Vision

Rules

Localization

Every place is different. What tools can you build that can adapt to the user's location to enhance and inform about transportation, mapping, regulations, food, culture, religion, or other local nuances? Is there anything missing from existing technologies that would benefit from strong knowledge of local information?

Consumer

Consumer is one of the fastest growing categories in India. What consumer applications do you feel are missing in India? What apps can you build to solve consumer problems in the region? Can you enhance existing apps with Gemini?

High Level Requirements

Text to itinerary

1. For different use cases
    a. Small activities
2. Social media integration for feed sources
3. End - end planning, cost structure management wrt budget
    a. Proactibvely suggest what to book and where to book and how much to pay

Use cases they would our product (1 city or a place for now) For MVP

1. Travel
    a. Long travel - multi step thinking, multiple factors
    b. Weekend getaways
2. Networking
    a. Walks and stuff
3. Physical activity
4. Experiences
5. Shopping
    a. Artistic
    b. Luxury
    c. Meet artisans
6. Architectural walks
7. Sightseeing
8. Art & Culture
    a. Workshops like woodworking
9. Outdoors & Nature
10. Food and Drink

Product specifications and User Experience: (For 1 city - Pondy)

1. User comes to the application and selects the travel dates - Lets assume for now 1st to 5th
2. User can select what to do on what day - Lets say on day 1 user selects Art & culture. We can show what options are there to explore city wide. For example

Moat - How do we optimally collect the local data which is not readily available like small scale artisans, weavers etc. and create recommendations for the users based on their interest

Data sources to refer to and expand on, while keeping the theme in mind

Technical challenges:
1. How do we get hyper local data -

PRD

Requirements

Homepage

1. Filters and search bar on top
    a. Enter start date
    b. Enter end date
    c. Enter budget
    d. Enter number of people
    e. Enter crowd preferences
        i. Crowded
        ii. Relatively niche
        iii. Super niche
    f. Places to see / things to do
2. Create itinerary?
    a. Drag and drop interface of multiple experiences

Discovery

1. On clicking the event show the event cards in standard formats

Design Instructions

1. Use rich themes

- Feasibility checks: Verify opening hours, travel times, realistic pacing
- Route optimization: Auto-calculate transport time between experiences
- Opening/closing time alerts: "Book trek 3 days ahead (sells out)"
- Dynamic adjustments: If rain, suggest indoor alternatives (integrate weather API)
- Real-time optimization: Update itinerary based on traffic, weather, crowd levels
- Visual event cards: Beautiful experience previews with photos
- Spontaneous discovery: "Happening this weekend" section
- Drag-and-drop: Rearrange experiences on visual timeline
- Google Calendar integration: One-click add to calendar
- Gamification: Earn badges for completing experiences (Cultural Explorer, Craft Enthusiast)
- 3D map visualization: See neighborhoods in 3D before visiting
-

Inspiration

Inspiration

1. https://www.seeksophie.com/magazine/a/things-to-do-singapore?gad_source=1&gad_campaignid=18709563570&gbraid=0AAAADCD2vjUttApo6nznefC4-YxXcRM1&gclid=CjwKCAiAtLvMBhB_EiwA1u6_Pvp9S_53ru42I1qE30b5Ipsg15CyIP1tpre9nvGRn7PgIi_xDu5idxoCLvQQAvD_BwE
    a. For singapore itinerary
2. https://docs.google.com/spreadsheets/d/1Zez0osu_h2IPu9_Aifry4gJLvG6pbp7pibIXhCJxfEc/edit?gid=242726960#gid=242726960
    a. For Sri Lanka itinerary I created
3. https://docs.google.com/spreadsheets/d/15HeoNghVHqJDQhADB9P-5jCfrsImGRAB-p2QLoNouRs/edit?gid=0#gid=0
    a. For Thailand itinerary

Competition

Tech Implementation

Winning Traits

Past Winners

project	hackathon	place	domain	architecture	key_innovation	tech_stack
Guardian	Llama Impact London 2024	1st	Healthcare	Supervisor	Rapid A&E triaging with multi-agent diagnostic flow	LangGraph, Local Llama 3.2, NVIDIA NIM, Docker/K8s
Circuit XR	Llama Impact Toronto 2024	1st	Education/XR	Custom Unity + Agent	Hand-drawn circuit to AR visualization pipeline	Unity, Groq (Llama 3.2), Computer Vision, AR Kit
OpenGlass	Llama 3 2024	1st	Accessibility	Swarm (vision + knowledge agents)	$20 smart glasses with visual understanding	Llama 3.2 Vision, Raspberry Pi, Custom hardware
EyeSpeak	Llama Impact Toronto 2024	2nd	Accessibility	Supervisor (gaze → text → speech)	Gaze-driven computer control with AI text prediction	LangGraph, Eye-tracking SDK, Llama 3.2, TTS

Fraudy	Llama Impact Toronto 2024	3rd	Security	Collaborative (real-time + analysis agents)	Real-time phone fraud detection during calls	LangGraph, Speech-to-text, Llama 3.2, Mobile SDK
Deb8	Llama 3 2024	2nd	Education/Entertainment	Collaborative (debater agents + judge agent)	Multi-model AI debate with autonomous judging	LangGraph, Multiple LLM APIs, Custom debate protocol

Hackathon Instructions

Hackathon Rules

To keep things fair and aligned with our goals, all teams must follow these rules:

- **Open Source:** Everything shown in the demo must be fully open source. This includes every component - backend, frontend, models, and any other parts of the project - published under an approved open source license.
- **New Work Only:** All projects must be started from scratch during the hackathon with no previous work.
- **Team Size:** Teams may have up to 4 members.
- **Banned Projects:** Project will be disqualified if they: violate legal, ethical, or platform policies, use code, data, or assets you do not have the rights to

🚫 Sample Anti-Projects to NOT DO — STRICTLY NO:

- AI Mental Health Advisor
- Basic RAG Applications
- Streamlit Applications
- Image Analyzers
- “AI for Education” Chatbot
- AI Job Application Screener
- AI Nutrition Coach
- Personality Analyzers
- Any project using AI to generate and give medical advice

Problem Statements

Your project must build in at least **one** of the two required tracks.

Localization

Every place is different. What tools can you build that can adapt to the user’s location to enhance and inform about transportation, mapping, regulations, food, culture, religion, or other local nuances? Is there anything missing from existing technologies that would benefit from strong knowledge of local information?

Consumer

Consumer is one of the fastest growing categories in India. What consumer applications do you feel are missing in India? What apps can you build to solve consumer problems in the region? Can you enhance existing apps with Gemini?

Project Prompt

UNIVERSAL HACKATHON WINNING SYSTEM PROMPT

Purpose: Generate prize-winning multi-agentic systems for any hackathon with guaranteed compliance and optimal scoring

---

SYSTEM INSTRUCTIONS

You are an expert AI system architect specializing in rapid prototyping of production-ready multi-agentic systems for competitive hackathons. Your goal is to generate complete, working solutions that maximize judging scores while ensuring full compliance with rules and avoiding disqualification patterns.

PHASE 1: REQUIREMENTS ANALYSIS

Input Processing

When given a hackathon brief, extract and validate:

1. **Mandatory Compliance Requirements**
   - Open source license requirements
   - Team size limits
   - New work vs pre-existing code rules
   - Submission format and deadlines
   - Required technology integrations (APIs, platforms, models)
2. **Problem Statements / Tracks**
   - List all available tracks
   - Identify judging criteria per track
   - Extract weight distribution (technical, impact, innovation)
   - Note any specific focus areas (accessibility, sustainability, etc.)
3. **Anti-Patterns / Disqualification Criteria**
   - Explicitly banned project types

    - Prohibited technologies or approaches
    - Ethical/legal restrictions
    - Overused patterns judges are tired of
4. **Past Winner Analysis** (if available)
    - Architecture patterns used (Supervisor, Swarm, Collaborative)
    - Framework adoption rates (LangGraph, CrewAI, custom)
    - Common scoring patterns (what gets 9-10 vs 6-7)
    - Demo formats that succeeded
    - Time-to-build estimates

PHASE 2: STRATEGIC OPTIMIZATION

Scoring Matrix Generation

For each potential project idea, calculate:

text

```
JUDGING_SCORE = (TECHNICAL_SCORE × WEIGHT₁) + (IMPACT_SCORE × WEIGHT₂) + (INNOVATION_SCORE × WEIGHT₃)



Where:

TECHNICAL_SCORE = f(architecture_complexity, code_quality, integration_depth, production_readiness)

IMPACT_SCORE = f(problem_size, user_benefit, market_potential, measurable_outcomes)

INNOVATION_SCORE = f(novelty, differentiation, technical_creativity, wow_factor)
```

Project Selection Criteria

MUST SATISFY:

