**🔐 Authentication & Authorization Structure**

**1. Authentication (API Consumers – Event Managers)**

* **Method**: Token-based (Laravel Sanctum or Passport)
* **Who**: Authenticated users (e.g. event managers, admin users)
* **Purpose**:
  + Create, update, delete events
  + View bookings
  + Manage attendees

**📦 Example Structure**

* /api/login – Authenticate and issue token
* /api/register – Create user account (optional)
* Use auth:sanctum middleware on routes like:

**2. Public Access (Unauthenticated Attendees)**

* **Who**: General users registering as attendees or booking events
* **Access Allowed**:
  + Register as attendee (POST /attendees)
  + Book an event (POST /bookings)
  + View events (GET /events, GET /events/{id})

**🔓 Open Routes**

Route::post('/attendees', [AttendeeController::class, 'store']);

Route::post('/bookings', [BookingController::class, 'store']);

Route::get('/events', [EventController::class, 'index']);

Route::get('/events/{id}', [EventController::class, 'show']);

**3. Authorization (Role-based)**

* **Strategy**: Use Laravel Policies or Gates
* **Roles**:
  + Admin – Full access
  + Event Manager – Manage only their events
  + Attendee – Only book events, no management privileges

**✅ Example Policy**

php artisan make:policy EventPolicy --model=Event

Then define:

public function update(User $user, Event $event)

{

return $user->id === $event->user\_id;

}

🔄 Summary

| **Role** | **Auth Required** | **Permissions** |
| --- | --- | --- |
| Admin/Manager | ✅ Yes | Manage events, view bookings |
| Attendee | ❌ No | Register, book events, view events |