1. **Plan Your MVP Development**

Now, focus on building your **Minimum Viable Product (MVP)**. Start with a **task list** or **roadmap** specifically for the MVP. Here's what you should prioritize:

**Backend (API):**

* Set up your **SQLite database** with tables for **Services**, **Bookings**, and **Users**.
* Create **API endpoints** for:
  + Listing available services.
  + Creating a new booking.
  + Fetching bookings for a user.
  + (Optional) basic user management functionality.
* Test your API using **Postman** or similar tools to ensure it's working before integrating it with the app.

**Frontend (MAUI App):**

* Create a simple **UI for listing services** and booking services.
* Integrate your **MAUI app** with the API using HttpClient.
* Implement basic **navigation** between screens (service listing, booking page, user details if necessary).
* Ensure that the UI is user-friendly and mobile-responsive.

1. **Create the Database Structure**

Start by designing the **database structure** for your MVP:

* **Tables**:
  + **Services** (e.g., ServiceId, ServiceName, Description, Price).
  + **Bookings** (e.g., BookingId, ServiceId, UserId, TimeSlot, Status).
  + **Users** (optional, depending on your MVP needs).
* Set up **Entity Framework** for **SQLite** and configure migrations to create these tables.

**Actionable Task:**

* Write the **initial migration** to set up your **SQLite database** for the MVP.

1. **Write and Test the Core API**

Focus on writing the core API endpoints for **Service Listing** and **Bookings**. The main goal here is to:

* Create endpoints that allow users to **fetch available services**.
* Build the **booking process** (e.g., selecting a service, booking a time slot).
* Implement basic **data validation** and **error handling**.

**Actionable Task:**

* Build, test, and deploy the **API** locally. Use tools like **Postman** to verify that the endpoints return the expected results.

1. **Develop the UI for the MAUI App**

Once your API is functional, start working on the **MAUI app**:

* Focus on creating a **clean, simple UI** for the MVP.
* Ensure that users can view available services and book them using the UI.
* Keep the UI minimal for now, focusing on functionality over design.

**Actionable Task:**

* Connect your **MAUI app** to the API and ensure that bookings are being processed and displayed correctly.

1. **Set Milestones and Keep Track**

Using your **PRINCE2-inspired plan**, divide your tasks into **manageable chunks** and set **clear milestones**:

* **Milestone 1**: API fully functional with core features (service listing, bookings).
* **Milestone 2**: Basic **MAUI app** UI integrated with the API.
* **Milestone 3**: Initial internal testing and bug fixing.
* **Milestone 4**: MVP deployment.

**Optional**: Use tools like **Trello**, **Notion**, or **GitHub Projects** to track tasks and progress.

1. **Prepare for Testing and Feedback**

As you approach MVP completion, you’ll want to prepare for:

* **Internal Testing**: Test all user flows (booking, service listing) to ensure there are no major bugs.
* **Gather Feedback**: If possible, have a few people test the app (e.g., your friend's girlfriend who inspired the project or other small business owners). Focus on usability and initial impressions.

**Actionable Task:**

* Set up **test cases** to ensure that booking functionality works as expected.

1. **Deployment (if ready)**

Once you’re satisfied with the MVP, you can consider **deploying**:

* Deploy the **API** (e.g., on **Azure**, **Heroku**, or another cloud platform).
* Publish the **app** to the App Store/Play Store for testing or internal release.

**Suggested Workflow:**

1. **Set up development environment** (1–2 days).
2. **Plan and implement database structure and API** (1 week).
3. **Develop and integrate the frontend (MAUI app)** (1–2 weeks).
4. **Test API and app integration** (ongoing).
5. **Deploy MVP** and gather feedback (target: December 11, 2024).

**Key Focus Areas:**

* **Start simple**: Focus on getting the core booking system working before moving to more complex features.
* **Keep testing**: Continuously test the API and the app to ensure everything works as expected.
* **Iterate**: Once the MVP is deployed, gather feedback, and make adjustments based on user input.

By following these steps, you’ll make steady progress towards launching your MVP and setting the foundation for the full version of **BookOrGetBooked**. Let me know if you want more detailed help on specific parts, like setting up the database, API structure, or app development!