + Prep (2 weeks)

Month 6: Alpha Testing
├─ Sprint 9: Alpha with 3 restaurants (2 weeks)
└─ Sprint 10: Iteration based on feedback (2 weeks)

Month 7: Beta Launch
```

```
├─ Sprint 11: Beta with 10 restaurants (2 weeks)
└─ Sprint 12: Final polish + Public launch (2 weeks)
```

2.2 Phase Breakdown

Phase	Duration	Key Deliverable	Success Criteria
Foundation	Weeks 1-6	Core features built	Menu + Tables working
Integration	Weeks 7-12	End-to-end flow	Customer can order, admin sees it
Testing	Weeks 13-16	Bug-free product	< 5 critical bugs
Alpha	Weeks 17-20	3 paying customers	90%+ satisfaction
Beta	Weeks 21-24	10 paying customers	95%+ retention
Launch	Weeks 25-28	20+ customers	Sustainable operations

---

3. Sprint-by-Sprint Breakdown

Sprint 1: Foundation & Authentication

Duration: Weeks 1-2
Goal: Setup infrastructure + user authentication

Week 1: Environment Setup

Person A (Product Lead):

- [ ] Create Figma workspace
- [ ] Design wireframes for login/register
- [ ] Define brand colors, fonts, logo
- [ ] Write copy for onboarding screens
- [ ] Create user flow diagrams

Person B (Backend Lead):

- [ ] Setup GitHub repo + branch strategy
- [ ] Setup PostgreSQL locally
- [ ] Create database schema (restaurants, users tables)
- [ ] Setup Express.js project structure
- [ ] Configure environment variables

Person C (Admin Frontend):

- [ ] Setup React + Vite project (admin)

- [ ] Install dependencies (React Router, TailwindCSS, React Query)
- [ ] Setup folder structure
- [ ] Create base layout components (Header, Sidebar)

Person D (Customer Frontend):

- [ ] Setup React + Vite project (customer)
- [ ] Install dependencies
- [ ] Setup folder structure
- [ ] Create mobile-first base layout

Person E (Full-Stack Support):

- [ ] Document setup process
- [ ] Help team with environment issues
- [ ] Create Docker Compose file (optional)
- [ ] Setup Git workflow guide

Deliverable: All team members have working dev environment

Week 2: Authentication Implementation

Person A:

- [ ] Finalize login/register UI designs
- [ ] Design onboarding flow (3 steps)
- [ ] Create style guide document
- [ ] Test auth flows on mobile

Person B:

- [ ] Build auth endpoints:
  - POST /auth/register
  - POST /auth/login
  - POST /auth/refresh
- [ ] Implement JWT generation
- [ ] Implement bcrypt password hashing
- [ ] Write auth middleware
- [ ] Test all endpoints with Thunder Client

Person C:

- [ ] Build Login page

- [ ] Build Register page
- [ ] Build Onboarding wizard (restaurant setup)
- [ ] Implement form validation (React Hook Form)
- [ ] Connect to backend API
- [ ] Setup auth store (Zustand)
- [ ] Implement protected routes

Person D:

- [ ] Not actively coding this sprint
- [ ] Learn React Query basics
- [ ] Review customer app requirements
- [ ] Start designing menu browsing UI (Figma)

Person E:

- [ ] Write integration tests for auth API
- [ ] Create Postman collection for all endpoints
- [ ] Help with CORS issues
- [ ] Document API response formats

Sprint 1 Demo:

- User can register a restaurant
- User can login
- Protected dashboard route works
- JWT token persists on refresh

Sprint 2: Menu Management

Duration: Weeks 3-4
Goal: Restaurant can create and manage their menu

Week 3: Categories & Items (Basic)

Person A:

- [ ] Design category manager UI
- [ ] Design item form (add/edit)
- [ ] Design item list/grid view
- [ ] User testing with 2 local cafes

Person B:

- [ ] Build database schema for menu_categories, menu_items
- [ ] Build API endpoints:
  - GET/POST /categories
  - GET/PUT/DELETE /categories/:id
  - GET/POST /items
  - GET/PUT/DELETE /items/:id
- [ ] Implement multi-tenancy filtering
- [ ] Add image upload endpoint (multer)

Person C:

- [ ] Build Categories page
  - List categories
  - Add/edit category modal
  - Delete confirmation
- [ ] Build Items page
  - Grid view with images
  - Search and filter
- [ ] Build Item form
  - Basic fields (name, description, price)
  - Image upload
  - Category selection

Person D:

- [ ] Continue learning React patterns
- [ ] Design customer menu view (mobile)
- [ ] Create reusable components (Card, Badge)

Person E:

- [ ] Setup image optimization (sharp library)
- [ ] Configure file storage structure
- [ ] Write tests for menu APIs
- [ ] Help with form validation

Week 3 Demo:

- Can create categories
- Can add items with photos

- Items display in grid
- Images upload successfully

Week 4: Variants, Addons & Advanced Features

Person A:

- [ ] Design variant selector UI
- [ ] Design addon groups interface
- [ ] Create video tutorial for menu setup
- [ ] Test with real menu data

Person B:

- [ ] Build schema for item_variants, addon_groups, addons
- [ ] Build variant endpoints:
  - POST/GET/PUT/DELETE /items/:id/variants
- [ ] Build addon endpoints:
  - POST/GET/PUT/DELETE /addon-groups
  - POST/GET/PUT/DELETE /addons
- [ ] Implement item-addon-group mapping
- [ ] Add bulk operations (reorder items)

Person C:

- [ ] Build Variants manager (inside item form)
  - Add variant groups (Size, Base, etc.)
