# AMINUTEMAN TECHNOLOGIES PVT LTD
# EXAMINATION TASK ASSESSMENT

**Multi-Agent AI Voice System Implementation Exam**

**Food Delivery Platform (DoorDash-Style)**

**Duration: 8 Days | Production-Ready System**

## Problem Statement

You are tasked with building a complete multi-agent AI voice calling system for a food delivery platform similar to DoorDash. The system must handle customer orders, restaurant coordination, delivery driver management, and customer support through intelligent voice agents with real-time CRM integration and order orchestration.

## System Requirements

### Core Business Flows to Implement

#### 1. Customer Order Placement (Inbound/Outbound)

- Voice-based order taking with natural conversation
- Menu navigation and item selection using voice
- Handle customizations and special dietary requests
- Address verification with Google Maps integration
- Real-time payment processing
- Order confirmation and ETA communication

#### 2. Restaurant Order Coordination (Outbound)

- Automated order notification calls to restaurants
- Confirmation of preparation time
- Handle inventory issues (out of stock items)
- Order modification coordination
- Pickup time reminders

#### 3. Delivery Driver Assignment (Outbound)

- Automated driver notification system
- Route optimization communication
- Pickup and delivery confirmation
- Real-time issue handling

**4. Customer Support & Updates (Inbound/Outbound)**

- Handle "where is my order" inquiries
- Process modification and cancellation requests
- Complaint resolution and refund processing
- Proactive delay notifications

**5. Post-Delivery Follow-up (Outbound)**

- Automated delivery confirmation
- Feedback and rating collection
- Issue resolution within 24 hours
- Promotional offer delivery

**Technical Implementation Tasks**

**Section 1: Agent Architecture Design**

**Task 1.1: Design Multi-Agent System** Create a complete agent architecture with the following agents:

1. **Customer Order Agent**
    - Capabilities: Menu search, item addition, customization, address verification, payment
    - Decision points: When to upsell, when to suggest alternatives, when to transfer
    - Context requirements: Customer history, preferences, current order state
    - Tool integrations: Menu search, CRM, payment gateway, maps API
2. **Restaurant Coordination Agent**
    - Capabilities: Order notification, time confirmation, inventory handling
    - Decision points: When to suggest substitutions, when to escalate delays

- Context requirements: Restaurant profile, current queue, historical prep times
- Tool integrations: Restaurant database, order management

system 3. **Driver Assignment Agent**

- Capabilities: Driver search, route communication, acceptance confirmation
- Decision points: Which driver to assign, incentive calculation, reassignment logic
- Context requirements: Driver locations, ratings, availability, traffic

data ○ Tool integrations: GPS tracking, route optimization, driver database 4.

**Delivery Tracking Agent**

- Capabilities: Real-time status updates, delay notification, address correction
- Decision points: When to notify customer, how to handle delivery

failures ○ Context requirements: Live driver location, ETA, customer availability

○ Tool integrations: GPS tracking, traffic API, customer notification 5. **Customer Support Agent**

- Capabilities: Refund processing, complaint handling, issue escalation
- Decision points: Refund vs credit, escalate to human, compensation amount
- Context requirements: Order history, customer LTV, issue

severity ○ Tool integrations: Payment gateway, ticketing system,

CRM

6. **Post-Delivery Agent**

- Capabilities: Satisfaction confirmation, feedback collection, promotion delivery
- Decision points: When to offer promotions, when to escalate

issues ○ Context requirements: Delivery status, customer feedback

history ○ Tool integrations: Review system, promotions database

**Deliverable:** Complete agent architecture document with interaction flows

**Task 1.2: Design State Management System** Implement a robust state management system that:

- Maintains conversation context across agent transitions
- Stores order state (items, pricing, delivery address)
- Preserves customer profile information
- Handles concurrent state updates safely
- Implements session persistence with appropriate TTL

