<https://gemini.google.com/u/1/app/21b8e85b3e611787>

Okay, I will now create a formal API Specification document for **Module 4a: Waypoint - Accommodations**, building upon the conceptual endpoints and database analysis we've discussed. This document will detail the endpoints for listing accommodations and retrieving accommodation details.

## **API Specification: Module 4a - Waypoint Accommodations (Version 1.0)**

### **1. Introduction**

This document specifies the API endpoints for accessing and managing accommodation-related data within the Via di Francesco Pilgrimage Platform. These endpoints are designed to support frontend applications and external services in providing comprehensive information about lodging options to pilgrims and enabling management by hosts and administrators.

The API relies on the underlying database schema, including the accommodations table, its related master tables (e.g., accommodation\_types\_master, amenities\_master), junction tables (e.g., accommodation\_amenities, accommodation\_room\_configurations), and views (e.g., accommodations\_capacity\_summary\_view and the newly specified localized views like v\_accommodation\_types\_localized).

### **2. General Considerations**

* **Base URL**: (Placeholder) /api/v1
* **Authentication**: All endpoints, unless specified as public, will require authentication via Supabase Auth (JWT Bearer token in the Authorization header).
* **Error Handling**: Standard HTTP status codes will be used:
  + 200 OK: Request successful.
  + 201 Created: Resource successfully created (for POST/PUT).
  + 204 No Content: Request successful, no response body.
  + 400 Bad Request: Invalid request parameters or body.
  + 401 Unauthorized: Authentication failed or not provided.
  + 403 Forbidden: Authenticated user does not have permission.
  + 404 Not Found: Resource not found.
  + 500 Internal Server Error: Server-side error. Error responses should include a JSON body with a message field.
* **Rate Limiting**: Appropriate rate limiting will be applied to prevent abuse (details TBD).
* **Localization**:
  + Endpoints supporting localization will accept a lang={language\_code} query parameter (e.g., lang=it).
  + If lang is provided and a translation exists, translatable fields will be returned in the requested language.
  + If lang is not provided, or if a translation for the requested language is not available, the content will default to the primary reference language (English).
  + The detailed accommodation endpoint will leverage the v\_<master\_table\_name>\_localized views to provide an all\_translations JSONB object for master data labels and descriptions. For direct translatable fields on entities like accommodations or accommodation\_reviews, the API will perform the necessary join to public.translations based on the lang parameter.
* **Pagination**: List endpoints will use page (default: 1) and limit (default: 20, max: 100) query parameters for pagination. Responses for paginated lists will include pagination metadata (e.g., total\_items, total\_pages, current\_page, per\_page).

### **3. Endpoint Specifications**

#### **3.1. List Accommodations**

* **Method & Path**: GET /accommodations
* **Description**: Retrieves a paginated list of accommodations. Supports filtering based on various criteria such as location, type, amenities, capacity, and opening months. Results are based on published and non-deleted waypoints.
* **Permissions**: Public.
* **Request**:
  + **Query Parameters**:
    - near\_waypoint\_id={waypoint\_id}: (Optional) integer. Waypoint ID to search near. Relies on waypoints.geom.
    - radius\_km={integer}: (Optional, requires near\_waypoint\_id) integer. Search radius in kilometers.
    - town\_id={integer}: (Optional) integer. Filter by waypoints.town\_id.
    - accommodation\_type\_codes={code1},{code2}: (Optional) string. Comma-separated list of accommodation\_types\_master.code values.
    - amenity\_codes={code1},{code2}: (Optional) string. Comma-separated list of amenities\_master.amenity\_code values (logical AND).
    - min\_beds={integer}: (Optional) integer. Minimum total beds, derived from accommodations\_capacity\_summary\_view.calculated\_total\_beds.
    - opening\_month={integer}: (Optional, 1-12) integer. Filters accommodations open in the specified month based on accommodations.opening\_months.
    - lang={language\_code}: (Optional) string. Two-letter ISO 639-1 language code (e.g., it, en).
    - sort\_by=name|distance|relevance: (Optional) string. Default: relevance. distance requires near\_waypoint\_id.
    - order=asc|desc: (Optional) string. Default: asc for name, asc for distance, desc for relevance.
    - page={integer}: (Optional) integer. Default: 1.
    - limit={integer}: (Optional) integer. Default: 20, Max: 100.
* **Response**:
  + **Success Status Code**: 200 OK
  + **Response Body**: A JSON object containing a list of accommodation summaries and pagination details.
  + JSON