  - Add options with price adjustments
- [ ] Build Addon Groups manager
  - Create addon groups
  - Add addons to groups
  - Link groups to items
- [ ] Implement drag-and-drop reordering
- [ ] Add availability toggle

Person D:

- [ ] Start building customer menu view
  - Category tabs
  - Item cards with images

- Veg/non-veg indicators

Person E:

- [ ] Create sample menu data seeder
- [ ] Performance testing (load 500 items)
- [ ] Optimize image loading
- [ ] Bug fixes from Week 3

Sprint 2 Demo:

- Complete menu management working
- Variants and addons functional
- Can upload 50 items in < 30 min
- Menu looks professional

Sprint 3: Tables & QR Generation

Duration: Weeks 5-6
Goal: Generate QR codes for each table

Week 5: Table Management

Person A:

- [ ] Design table manager UI
- [ ] Design QR display/download interface
- [ ] Design QR code templates for print
- [ ] Create setup guide for restaurants

Person B:

- [ ] Build schema for restaurant_tables
- [ ] Build table endpoints:
  - GET/POST /tables
  - GET/PUT/DELETE /tables/:id
- [ ] Implement QR generation (qrcode library)
- [ ] Store QR images in filesystem
- [ ] Build bulk QR download (ZIP)

Person C:

- [ ] Build Tables page

- List all tables
- Add/edit table modal
- Table sections (Indoor, Outdoor, etc.)

☐ Build QR viewer

- Display QR image
- Download single QR
- Regenerate QR option

☐ Build bulk download feature

Person D:

☐ Build table landing page for customer

- Extract restaurant + table from URL
- Display "Ordering for Table X"
- Welcome screen

Person E:

☐ Design QR print template (PDF)
☐ Implement bulk QR generation script
☐ Test QR codes on multiple devices
☐ Document QR URL structure

Week 5 Demo:

- Can create tables
- QR codes generate correctly
- QR scan opens correct restaurant+table
- Can download all QRs as ZIP

Week 6: Polish & Integration

Person A:

☐ User acceptance testing with 2 cafes
☐ Gather feedback on UI/UX
☐ Update designs based on feedback
☐ Create onboarding video

Person B:

- [ ] Add restaurant settings endpoint
  - Update logo, colors, hours
- [ ] Implement table status management
- [ ] Add analytics placeholders
- [ ] Performance optimization

Person C:

- [ ] Build Settings page
  - Restaurant info
  - Branding (logo, colors)
  - Operating hours
- [ ] Polish all existing pages
- [ ] Add loading states everywhere
- [ ] Improve error messages

Person D:

- [ ] Continue customer app
  - Menu browsing with categories
  - Item cards with photos
  - Search functionality

Person E:

- [ ] Integration testing (end-to-end)
- [ ] Setup CI/CD pipeline (GitHub Actions)
- [ ] Performance testing
- [ ] Bug fixes

Sprint 3 Demo:

- Admin can setup entire restaurant in 20 min
- QR codes work perfectly
- All pages load fast
- No critical bugs

Sprint 4: Customer Ordering Interface

Duration: Weeks 7-8
Goal: Customers can browse menu and place orders

Week 7: Menu Browsing & Cart

Person A:

- [ ] Design item detail modal
- [ ] Design cart interface
- [ ] Design order confirmation screen
- [ ] Test on multiple phone sizes

Person B:

- [ ] Build customer API endpoints:
  - GET /public/menu/:restaurantSlug/:tableToken
  - (Returns restaurant, table, full menu)
- [ ] Optimize menu API (include variants, addons)
- [ ] Add menu caching (Redis optional)

Person C:

- [ ] Help Person D with complex components
- [ ] Build reusable UI components for customer app

Person D:

- [ ] Build menu browsing:
  - Category tabs (horizontal scroll)
  - Item list with lazy loading
  - Veg/non-veg filter
  - Search functionality
- [ ] Build item detail view:
  - Full description
  - Photo zoom
  - Variant selector
