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

## **New Table:** public.accommodation\_reviews **(Version 1.0)**

**(Module 6: User Interaction; closely related to Module 4a: Accommodations)**

### **1. Purpose & Primary Use-Cases**

The public.accommodation\_reviews table stores user-submitted reviews specifically for accommodations. It allows pilgrims to share their experiences, including textual feedback, an optional overall vote, and the date of their stay. The table supports multilingual content, moderation workflows, and audit trails.

**Key user-story touchpoints:**

* Pilgrim (Anna): Reading reviews for an accommodation to help make lodging choices.
* Pilgrim (Anna): Submitting a review of an accommodation where they stayed.
* Accommodation Host (Marco): (Potentially, V2+) Viewing reviews for their establishment.
* Platform Administrator/Moderator (Sofia): Reviewing and approving/rejecting submitted reviews.
* System/UI: Displaying approved reviews on accommodation detail pages.

### **2. Schema (Markdown Table)**

| **column** | **data\_type** | **constraints** | **description** |
| --- | --- | --- | --- |
| id | bigint | Primary Key, Generated always as identity | Unique identifier for the review. |
| accommodation\_waypoint\_id | bigint | Not Null, Foreign Key to public.accommodations(waypoint\_id) ON DELETE CASCADE | Links to the specific accommodation being reviewed. |
| profile\_id | uuid | Not Null, Foreign Key to public.profiles(id) ON DELETE CASCADE | Author of the review (implicitly the creator of the content). |
| review\_title | text | Nullable, CHECK (char\_length(review\_title) &lt;= 255) | Optional title for the review. (Translatable via public.translations) |
| review\_body | text | Not Null, CHECK (char\_length(review\_body) > 0 AND char\_length(review\_body) &lt;= 2000) | The main content of the review. (Translatable via public.translations) |
| language\_code | text | Not Null, Foreign Key to public.languages\_master(language\_code) ON DELETE RESTRICT | Language the review was originally written in. |
| stay\_date | date | Nullable | Optional: Date when the pilgrim stayed at the accommodation. |
| overall\_vote | public.vote\_type\_enum | Nullable | Optional: A simple overall vote (e.g., 'up', 'down') for the stay. (Assumes vote\_type\_enum exists from Module 6). |
| moderation\_status | public.content\_moderation\_status\_enum | Not Null, Default 'pending\_approval' | Moderation status of the review (e.g., 'pending\_approval', 'approved\_visible', 'rejected'). (Assumes content\_moderation\_status\_enum exists from Module 6). |
| is\_publicly\_visible | boolean | Not Null, Generated always as (moderation\_status = 'approved\_visible' AND deleted\_at IS NULL) STORED | Auto-calculated flag indicating if the review should be visible to the public. (Requires PostgreSQL 12+) |
| moderated\_by\_profile\_id | uuid | Nullable, Foreign Key to public.profiles(id) ON DELETE SET NULL | Profile ID of the admin/moderator who actioned this review. |
| moderation\_timestamp | timestamp with time zone | Nullable | Timestamp of when the moderation action was taken. |
| moderation\_notes\_internal | text | Nullable | Internal notes for moderation team regarding this review. |
| created\_at | timestamp with time zone | Not Null, Default now() | Timestamp of when this review record was created. |
| updated\_at | timestamp with time zone | Not Null, Default now() | Timestamp of when this review record was last updated (auto-updated by trigger). |
| updated\_by\_profile\_id | uuid | Nullable, Foreign Key to public.profiles(id) ON DELETE SET NULL | Profile ID of the user/admin who last updated this review row (e.g., user editing their own review if allowed, or admin edit). |
| deleted\_at | timestamp with time zone | Nullable | Timestamp for soft deletion by user or admin. Reviews with deleted\_at set are not publicly visible. |

*(Assumes public.vote\_type\_enum and public.content\_moderation\_status\_enum are globally defined or defined within Module 6 as per 4.x todos.docx)*

### **3. PostgreSQL DDL**

SQL

-- Assumes ENUM types public.vote\_type\_enum and public.content\_moderation\_status\_enum exist.

-- Assumes public.accommodations, public.profiles, public.languages\_master tables exist.

-- Assumes public.set\_current\_timestamp\_updated\_at() function exists.

-- Assumes public.cleanup\_related\_translations(TEXT, TEXT) function exists for the AFTER DELETE trigger.

