Therapeutic Chatbot - Vibe Coding Roadmap 🚀

Prerequisites Setup (Day 1-2)

Development Environment

bash

```bash
# Create project folder
mkdir therapeutic-chatbot
cd therapeutic-chatbot

# Create virtual environment
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate

# Install dependencies
pip install fastapi uvicorn python-multipart
pip install google-generativeai
pip install pinecone-client
pip install python-dotenv
pip install pydantic
```

API Keys Setup

1. **Google AI Studio**: Get your Gemini API key
2. **Pinecone**: Create free account, get API key
3. Create `.env` file:

env

```env
GEMINI_API_KEY=your_gemini_key_here
PINECONE_API_KEY=your_pinecone_key_here
PINECONE_ENV=your_pinecone_environment
```

Phase 1: Basic Chat System (Week 1)

Goal: Get a simple chatbot talking with one persona

Day 1-2: FastAPI Basic Setup

What you're building: A simple API that receives messages and sends responses

python

```python
# main.py - Your starting point
from fastapi import FastAPI
import google.generativeai as genai
import os
from dotenv import load_dotenv

load_dotenv()
genai.configure(api_key=os.getenv("GEMINI_API_KEY"))

app = FastAPI()

@app.get("/")
def read_root():
    return {"message": "Therapeutic Chatbot API is running!"}

@app.post("/chat")
def chat_endpoint(message: str):
    # This is where the magic happens
    return {"response": "Hello! I'm your therapeutic chatbot."}
```

Test it: Run with `uvicorn main:app --reload`

Day 3-4: Add Gemini Integration

What you're adding: Actual AI responses

python

```python
# Add this to your chat_endpoint
model = genai.GenerativeModel('gemini-2.0-flash-exp')
response = model.generate_content(message)
return {"response": response.text}
```

Day 5-7: Add Professional Therapist Persona

What you're adding: Your first persona with basic prompt engineering

Create a simple prompt template and test it extensively.

Phase 2: Persona System (Week 2)

Goal: Three working personas that feel different

Day 8-10: Persona Architecture

What you're building: A system to switch between different AI personalities

Key Files:

- `personas.py` - All your persona prompts
- `chat_handler.py` - Logic for handling different personas
- Update `main.py` - Add persona selection

Day 11-14: Develop Each Persona

Focus: One persona per day, lots of testing

1. **Professional Therapist**: Structured, CBT-focused
2. **Friendly Companion**: Warm, supportive
3. **Gen-Z Friend**: Casual, validating with current slang

Vibe Coding Tip: Chat with each persona yourself for hours. If it doesn't feel right to you, keep tweaking the prompts.

Phase 3: Memory & Context (Week 3)

Goal: Chatbot remembers conversations

Day 15-17: Basic Memory System

What you're building: Store and retrieve conversation history

Start Simple: Use JSON files before moving to Pinecone

- `memory_handler.py` - Save/load conversations
- Update chat logic to include conversation history

Day 18-21: Pinecone Integration

What you're adding: Professional vector database for memory

Key Concept: Each conversation gets stored as embeddings so the bot can remember context and past sessions.

Phase 4: Safety First (Week 4)

Goal: Crisis detection that actually works

Day 22-24: Basic Crisis Detection

What you're building: A safety net that catches concerning messages

python

```python
# crisis_detector.py - Start simple
crisis_keywords = ["suicide", "kill myself", "end it all", "not worth living"]

def check_for_crisis(message):
    for keyword in crisis_keywords:
        if keyword in message.lower():
            return True
    return False
```

Day 25-28: Enhanced Safety

What you're adding:

- Sentiment analysis
- Context-aware detection
- Proper crisis response protocols

Phase 5: Frontend Magic (Week 5-6)

Goal: Beautiful, usable chat interface

Day 29-35: React Chat Interface

What you're building: The user-facing chat app

Key Features:

- Persona selection cards
- Chat bubbles that look good
- Typing indicators
- Responsive design

Day 36-42: Polish & Connect

What you're adding:

- Connect frontend to your FastAPI backend
- Add loading states
- Error handling
- Mobile responsiveness

Vibe Coding Philosophy 🎯

The "Feel Right" Approach

1. **Chat with your bot constantly** - If responses feel off, fix them immediately
2. **One feature at a time** - Get each part working perfectly before moving on
3. **Test with real scenarios** - Use actual therapeutic conversations as test cases
4. **Iterate based on gut feeling** - If something feels weird, it probably is

Daily Vibe Check Questions

- Does this persona sound authentic?
- Would I want to talk to this bot if I was struggling?
- Is the safety system actually protective?
- Does the interface feel welcoming?

Week-by-Week Milestones

Week 1 ✅

- [ ] FastAPI running
- [ ] Gemini responding
- [ ] One persona working well

Week 2 ✅

- [ ] All three personas distinct and authentic
- [ ] Smooth persona switching
- [ ] Consistent personality maintenance

Week 3 ✅

- [ ] Conversation memory working
- [ ] Context awareness in responses
- [ ] Pinecone integration complete

Week 4 ✅

- [ ] Crisis detection functional
- [ ] Safety protocols tested
- [ ] Emergency response system

Week 5-6 ✅

- [ ] Frontend chat interface
- [ ] Full system integration
- [ ] User testing ready

Pro Tips for Vibe Coding Success

1. Start Extremely Simple

Don't try to build everything at once. Get one thing working perfectly, then add the next piece.

2. Use Real Test Data

Create a collection of real-ish therapeutic conversations to test against. Make your personas respond to actual mental health scenarios.

3. Iterate Like Crazy

Every time something feels "off," stop and fix it. Trust your instincts about what feels natural in conversation.

4. Document Your Prompts

Keep a separate file with all your prompt experiments. You'll want to A/B test different approaches.

5. Focus on the Experience

The goal isn't just a working chatbot—it's a chatbot that people actually want to talk to when they're struggling.

Emergency Resources Integration

Build in proper crisis resources from day one:

- National Suicide Prevention Lifeline: 988
- Crisis Text Line: Text HOME to 741741
- Local emergency services: 911

Success Metrics to Track

Week 1: Technical

- API response time < 3 seconds
- Zero crashes during testing
- Gemini integration working

Week 2: Personality

- Each persona feels distinct
- Responses stay in character

- Natural conversation flow

Week 3: Intelligence

- Remembers previous conversations
- Context-appropriate responses
- Progressive relationship building

Week 4: Safety

- 100% crisis keyword detection
- Appropriate escalation responses
- User safety prioritized

Week 5-6: Experience

- Intuitive interface
- Engaging user experience
- Ready for real user testing

Final Vibe Check

At the end of 6 weeks, you should have:

- A therapeutic chatbot you'd actually use
- Three distinct, helpful personas
- Rock-solid safety features
- A beautiful, functional interface
- The confidence to show it to others

Remember: The goal isn't perfection—it's creating something genuinely helpful that people would want to use during difficult times. Focus on making it feel human, safe, and caring above all else.