  - Addon selector (single/multi-choice)
  - Quantity controls
- [ ] Build cart:
  - Cart button (floating)

- Cart summary sheet
- Edit items in cart
- Special instructions field

Person E:

- [ ] Setup customer app state management
- [ ] Implement cart persistence (localStorage)
- [ ] Mobile testing on real devices
- [ ] PWA setup (basic)

Week 7 Demo:

- Customer scans QR → sees menu
- Can browse all items
- Can select variants/addons
- Can add to cart
- Cart persists on refresh

Week 8: Order Placement

Person A:

- [ ] Design order success animation
- [ ] Design order status tracking UI
- [ ] Test full flow with users
- [ ] Gather feedback

Person B:

- [ ] Build order creation:
  - POST /public/orders
  - Validate order data
  - Calculate totals correctly
  - Store order items + addons
- [ ] Build order status endpoint:
  - GET /public/orders/:id
- [ ] Generate order numbers (ORD-20241204-001)

Person C:

- [ ] Not actively coding
- [ ] Test customer app thoroughly
- [ ] Document bugs

Person D:

- [ ] Build order confirmation screen:
  - Review order summary
  - Table confirmation
  - Place order button
- [ ] Build success screen:
  - Order received message
  - Order number
  - Estimated time
  - "Order More" button
- [ ] Build order status page:
  - Current status (Preparing, Ready, etc.)
  - Progress indicator
- [ ] Add error handling for failed orders

Person E:

- [ ] Implement order retry logic (network failures)
- [ ] Add offline detection
- [ ] Performance optimization (Lighthouse)
- [ ] Bug fixes

Sprint 4 Demo:

- Customer can place complete order
- Order appears in system
- Success screen works
- Can track order status
- Works on slow networks

Sprint 5: Orders Dashboard (Admin)

Duration: Weeks 9-10
Goal: Restaurant staff can manage orders

Week 9: Live Orders View

Person A:

- [ ] Design live orders dashboard
- [ ] Design order card layout
- [ ] Design status update controls
- [ ] Create staff training guide

Person B:

- [ ] Build order management endpoints:
  - GET /orders (with filters)
  - GET /orders/:id (detailed view)
  - PUT /orders/:id/status
  - PUT /orders/:id/cancel
- [ ] Implement order filtering (status, table, date)
- [ ] Add order statistics endpoint
- [ ] Optimize queries with indexes

Person C:

- [ ] Build Live Orders page:
  - Order cards in columns (Pending, Preparing, Ready)
  - Drag-and-drop status change (optional)
  - Real-time updates (polling for now)
- [ ] Build Order detail view:
  - Full order info
  - Line items with customizations
  - Status timeline
  - Action buttons
- [ ] Build filters:
  - By status
  - By table
  - By date

- Search by order number

Person D:

- [ ] Help with customer app polish
- [ ] Fix bugs from Sprint 4
- [ ] Start marketing site structure

Person E:

- [ ] Setup notification sounds
- [ ] Implement order alerts
- [ ] Performance testing (100 orders)
- [ ] Bug fixes

Week 9 Demo:

- Orders appear in dashboard
- Can filter and search orders
- Can update order status
- Updates are fast (< 1 second)

Week 10: Real-time + Analytics

Person A:

- [ ] Design analytics dashboard
- [ ] Design charts (bestsellers, sales)
- [ ] User testing with real orders
- [ ] Feedback iteration

Person B:

- [ ] Setup Socket.io server
- [ ] Implement WebSocket authentication
- [ ] Emit events:
  - new_order
  - order_updated
  - menu_updated
- [ ] Build analytics endpoints:
  - GET /analytics/dashboard

- GET /analytics/sales
- GET /analytics/items

Person C:

- [ ] Integrate Socket.io client
- [ ] Implement real-time order updates
- [ ] Add browser notifications
- [ ] Build Analytics page:
  - Today's summary cards
  - Bestsellers list
  - Sales chart (basic)
