I want you to generate a backend codebase using **Django + Django REST Framework (DRF)** with **PostgreSQL** for my startup called **MyConsultia**.

✅**Required Features**:

1. **Google OAuth2 Authentication**:
2. Use `dj-rest-auth` or `django-allauth` to integrate Google login.
3. Every user authenticated through Google must be saved in the PostgreSQL database.
4. **Custom User Profile**:
5. Each user has a profile containing:
   - `full_name`
   - `role` (choices: `CLIENT`, `EXPERT`, `ADMIN`)
   - `bio`
   - `profile_picture`
6. **Notification System**:
7. Internal notifications stored in the database
8. Email notifications sent automatically for important events like meeting creation
9. **Review System**:
10. Clients can leave a review on experts:
    - `reviewer` (FK)
    - `expert` (FK)
    - `rating` (1 to 5)
    - `comment`
    - `created_at`
11. **Schedule System**:
12. Experts define availability:
    - `available_day` (e.g., "Monday")
    - `start_time`
    - `end_time`
13. **Meeting System**:
14. Clients can book meetings with experts

15. Integrate **Google Calendar API** to automatically create calendar events
16. Send automated emails with meeting details

17. Meeting model fields:

    - `client` (FK)
    - `expert` (FK)
    - `day` (Date)
    - `start_datetime`
    - `end_datetime`
    - `google_event_id`
    - `status` (choices: `SCHEDULED`, `COMPLETED`, `CANCELED`)

18. **Security**:

19. Use JWT (via `djangorestframework-simplejwt`) for all authenticated routes
20. Custom permissions for role-based access
21. Secure routes using `IsAuthenticated`, `IsExpert`, `IsClient`, etc.

✔️**Tech Stack & Structure**:

- Database: **PostgreSQL**
- Authentication: Google OAuth2
- Django apps: `users`, `profiles`, `reviews`, `schedules`, `meetings`, `notifications`
- Each app should have:
- `models.py`
- `serializers.py`
- `views.py` (ViewSets or APIViews)
- `urls.py`
- `settings.py` configured for:
- PostgreSQL
- Google OAuth credentials
- Email backend (SMTP or console)
- Include permission classes for each role
- Google Calendar API integration for creating and deleting events
- Email sending example using `send_mail()`

♐**Required API Endpoints**:

- `/auth/google/` – Google login endpoint
- `/profile/` – Get and update user profile
- `/reviews/` – Add and list reviews
- `/schedules/` – CRUD operations for expert availability
- `/meetings/` – CRUD meetings with Google integration and email

- `/notifications/` – List and mark as read

The code must be clean, well-structured, extensible, and clearly commented.