Creating a mobile app for IoT using Python isn't common, as mobile apps are typically built with languages like Java, Kotlin (for Android) or Swift (for iOS). However, you can use Python for IoT-related tasks, such as developing a backend server or handling IoT device communication.

To proceed:

Choose the Mobile Platform: Decide if you want to create the app for Android or iOS, or both. You might consider using Kotlin or Swift for the app's front-end development.

Select an IoT Framework: For IoT functionality, you can use Python libraries like MQTT or WebSocket for communication between the app and IoT devices.

Create the Backend: Develop a Python server that handles communication with IoT devices. You can use frameworks like Flask or Django for this purpose.

Database Integration: If your app requires a database to store IoT data, use Python libraries like SQLAlchemy for database connectivity.

User Interface: Build the app's user interface using native languages like Kotlin or Swift. However, if you prefer using Python, you can explore frameworks like Kivy for cross-platform app development.

Connect Front-end and Back-end: Establish communication between the app's front-end and the Python back-end, which handles IoT interactions.

Testing and Debugging: Thoroughly test your app to ensure it functions correctly and securely. Debug and fix any issues that arise.

Deployment: Prepare your app for deployment on the Google Play Store (for Android) or Apple App Store (for iOS).

Remember that using Python for IoT may not be the most common approach for mobile app development, so you might face limitations and challenges. Consider other programming languages if you find Python less suitable for certain tasks in your project