  - Export to CSV

Person D:

- [ ] Make customer app listen for order updates
- [ ] Update order status in real-time
- [ ] Polish animations

Person E:

- [ ] Test real-time across multiple browsers
- [ ] Implement fallback to polling (if Socket fails)
- [ ] Performance optimization
- [ ] Bug fixes

Sprint 5 Demo:

- Orders update in real-time
- Notifications work
- Analytics show correct data
- System handles 50+ concurrent users

Sprint 6: Polish & Integration

Duration: Weeks 11-12
Goal: Bug-free, polished product

Week 11: Bug Fixes & Polish

Everyone:

- [ ] Fix all P0 and P1 bugs
- [ ] Improve error messages
- [ ] Add loading states everywhere
- [ ] Improve mobile responsiveness
- [ ] Add helpful tooltips
- [ ] Write user-facing docs

Person A:

- [ ] Create help center content
- [ ] Make setup tutorial videos
- [ ] Design empty states
- [ ] Polish all UI screens

Person B:

- [ ] Performance optimization
- [ ] Security audit
- [ ] API documentation (Swagger/OpenAPI)
- [ ] Database optimization

Person C:

- [ ] Code refactoring
- [ ] Add keyboard shortcuts
- [ ] Improve accessibility
- [ ] Polish animations

Person D:

- [ ] Customer app final polish
- [ ] A/B test different layouts
- [ ] Optimize images

Person E:

- [ ] Integration testing
- [ ] Load testing
- [ ] Browser compatibility testing
- [ ] Mobile device testing

Week 12: Pre-Launch Prep

Person A:

- [ ] Create demo account with sample data
- [ ] Record demo videos
- [ ] Write launch blog post
- [ ] Prepare support docs

Person B:

- [ ] Setup production server (Hostinger)
- [ ] Configure Nginx
- [ ] Setup SSL certificates
- [ ] Configure automated backups
- [ ] Setup monitoring

Person C & D:

- [ ] Build deployment scripts
- [ ] Test production deployment
- [ ] Setup staging environment
- [ ] Final code review

Person E:

- [ ] Security testing
- [ ] Performance benchmarks
- [ ] Create runbook for operations
- [ ] Setup error monitoring

Sprint 6 Demo:

- Feature-complete product
- < 5 critical bugs remaining
- Fast and responsive
- Ready for alpha testing

Sprint 7-8: Marketing Site

Duration: Weeks 13-16
Goal: Professional landing page to attract customers

Weeks 13-14: Landing Page

Person A:

- [ ] Design homepage (Figma)
- [ ] Write all copy
- [ ] Source images/illustrations
- [ ] Design "For Cafes" page
- [ ] Design pricing page

Person D:

- [ ] Build marketing site:
  - Homepage with sections:
    - Hero with CTA
    - How it works
    - Features grid
    - Social proof
    - Use cases
    - Pricing teaser
    - FAQ
    - Footer
  - For Cafes page (detailed sales page)
  - Pricing page
  - Contact form
- [ ] Implement animations (Framer Motion)
- [ ] SEO optimization (meta tags, schema)
- [ ] Integrate WhatsApp CTA
- [ ] Add lead capture forms

Person E:

- [ ] Setup Google Analytics
- [ ] Setup Facebook Pixel (optional)
- [ ] Performance optimization (Lighthouse 90+)
- [ ] Deploy marketing site

Others:

- [ ] Review and test

- [ ] Provide feedback
- [ ] Share on social media

Weeks 15-16: Pre-Alpha Prep

Person A:

- [ ] Identify 5 potential alpha restaurants
- [ ] Schedule demos
- [ ] Create pitch deck
- [ ] Prepare onboarding materials

Person B:

- [ ] Create alpha restaurant accounts
- [ ] Setup monitoring dashboards
- [ ] Prepare for scale
- [ ] Create backup/restore procedures

Everyone:

- [ ] Final testing
- [ ] Bug fixing
- [ ] Documentation updates
- [ ] Team training on support

Deliverable: Ready to onboard first alpha customers

Sprint 9-10: Alpha Testing

Duration: Weeks 17-20
Goal: 3 restaurants actively using the system

Weeks 17-18: Alpha Launch

Person A:

- [ ] Onboard 3 restaurants
  - In-person setup (2 hours each)
