**Freelancer and Prospect Management Telegram Bot Overview**

**1. Purpose of the Bot**

This bot is designed to:

1. **Register freelancers** and store their information.
2. **Collect prospects** from freelancers.
3. **Send reports** (daily, weekly, monthly) to the admin.
4. **Notify freelancers** to add prospects weekly.
5. **Allow the admin** to broadcast messages and verify freelancers.

It is built with **Python** and uses the **Telegram Bot API**.

**2. Admin and User Setup**

* **Admin ID:**
  + Only the admin can access special features.
  + Admin ID is set in the code: ADMIN\_ID = 1971668737.
* **Users:**
  + Freelancers register via the bot and can add prospects, see their profile, or download their prospect list.

**3. Excel Files Used**

1. **freelancers.xlsx** – stores all freelancer information:  
   | Full Name | Phone | City | National ID | TIN Number | Telegram ID | Date Registered | Verified |
2. **prospects.xlsx** – stores all prospect information:  
   | Freelancer Telegram ID | Prospect Name | Phone Number | Interest | Comment | Date Added |
3. **Officer files (officer1.xlsx … officer4.xlsx)** – used for assigning freelancers in a **round-robin** way among officers.

* The bot **creates these files automatically** if they do not exist.

**4. Conversation and User Flow**

The bot uses **Conversation Handlers**, which are structured multi-step dialogues:

**A. Registration**

* **Steps:**
  1. Name
  2. Phone
  3. City
  4. National ID
  5. TIN Number
* **Data saved to:** freelancers.xlsx
* **Checks:** prevents duplicate registration.

**B. Freelancer Menu**

After registration, a freelancer can:

1. **Add Prospect** – guides them to enter name, phone, interest, comment.
   * Prevents duplicates added in last 90 days.
   * Saves to prospects.xlsx.
2. **Download Prospect List** – generates a **PDF** of all their prospects.
3. **See Profile** – shows their name, Telegram ID, and verification status.

**C. Admin Menu**

* Download **all freelancers** or **all prospects**.
* Broadcast messages to all freelancers.
* Verify or unverify freelancers using commands /verify <telegram\_id> or /unverify <telegram\_id>.

**5. Assigning Freelancers to Officers**

* Uses a **round-robin algorithm**:
  + Counts current freelancers in each officer file.
  + Assigns new freelancer to the officer with the least assigned freelancers.
* Updates the corresponding officerX.xlsx file.

**6. Reports and Analytics**

**Daily Reports**

* Sent **every day at 6 AM**.
* Includes:
  + New freelancers (last 24 hours).
  + New prospects (last 24 hours).
  + Trend PDFs (weekly and monthly):
    - Graph of freelancers and prospects over time.
* Admin receives the reports as **PDFs and Excel files**.

**Weekly Prospect Reminder**

* Runs **every Monday at 8 AM**.
* Sends reminder to freelancers who have **not added a prospect in the last 7 days**.
* Helps maintain engagement and consistent data collection.

**7. File and Data Management**

* Excel files are used as the **primary database**.
* PDFs are generated using **FPDF** for reports and prospect lists.
* Trend graphs are generated using **Matplotlib** and then embedded into PDFs.

**8. Bot Implementation Details**

* Uses **python-telegram-bot** library for interactions.
* Uses **asyncio** for scheduling tasks (daily and weekly reminders).
* Conversation handlers ensure a **step-by-step flow** for registration and prospect entry.
* Buttons (inline and keyboard) are used for user-friendly navigation.

**9. Scheduling Tasks**

| **Task** | **When** | **How** |
| --- | --- | --- |
| Daily Admin Reports | 6 AM every day | scheduler() function waits until 6 AM, then sends reports. |
| Weekly Prospect Reminder | Monday 8 AM | weekly\_prospect\_reminder() calculates next Monday 8 AM and notifies freelancers. |

* Both tasks run **in the background** using asyncio.

**10. Security and Access Control**

* Only the **admin ID** can access admin features.
* Freelancers cannot access or modify other freelancer’s data.
* Telegram ID is used as **unique identifier**.

**11. Error Handling**

* The bot catches exceptions during:
  + Sending messages (e.g., user blocked bot).
  + Reading Excel files.
  + Duplicate entries.
* Errors are logged in console for admin review.

**12. How Leaders Can Use the Bot**

1. **Check daily trends:** Admin receives reports every day.
2. **Verify freelancers:** /verify <telegram\_id> to mark a freelancer as verified.
3. **Send messages to all freelancers:** Use the broadcast feature.
4. **Monitor prospect collection:** Weekly reminders keep freelancers on track.

**13. Next Steps / Recommendations**

* Add **data backup** for Excel files.
* Add **notification logs** to track delivery of weekly reminders.
* Optionally migrate to a **database** (like SQLite or PostgreSQL) for scalability.