Project Title: AI Weather Chatbot

Project Summary

For my programming task, I built a weather chatbot using artificial intelligence to create a more adaptive and user-friendly experience. I used Claude, an AI assistant developed by Anthropic, not just for code generation but as a consistent collaborator throughout the process. Claude helped me write clean Python code with built-in type hints and error handling, and also supported me in outlining the overall architecture of the chatbot. One of the challenges I ran into was handling API timeouts, but with Claude's guidance, I implemented fallback strategies that allowed the app to keep functioning even when external services failed. I also designed the interface and menu structure to ensure users could navigate the app easily.

Approach to AI Prompting

To get the most out of the AI, I used a mix of prompting techniques. I started with a simple weather function and built up features step by step, including error handling, type annotations, and detailed documentation. This gradual approach helped me stay in control of the process. I also used clearly formatted templates to guide the AI when defining functions, which made the outputs more consistent. Whenever I faced a specific problem, like how to handle failed API calls, I made sure to explain the situation clearly so the AI could respond with focused and relevant solutions. For elements tied to user experience, I described my goals casually, which helped the AI align with what I had in mind. When working on more advanced features, I often asked for deeper explanations to understand how everything fit together.

Designing for Different Users

One of the features I was most excited to develop was the multi-mode interaction system. I created four distinct modes: a conversation-only option, a weather-only mode that gives basic forecasts, a visual mode that displays data through charts, and a full mode that combines all features. This structure gave users flexibility in how they wanted to interact with the chatbot, depending on their needs. Switching between modes was designed to be seamless, and the system could tell the difference between a casual question and a weather request without confusion. It was rewarding to see how these design choices led to a smoother and more personal user experience.

Handling Real-World Challenges

Building a dependable error-handling system was another major win. I created a three-level fallback sequence that began with HTTPS, switched to HTTP when necessary, and fell back on demo data if all else failed. This meant the chatbot never fully crashed, even when APIs were unavailable. It remained usable under any conditions, which is exactly what I hoped to achieve.

Creating that kind of stability helped the app feel like something people could rely on, not just experiment with.

What Could Be Improved

I could focus on expanding the chatbot's natural language processing skills. The current version works fine, but it could be stronger at understanding what users really mean. I would add features like intent classification and more accurate entity recognition to improve how the chatbot interprets requests involving locations, dates, or types of weather. It would also be useful to implement a way for the chatbot to remember previous parts of a conversation so it can respond appropriately to follow-up questions like "How about tomorrow?"

Ideas for Future Expansion

There are several other improvements I would love to implement. Introducing a caching system could cut down on repeated API calls, making everything faster. Auto-detecting a user's location would help tailor the experience from the very beginning without asking for extra input. I'd also like to make the charts more interactive by allowing users to zoom or hover for more details. And of course, offering multilingual support would make the app more accessible to users from different backgrounds.

Key Takeaways

One of the most important things I learned from this project was how valuable it is to treat AI as a genuine partner in development. When you engage with it thoughtfully and give it structured direction, it becomes much more than a tool. Planning the system thoroughly from the start, especially with the user journey in mind, made it easier to build something coherent and functional. I also saw how essential it is to design for real-world conditions. A solid fallback strategy kept things working smoothly, even when unexpected issues came up.

Technical Skills Gained

On the technical side, I got hands-on experience with API integration, advanced error handling, interface design across multiple modes, and natural language processing. I also worked with data visualisation in a way that made the output easier to understand. Many of the methods I used, like modular development and incremental collaboration with AI, are lessons I'll carry into future work. This project was more than just a coding exercise. It was a chance to create something that people could actually use and enjoy.

Final Reflection

This assignment gave me the opportunity to build something real with AI, not just lines of code, but a complete, thoughtful solution. Every piece, from the layout of the menu to the way different interaction modes worked together, was designed with the user in mind. It taught me that when you combine smart AI prompting with careful design and real testing, the result is something not only functional, but also meaningful.