{

"pagination": {

"total\_items": 150,

"total\_pages": 8,

"current\_page": 1,

"per\_page": 20

},

"accommodations": [

{

"waypoint\_id": 101,

"waypoint\_name": "Rifugio Cuor Contento", // Translated if lang specified

"waypoint\_slug": "rifugio-cuor-contento",

"latitude": 43.567,

"longitude": 12.345,

"elevation\_meters": 789,

"town\_name": "Villaggio Montano", // Translated, from joined towns table

"primary\_image\_thumbnail\_url": "path/to/thumb\_s.webp", // Derived from waypoints.primary\_thumbnail\_media\_id

"accommodation\_type": {

"code": "rifugio\_mountain\_hut",

"label": "Rifugio Alpino" // Translated

},

"booking\_availability\_status": { // Simplified for list view

"code": "open\_bookings\_available",

"label": "Aperto - Prenotazioni Disponibili", // Translated

"is\_positive\_status": true

},

"capacity\_summary": {

"calculated\_total\_beds": 25

},

"brief\_description": "Accogliente rifugio alpino con vista mozzafiato...", // From waypoints.short\_narrative\_for\_dynamic\_lists, translated

"highlight\_amenity\_icons": ["wifi", "local\_laundry\_service"] // Example: Top 2-3 relevant amenity icon\_identifiers

}

// ... other accommodation summaries

]

}

#### **3.2. Get Accommodation Details**

* **Method & Path**: GET /accommodations/{waypoint\_id}
* **Description**: Retrieves comprehensive details for a single accommodation, identified by its waypoint\_id. Includes linked entities like amenities, room configurations, payment methods, meal services, and recent reviews.
* **Permissions**: Public (for published and non-deleted waypoints).
* **Request**:
  + **Path Parameters**:
    - waypoint\_id: integer (BIGINT in DB). The ID of the waypoint that is an accommodation. **Required**.
  + **Query Parameters**:
    - lang={language\_code}: (Optional) string. Two-letter ISO 639-1 language code.
    - reviews\_limit={integer}: (Optional) integer. Number of recent reviews to return. Default: 3.
* **Response**:
  + **Success Status Code**: 200 OK
  + **Response Body**: A JSON object containing the detailed accommodation information. (Refer to the detailed example provided in the previous "PHASE-2 PROMPT" response, which remains largely applicable here, but ensure field names match the finalized database schema, especially for master data labels which should be consistently named e.g., label and not name where appropriate).
    - **Key Sections in Response**:
      * waypoint\_id, waypoint\_name (translated), waypoint\_slug, geom (GeoJSON), primary\_image (with alt\_text, caption translated, and image\_variants\_json).
      * accommodation\_details: Object containing fields from the accommodations table. Translatable fields (e.g., host\_name\_or\_organization, booking\_notes) should be in the requested language. Linked master data (e.g., accommodation\_type, booking\_availability\_status) should provide their code and translated label (potentially using the all\_translations field from the new localized views e.g., v\_accommodation\_types\_localized.all\_translations).
      * capacity\_summary: From accommodations\_capacity\_summary\_view.
      * amenities: Array of objects, each from accommodation\_amenities joined with amenities\_master (or its localized view v\_amenities\_localized). Includes amenity\_code, translated name, icon\_identifier, and translated notes\_on\_amenity.
      * room\_configurations: Array of objects, each from accommodation\_room\_configurations joined with room\_types\_master (or v\_room\_types\_localized). Includes room\_type (with code, translated name), count\_of\_this\_room\_type, pricing details, and translated price\_notes / room\_specific\_notes.
      * payment\_methods: Array of objects, each from accommodation\_payment\_methods joined with payment\_methods\_master (or v\_payment\_methods\_localized). Includes code, translated label, icon\_identifier, and translated notes\_on\_method.
      * meal\_services: Array of objects, each from accommodation\_meal\_services joined with meal\_services\_master (or v\_meal\_services\_localized). Includes service code, translated name, pricing, and translated availability\_and\_timing\_notes.
      * reviews: Array of recent accommodation\_reviews (respecting is\_publicly\_visible), with translated review\_title and review\_body. Includes pilgrim\_display\_name (derived from profile\_id), stay\_date, overall\_vote, created\_at.

