MemoryAnchor - Product Requirements Document

Executive Summary

ProductName: MemoryAnchor

Version: 1.0
Team: Yash, Moulshree, Rahul, Vinay
DocumentDate: July 25, 2025
TargetCompletion: Hackathon Submission Deadline

MemoryAnchor is an agentic AI application designed to assist early-stage Alzheimer's patients in maintaining independence and emotional comfort through personalized memory support, routine reminders, and conversational guidance.

1. Problem Statement

1.1 Market Need

Early-stage Alzheimer's patients face significant challenges in:

- Maintaining daily routines and medication schedules
- Remembering family members, appointments, and personal information
- Managing anxiety and confusion from memory lapses
- Preserving independence while ensuring safety

1.2 Target Audience

PrimaryUsers:

- Early-stage Alzheimer's patients (mild cognitive impairment)
- Age range: 60-85 years
- Retain basic smartphone/tablet interaction capabilities

SecondaryUsers:

- Family caregivers and medical professionals
- Adult children managing parent care

2. Product Vision & Goals

2.1 Vision Statement

To create a deeply personalized, emotionally intelligent AI companion that helps early-stage Alzheimer's patients maintain their independence, dignity, and emotional well-being through proactive

memory and conversational support.

2.2 Success Metrics

HackathonJudgingCriteriaAlignment:

- **TechnicalExcellence:** Clean, robust code with minimal bugs
- **SolutionArchitecture:** Well-documented, maintainable codebase
- **InnovativeGeminiIntegration:** Creative API usage enhancing user experience
- **SocietalImpact&Novelty:** Meaningful solution for underserved population

ProductKPIs:

- User engagement frequency (daily interactions)
- Memory cue accuracy and relevance
- Reminder completion rates
- User emotional comfort scores (qualitative feedback)

3. Product Features & Requirements

3.1 Core Features (MVP)

3.1.1 MemoryManagementSystem

- **PersonalMemoryStorage:** JSON-based storage for family, events, routines
- **MemoryRetrieval:** Context-aware recall of personal information
- **MemoryCues:** Gentle prompts about people, places, and events

3.1.2 Routine&ReminderSystem

- **DailyRoutineReminders:** Meals, medications, appointments
- **ProactiveNotifications:** Based on habit tracking and patterns
- **Multi-modalAlerts:** Audio, visual, and text-based reminders

3.1.3 ConversationalInterface

- **NaturalLanguageProcessing:** Voice and text input support
- **EmotionallyIntelligentResponses:** Comforting, patient, reassuring tone
- **Context-AwareConversations:** Leveraging personal memory for familiarity

3.1.4 IntegrationCapabilities

- **EmailIntegration:** Gmail API for calendar and communication monitoring
- **CalendarSync:** Appointment and event tracking

- **Emergency Notifications:** Caregiver alerts for missed critical reminders

3.2 Technical Architecture

3.2.1 Agent-Based Architecture

```
Master Agent (Planner)
 ├── Memory Retrieval Agent
 ├── Reminder & Notification Agent
 ├── Conversation Agent
 └── Emergency Alert Agent
```

3.2.2 System Components

- **app.py:** Entry point and route handling
- **planner.py:** Intent parsing and task delegation
- **executor.py:** Task execution and Gemini API calls
- **memory.py:** Long-term and short-term memory management •

tools.py: Notification, email, and reminder utilities

3.2.3 Data Architecture

```
json

{
  "personal_memory": {
    "people": [{"name": "Anjali", "relation": "daughter", "notes": "visits every Sunday"}],
    "routines": [{"activity": "medication", "time": "08:00", "frequency": "daily"}],
    "appointments": [{"type": "doctor", "date": "2025-07-28", "provider": "Dr. Smith"}]
  }
}
```

3.3 Gemini API Integration

3.3.1 Core Usage Patterns

- **Intent Recognition:** Parse user input for memory queries vs. reminder requests
- **Response Generation:** Create empathetic, contextual responses
- **Memory Summarization:** Condense conversation history for context management
- **Emotional Support:** Generate comforting responses during confusion or distress

3.3.2 Prompt Engineering Strategy

- Strict grounding with factual user data

- Explicit instructions to avoid hallucination
- Context injection with relevant personal memories
- Emotional tone guidelines for patient interaction

4. User Experience & Workflows

4.1 Primary User Workflows

4.1.1 InformationStorageWorkflow

```
User Input: "My granddaughter Sarah visits on weekends"
⇓ Planner.py (intent: store_info)
⇓ Executor.py (updates memory.py)
⇓ Memory.py (logs: person, relation, schedule)
⇓ Gemini confirmation + contextual follow-up
```

4.1.2 MemoryRetrievalWorkflow

```
User Input: "What was my granddaughter's name again?"
⇓ Planner.py (intent: memory_query)
⇓ Memory Retrieval Agent (searches personal_memory)
⇓ Gemini generates warm, familiar response
⇓ Output: "Your granddaughter's name is Sarah!"
```