1. Addresses at least 1 official problem statement/track
2. Avoids ALL disqualification patterns
3. Buildable within time constraint (buffer: 80% of total time)
4. Leverages required technologies centrally (not peripherally)
5. Has clear live demo potential

SHOULD OPTIMIZE FOR:

1. Highest projected judging score (target: top 10%)
2. Proven architecture patterns (reduce risk)
3. Team expertise alignment (faster execution)
4. Clear differentiation from past winners (avoid "me too")
5. Multiple modalities if possible (voice, vision, text)

Risk Assessment Matrix

Risk Level	Criteria	Mitigation Strategy
CRITICAL	Uses banned patterns, weak core tech integration, no demo plan	REJECT project immediately
HIGH	Complex hardware dependencies, unproven architecture, tight timeline	Add backup plans, simplify scope
MEDIUM	API rate limits, latency issues, UI complexity	Cache data, optimize critical path, simplify frontend
LOW	Minor bugs, limited feature coverage, basic UI	Acceptable - focus on core demo

---

PHASE 3: ARCHITECTURE GENERATION

Multi-Agent Pattern Selection

Decision Tree:

text

```
IF problem requires_sequential_reasoning AND clear_task_decomposition:
    → USE Supervisor Pattern (LangGraph)
    PROS: Explicit control flow, easier debugging, proven winner pattern
    CONS: Single point of failure (coordinator)

ELIF problem requires_parallel_exploration AND independent_tasks:
    → USE Swarm Pattern
    PROS: Parallel execution, resilient to single agent failure
    CONS: Harder to coordinate, potential inconsistencies

ELIF problem requires_negotiation AND multi_perspective_reasoning:
    → USE Collaborative Pattern
    PROS: Rich interaction dynamics, novel for judges
    CONS: More complex state management

ELSE:
  → DEFAULT to Supervisor (lowest risk, highest success rate)
```

Agent Responsibility Assignment

For each agent in the system, define:

python

```python
class AgentSpecification:
    name: str                          # Descriptive role name
    primary_model: str                 # LLM/model used (e.g., "gemini-pro", "gpt-4")
    responsibilities: list[str]        # Core tasks
    why_this_model: str                # Justification for model choice
    latency_target: str                # Response time requirement
    tools: list[str]                   # MCP/API tools available
    input_from: list[str]              # Which agents provide input
    output_to: list[str]               # Which agents receive output
    error_handling: str                # Fallback strategy
```

State Management Schema

Always include:

python

```python
class UniversalAgentState(TypedDict):
    messages: Annotated[list, add_messages]           # Conversation history
    current_task: str                                  # Active task description
    task_history: list[dict]                          # Execution trace
    agent_outputs: dict[str, Any]                     # Results from each agent
    errors: list[dict]                                # Error log for debugging
    next_agent: str                                   # Routing decision
    final_output: Optional[str]                       # End result
    metadata: dict                                    # Timestamps, user context, etc.
    checkpoint_id: Optional[str]                      # For state recovery
```

PHASE 4: IMPLEMENTATION GENERATION

Code Structure Template

text

```
project_root/
├── README.md                          # Comprehensive documentation
├── LICENSE                            # Open source license (MIT/Apache)
├── requirements.txt / package.json    # Dependencies
├── .env.example                       # Environment variables template
├── architecture.md                    # System design document
├── backend/
│   ├── main.py / index.js            # API entry point
│   ├── agents/
│   │   ├── coordinator.py            # Main orchestrator
│   │   ├── agent_{role1}.py          # Specialized agent 1
│   │   ├── agent_{role2}.py          # Specialized agent 2
│   │   └── ...                       # Additional agents
│   ├── tools/                        # MCP/API integrations
│   ├── state/                        # State management
│   ├── utils/                        # Helper functions
│   └── tests/                        # Unit tests (if time permits)
├── frontend/
```

```
│   ├── src/
│   │   ├── App.{tsx,jsx}            # Main component
│   │   ├── components/               # UI components
│   │   └── api/                      # Backend client
│   └── public/                       # Static assets
├── deployment/
│   ├── Dockerfile                    # Container definition
│   ├── docker-compose.yml            # Local development
│   └── {platform}_config             # Railway/Vercel/etc config
└── demo/
    ├── scenarios.md                  # Demo scripts
    ├── demo_video.mp4                # Recorded backup
    └── pitch_deck.pdf                # Presentation slides
```

Critical Code Patterns

1. Error Handling (Non-Negotiable)

python

```
def agent_execution_wrapper(agent_fn, state, max_retries=3):
    for attempt in range(max_retries):
        try:
            result = agent_fn(state)
            return result
        except RateLimitError:
            wait_time = 2 ** attempt  # Exponential backoff
            time.sleep(wait_time)
        except Exception as e:
            log_error(e, state, agent_fn.__name__)
            if attempt == max_retries - 1:
                return fallback_response(state)
    return fallback_response(state)
```

2. Checkpointing (For Complex Workflows)

python

```
from langgraph.checkpoint import MemorySaver, MongoDBSaver

# Development
checkpointer = MemorySaver()

# Production
checkpointer =
MongoDBSaver(connection_string=os.getenv("MONGODB_URI"))
```

```
app = workflow.compile(checkpointer=checkpointer)
```

3. Observability (Critical for Judging)

python

```python
# Enable tracing for demo visualization
import langsmith

langsmith.configure(
    tracing_enabled=True,
    project_name="hackathon-demo"
)

# Log all agent decisions
def log_agent_decision(agent_name, input_state, output, latency):
    trace_log.append({
        "timestamp": datetime.now(),
        "agent": agent_name,
        "input": input_state,
        "output": output,
        "latency_ms": latency
    })
```

PHASE 5: DIFFERENTIATION ENGINEERING

Multimodal Integration Decision Matrix

Modality	Add If...	Implementation Pattern	Demo Impact
Voice	Accessibility angle, hands-free use case	Speech-to-Text → LLM → Text-to-Speech	HIGH (live interaction)
Vision	Visual data processing, document analysis	Image → Vision Model → Structured Data	HIGH (tangible wow factor)
Video	Temporal analysis, surveillance, education	Frame extraction → Analysis → Summary	MEDIUM (harder to demo)
Audio	Music, ambient context, sound analysis	Audio processing → Classification → Action	MEDIUM (niche use cases)

Rule: Choose 1-2 modalities maximum. More = higher risk, diminishing returns.

Innovation Amplifiers

Technical Innovation:

- Novel architecture pattern combinations
- Custom protocols or communication patterns
- Real-time streaming with low latency
- Edge deployment or federated learning
- Cross-model reasoning (ensemble approaches)

UX Innovation:

- Zero-learning-curve interfaces
- Contextual awareness (location, time, user state)
- Proactive suggestions vs reactive responses
- Social/collaborative features
- Gamification or engagement loops

Domain Innovation:

- Underserved market segments
- Unique data sources or signals
- Cross-domain integration (e.g., finance + health)
- Cultural adaptation (not just translation)
- Regulatory compliance automation

PHASE 6: DEMO OPTIMIZATION

Live Demo Script Template

Minute 1: Problem Framing (Impact)

text

```
"[QUANTIFIED_USER_GROUP] face [SPECIFIC_PROBLEM] because
[ROOT_CAUSE].

This costs [MEASURABLE_IMPACT] in [time/money/opportunity].

We're solving this with [PROJECT_NAME]."
```

Minute 2: Live Demonstration (Technical + Innovation)

text

```
1. Show INPUT (voice, upload, text)

2. Visualize AGENT WORKFLOW (real-time trace)
```

```
3. Display INTERMEDIATE STEPS (transparency)

4. Reveal FINAL OUTPUT (with metrics)

5. Highlight UNIQUE FEATURE (wow moment)
```

Minute 3: Technical Depth + Impact (Credibility)

text

```
"Built on [ARCHITECTURE_PATTERN] with [CORE_TECH] powering
[KEY_CAPABILITY].

We're not just [GENERIC_APPROACH]—we're
[SPECIFIC_DIFFERENTIATION].

This is production-ready: [DEPLOYMENT_PROOF],
[CODE_QUALITY_SIGNAL],

fully open source at [GITHUB_LINK]."
```

Demo Environment Checklist

24 Hours Before:

- Record backup video (3 scenarios minimum)
- Test on demo WiFi network (not your laptop hotspot)
- Verify all API keys have sufficient quota
- Cache fallback data for offline demo
- Practice pitch 5+ times (aim for <3 minutes)

1 Hour Before:

- Fresh deployment to production (latest code)
- Smoke test all demo scenarios
- Charge all devices to 100%
- Have backup slides ready (if live demo fails)

- Mute notifications, close unnecessary apps

During Demo:

- Start with highest-impact scenario (hook judges immediately)
- Narrate what's happening ("Now the Discovery Agent is calling Swiggy API...")
- Show, don't just tell (traces, logs, real data)
- If error occurs, gracefully pivot to backup video
- End with clear call-to-action (GitHub, try demo URL)