**Requirements:**

- Use Redis for active session storage
- Design schema for session data
- Implement context handoff mechanism between agents
- Handle session recovery after disconnections
- Clean up expired sessions automatically

**Deliverable:** State management system design and Redis schema

**Task 1.3: Agent Orchestration Logic** Build the orchestration layer that:

- Routes incoming calls to appropriate agents
- Handles agent-to-agent transitions seamlessly
- Maintains conversation continuity during handoffs
- Implements load balancing across agent instances
- Provides failover mechanisms

**Requirements:**

- Intent classification system
- Agent capability registry
- Dynamic routing based on context
- Handoff summary generation
- Transition logging for analytics

**Deliverable:** Working orchestrator with routing logic

**Section 2: Voice Infrastructure Implementation**

**Task 2.1: Twilio Voice Integration** Implement complete Twilio voice infrastructure: **Inbound Call Handling:**

- Webhook endpoint to receive calls
- Call SID tracking and session creation
- Caller ID lookup and customer identification
- Media stream initialization

**Outbound Call Handling:**

- Programmatic call initiation
- Machine detection (voicemail detection)
- Call status tracking
- Retry logic for failed calls

**Media Streaming:**

- WebSocket connection for real-time audio
- Bidirectional audio streaming
- Call recording with consent
- Call quality monitoring

**Requirements:**

- Handle 100+ concurrent calls
- <500ms connection establishment time
- Proper error handling and retries
- TRAI/TCPA compliance implementation

**Deliverable:** Complete Twilio integration with webhooks

**Task 2.2: Real-Time Audio Processing Pipeline** Build audio processing system with: **Speech-to-Text (Deepgram):**

- Real-time streaming transcription
- Interim and final results handling
- Voice Activity Detection (VAD)
- Silence detection and utterance completion
- Multi-language support (English + Hindi/Tamil)

**Text-to-Speech (ElevenLabs):**

- Natural voice synthesis
- Streaming audio output
- Low-latency response (<500ms)
- Emotion and tone control
- Voice consistency across agents

**Audio Processing Features:**

- Noise reduction
- Echo cancellation
- Volume normalization
- Barge-in/interruption handling

**Requirements:**

- Total STT latency <300ms
- Total TTS latency <500ms
- Handle audio quality issues gracefully
- Support 8kHz and 16kHz sample rates

**Deliverable:** Working audio processing pipeline

**Task 2.3: Interrupt and Barge-in Handling** Implement sophisticated interrupt handling:

- Detect user speech while AI is speaking

- Immediately stop current TTS output
- Clear any queued responses
- Resume listening for user input
- Maintain conversation context

**Requirements:**

- <200ms interrupt detection time
- Graceful cancellation of ongoing speech
- No audio artifacts during interruption • Context preservation

**Deliverable:** Interrupt handling system

**Section 3: LLM Integration & Agent Intelligence Task 3.1: Choose and Implement LLM Strategy Option A: API-Based (For 8-Day Timeline)**

- Primary: Claude Sonnet 4.5 via Anthropic API • Fallback: GPT-4o-mini for cost optimization • Function calling for tool execution
- Streaming responses for lower latency

**Option B: Self-Hosted SLM (For Future Scale)** • Primary: Llama 3.3 70B Instruct

- Lightweight: Qwen 2.5 14B for simple agents
- Fine-tuning strategy for domain adaptation • vLLM for optimized inference

**Requirements:**

- Function calling support
- <1.5s response time

- Cost per call <$0.05 for LLM portion
- Fallback mechanism for API failures

**Decision:** Document which approach you're taking and why

**Deliverable:** Working LLM integration with function calling

**Task 3.2: Prompt Engineering for Each Agent** Create production-ready system prompts for all 6 agents:

**Each prompt must include:**

- Clear role definition and capabilities
- Conversation style guidelines
- Tool usage instructions
- Transfer criteria to other agents
- Error handling instructions
- Example conversation flows