  - Menu migration
  - QR printing

- Staff training

- [ ] Daily check-ins
- [ ] Gather feedback
- [ ] Document issues

Everyone:

- [ ] Monitor alpha restaurants daily
- [ ] Fix bugs within 24 hours
- [ ] Add small improvements
- [ ] Customer support

Metrics to Track:

- Orders per day per restaurant
- System uptime
- Bug reports
- User satisfaction (NPS)
- Time to setup restaurant

Weeks 19-20: Iteration

Based on alpha feedback:

- [ ] Fix top 10 pain points
- [ ] Add most requested features
- [ ] Improve onboarding flow
- [ ] Update documentation

Person A:

- [ ] Create case studies from alpha
- [ ] Get testimonials
- [ ] Take photos/videos
- [ ] Prepare for beta launch

Sprint 10 Demo:

- 3 restaurants using daily
- > 90% satisfaction
- < 3 critical bugs per week

- Ready to scale to 10 restaurants

Sprint 11-12: Beta Launch

Duration: Weeks 21-24
Goal: 10 paying restaurants, refined product

Weeks 21-22: Beta Onboarding

Onboarding Plan:

- Week 21: Onboard 4 restaurants
- Week 22: Onboard 3 more restaurants

Person A:

- [ ] Lead all onboarding (in-person)
- [ ] Create self-serve onboarding guide
- [ ] Build knowledge base
- [ ] Setup support system (WhatsApp + Docs)

Everyone:

- [ ] Rotating support shifts
- [ ] Bug fixes within 12 hours
- [ ] Weekly feature releases
- [ ] Customer success calls

Person B:

- [ ] Scale infrastructure if needed
- [ ] Monitor performance
- [ ] Optimize slow queries

Weeks 23-24: Final Iteration + Launch Prep

Goals:

- Stabilize system
- < 2% bug rate
- 95%+ retention

Person A:

- [ ] Finalize pricing
- [ ] Prepare contracts
- [ ] Get legal terms reviewed
- [ ] Plan public launch campaign

Everyone:

- [ ] Polish all rough edges
- [ ] Update all documentation
- [ ] Create FAQ from support questions
- [ ] Prepare for scale

Sprint 12 Demo:

- 10 paying restaurants
- 95%+ retention
- < 5 bugs per week
- Smooth operations
- Ready for public launch

Week 25+: Public Launch

Not a sprint, but ongoing:

Launch Activities:

1. Soft launch in Bangalore (target: 20 more restaurants in 2 months)
2. Content marketing (blogs, videos)
3. Instagram marketing
4. Referral program
5. Local cafe partnerships
6. Food blogger outreach

Ongoing:

- Weekly feature releases
- Continuous optimization

- Customer success
- Metric tracking

---

4. Risk Management

4.1 Technical Risks

Risk	Probability	Impact	Mitigation
Team lacks coding skills	High	High	Heavy AI usage, pair programming, extensive docs
Real-time sync issues	Medium	Medium	Use Socket.io (proven), polling fallback
Multi-tenancy bugs	Medium	High	Extensive testing, code review checklist
Performance issues	Low	Medium	Regular load testing, optimization sprints
Security vulnerabilities	Low	High	Security checklist, code reviews, penetration testing

4.2 Business Risks

Risk	Probability	Impact	Mitigation
Can't find alpha restaurants	Medium	High	Start with family/friend connections
Low retention	Medium	High	Weekly check-ins, fast bug fixes, excellent support
Feature creep delays launch	High	High	Strict scope, Phase 2 for everything else
Competition	Low	Medium	Focus on better UX + faster onboarding

4.3 Team Risks

Risk	Probability	Impact	Mitigation
Team member drops out	Low	High	Cross-train, document everything
Burnout	Medium	Medium	Realistic timelines, breaks, celebrate wins
Poor collaboration	Low	Medium	Daily standups, clear ownership

---

5. Success Metrics

5.1 Development Metrics

Sprint Velocity:

- Track story points completed per sprint
- Aim for consistent velocity by Sprint 4

Code Quality:

- < 5 critical bugs per sprint
- All PRs reviewed within 24 hours
- 80%+ test coverage (Phase 2 goal)

Deployment:

- Zero-downtime deployments
- < 1 hour to hotfix critical bugs

5.2 Product Metrics

Phase 1 (Alpha):

- 3 restaurants onboarded ✅
- ▌ 50 orders/day total across all
- 90%+ satisfaction (NPS > 50)
- < 5 critical bugs per week

Phase 2 (Beta):

- 10 restaurants onboarded ✅
- ▌ 200 orders/day total
- 95%+ retention after 1 month
- < 2% bug rate

Phase 3 (Launch):

- 20+ restaurants
- ₹2 lakhs MRR
- 95%+ retention
- < 1% churn rate

5.3 Business Metrics

Customer Acquisition:

- CAC < ₹10,000
- Onboarding time < 2 hours
- Setup completion rate > 80%

Revenue:

- Month 6: ₹30,000 MRR (3 restaurants × ₹10,000)
- Month 9: ₹1,00,000 MRR (10 restaurants × ₹10,000)
- Month 12: ₹2,00,000 MRR (20 restaurants × ₹10,000)

Retention:

- Month 1-3: 85%+ retention
- Month 4-6: 90%+ retention
- Month 7+: 95%+ retention

---

6. Post-MVP Plan

6.1 Phase 2 Features (Months 7-12)

Q3 (Months 7-9):

1. **POS + Billing**
   - Full dine-in billing
   - Payment gateway (UPI, cards)
   - GST compliance
   - Bill splitting
2. **Kitchen Display System (KDS)**
   - Digital KOT
   - Kitchen view (tablets)
   - Order queue management
3. **Staff Management**
   - Roles & permissions
   - Attendance tracking
   - Performance metrics

Q4 (Months 10-12): 4. **Inventory Management**

- Stock tracking
- Low stock alerts
- Recipe costing

- Waste tracking

5. **CRM & Loyalty**
    - Customer database
    - Loyalty points
    - Offers/coupons
    - SMS/WhatsApp campaigns
6. **Advanced Analytics**
    - Profit margins
    - Peak hours analysis
    - Staff efficiency
    - Customer behavior

6.2 Phase 3 Features (Year 2)

1. **Multi-outlet Management**
    - Central dashboard
    - Outlet-specific menus
    - Consolidated reporting
    - Franchise features
2. **Online Ordering**
    - Branded ordering website
    - Delivery management
    - Integration with Swiggy/Zomato
    - ONDC integration
3. **Enterprise Features**
    - White-labeling
    - API access
    - Advanced permissions
    - SLA guarantees

6.3 Growth Strategy

Months 7-12: Bangalore Domination

- Goal: 100 restaurants in Bangalore

- Strategy: Word-of-mouth, partnerships, content marketing

Year 2: Expand to 3 More Cities

- Hyderabad, Pune, Mumbai
- Target: 500 restaurants total

Year 3: Pan-India + Enterprise

- All major cities
- Target: 2,000 restaurants
- Focus on chains and franchises

---

7. Budget & Resources

7.1 Monthly Operational Costs (Phase 1)

Item	Cost
Hostinger VPS	₹700
Domain	₹50 (₹500/year)
Email Service	₹0 (Phase 2)
Monitoring Tools	₹0 (free tier)
Total	**₹750/month**

7.2 One-Time Costs

Item	Cost	When
Logo Design	₹5,000	Month 1
Marketing Videos	₹10,000	Month 4
Legal (T&C, Privacy)	₹15,000	Month 5
Total	**₹30,000**	

7.3 Tools & Software (Free Tier)

- **Development:** VS Code, GitHub (free)
- **Design:** Figma (free for 3 users)
- **Project Management:** Notion (free)
- **Communication:** WhatsApp, Slack (free)
- **Analytics:** Google Analytics (free)

- **Monitoring:** UptimeRobot (free for 50 monitors)
- **Error Tracking:** Sentry (free for 5K errors/month)

---

8. Weekly Cadence

8.1 Team Rituals

Monday:

- 9 AM: Weekly planning (1 hour)
  - Review last week
  - Set goals for this week
  - Assign tasks
- Sprint planning (if starting new sprint)

Tuesday-Friday:

- 9 AM: Daily standup (15 min)
- Coding + AI-assisted development
- Code reviews (continuous)
- Testing (continuous)

Saturday:

- Flexible working
- Catch up on pending tasks
- Personal learning

Sunday:

- Sprint review (if ending sprint)
- Demo to team (1 hour)
- Sprint retrospective (30 min)
- Planning for next week

8.2 Communication Guidelines

Synchronous (Real-time):

- Daily standup