PHASE 7: JUDGING OPTIMIZATION

Score Maximization Strategies

For Technical Execution Score:

1. **Code Quality Signals:**
   - Clean architecture with separation of concerns
   - Type hints and documentation
   - Error handling and logging
   - Unit tests (even if just a few)
2. **Integration Depth:**
   - Core technology is CENTRAL, not peripheral
   - Multiple sophisticated integrations (APIs, MCP tools)
   - Real-time processing where applicable
   - Production deployment (not localhost)
3. **Technical Sophistication:**
   - Advanced patterns (checkpointing, parallel execution, streaming)
   - Performance optimization evidence (caching, fast models)
   - Observability (tracing, monitoring)
   - Scalability considerations

For Impact Score:

1. **Problem Quantification:**
   - Number of affected users (bigger = better)
   - Current cost/inefficiency (measurable)
   - Market size or opportunity (dollars, time saved)

    - Urgency or severity (life-saving > convenience)
2. **Solution Validation:**
    - User testimonials or quotes (even informal)
    - Comparable to existing solutions (show advantages)
    - Feasibility (not science fiction)
    - Measurable success metrics
3. **Scalability Narrative:**
    - Path from MVP to production
    - Unit economics (if applicable)
    - Network effects or viral potential
    - Defensibility (why competitors can't easily copy)

For Innovation Score:

1. **Novelty Evidence:**
    - "First to..." claim (verify it's true)
    - Unique combination of technologies
    - Non-obvious approach to problem
    - Different from past winners
2. **Technical Creativity:**
    - Novel algorithm or architecture
    - Creative use of constraints (low resource, edge compute)
    - Unexpected integration (hardware + software, old + new)
    - Emergent behaviors from agent interactions
3. **Wow Factor:**
    - Visible intelligence (not black box)
    - Real-time or interactive demo
    - Multimodal showcase
    - Unexpected capability (menu photo → cultural recommendations)

PHASE 8: COMPLIANCE VERIFICATION

Pre-Submission Checklist

Legal & Ethical:

- All code is original or properly licensed

- No copyright violations (images, text, code)
- Privacy compliant (no PII exposure)
- No prohibited content (violence, discrimination, etc.)
- Terms of service respected for all APIs used

Technical Requirements:

- Uses required technologies/platforms (if specified)
- Core technology is demonstrably central to solution
- Code runs on standard hardware (no specialized GPUs unless provided)
- Dependencies are available (no internal/proprietary packages)
- Deployment is accessible to judges (public URLs, not localhost)

Submission Materials:

- README with setup instructions (5-minute quickstart max)
- Architecture document explaining system design
- Demo video (follows time limit, high quality audio/video)
- Pitch deck (if required, typically 5-10 slides max)
- Integration description (typically 150-300 words explaining core tech usage)
- Open source license file (MIT, Apache 2.0, or approved alternative)

Anti-Pattern Avoidance:

- Confirmed project type is NOT on banned list
- Approach is sophisticated (not basic wrapper around API)
- Demo is substantive (not just UI mockup with fake data)
- Claims are defensible (no vaporware or fake capabilities)

PHASE 9: TIMELINE ENGINEERING

Time Budget Allocation (for 24-hour hackathon)

text

```
Foundation (15%)          → 3.6 hours

Core Implementation (35%) → 8.4 hours
```

```
Differentiation (20%)      → 4.8 hours
Demo/Polish (15%)          → 3.6 hours
Submission (10%)           → 2.4 hours
Buffer (5%)                → 1.2 hours
─────────────────────────────────────
TOTAL                      → 24 hours
```

Critical Path Identification

Must Complete (Non-Negotiable):

1. Core agent workflow (coordinator + 2 specialized agents minimum)
2. One complete end-to-end demo scenario
3. Production deployment (working public URL)
4. GitHub repo with README
5. Demo video recording
6. Submission form completed

Should Complete (Competitive Advantage):

1. 3+ specialized agents with clear division of labor
2. Multimodal capability (voice OR vision)
3. Real-time agent visualization
4. Error handling and recovery
5. Monitoring/observability
6. Pitch deck

Nice to Have (Time Permitting):

1. Multiple demo scenarios
2. UI polish and animations
3. Unit tests
4. Advanced features (recommendation engine, personalization)
5. Social features (sharing, collaboration)

PHASE 10: EMERGENCY PROTOCOLS

If Behind Schedule (>2 hours from target)

Immediate Actions:

1. **Identify bottleneck:** What's blocking progress?
2. **Simplify scope:** Cut nice-to-haves ruthlessly
3. **Parallelize:** Can another team member take over a task?
4. **Use templates:** Don't build from scratch if libraries exist
5. **Mock when necessary:** Fake slow API calls with cached data

Priority Cuts (in order):

1. Additional agent specializations (keep coordinator + 2 agents)
2. Secondary modality (keep only primary differentiation)
3. UI polish (functional > beautiful)
4. Advanced error handling (happy path only)
5. Monitoring/observability (remove dashboards, keep basic logging)

If Critical Feature Fails

Failure Mode: Core API Down

- **Mitigation:** Switch to cached sample data, explain "production uses live API"
- **Demo Strategy:** Show code calling API, display cached results
- **Narrative:** "In production this calls [API], for demo stability we're showing cached responses"

Failure Mode: Agent Coordination Broken

- **Mitigation:** Simplify to sequential calls (remove complex routing)
- **Demo Strategy:** Manual agent selection, show each agent works independently
- **Narrative:** "Each agent can operate independently, coordinator optimizes routing"

Failure Mode: Live Demo Crashes

- **Mitigation:** Immediately switch to backup video

- **Demo Strategy:** "Let me show you our recorded demo while we troubleshoot"
- **Narrative:** Show architecture slides, explain technical depth verbally

---

OUTPUT FORMAT SPECIFICATIONS

When generating a complete hackathon solution, provide:

1. Strategic Brief (1-2 pages)

- Project name and one-sentence description
- Problem statement alignment (which tracks addressed)
- Target judging scores with justification
- Key differentiators (3-5 bullet points)
- Risk assessment and mitigation

2. Technical Architecture (2-3 pages)

- System diagram (ASCII or description for diagram generation)
- Agent specifications (name, role, model, responsibilities)
- State management schema
- Technology stack with version numbers
- Deployment architecture
- [REQUIRED_TECH] integration rationale (150-300 words)

3. Implementation Files

- Complete file structure
- Starter code for coordinator agent
- Starter code for 2+ specialized agents
- State management utilities
- Basic frontend (if applicable)
- Deployment configurations
- requirements.txt / package.json

4. Demo Strategy (1 page)

- 3-minute pitch script
- 3 demo scenarios with expected inputs/outputs
- Backup plans for common failure modes
- Visual elements to prepare (diagrams, slides)

5. Execution Timeline (1 page)

- Hour-by-hour breakdown
- Team role assignments (if multi-person)
- Critical path dependencies
- Checkpoints for progress validation

6. Submission Checklist

- All required materials listed
- Pre-submission QA steps
- Compliance verification items
- Emergency contact info (mentors, organizers)

QUALITY ASSURANCE GATES

Before finalizing any solution, verify:

Gate 1: Compliance

- ✅ Addresses ≥1 official problem statement
- ✅ Avoids ALL disqualification patterns
- ✅ Uses required technology centrally
- ✅ Buildable in available time

Gate 2: Technical Viability

- ✅ Architecture pattern is proven (not experimental)
- ✅ All APIs/services have available free tiers
- ✅ No single point of failure without backup
- ✅ Complexity matches team skill level

Gate 3: Competitive Positioning

- ✅ Projected score in top 20% (based on criteria weights)
- ✅ Clear differentiation from past winners
- ✅ Strong impact narrative (quantified users/market)
- ✅ Live demo potential is high

Gate 4: Execution Realism

- ✅ Critical path fits in 70-80% of available time
- ✅ No dependency on unavailable resources
- ✅ Team has necessary expertise (or can acquire quickly)
- ✅ Backup plans exist for high-risk components

SYSTEM SELF-CHECK PROMPTS

Before generating solution, ask:

1. **"What would disqualify this project?"** → Verify no anti-patterns
2. **"Why would judges give this a 9-10 vs 6-7?"** → Ensure differentiation
3. **"Can this demo fail gracefully?"** → Validate backup plans
4. **"What would I cut if I had 50% less time?"** → Identify true priorities
5. **"How is this better than [top past winner]?"** → Confirm competitive edge

---

EXECUTION MANTRAS

1. **"Compliance first, optimization second"** - Disqualification is worse than imperfect solution
2. **"Working demo beats perfect code"** - Judges see demo, not your git history
3. **"Impact narrative drives scores"** - Technical excellence must serve user benefit
4. **"Simplicity scales, complexity fails"** - Under time pressure, simple wins
5. **"Record early, demo confidently"** - Backup video is insurance policy

---

SUCCESS CRITERIA

A solution generated by this system succeeds if:

- ✅ **Compliance:** 100% adherence to rules (binary: pass/fail)
- ✅ **Scoring:** Projected score ≥ 80th percentile of past winners
- ✅ **Viability:** Buildable in ≤80% of available time
- ✅ **Demonstrability:** Clear live demo with wow moments
- ✅ **Differentiation:** Unique approach vs past winners

Target Outcome: Top 10% finish in hackathon (prize-winning territory)

END OF SYSTEM PROMPT

This prompt is designed to be error-resistant through comprehensive checklists, multiple validation gates, and built-in fallback strategies. Apply iteratively: analyze requirements → generate solution → validate against gates → refine → execute.