**Quality Requirements:**

- Consistent personality across agents
- Natural, conversational tone
- Efficient (minimize unnecessary back-and-forth)
- Handle edge cases and ambiguity
- Never hallucinate information

**Deliverable:** Complete prompt library for all agents

**Task 3.3: Function/Tool Registry** Implement comprehensive tool system: **Customer-facing tools:**

- get_customer_profile(phone_number)
- search_menu(restaurant_id, query, filters)
- get_item_details(item_id)
- add_to_order(item_id, quantity, customizations)

- remove_from_order(item_id)
- verify_address(address)
- calculate_total(apply_promotions)
- place_order(payment_method_id)
- get_order_status(order_id)
- process_refund(order_id, reason, amount)

**Restaurant tools:**

- notify_restaurant(restaurant_id, order_details)
- confirm_preparation_time(order_id, minutes)
- handle_unavailable_item(order_id, item_id)
- update_restaurant_status(restaurant_id, status)

**Driver tools:**

- find_available_drivers(location, radius)
- assign_driver(driver_id, order_id)
- send_pickup_details(driver_id, restaurant_info)
- update_driver_location(driver_id, coordinates)
- confirm_delivery(order_id)

**Requirements:**

- Type-safe function definitions
- Input validation
- Error handling
- Logging and monitoring
- Response caching where appropriate

**Deliverable:** Complete tool registry with implementations

**Section 4: Database & Data Management**

**Task 4.1: Database Schema Design** Design and implement PostgreSQL schema:

**Required Tables:**

- customers (profile, addresses, payment methods)
- restaurants (profile, menu, operating hours) • menu_items (details, pricing, availability) • orders (items, pricing, status, timeline)
- drivers (profile, location, availability)
- call_sessions (transcript, metrics, outcomes) • agent_transitions (handoff tracking)

**Requirements:**

- Proper indexing for performance
- Foreign key relationships
- JSONB for flexible data (addresses, customizations) • Timestamp tracking for all records
- Support for soft deletes

**Deliverable:** Complete database schema with migrations

**Task 4.2: CRM Integration** Build CRM integration system: • Real-time customer data retrieval during calls • Order history and preferences lookup
- Address and payment method management
- Post-call data synchronization
- Customer lifetime value calculation

**Requirements:**

- <100ms lookup time
- Caching strategy for frequent lookups
- Webhook support for real-time updates
- Error handling for CRM unavailability

**Deliverable:** Working CRM integration

**Task 4.3: Vector Database for Semantic Search** Implement Qdrant vector database for:

- Menu item semantic search ("something spicy and healthy")
- Restaurant recommendations
- FAQ and knowledge base for support agent
- Similar order suggestions

**Requirements:**

- Generate embeddings using OpenAI ada-002
- Create collections for menu items and FAQs
- Implement hybrid search (semantic + filters)
- Cache frequent queries

**Deliverable:** Vector DB integration with semantic search

**Section 5: Order Management & Payment Processing**

**Task 5.1: Order Lifecycle Management** Implement complete order flow: **Order States:**

- cart → placed → confirmed → preparing → ready → picked_up → in_transit → delivered → completed

**State Transitions:**

- Automated status updates
- Webhook notifications
- Real-time tracking
- Failure handling and rollbacks

**Requirements:**

- Atomic state transitions
- Event logging for audit trail
- Notification triggers at each state
- Handle concurrent updates

**Deliverable:** Order management system

**Task 5.2: Payment Processing** Integrate payment gateway (Stripe/Razorpay): • Store and retrieve payment methods securely

- Process payments during order placement
- Handle payment failures and retries
- Process refunds automatically
- Support multiple payment methods

**Requirements:**

- PCI compliance
- Encryption for sensitive data
- Idempotency for payment requests
- Detailed transaction logging

**Deliverable:** Payment processing system

**Task 5.3: Real-Time Order Tracking** Build tracking system: • Driver GPS location streaming

- ETA calculation with traffic data
- Proactive delay notifications
- Customer notification system (SMS/Email)
- Delivery proof capture