4.1.3 ReminderSettingWorkflow

```
User Input: "Remind me to call John at 3PM"
⇓ Intent parsing (reminder request)
⇓ Scheduler.py (APScheduler integration)
⇓ Notification at specified time
⇓ Confirmation and habit tracking
```

4.2 User Interface Requirements

- **Simple,LargeTextInterface:** Accessibility for vision impairments
- **VoiceInput/Output:** Hands-free interaction capability
- **ClearVisualIndicators:** Status, notifications, and confirmation states
- **EmergencyContactAccess:** One-touch caregiver communication

5. Technical Specifications

5.1 Technology Stack

- **Backend:** Python, Flask/FastAPI
- **AI Integration:** Google Gemini API
- **Scheduling:** APScheduler
- **Notifications:** Firebase Cloud Messaging
- **Email Integration:** Gmail API with OAuth 2.0
- **Storage:** JSON file-based (MVP), Vector database (future)
- **Authentication:** OAuth 2.0 for service integrations

5.2 Security & Privacy Requirements

- **Data Encryption:** All personal memory data encrypted at rest
- **API Security:** Secure Gemini API key management via environment variables
- **OAuth Implementation:** Secure email and calendar access
- **Local Storage Priority:** Minimize cloud data transmission
- **Emergency Access:** Secure caregiver notification protocols

5.3 Performance Requirements

- **Response Time:** < 2 seconds for memory queries
- **Notification Reliability:** 99%+ delivery rate for critical reminders
- **Offline Capability:** Basic functionality without internet connection
- **Battery Optimization:** Efficient mobile device resource usage

6. Risk Assessment & Mitigation

6.1 Technical Risks

6.1.1 LLM Hallucinations

Risk: AI generating incorrect personal information

Mitigation:

- Strict prompt engineering with factual grounding
- Memory verification against stored data
- Human-in-the-loop for critical information

6.1.2 Context Window Limitations

Risk: Loss of conversation context over time

Mitigation:

- Periodic conversation summarization Categorized memory storage
- Semantic search for relevant context retrieval

6.1.3 Notification Reliability

Risk: Missing critical medication or appointment reminders

Mitigation:

- Multi-channel notification system
- Redundant reminder protocols
- Caregiver escalation for missed acknowledgments

6.2 Privacy & Ethical Risks

Risk: Sensitive health data exposure through API usage

Mitigation:

- Local-first data storage approach
- Minimal PII transmission to external services
- Clear privacy disclosure and consent mechanisms

7. Development Plan & Milestones

7.1 Development Phases

Phase 1: Core MVP (Hackathon Submission)

Basic memory storage and retrieval

Simple reminder system
Gemini integration for conversational interface
Demonstration-ready prototype

Phase 2: Enhanced Features (Post-Hackathon)

Advanced notification system

Email and calendar integration
Habit tracking and proactive suggestions
Emergency caregiver alerts

Phase 3: Production Ready (Future Scope)

Vector database for semantic memory search

Mobile application development
Clinical validation and medical partnerships
Scalable cloud infrastructure

8. Success Criteria & Validation

8.1 Hackathon Success Metrics

- **TechnicalDemonstration:** Functional prototype with core features
- **CodeQuality:** Clean, documented, maintainable codebase
- **InnovationShowcase:** Creative Gemini API usage examples
- **ImpactPresentation:** Clear articulation of societal benefit

8.2 User Validation Methods

- **CognitiveAssessmentIntegration:** Memory screening questionnaire results
- **UsageAnalytics:** Interaction frequency and pattern analysis
- **Caregiver Feedback:** Family member satisfaction surveys
- **ClinicalMetrics:** Medication adherence and appointment attendance rates

9. Future Roadmap

9.1 Short-term Enhancements (3-6 months)

- Advanced habit learning algorithms
- Multi-language support
- Wearable device integration
- Enhanced emergency protocols

9.2 Long-term Vision (1-2 years)

- Clinical trial partnerships
- Healthcare system integration
- Advanced AI personality customization
- Expanded cognitive health monitoring

10. Appendix

10.1 Memory Screening Questionnaire Integration

The product incorporates a 25-question cognitive assessment tool covering:

- Identity and temporal orientation

- Short-term and long-term memory
- Spatial awareness and routine memory
- Behavioral and emotional indicators

ScoringSystem: 30-point scale with interpretation guidelines for cognitive function assessment.

10.2 Sample User Interactions

- "Remind me to call John tomorrow at 10."
- "What did I do yesterday afternoon?"
- "Who is Emma again?"
- "What's my next appointment?"
- "I'm feeling a bit confused."

This PRD serves as the foundational document for MemoryAnchor development and hackathon submission.