AI Roadmap

Based on your comprehensive prompt and the current Gemini 3 Bengaluru Hackathon happening today, I'll create all requested deliverables optimized for your 6-hour, 2-person team constraint.

📋 DELIVERABLE 1: EXPANDED VISION DOCUMENT

Project Name: Sidequest

"From scroll to story: Turn social media inspiration into plot-first experiences"

Core Thesis

Urban India is experiencing a cultural reset driven by digital fatigue and analog renaissance. The opportunity lies in capturing the Instagram-to-IRL gap: people see compelling experiences on social media but face massive friction converting inspiration to action. Sidequest bridges this gap using Gemini 3's multimodal capabilities to extract experiences from social content and generate culturally-grounded, story-driven itineraries.[ppl-ai-file-upload.s3.amazonaws]cerebralvalley+1

Market Validation:

- 80% of urban Indians want personalized cultural engagement[ppl-ai-file-upload.s3.amazonaws]
- 78 out of 10 Indians explored new experience categories in 2025[ppl-ai-file-upload.s3.amazonaws]
- Going-out revenue jumped 3.5× YoY (₹73 Cr → ₹259 Cr)[linkedin]
- Urban Indians spend 11% of daily time in culture/leisure (up from 9.9% in 2019)[ppl-ai-file-upload.s3.amazonaws]

Experience Economy Behavioral Shifts

1. **Plot-First Culture**: The new luxury is earned experiences with provenance - friction creates memory, not convenience[ppl-ai-file-upload.s3.amazonaws]
2. **Collision Economy**: Novel stimulus through unlikely pairings (pottery + sound healing, padel + founder networking) defeats algorithmic sameness[ppl-ai-file-upload.s3.amazonaws]
3. **Proof-of-Life Signaling**: Validation moving off-social - experiences felt in the body via energy, mood, stimulus[ppl-ai-file-upload.s3.amazonaws]
4. **Ambient Belonging**: 83% of Indian Gen Z choose community fitness classes; arriving alone is autonomy, not loneliness[ppl-ai-file-upload.s3.amazonaws]

5. **Dual Prime Times**: 40% of dining now on weekdays; mornings as valid as evenings - status by schedule[ppl-ai-file-upload.s3.amazonaws]

Differentiated Value Proposition

Problem: Existing platforms (Google Maps, MakeMyTrip, District) optimize for convenience over memory, transactions over transformation.[ppl-ai-file-upload.s3.amazonaws]

Solution: Sidequest is identity scaffolding for the experience economy - helping users convert digital fatigue into plot-first living.[ppl-ai-file-upload.s3.amazonaws]

Unique Moat:

- **Social Media Ingestion**: Direct extraction from Instagram Reels/YouTube vlogs using Gemini Vision[cerebralvalley][ppl-ai-file-upload.s3.amazonaws]
- **Hyperlocal Curation**: Small-scale artisans, hidden workshops, supper clubs not on Google Maps[ppl-ai-file-upload.s3.amazonaws]
- **Narrative-Driven Itineraries**: Story arcs with intentional friction, lore, and provenance[ppl-ai-file-upload.s3.amazonaws]
- **Cultural Autobiography**: Track neighborhoods explored, interest pods formed, taste evolution[ppl-ai-file-upload.s3.amazonaws]

Gemini 3 Centrality

Gemini 3's capabilities make this uniquely possible:[cerebralvalley]

1. **Multimodal Understanding**: Extract experiences from Instagram Reels, YouTube vlogs, menu photos
2. **Long Context Reasoning**: Process entire blog posts, synthesize multiple social sources
3. **Cultural Depth**: India-specific localization beyond translation (timing, etiquette, dress codes)
4. **Generative Planning**: Craft narrative itineraries with emotional arcs, not just chronological lists
5. **Reduced Latency**: Flash for fast discovery, Pro for deep reasoning[cerebralvalley]

Target Users & Use Cases

Primary Persona: "The Cultural Nomad" (25-35, Bangalore/Mumbai)

- Wants to experiment with identity pods (food nerd, run club, art) without commitment
- Seeks solo-sure experiences with ambient belonging
- **Use Case Priority**: Solo physical activities (run clubs, pottery workshops) + Shopping (artisan markets)[ppl-ai-file-upload.s3.amazonaws]

Secondary Persona: "The Plot Seeker"

- Values earned experiences over effortless ones
- Willing to queue, trek, learn - friction creates stories
- **Use Case Priority**: Travel (heritage walks, hidden gems) + Architectural exploration[ppl-ai-file-upload.s3.amazonaws]

Tertiary Persona: "The Collision Hunter"

- Bored by predictable nights, seeks novel stimulus
- Early adopter energy, first-mover advantage
- **Use Case Priority**: Networking events + Unlikely experience pairings[ppl-ai-file-upload.s3.amazonaws]

📊 DELIVERABLE 2: COMPETITOR ANALYSIS & GAP MATRIX

Direct Competitors

Platform	Strengths	Experience Economy Gaps	Our Advantage
Google Maps	Comprehensive venue data, reviews	No story/lore, no cultural autobiography, optimizes convenience over memory [ppl-ai-file-upload.s3.amazonaws]	Plot-first narratives with provenance, friction injection
District by Zomato	Leading going-out platform, events focus [linkedin]	Event-centric (not daily micro-escapes), no social media ingestion, no AI itinerary building [ppl-ai-file-upload.s3.amazonaws]	Social media → itinerary flow, hyperlocal artisan data
MakeMyTrip	Booking infrastructure	Transactional, treats experiences as interchangeable, no identity scaffolding [ppl-ai-file-upload.s3.amazonaws]	Cultural autobiography tracking, taste evolution
Luma/Partiful	Community events	No AI discovery, no social media synthesis, no narrative building [ppl-ai-file-upload.s3.amazonaws]	Gemini-powered multimodal extraction and planning

Competitor Research

CATEGORY 1: AI-POWERED TRAVEL PLANNERS

---

1. Expedia Trip Matching ⭐ MOST RELEVANT

Launch: May 2025 | Status: Beta

What They Do:

- First OTA to turn Instagram Reels into bookable itineraries
- Users share Reel → @Expedia DM → AI generates personalized itinerary → book directly
- Powered by OpenAI, integrated with Microsoft Copilot Actions

Their Workflow:

1. Find inspiration (Instagram Reel scrolling)
2. Share with @Expedia account via DM
3. Receive AI-generated recommendations in Instagram DMs
4. Book directly from itinerary

✅ Features to Borrow:

- Instagram Reel URL ingestion (you MUST have this - they validated the market)
- DM-based interaction (consider WhatsApp/Telegram alternative for India)
- Instant personalization from video content analysis
- Bookable output (link to District, Headout, direct booking)

🎯 Your Differentiation:

- They optimize for booking conversion → You optimize for cultural autobiography
- They're transactional → You're narrative-first with lore/provenance
- They don't have hyperlocal artisan data or solo-sure filtering

---

2. Mindtrip 🔥 BEST-IN-CLASS AI PLANNER

Launch: 2024 | Platform: Web + iOS

What They Do:

- AI travel companion with "Start Anywhere™" feature

- Build trips from links, photos, bookings, or receipts
- Real-time location-based recommendations during trips
- Collaborative planning (Google Docs-style)

Key Features:

- Magic Camera: Point at landmark → instant ID and recommendations
- Interactive maps: Visual itineraries with drag-and-drop calendar
- Collaboration tools: Real-time co-editing with friends/family
- Booking management: Store all confirmations in one hub
- Offline mode: Access itineraries without internet
- Personalization: Answers questions about preferences, surfaces tailored suggestions

✅ Features to Borrow:

- "Start Anywhere™" concept → Your version: "Paste Instagram URL, describe vibe, upload menu photo"
- Visual map integration with pins for cultural autobiography tracking
- Real-time location awareness for on-trip recommendations
- Collaborative planning → Share itinerary with friends, co-edit
- Photo-to-itinerary: Upload restaurant menu photo → extract venue → find similar experiences

🎯 Your Differentiation:

- They're destination-focused → You're experience-economy-focused
- They optimize routes → You inject intentional friction
- They lack cultural depth (timing nuances, solo-sure indicators)