**Requirements:**

- Real-time location updates
- Accurate ETA using Google Maps API
- Geofencing for delivery confirmation
- Photo capture for proof of delivery

**Deliverable:** Order tracking system

**Section 6: Advanced Features & Optimization**

**Task 6.1: Sentiment Analysis** Implement real-time sentiment analysis: • Analyze customer tone during calls

• Detect frustration or dissatisfaction

• Trigger escalation to support

• Adjust agent response tone

• Track sentiment metrics

**Requirements:**

• Real-time analysis during conversation

• <100ms processing time

• Integration with agent decision-making

**Deliverable:** Sentiment analysis system

**Task 6.2: Call Summarization** Build automatic call summarization: • Generate concise summaries post-call

• Extract key information (order details, issues)

• Store in database for future reference

• Use for agent training and quality assurance

**Requirements:**

• Accurate extraction of key details

• Structured output format

• Integration with CRM

**Deliverable:** Call summarization feature

**Task 6.3: Cost Optimization** Implement cost reduction strategies: • Use cheaper TTS (Deepgram Aura) for confirmations • Switch to GPT-4o-mini for simple queries

- Cache frequent LLM responses
- Batch process non-urgent operations
- Optimize prompt lengths

**Target:** Reduce cost per call by 30%

**Deliverable:** Cost optimization report

**Task 6.4: Multi-Language Support** Implement support for 3+ languages:
- English (primary)
- Hindi
- Tamil
- Language detection
- Seamless switching

**Requirements:**

- Language-specific STT/TTS models
- Translated prompts
- Bilingual menu support

**Deliverable:** Multi-language system

**Section 7: Monitoring, Compliance & Security**

**Task 7.1: Monitoring Dashboard** Build comprehensive monitoring:
- Real-time call metrics (volume, success rate, duration)
- Agent performance tracking
- System health monitoring
- Cost tracking per call
- Error rate and alerting

**Tools:** Prometheus + Grafana

**Key Metrics:**

- Calls per minute

- Average call duration
- Agent success rates
- STT/TTS/LLM latency
- Order completion rate
- Customer satisfaction score

**Deliverable:** Grafana dashboard

**Task 7.2: Compliance**

**Implementation TRAI Compliance**

**(India):**

- DND registry check before calling
- Call only between 9 AM - 9 PM IST • Recording consent at call start
- Opt-out mechanism
- Complaint handling system

**TCPA Compliance (USA):**

- Prior express consent verification
- Opt-out in every call
- Respect calling hours (8 AM - 9 PM local) • Accurate caller ID

**GDPR Compliance (EU):**

- PII data encryption
- Right to access data
- Right to deletion
- Consent management
- Data minimization

**Deliverable:** Compliance system with documentation

**Task 7.3: Security Implementation** Implement security measures: • API rate limiting

• Webhook signature verification

• Data encryption at rest and in transit

• PII anonymization in logs

• Session token security

• SQL injection prevention

• XSS protection

**Deliverable:** Security audit checklist

**Section 8: Testing & Quality Assurance**

**Task 8.1: Unit Testing** Write comprehensive unit tests: • Agent response generation

• Tool execution

• State management

• Payment processing

• Order lifecycle

• Audio processing components

**Target:** >80% code coverage

**Deliverable:** Test suite with coverage report

**Task 8.2: Integration Testing** Test complete workflows: • End-to-end order placement

• Agent handoffs

• Payment processing

• Refund processing

• Multi-agent coordination

- Error handling

**Deliverable:** Integration test suite

**Task 8.3: Load Testing** Perform load testing:
- 100 concurrent calls baseline
- 500 concurrent calls stretch goal
- Identify bottlenecks
- Test failover mechanisms
- Measure response times under load

**Tools:** Locust or K6

**Deliverable:** Load test results and analysis

**Task 8.4: Call Quality Testing** Test conversation quality:
- Natural conversation flow
- Accurate speech recognition
- Natural TTS output
- Interruption handling
- Context preservation
- Edge case handling