- Urgent bugs
- Complex discussions

Asynchronous (Slack/WhatsApp):

- Updates on tasks
- Questions
- Code reviews
- Documentation

Documentation:

- All decisions in Notion
- Code comments for complex logic
- API docs (Swagger)
- User guides

---

9. Quality Assurance

9.1 Testing Strategy

Manual Testing (Every Sprint):

- [ ] Admin dashboard (all features)
- [ ] Customer app (all features)
- [ ] Cross-browser (Chrome, Safari, Firefox)
- [ ] Mobile (iOS, Android)
- [ ] Edge cases

Automated Testing (Phase 2):

- Unit tests (Jest)
- Integration tests (Supertest)
- E2E tests (Cypress)

Performance Testing:

- Lighthouse scores (weekly)
- Load testing (before each phase)

- Database query optimization

9.2 Bug Prioritization

Priority	SLA	Examples
P0 - Critical	Fix within 2 hours	System down, data loss, security breach
P1 - High	Fix within 24 hours	Feature broken, orders not working
P2 - Medium	Fix within 1 week	UI bugs, performance issues
P3 - Low	Fix when convenient	Minor UI glitches, nice-to-haves

---

10. Launch Checklist

10.1 Pre-Alpha Checklist

- [ ] All Phase 1 features complete
- [ ] Security audit passed
- [ ] Performance benchmarks met
- [ ] Documentation complete
- [ ] Support process defined
- [ ] 3 alpha restaurants identified
- [ ] Onboarding materials ready

10.2 Pre-Beta Checklist

- [ ] Alpha feedback incorporated
- [ ] No critical bugs
- [ ] 90%+ alpha retention
- [ ] Marketing site live
- [ ] Pricing finalized
- [ ] 10 beta restaurants identified
- [ ] Support team trained

10.3 Pre-Public Launch Checklist

- [ ] Beta feedback incorporated
- [ ] 95%+ beta retention
- [ ] Legal docs ready (T&C, Privacy)
- [ ] Payment processing works
- [ ] Monitoring set up
- [ ] Backup/restore tested
- [ ] Marketing campaign ready
- [ ] Press kit prepared

- [ ] 20+ pipeline restaurants

---

11. Key Performance Indicators (KPIs)

11.1 Track Weekly

Metric	Target
Active restaurants	Growing
Orders per restaurant	> 20/day
System uptime	> 99.5%
Critical bugs	< 3/week
Customer satisfaction	> 4.5/5
Support response time	< 2 hours

11.2 Track Monthly

Metric	Target
New restaurants	+5-10
MRR	Growing 20% MoM
Churn rate	< 5%
CAC	< ₹10,000
LTV	> ₹1,20,000 (₹10k × 12 months)
Onboarding time	< 2 hours

---

12. Conclusion

This roadmap is designed to be **realistic for an AI-assisted development team** with limited coding experience. The key to success:

1. **Use AI heavily** - Don't hesitate to use Claude, Cursor, Copilot for every task
2. **Start small** - Focus on MVP, defer everything else
3. **Ship fast** - 2-week sprints, continuous deployment
4. **Listen to users** - Alpha/beta feedback is gold
5. **Stay focused** - Resist feature creep
6. **Support each other** - Team success > individual heroics

Remember: The goal of Phase 1 is not perfection. It's to get 20 paying restaurants who love the product. Everything else can be built in Phase 2.

Let's build! 🚀

END OF ROADMAP

Appendix A: Sprint Template

```markdown
### Sprint X: [Name]
**Duration:** Weeks X-Y
**Goal:** [One-sentence goal]

#### Week X: [Focus]

**Person A:**
- [ ] Task 1
- [ ] Task 2

**Person B:**
- [ ] Task 1
- [ ] Task 2

... (repeat for all team members)

**Sprint Demo:**
- Deliverable 1
- Deliverable 2
```

Appendix B: Daily Standup Template

```markdown
```

Date: [YYYY-MM-DD]

Person A:
- Yesterday: [What you completed]
- Today: [What you'll work on]
- Blockers: [Any issues]

... (repeat for all team members)

Appendix C: Sprint Retrospective Template

markdown

Sprint X Retrospective

What went well:
- [Thing 1]
- [Thing 2]

What didn't go well:
- [Thing 1]
- [Thing 2]

Action items for next sprint:
- [Action 1]
- [Action 2]