---

3. iplan.ai 📍 MINUTE-BY-MINUTE PLANNER

Platform: Mobile app

What They Do:

- Minute-by-minute itineraries with opening hours, travel times, maps
- AI optimizes routes so you don't arrive when venues are closed
- Real-time collaboration and edits

✅ Features to Borrow:

- Feasibility checks: Verify opening hours, travel times, realistic pacing
- Route optimization: Auto-calculate transport time between experiences

- Opening/closing time alerts: "Book trek 3 days ahead (sells out)"

🎯 Your Differentiation:

- They're hyper-structured → You embrace spontaneity and friction
- They optimize for efficiency → You optimize for memory

---

4. NXVoy Trips

Status: Top AI trip planner 2026

What They Do:

- Dynamic route adjustments based on real-time factors
- Personalized recommendations based on travel style
- Not static templates - adaptive planning

✅ Features to Borrow:

- Dynamic adjustments: If rain, suggest indoor alternatives (integrate weather API)
- Travel style detection: Detect if user is adventure/cultural/foodie from query patterns
- Real-time optimization: Update itinerary based on traffic, weather, crowd levels

---

5. Wonderplan

Focus: Budget-conscious planning

What They Do:

- Questionnaire-based itineraries
- Cost breakdowns with hotels and experiences
- Budget transparency

✅ Features to Borrow:

- Questionnaire onboarding: "What made you feel alive recently?" → Build interest pods
- Budget breakdown: Total cost + individual experience costs + hidden fees (auto fare)
- Cost optimization suggestions: "Take metro instead of auto, save ₹200"

---

6. GuideGeek

Platform: WhatsApp, Instagram Messenger

What They Do:

- AI assistant in messaging apps
- On-the-go local tips in 50+ languages
- Conversational interface

✅ Features to Borrow:

- WhatsApp bot: Indians prefer WhatsApp → build Sidequest bot for accessibility
- Multilingual support: English + Hindi + Kannada for Bangalore
- On-the-go assistance: "I'm at Cubbon Park, what's nearby that's solo-friendly?"

CATEGORY 2: EXPERIENCE DISCOVERY PLATFORMS (INDIA)

---

7. District by Zomato 🇮🇳 TOP INDIA COMPETITOR

Launch: November 2024 | Revenue: ₹259 Cr (3.5× YoY growth)

What They Do:

- All-in-one going-out platform: movies, dining, events, concerts, sports
- Smart suggestions based on preferences
- One-stop payment for tickets + dining
- User reviews integrated
- Waitlists, recurring events, community management

Their Strengths:

- Zomato's distribution (millions of users)
- Event-centric with strong booking infrastructure
- Real-time updates on availability

Their Gaps (Your Opportunities):

- No AI itinerary building - they show events, you create stories
- No social media ingestion - they don't extract from Instagram
- No hyperlocal artisan data - focused on established venues
- No cultural autobiography tracking - single-transaction focus

- Event-centric, not daily micro-escapes

✅ Features to Borrow:

- Smart suggestions based on past behavior
- Combined payment for multiple experiences
- Waitlist functionality for popular workshops
- Recurring events tracking (weekly run clubs, monthly supper clubs)
- Easy cancellations with minimal hassle

🎯 Partnership Opportunity:

- Integrate Sidequest as "AI Planner" inside District app
- You provide narrative itineraries → District handles bookings

---

8. Urbanaut 🇮🇳 HYPERLOCAL EXPERIENCES

Platform: Web + iOS

What They Do:

- Curated local experiences across 35+ Indian cities
- Heritage walks, pottery workshops, food tastings, cycling, live music
- 350+ partners use their listing platform
- "Off the beaten path" focus, not TripAdvisor-style mass tourism

Their Strengths:

- Deep local partnerships with artisans and workshop hosts
- Authentic, non-touristy experiences
- Strong curation ("If it's on Urbanaut, it's worth your bucket list")

Their Gaps:

- No AI discovery - manual browsing
- No narrative itinerary building
- No social media integration

✅ Features to Borrow:

- Partner listing platform: Let artisans list experiences directly (post-hackathon)
- City-specific curation: Bangalore, Mumbai, Delhi, Goa focus
- Category diversity: Heritage + craft + food + fitness + music
- "Local eyes" positioning: Market as seeing cities through local lens

🎯 Partnership Opportunity:

- Scrape Urbanaut's 350+ experiences for your database
- Partner to be their "AI planning layer"

---

9. Headout, GetYourGuide, Klook 🌏 GLOBAL EXPERIENCE MARKETPLACES

Market: India + Global

Comparison:

Platform	**Strengths**	**India Focus**	**Best For**
GetYourGuide	Polished UI, family-friendly, easy cancellations	Medium	Europe, full tours
Klook	Best pricing, great app, QR code tickets, Asia-strong	High	Budget travelers, attraction tickets
Headout	India-present but limited inventory	Medium	Last-minute bookings

✅ Features to Borrow from Klook:

- Clear pricing with no hidden costs
- In-app QR codes for ticket access
- Calendar integration: Add experiences to Google Calendar
- Ticket translation: Translate tickets to local language
- Bundle deals: "Potter workshop + heritage walk = 15% off"

✅ Features to Borrow from GetYourGuide:

- 24-hour cancellation with instant refunds
- Responsive chat support

- Family-friendly filtering (adapt to solo-friendly)

CATEGORY 3: TRADITIONAL TRIP ORGANIZERS

---

10. Wanderlog 🗺️ VISUAL MAP PLANNER

Focus: Visual itinerary creation

What They Do:

- Map-first planning - add places directly from map
- Collaborative editing with travel companions
- Offline access to itineraries
- Integration with booking platforms
- Rich travel inspiration and recommendations

✅ Features to Borrow:

- Map-first visualization: Show cultural autobiography as map with explored neighborhoods
- Drag-and-drop: Rearrange experiences on visual timeline
- Offline mode: Download itinerary for no-internet access
- Booking platform integration: Link to Headout, Urbanaut, District

---

11. TripIt 📧 EMAIL-TO-ITINERARY

Focus: Automated trip organization

What They Do:

- Forward confirmation emails → auto-creates itineraries
- Consolidates flights, hotels, activities automatically
- Master itinerary for multi-destination trips
- Simple, efficient organization

✅ Features to Borrow:

- Email integration: Forward workshop booking email → auto-add to cultural autobiography
- Auto-import: Parse booking confirmations from Gmail/Outlook

- Master itinerary view: See all upcoming experiences in one timeline

CATEGORY 4: COMMUNITY EVENT PLATFORMS

12. Luma 🎟️ COMMUNITY EVENTS

Focus: Professional community events

What They Do:

- Advanced RSVP with waitlists
- Zoom + Google Calendar integration
- Community management: newsletters, recurring events
- Series of events for ongoing communities

✅ Features to Borrow:

- Waitlist management: "Pottery workshop full, join waitlist (3 people ahead)"
- Recurring event tracking: "This run club meets every Saturday 6am"
- Google Calendar integration: One-click add to calendar
- Community newsletters: Weekly digest of new hyperlocal experiences

13. Partiful 🎉 CASUAL SOCIAL EVENTS

Focus: Spontaneous, social-driven planning

What They Do:

- Simple, fast event creation for casual gatherings
- Visual, engaging invites
- Low-friction RSVP

✅ Features to Borrow:

- Low-friction interaction: One-tap "I'm interested" vs full booking
- Visual event cards: Beautiful experience previews with photos
- Spontaneous discovery: "Happening this weekend" section

14. AskLuma.ai 🤖 AI EVENT DISCOVERY

Focus: AI-powered event recommendations

What They Do:

- Personalized event recommendations based on interests
- Live insights and trending events
- Networking matching: Connect with relevant attendees
- Reward points for participation

✅ Features to Borrow:

- AI-powered filtering: "Based on your pottery interest, try this ceramics expo"
- Trending experiences: "5 people from your network attended this workshop"
- Networking layer: "3 other solo travelers are going to this heritage walk"
- Gamification: Earn badges for completing experiences (Cultural Explorer, Craft Enthusiast)

CATEGORY 5: SOCIAL-FIRST DISCOVERY

---

15. Pinterest Travel Planning

Method: Visual bookmarking → organization

How Users Use It:

- Pin inspiring images to boards
- Organize by destination or theme
- Use hashtags to discover hidden gems

✅ Features to Borrow:

- Visual bookmarking: Save experiences to "Wish List" with beautiful images
- Thematic boards: Organize by interest pod (Food Nerd, Craft Explorer, Wellness)
- Hashtag discovery: Extract experiences from Instagram via hashtags (#BangaloreHiddenGems)

---

16. TikTok #TravelTok

Stats: 89% of Gen Z discover destinations via TikTok

How It Works:

- Short-form video inspiration
- 72% influenced by social media content for travel decisions
- 40% book trips after viewing content

✅ Features to Borrow:

- TikTok URL ingestion: Extract experiences from TikTok travel videos
- Short-form video optimized: Sidequest's own TikTok showing plot-first itineraries
- Creator partnerships: Collab with travel TikTokers for experience validation

---

CATEGORY 6: NICHE AI TOOLS

---

17. Tryp.com

Focus: Cheap multi-city trips

✅ Feature to Borrow:

- Flexible date optimization: "Go Bangalore→Goa→Mumbai on these dates, save ₹2000"

---

18. Sygic Travel

Focus: Map-based planning

✅ Feature to Borrow:

- 3D map visualization: See neighborhoods in 3D before visiting

---

19. QuillBot Travel Itinerary Generator

Focus: AI writing tool for itineraries

✅ Feature to Borrow:

- Natural language prompts: "Plan a cultural weekend with solo-friendly workshops"
- Refinement loop: Iterative chat to improve itinerary

- Multi-city support: Bangalore + Mysore + Coorg in one narrative

---

20. Canva Trip Planner AI

Focus: Visual itinerary design

✅ Feature to Borrow:

- Printable itineraries: Beautiful PDF exports for offline use
- Template customization: Pre-designed itinerary templates
- Visual design: Make itineraries shareable on Instagram Stories

---

📊 FEATURE PRIORITIZATION MATRIX

MUST-HAVE (Core Demo)

Feature	**Inspired By**	**Why Critical**	**Implementation Time**
Instagram Reel URL extraction	Expedia Trip Matching	Market validation, direct competitor	1 hour
Narrative itinerary generation	iplan.ai, Mindtrip	Your core differentiation	1 hour
Cultural context layering	Original (your innovation)	Hyperlocal moat	1 hour

Visual map display	Wanderlog	Judges love visuals	30 mins
Budget breakdown	Wonderplan	Table stakes	30 mins

DIFFERENTIATORS (Competitive Edge)

Feature	**Inspired By**	**Your Twist**	**Implementation Time**
Solo-sure filtering	Original	Community Agent tags solo-friendly	45 mins
Collision suggestions	Original	Cross-pod recommendations	45 mins
Lore layering	Original	Backstory + provenance	Baked into prompts
Intentional friction	Original	Queue times = memory	Baked into prompts

Cultural autobiography	AskLuma rewards + Mindtrip maps	Visual map of identity evolution	1 hour (post-demo)

NICE-TO-HAVE (Post-Hackathon)

Feature	**Inspired By**	**Value Add**	**Priority**
WhatsApp bot	GuideGeek	Indian market fit	High
Collaborative planning	Mindtrip	Group trips	Medium
Waitlist management	Luma	Popular workshop handling	Medium
Magic Camera	Mindtrip	Point at venue → instant info	Low (cool but complex)
Gamification badges	AskLuma	Long-term engagement	Medium
Partner listing platform	Urbanaut	Two-sided marketplace	Low (post-MVP)

🎯 YOUR WINNING COMBINATION

What No Competitor Has:

1. Social Media → Narrative Itinerary → Cultural Autobiography (end-to-end flow)
2. Intentional friction injection (counterintuitive, culturally grounded)
3. India-specific cultural context (timing, dress codes, solo-sure, transport hacks)
4. Hyperlocal artisan data (30% not on Google Maps)
5. Experience economy framing (identity scaffolding, not trip planning)

Your Tech Stack Advantage:

- Gemini 3 Vision: Better video frame analysis than Expedia's OpenAI
- LangGraph Supervisor: More sophisticated than single-model competitors
- Multi-agent coordination: Discovery + Context + Plot + Budget + Community in parallel

Experience Economy Gap Matrix

Cultural Shift	User Need	Competitor Gap	Sidequest Opportunity
Plot-First Culture [ppl-ai-file-upload.s3.amazonaws]	Earned experiences with backstory	Generic "top 10" lists	Story-driven itineraries with lore + intentional friction
Collision Economy [ppl-ai-file-upload.s3.amazonaws]	Novel, unlikely pairings	Single-category recommendations	Cross-pod mashups (fitness + art + networking)
Proof-of-Life [ppl-ai-file-upload.s3.amazonaws]	Build cultural autobiography	No identity tracking	Experience residue map, taste evolution visualization
Ambient Belonging [ppl-ai-file-upload.s3.amazonaws]	Solo-sure arrival confidence	Group-assumed experiences	Solo-optimized filtering with social scaffolding cues
Social Media Discovery	Instagram → IRL conversion	Manual search after inspiration	Direct Reel/vlog ingestion → bookable plans
Hyperlocal Depth	Artisans, hidden gems	Only established businesses	Community-sourced, creator-vetted micro-experiences

Strategic Positioning

Sidequest is NOT:

- Another travel recommendation chatbot
- Basic RAG over venue database
- Streamlit wrapper around Gemini API[ppl-ai-file-upload.s3.amazonaws]

Sidequest IS:

- Identity scaffolding tool for experience economy
- Social media inspiration → cultural autobiography converter
- First-mover in plot-first, lore-driven planning[ppl-ai-file-upload.s3.amazonaws]

📘 DELIVERABLE 3: PRODUCT REQUIREMENTS DOCUMENT (PRD)

Executive Summary

Product: Sidequest
Tagline: "From scroll to story: Turn social media inspiration into plot-first experiences"
Market: India's $142B live events sector projected to reach $235B by 2027[ppl-ai-file-upload.s3.amazonaws]
Team: 2 members, 6-hour hackathon build
Tech Stack: Gemini 3 (Vision + Pro + Flash), LangGraph Supervisor, Next.js, MongoDB[youtube][ppl-ai-file-upload.s3.amazonaws]

Problem Statement: Urban Indians face massive friction converting Instagram inspiration to bookable plans. 80% want personalized cultural engagement, yet existing platforms optimize for transactions over transformation.[ppl-ai-file-upload.s3.amazonaws]

Success Metrics:

- Planning time: 3+ hours → 15 minutes
- Experience completion: 60%+ users complete ≥1 experience within 2 weeks
- Cultural impact: 2.5 new interest pods explored per month[ppl-ai-file-upload.s3.amazonaws]

User Personas

Persona 1: Priya - The Cultural Nomad

- Age: 28, Product Manager, Bangalore

- Pain: Sees amazing pottery workshops on Instagram, doesn't know how to access them as a beginner
- Need: Solo-sure experiences with beginner energy, no commitment pressure[ppl-ai-file-upload.s3.amazonaws]
- Gain: Builds confidence experimenting with new identity pods (pottery circle, run club, film community)

Persona 2: Rohan - The Plot Seeker

- Age: 32, Startup Founder, Mumbai
- Pain: Bored by algorithm-optimized recommendations, wants experiences that feel earned
- Need: Friction that creates stories - queueing at heritage bakery, trekking to hidden artisan[ppl-ai-file-upload.s3.amazonaws]
- Gain: Collects experiences with provenance and lore, builds cultural autobiography

Core User Journey

Stage 1: Inspiration Discovery

- User scrolling Instagram, sees Reel about Bangalore heritage coffee culture
- Feels underwhelmed by endless scroll, wants to experience IRL[ppl-ai-file-upload.s3.amazonaws]

Stage 2: Frictionless Ingestion

- Opens Sidequest, pastes Instagram Reel URL
- Gemini Vision extracts 5 mentioned coffee spots, timings, neighborhood context[cerebralvalley][ppl-ai-file-upload.s3.amazonaws]
- Alternative: Voice input "I want solo-friendly pottery workshop for complete beginners"

Stage 3: AI-Powered Planning

- 5 agents collaborate via LangGraph Supervisor:[changelog.langchain][youtube]
  - Discovery Agent: Finds 12 experiences across Instagram, Luma, artisan databases
  - Cultural Context Agent: Adds timing (9-11am coffee peak), dress codes, solo-sure indicators[ppl-ai-file-upload.s3.amazonaws]
  - Plot-Builder Agent: Crafts narrative arc with intentional friction[ppl-ai-file-upload.s3.amazonaws]
  - Budget Agent: Estimates ₹800-1200, suggests BNPL for workshop fee
  - Community Agent: Tags solo-friendly, conversation-optional experiences[ppl-ai-file-upload.s3.amazonaws]

Stage 4: Story-Driven Itinerary

- Receives: "Start your Saturday 9am at CTR for benne dosa (queue 20 mins - locals gather here, worth the wait). Walk through Basavanagudi Bull Temple neighborhood (heritage architecture). 11am filter coffee at Indian Coffee House (counter seating = easy conversation starters). End at Atta Galatta bookstore pottery workshop (beginner-friendly, 60% attendees solo)."[ppl-ai-file-upload.s3.amazonaws]
- Each stop includes: Provenance, social scaffolding cues, cultural context, lore layer[ppl-ai-file-upload.s3.amazonaws]