**Deliverable:** Call quality report

**Section 9: Deployment & DevOps**

**Task 9.1: Containerization** Dockerize all services:

- FastAPI application
- Agent orchestrator
- Audio processor
- Background workers
- Database setup

**Deliverable:** Docker Compose setup

**Task 9.2: Kubernetes Deployment** Create Kubernetes manifests:

- Deployment configs
- Service definitions
- ConfigMaps and Secrets
- Horizontal Pod Autoscaler
- Ingress configuration
- Resource limits

**Deliverable:** K8s manifests

**Task 9.3: CI/CD Pipeline** Set up automated deployment:

- Automated testing on commit
- Docker image building
- Staging deployment
- Production deployment with approval
- Rollback capability

**Tools:** GitHub Actions or GitLab CI

**Deliverable:** Working CI/CD pipeline

**Section 10: Documentation & Demo**

**Task 10.1: Technical Documentation** Create comprehensive documentation:

- System architecture overview
- API documentation (OpenAPI/Swagger)
- Agent behavior documentation
- Database schema documentation
- Deployment guide
- Troubleshooting guide

• Runbook for operations

**Deliverable:** Complete documentation

**Task 10.2: Demo Preparation** Prepare system demonstration: • Record 15-minute demo video showing:

- Customer placing an order
- Restaurant coordination
- Driver assignment
- Order tracking
- Customer support interaction
- Post-delivery follow-up

• Show monitoring dashboards

• Demonstrate agent handoffs

• Show real-time state management

**Deliverable:** Demo video and presentation

**Task 10.3: Cost Analysis** Provide detailed cost breakdown: • Per-call cost calculation

• Monthly projections at different scales (10K, 100K, 500K calls) • Cost optimization strategies

• ROI analysis

• Break-even analysis for SLM migration

**Deliverable:** Cost analysis spreadsheet

**Technology Stack Specification**

**Voice Infrastructure**

• **Twilio Programmable Voice** - Call handling and media

streaming • Cost: $0.0085/min inbound, $0.0140/min outbound

**Speech Recognition**

- **Deepgram Nova-2** - Real-time STT
- Cost: $0.0043/min
- Latency: <300ms

**Text-to-Speech**

- **ElevenLabs** (Primary) - Natural voice synthesis
- Cost: $0.18/1K characters
- **Deepgram Aura** (Fallback) - Cost optimization
- Cost: $0.015/1K characters

**Language Models**

**Option A - API (Recommended for 8 days):**

- **Claude Sonnet 4.5** - Main LLM
- Cost: $3/$15 per 1M tokens
- **GPT-4o-mini** - Fallback
- Cost: $0.15/$0.60 per 1M tokens

**Option B - Self-Hosted (For scale):**

- **Llama 3.3 70B** - Main model
- **Qwen 2.5 14B** - Lightweight
- Infrastructure: 2x A100 GPUs

**Backend Framework**

- **Python 3.11+ with FastAPI** - Async API server
- **Redis** - Session and cache management
- **PostgreSQL 15** - Primary database
- **Qdrant** - Vector database

**Message Queue**

- **RabbitMQ** - Event processing

**Monitoring**

- **Prometheus** - Metrics
- **Grafana** - Dashboards
- **Sentry** - Error tracking
- **ClickHouse** - Analytics

**Deployment**

- **Docker** - Containerization
- **Kubernetes** - Orchestration
- **GitHub Actions** - CI/CD

**External APIs**

- **Google Maps API** - Address verification
- **Stripe/Razorpay** - Payment processing
- **Twilio SMS** - Notifications

**8-Day Implementation Schedule**

**Day 1: Foundation**

**Focus:** Infrastructure setup and basic call flow

- Set up all accounts (Twilio, Deepgram, ElevenLabs, Claude) • Initialize project structure
- Set up PostgreSQL and Redis
- Implement basic Twilio webhooks
- Test basic call answering