### **4. Data Models / DTOs (Conceptual Examples)**

* **AccommodationSummaryObject**:
  + waypoint\_id, waypoint\_name, waypoint\_slug, latitude, longitude, elevation\_meters, town\_name, primary\_image\_thumbnail\_url, accommodation\_type (object with code, label), booking\_availability\_status (object with code, label, is\_positive\_status), capacity\_summary (object with calculated\_total\_beds), brief\_description, highlight\_amenity\_icons (array of strings).
* **AccommodationDetailObject**:
  + Includes all fields from AccommodationSummaryObject plus:
  + geom (GeoJSON).
  + primary\_image (full media object).
  + accommodation\_details (full object from accommodations table with master data expanded and translatable fields in requested language).
  + amenities (array of full AmenityLinkObject).
  + room\_configurations (array of full RoomConfigurationObject).
  + payment\_methods (array of full PaymentMethodLinkObject).
  + meal\_services (array of full MealServiceLinkObject).
  + reviews (array of ReviewObject).
  + translations\_available (array of language codes).
* **LocalizedMasterDataObject (Generic structure for master data within details)**:
  + code
  + label (translated based on lang param)
  + description (translated, optional)
  + icon\_identifier (optional)
  + all\_translations (JSONB object, if fetching directly from localized views and exposing all translations)
* **AmenityLinkObject, RoomConfigurationObject, PaymentMethodLinkObject, MealServiceLinkObject, ReviewObject**: Structures based on the example JSON response for the details endpoint, with translatable text fields provided in the requested language.

### **5. Database Support & Optimization Notes**

* **Localized Views**: The newly specified views (e.g., v\_accommodation\_types\_localized, v\_amenities\_localized, etc.) are crucial for efficiently providing translated labels and descriptions for master data within the /accommodations/{waypoint\_id} detail endpoint. The API backend should query these views.
* **Database Functions**: For the /accommodations/{waypoint\_id} endpoint, which involves extensive joins and data aggregation (including amenities, room configurations, payment methods, meal services, reviews, and translations for multiple fields), creating a **PostgreSQL function** (e.g., get\_public\_accommodation\_details(p\_waypoint\_id BIGINT, p\_lang TEXT) RETURNS JSONB) is highly recommended. This function would:
  + Encapsulate the complex query logic.
  + Perform necessary joins to waypoints, accommodations, all relevant junction and master tables (or their localized views).
  + Handle translation lookups based on p\_lang.
  + Aggregate related collections (amenities, rooms, etc.) into JSON arrays.
  + Structure the final JSONB output to match the API response, reducing processing load on the API application layer.
  + This function can be optimized within the database and can also more easily respect RLS policies if defined with appropriate security contexts (e.g., SECURITY DEFINER with caution, or ensuring RLS is applied to all underlying tables accessed by a SECURITY INVOKER function).
* **Indexes**:
  + The GIN index on accommodations.opening\_months (idx\_accommodations\_opening\_months) is essential for the opening\_month filter on the list endpoint.
  + Indexes on label/name columns of all master tables (e.g., idx\_atm\_label on accommodation\_types\_master) support server-side sorting for list endpoints.
  + All Foreign Keys involved in joins should be indexed on the referencing table.
  + The accommodations\_capacity\_summary\_view performance depends on indexes on accommodation\_room\_configurations (accommodation\_waypoint\_id, room\_type\_id) and room\_types\_master (id).
* **Filtering Logic**:
  + Filtering by multiple amenity\_codes (e.g., must have Wi-Fi AND kitchen) will require careful SQL construction, potentially using array operations or multiple EXISTS subqueries if implemented directly, or handled within the aforementioned database function.
* **RLS**: All queries executed by the API (whether direct or via database functions) will be subject to Row-Level Security policies, ensuring data is only exposed according to user roles and permissions (e.g., only published waypoints, hosts see only their own data for management endpoints not specified here).

This formal API specification should provide a solid foundation for developing the accommodation-related features.