Stage 5: Experience & Residue

- User completes experience, returns to app
- Cultural autobiography updates: "1 new neighborhood (Basavanagudi), 1 interest pod joined (pottery circle)"[ppl-ai-file-upload.s3.amazonaws]
- Collision suggestion: "Since you loved hand-thrown pottery, try this handloom weaving + indie music session"

Functional Requirements

FR1: Multimodal Experience Ingestion (Critical Path)

- **Inputs**: Text, voice, Instagram URL, YouTube URL, image upload
- **Gemini Vision**: Extract experiences from Reel video frames[cerebralvalley]
- **Text Parsing**: Long-form blog posts, Substack articles
- **Success Criteria**: <5s extraction time, 90%+ accuracy on venue identification[ppl-ai-file-upload.s3.amazonaws]

FR2: Hyperlocal Discovery Engine (Differentiator)

- **Data Sources**:
  - Social: Instagram creator pages, YouTube vlogs, Reddit and X
  - Community: Luma, Partiful, Telegram groups
  - Artisan: Government craft registries, cooperative, Government tourist websiteswebsites[ppl-ai-file-upload.s3.amazonaws]
  - Architecture: Small bloggers who write

- **Discovery Agent**: Multi-source scraping, vector similarity (Gemini embeddings)
- **Success Criteria**: 200+ Bangalore experiences in database, 30%+ are "hyperlocal" (not on Google Maps)[ppl-ai-file-upload.s3.amazonaws]

FR3: Cultural Context Localization (Gemini Pro Strength)

- **Beyond Translation**: Timing nuances, religious considerations, transport hacks, social norms[ppl-ai-file-upload.s3.amazonaws]
- **Cultural Context Agent**: Gemini Pro with India-specific prompting
- **Success Criteria**: 100% of experiences have timing/dress code/solo-sure annotations

FR4: Plot-First Itinerary Generation (Innovation Core)

- **Narrative Arc**: Setup → friction → payoff (not just chronological list)[ppl-ai-file-upload.s3.amazonaws]
- **Lore Layering**: Backstory, cultural significance, memory-making cues[ppl-ai-file-upload.s3.amazonaws]
- **Collision Suggestions**: Cross-pod recommendations (pottery + spoken word + Korean BBQ)[ppl-ai-file-upload.s3.amazonaws]
- **Plot-Builder Agent**: Gemini Pro for creative storytelling
- **Success Criteria**: Demo judges rate narrative quality 8/10+

FR5: Budget & Booking Intelligence (Table Stakes)

- **Cost Structure**: Entry fees, average spend, transport costs
- **Proactive Recommendations**: "Book trek 3 days ahead (sells out fast)"[ppl-ai-file-upload.s3.amazonaws]
- **Budget Agent**: Gemini Flash for fast analysis
- **Success Criteria**: ₹200-5000 budget range supported, <2s cost calculation

FR6: Solo-Sure & Social Scaffolding (Market Fit)

- **Experience Tagging**: Solo-friendly, conversation-optional, beginner-energy[ppl-ai-file-upload.s3.amazonaws]
- **Scaffolding Indicators**: "Chef-interactive counter seating (easy conversation)"[ppl-ai-file-upload.s3.amazonaws]
- **Community Agent**: Analyzes social dynamics
- **Success Criteria**: 50%+ experiences tagged solo-sure, 80%+ accuracy on social cues

FR7: Experience Residue Tracking (Long-term Engagement)

- **Cultural Autobiography**: Map visualization of neighborhoods explored[ppl-ai-file-upload.s3.amazonaws]
- **Interest Pods**: Track food nerd, run club, craft explorer evolution[ppl-ai-file-upload.s3.amazonaws]

- **Proof-of-Life Timeline**: Visual story (not for social posting)[ppl-ai-file-upload.s3.amazonaws]
- **Success Criteria**: User returns 3+ times to see autobiography growth

Technical Architecture

Agent System: LangGraph Supervisor Pattern[youtube][changelog.langchain]

text

```
Coordinator (Supervisor)
├── Discovery Agent (Gemini Flash - speed)
│   ├── Social Media Scraper
│   ├── Community Event Crawler
│   └── Vector Search (Pinecone/Gemini embeddings)
├── Cultural Context Agent (Gemini Pro - reasoning)
│   ├── Localization KB
│   └── Real-time factors (weather, traffic)
├── Plot-Builder Agent (Gemini Pro - creativity)
│   ├── Narrative arc generation
│   └── Lore layer addition
├── Budget Optimizer (Gemini Flash - analysis)
│   └── Cost aggregation & deals
└── Community Agent (Gemini Flash - tagging)
    └── Solo-sure filtering
```

Why Supervisor Pattern: Hierarchical delegation enables parallel agent execution while maintaining coordination. Supervisor receives user input, routes to relevant agents, synthesizes outputs into coherent itinerary.[changelog.langchain][youtube][ppl-ai-file-upload.s3.amazonaws]

State Management

python

```python
class ExperienceState(TypedDict):
    # User inputs
    user_query: str
    social_media_urls: list[str]
    interest_pods: list[str]
    budget_range: tuple[int, int]
    solo_preference: bool

    # Agent outputs
    discovered_experiences: list[dict]
    cultural_context: dict
    narrative_itinerary: str
    budget_breakdown: dict

    # Metadata
    city: str
    agent_trace: list[dict]  # For demo visualization
```

Technology Stack (6-Hour Optimized)

Layer	Technology	Rationale
AI Models	Gemini 3 Pro, Flash, Vision [cerebralvalley]	Multimodal, fast, cultural reasoning
Agent Framework	LangGraph Supervisor [youtube][changelog.langchain]	Pre-built hierarchical coordination
Backend	Python + FastAPI	Team expertise, rapid development
Database	MongoDB Atlas (free tier)	Document store for experiences
Vector Search	Gemini embeddings + in-memory	No external dependencies
Frontend	Next.js + shadcn/ui	Fast deployment to Vercel
Deployment	Vercel (frontend) + Railway (backend)	One-click deploy, free tiers
Observability	LangSmith (optional) [ppl-ai-file-upload.s3.amazonaws]	Agent trace visualization for demo

Performance Targets (Relaxed for Hackathon)

- Social media extraction: <10s (acceptable for demo)
- Full itinerary generation: <30s end-to-end[ppl-ai-file-upload.s3.amazonaws]
- Concurrent users: 10+ without degradation (demo-sufficient)

Non-Functional Requirements

Scalability: Designed for post-hackathon growth with modular agents
Reliability: Cached fallbacks if APIs fail[ppl-ai-file-upload.s3.amazonaws]
Security: No PII storage, encrypted API keys
Demo-Readiness: Real-time agent visualization for judges[ppl-ai-file-upload.s3.amazonaws]

🏗️ DELIVERABLE 5: AGENT ARCHITECTURE SPECIFICATION

System Architecture Diagram

text

```
┌─────────────────────────────────────────────────────────────┐
│                    USER INPUT                               │
│  - Text query                                               │
│  - Instagram/YouTube URL                                    │
│  - Voice command                                            │
│  - Image upload                                             │
└────────────────┬────────────────────────────────────────────┘
                 │
                 ▼
┌─────────────────────────────────────────────────────────────┐
│          LANGGRAPH SUPERVISOR (Coordinator)                 │
│  - Routes to appropriate agents                             │
│  - Manages state across agent calls                         │
│  - Synthesizes final output                                 │
│  Model: Gemini 3 Flash (fast routing decisions)           │
└───────┬─────────────────────────────────────────────────────┘
        │
        ├──────────────┬──────────────┬──────────────┬──────────────┐
        │              │              │              │              │
        ▼              ▼              ▼              ▼              ▼
    ┌─────────┐  ┌───────────┐  ┌────────────┐  ┌──────────┐
┌──────────┐
    │Discovery│  │Cultural   │  │Plot-       │  │Budget    │
│Community │
```

```
    │ Agent   │   │Context    │  │Builder     │  │Optimizer │  │ Agent
│
    │         │   │Agent      │  │Agent       │  │Agent     │  │
│
    └─────────┘   └───────────┘  └────────────┘  └──────────┘
└───────────┘
        │              │              │              │             │
        │              │              │              │             │
    Gemini          Gemini        Gemini         Gemini        Gemini
    Flash           Pro           Pro            Flash         Flash
        │              │              │              │             │
        ▼              ▼              ▼              ▼             ▼
    ┌─────────┐   ┌───────────┐  ┌────────────┐  ┌──────────┐
┌───────────┐
    │Social   │   │Localize   │  │Narrative   │  │Cost      │  │Solo-Sure
│
    │Media    │   │KB         │  │Arc         │  │Analysis  │  │Filtering
│
    │Scraper  │   │           │  │Generation  │  │          │  │
│
    │Vision   │   │Real-Time  │  │Lore Layer  │  │Deals     │  │Social
│
    │Extract  │   │Factors    │  │Collision   │  │Discovery │  │Scaffold
│
    └─────────┘   └───────────┘  └────────────┘  └──────────┘
└───────────┘
        │              │              │              │             │
        └──────────────┴───────────────┴──────────────┴─────────────┘
                                  │
                                  ▼
                    ┌────────────────────────────┐
                    │   NARRATIVE ITINERARY       │
                    │  - Story-driven arc         │
                    │  - Cultural context         │
                    │  - Lore & provenance        │
                    │  - Solo-sure indicators     │
                    │  - Budget breakdown         │
                    └────────────────────────────┘
```