**Deliverables:**

- Working API server
- Basic call answering capability
- Database schema created

**Day 2: Audio Pipeline**

**Focus:** Real-time audio processing

- Implement Deepgram STT integration
- Build ElevenLabs TTS integration
- Create audio streaming pipeline
- Implement VAD and interruption handling • Test end-to-end audio flow

**Deliverables:**

- Working STT/TTS pipeline
- Interrupt handling functional
- Audio quality validated

**Day 3: Agent Framework & Customer Order**

**Agent Focus:** Core agent infrastructure

- Build base agent class
- Implement LLM service wrapper
- Create tool registry system
- Build state management
- Implement customer order agent
- Build menu search with vector DB

**Deliverables:**

- Base agent framework
- Customer order agent working
- Menu search functional

**Day 4: Restaurant & Driver**

**Agents Focus:** Multi-party coordination

- Build restaurant coordination agent • Implement driver assignment agent •

Create order notification system •

Build driver search algorithm • Test

agent coordination

**Deliverables:**

- Restaurant agent working
- Driver agent working
- Order lifecycle automated

**Day 5: Support & Tracking**

**Focus:** Customer service

features • Build delivery

tracking agent

- Implement customer support agent •

Create refund processing

- Build proactive notification system •

Implement real-time ETA updates

**Deliverables:**

- Tracking agent operational
- Support agent handling issues
- Refund automation working

**Day 6: Advanced Features**

**Focus:** Polish and

optimization • Build

post-delivery agent

- Implement sentiment analysis •

Add call summarization

- Create RAG for FAQ

- Build analytics dashboard
- Optimize costs

**Deliverables:**

- All 6 agents complete
- Analytics working
- Cost optimizations applied

**Day 7: Testing & Compliance**

**Focus:** Quality and compliance • Write unit tests

- Integration testing
- Load testing (100+ concurrent)
- Implement TRAI compliance • Add security measures
- Bug fixes

**Deliverables:**

- Test suite complete
- Compliance features working • System stable under load

**Day 8: Documentation & Deployment**

**Focus:** Production readiness

- Write all documentation
- Create deployment configs
- Set up CI/CD pipeline
- Deploy to staging
- Record demo video
- Prepare presentation

**Deliverables:**

- Complete documentation
- System deployed
- Demo video ready
- Presentation complete

**Success Criteria**

Your implementation will be evaluated on:

**Functional Requirements**

✓ All 6 agent types operational and coordinating seamlessly ✓ Complete end-to-end order flow working via voice ✓ Real-time CRM integration functional ✓ Payment processing working securely ✓ Order tracking with live updates ✓ Support and refund automation working

**Performance Requirements**

✓ Total response time <2.5 seconds ✓ System handles 100+ concurrent calls ✓ Call success rate >95% ✓ Agent handoff success rate >98% ✓ Order completion rate >80% ✓ STT accuracy >90%

**Business Requirements**

✓ Cost per call <$0.25 ✓ Natural conversation quality ✓ Customer satisfaction score >4/5 ✓ System uptime >99%

**Technical Requirements**
AMINUTEMAN TECHNOLOGIES PVT LTD
EXAMINATION TASK ASSESSMENT
09-11-2025

✓ Clean, maintainable code ✓ Comprehensive error handling ✓ Proper logging and monitoring ✓ Security best practices implemented ✓ Scalable architecture ✓ Complete documentation

**Compliance Requirements**

✓ TRAI compliance for India ✓ Recording consent implementation ✓ PII data protection ✓ Opt-out mechanisms

**Submission Requirements**

**Code Deliverables**

1. Complete source code in Git repository
2. Database migration scripts
3. Docker and Kubernetes configs
4. CI/CD pipeline configuration
5. Environment configuration templates