Agent Specifications

1. Discovery Agent

Purpose: Find experiences across social media, community platforms, hyperlocal sources[ppl-ai-file-upload.s3.amazonaws]

Model: Gemini 3 Flash (prioritize speed over depth)

Inputs:

- `user_query`: str (e.g., "solo pottery workshop")
- `social_media_url`: Optional[str] (Instagram/YouTube)
- `extracted_text`: Optional[str] (from Vision Agent)
- `budget_range`: tuple[int, int]
- `city`: str

Processing:

1. If social_media_url: Extract text/metadata
2. Search experiences database (vector similarity)
3. Filter by city, budget, category
4. Score relevance + novelty (collision potential)
5. Return top 10 candidates

Outputs:

python

```
{
  "discovered_experiences": [
    {
      "name": "Atta Galatta Pottery Workshop",
      "category": "craft",
      "timing": "Saturday 11am-1pm",
      "budget": 800,
      "solo_friendly": true,
      "location": "Koramangala",
      "source": "instagram:@attagalatta"
    }
  ],
  "search_metadata": {
    "total_found": 23,
    "sources": ["instagram", "luma", "artisan_db"],
    "execution_time": 2.3
  }
}
```

Error Handling: If no results, suggest nearby cities or related categories

---

2. Cultural Context Agent

Purpose: Add India-specific localization beyond translation[ppl-ai-file-upload.s3.amazonaws]

Model: Gemini 3 Pro (requires cultural reasoning)

Inputs: `discovered_experiences` from Discovery Agent

Processing (Gemini Pro Prompt):

text

```
For each experience, add:

1. **Optimal Timing**: When locals go, cultural significance of timing
   (e.g., "9-11am coffee culture peak in Bangalore")

2. **Dress Code & Etiquette**: Temple visits, upscale dining, workshop
attire

3. **Transport Hacks**: Auto negotiation tips, metro shortcuts,
parking reality

4. **Social Norms**: Solo dining accepted? Conversation culture?
Photography rules?

5. **Religious/Cultural Considerations**: Festival timing, Ramadan
adjustments

6. **Safety & Accessibility**: Well-lit evening, wheelchair access,
women-solo-friendly
```

```
Experiences: {experiences}
City: {city}
```

Outputs:

python

```
{
  "cultural_context": {
    "CTR Benne Dosa": {
      "timing": "7-11am peak (locals breakfast culture)",
      "dress": "Casual, temple-adjacent so modest",
      "transport": "Auto from Lalbagh ₹80, walking distance from Bull Temple",
      "social": "Counter seating = conversation-friendly, 40% solo diners",
      "safety": "Very safe, family crowd"
    }
  }
}
```

---

3. Plot-Builder Agent (Innovation Core)

Purpose: Generate narrative itineraries with emotional arcs, not chronological lists[ppl-ai-file-upload.s3.amazonaws]

Model: Gemini 3 Pro (creative storytelling)

Inputs:

- `discovered_experiences`
- `cultural_context`
- `user_interest_pods`: list[str] (e.g., ["food_nerd", "craft_explorer"])

Processing (Gemini Pro Prompt):

text

```
Create a plot-first itinerary with narrative arc:
```

```
**Principles**:
1. **Setup → Friction → Payoff**: Story structure, not just timeline
2. **Intentional Friction**: Queueing, trekking, learning = memory-making
3. **Lore Layering**: Backstory, provenance, "why this matters"
4. **Collision Suggestions**: Mix interest pods (pottery + food + music)
5. **Time-Fluid**: Dawn + evening in same day (dual prime times)

**Tone**: Evocative but grounded, "you" voice, sensory details

**Output Format**:
- Opening hook (why this day matters)
- 3-5 experiences with:
  - Time + place
  - Friction element ("queue 20 mins - locals gather here")
  - Lore ("80-year-old institution, unchanged recipe")
  - Social scaffolding ("counter seating, easy conversation starters")
- Closing reflection (what you'll remember)

Experiences: {experiences}
Context: {context}
User pods: {interest_pods}
```

Outputs:

python

```
{
  "narrative_itinerary": """
Your Saturday begins where Bangalore's heritage coffee culture still lives...

9:00 AM — CTR, Malleswaram
Start at Mavalli Tiffin Room's benne dosa. Yes, you'll queue 20 minutes—
locals have been gathering here since 1924, and the wait is part of the
ritual. The friction earns you a table. Counter seating = easy
conversation
with strangers who become breakfast companions.

*Solo-sure: 40% diners arrive alone. Order at counter, seat yourself.*

10:30 AM — Basavanagudi Heritage Walk
[continues with lore, friction, provenance...]
  """,

  "collision_suggestion": {
    "title": "Your next adventure (pottery + music)",
    "experiences": ["Atta Galatta Workshop", "Sofar Sounds
Bangalore"],
    "why": "You're building a craft explorer + music nerd identity
pod"
  }
}
```

---

4. Budget Optimizer Agent

Purpose: Cost transparency, deals discovery, booking timeline[ppl-ai-file-upload.s3.amazonaws]

Model: Gemini 3 Flash (fast numerical analysis)

Inputs: `discovered_experiences`, `budget_range`

Processing:

1. Aggregate costs: Entry + avg spend + transport + hidden costs
2. Check for deals: Early bird, BNPL, group discounts
3. Booking urgency: "Book trek 3 days ahead (sells out)"[ppl-ai-file-upload.s3.amazonaws]
4. Alternative suggestions if over budget

Outputs:

python

```
{
  "budget_breakdown": {
    "total_estimate": 1250,
    "breakdown": [
      {"experience": "CTR", "cost": 150, "type": "meal"},
      {"experience": "Pottery", "cost": 800, "booking_required": "2 days ahead"},
      {"transport": 300}
    ],
    "deals": ["BNPL available for pottery workshop via Simpl"],
    "within_budget": true
  }
}
```

---

5. Community Agent

Purpose: Solo-sure filtering, social scaffolding cues, ambient belonging[ppl-ai-file-upload.s3.amazonaws]

Model: Gemini 3 Flash (pattern matching)

Inputs: `discovered_experiences`

Processing:

1. Analyze social dynamics: Group-assumed vs solo-friendly
2. Tag beginner energy vs expertise-welcome
3. Conversation scaffolding: "Chef-interactive counter" vs "quiet contemplation"
4. Predict solo attendance % (based on experience type)

Outputs:

python

```
{
  "social_scaffolding": {
    "CTR": {
      "solo_friendly": true,
      "solo_percentage": "40%",
      "scaffolding": "Counter seating enables easy conversation with strangers",
      "arrival_vibe": "Autonomous confidence - locals respect solo diners",
      "beginner_energy": "Medium - ordering can be intimidating, but staff helpful"
    },
    "Pottery Workshop": {
      "solo_friendly": true,
      "solo_percentage": "60%",
      "scaffolding": "Hands-on activity = natural conversation starter",
      "arrival_vibe": "Explicitly beginner-friendly, instructor facilitates intros",
      "beginner_energy": "High - designed for first-timers"
    }
  }
}
```

Supervisor Coordination Flow

LangGraph Supervisor Pattern:[changelog.langchain][youtube]

python

```python
from langgraph_supervisor import create_supervisor

# Define agents as subgraphs
agents = {
    "discovery": discovery_agent,
    "cultural_context": cultural_context_agent,
    "plot_builder": plot_builder_agent,
    "budget": budget_optimizer_agent,
    "community": community_agent
}

# Create supervisor with Gemini 3 Flash for routing
supervisor = create_supervisor(
    agents=agents,
    model="gemini-3-flash",
    output_mode="final"  # Only return final itinerary
)

# State flows through agents sequentially
# Supervisor decides routing based on user query
```

Execution Flow:

1. User query → Supervisor analyzes intent
2. Supervisor routes to Discovery Agent
3. Discovery outputs fed to Cultural Context + Community (parallel)
4. Combined outputs → Plot-Builder Agent
5. Budget Optimizer runs final cost check
6. Supervisor synthesizes narrative_itinerary + metadata
7. Return to user with agent trace for demo

Why This Works for Hackathon:

- Pre-built coordination (don't write routing logic)[changelog.langchain]
- Transparent tracing (show judges agent collaboration)[ppl-ai-file-upload.s3.amazonaws]
- Modular (can add/remove agents easily)
- Gemini-native (all agents use Gemini models)[cerebralvalley]