**Documentation Deliverables**

1. System architecture document
2. API documentation (Swagger/OpenAPI)
3. Database schema documentation
4. Agent behavior documentation
5. Deployment guide
6. Troubleshooting guide
7. Operations runbook

**Testing Deliverables**

1. Unit test suite with coverage report
2. Integration test suite
3. Load test results and analysis
4. Call quality testing report

**Demo Deliverables**

1. 15-minute demo video covering all

features 2. Live system demonstration capability

3. Presentation deck explaining architecture

**Analysis Deliverables**

1. Cost analysis spreadsheet
2. Performance benchmarks
3. Scalability analysis
4. Security audit checklist

**Evaluation Criteria**

**Code Quality (25%)**

- Clean, readable code
- Proper error handling
- Type hints and documentation
- Following best practices
- DRY principle
- Separation of concerns

**Architecture (25%)**

- Scalable design
- Proper component separation
- State management
- Database design
- API design
- Event-driven architecture

**Functionality (25%)**

- All features working
- Natural conversation flow
- Seamless agent handoffs
- Accurate speech recognition
- Quality TTS output
- Real-time responsiveness

**Production Readiness (15%)**

- Monitoring and alerting
- Error tracking
- Logging
- Security measures

- Compliance implementation
- Deployment automation

**Documentation (10%)**

- Complete technical docs
- Clear setup instructions
- API documentation
- Troubleshooting guides
- Code comments

**Tips for Success**

**Day 1-2: Get the basics rock solid**

- Don't move forward until basic call flow works perfectly
- Test audio quality extensively
- Ensure low latency from the start

**Day 3-4: Focus on agent intelligence**

- Invest time in prompt engineering
- Test edge cases thoroughly
- Make sure context preservation works

**Day 5-6: Build for scale**

- Think about performance from the start
- Implement caching strategically
- Monitor resource usage

**Day 7: Test everything**

- Don't skip load testing
- Test failure scenarios
- Verify compliance requirements

**Day 8: Polish and present**

- Good documentation is crucial

- Demo video should be professional
- Presentation should tell a story

**Common Pitfalls to Avoid**

1. **Latency creep** - Monitor and optimize latency from Day

1 2. **Poor error handling** - Every external API call can fail

3. **Weak prompts** - Invest time in prompt engineering 4.

**No caching** - Cache aggressively where appropriate 5.

**Ignoring costs** - Track costs from the beginning

6. **Bad state management** - Context loss breaks user

experience 7. **No monitoring** - You can't fix what you can't see

8. **Skipping tests** - Technical debt compounds

quickly 9. **Poor documentation** - Future you will

thank you

10. **Overengineering** - Focus on MVP first, iterate later

**Resources & References**

**Documentation**

- Deepgram: https://developers.deepgram.com/
- ElevenLabs: https://elevenlabs.io/docs
- Anthropic Claude: https://docs.anthropic.com/
- OpenAI: https://platform.openai.com/docs

**Tools**

- FastAPI: https://fastapi.tiangolo.com/
- Redis: https://redis.io/documentation
- PostgreSQL: https://www.postgresql.org/docs/
- Qdrant: https://qdrant.tech/documentation/

**Deployment**

- Docker: https://docs.docker.com/
- Kubernetes: https://kubernetes.io/docs/
- GitHub Actions: https://docs.github.com/actions

**Final Notes**

This is an ambitious 8-day project that requires:

- Strong technical skills in Python and async programming
- Experience with API integrations
- Understanding of voice AI systems
- Ability to work under time pressure
- Focus on MVP while maintaining quality

**Remember:** The goal is not perfection, but a working, production-ready system that demonstrates all core capabilities. Prioritize functionality over features, and scalability over optimization.

Good luck with your implementation!
AMINUTEMAN TECHNOLOGIES PVT LTD EXAMINATION TASK
ASSESSMENT 09-11-2025

**Submission Deadline**

**Date: 18th November**

Submit via:

- Git repository link
- Demo video link
- Documentation package
- Deployed system URL (if applicable)