CREATE TABLE public.accommodation\_reviews (

id BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,

accommodation\_waypoint\_id BIGINT NOT NULL REFERENCES public.accommodations(waypoint\_id) ON DELETE CASCADE,

profile\_id UUID NOT NULL REFERENCES public.profiles(id) ON DELETE CASCADE,

review\_title TEXT NULL CHECK (char\_length(review\_title) <= 255),

review\_body TEXT NOT NULL CHECK (char\_length(review\_body) > 0 AND char\_length(review\_body) <= 2000),

language\_code TEXT NOT NULL REFERENCES public.languages\_master(language\_code) ON DELETE RESTRICT,

stay\_date DATE NULL,

overall\_vote public.vote\_type\_enum NULL,

moderation\_status public.content\_moderation\_status\_enum NOT NULL DEFAULT 'pending\_approval',

is\_publicly\_visible BOOLEAN NOT NULL GENERATED ALWAYS AS (moderation\_status = 'approved\_visible' AND deleted\_at IS NULL) STORED,

moderated\_by\_profile\_id UUID NULL REFERENCES public.profiles(id) ON DELETE SET NULL,

moderation\_timestamp TIMESTAMPTZ NULL,

moderation\_notes\_internal TEXT NULL,

created\_at TIMESTAMPTZ NOT NULL DEFAULT now(),

updated\_at TIMESTAMPTZ NOT NULL DEFAULT now(),

updated\_by\_profile\_id UUID NULL REFERENCES public.profiles(id) ON DELETE SET NULL,

deleted\_at TIMESTAMPTZ NULL

);

COMMENT ON TABLE public.accommodation\_reviews IS 'Stores user-submitted reviews for accommodations, including text, ratings, and moderation status. Version 1.0';

COMMENT ON COLUMN public.accommodation\_reviews.id IS 'Unique identifier for the review.';

COMMENT ON COLUMN public.accommodation\_reviews.accommodation\_waypoint\_id IS 'FK to accommodations.waypoint\_id. Links to the specific accommodation being reviewed.';

COMMENT ON COLUMN public.accommodation\_reviews.profile\_id IS 'FK to profiles.id. Author of the review (implicitly the creator of the content).';

COMMENT ON COLUMN public.accommodation\_reviews.review\_title IS 'Optional title for the review. (Translatable via public.translations)';

COMMENT ON COLUMN public.accommodation\_reviews.review\_body IS 'The main content of the review. (Translatable via public.translations)';

COMMENT ON COLUMN public.accommodation\_reviews.language\_code IS 'FK to languages\_master.language\_code. Language the review was originally written in.';

COMMENT ON COLUMN public.accommodation\_reviews.stay\_date IS 'Optional: Date when the pilgrim stayed at the accommodation.';

COMMENT ON COLUMN public.accommodation\_reviews.overall\_vote IS 'Optional: A simple overall vote (e.g., ''up'', ''down'') for the stay. Uses public.vote\_type\_enum.';

COMMENT ON COLUMN public.accommodation\_reviews.moderation\_status IS 'Moderation status of the review. Uses public.content\_moderation\_status\_enum.';

COMMENT ON COLUMN public.accommodation\_reviews.is\_publicly\_visible IS 'Auto-calculated flag: TRUE if moderation\_status is ''approved\_visible'' AND deleted\_at IS NULL.';

COMMENT ON COLUMN public.accommodation\_reviews.moderated\_by\_profile\_id IS 'FK to profiles.id. Admin/moderator who actioned this review.';

COMMENT ON COLUMN public.accommodation\_reviews.moderation\_timestamp IS 'Timestamp of when the moderation action was taken.';

COMMENT ON COLUMN public.accommodation\_reviews.moderation\_notes\_internal IS 'Internal notes for moderation team regarding this review.';

COMMENT ON COLUMN public.accommodation\_reviews.created\_at IS 'Timestamp of when this review record was created.';

COMMENT ON COLUMN public.accommodation\_reviews.updated\_at IS 'Timestamp of when this review record was last updated (auto-updated by trigger).';

COMMENT ON COLUMN public.accommodation\_reviews.updated\_by\_profile\_id IS 'FK to profiles.id. User/admin who last updated this review row.';

COMMENT ON COLUMN public.accommodation\_reviews.deleted\_at IS 'Timestamp for soft deletion. Reviews with deleted\_at set are not publicly visible.';

-- Indexes

CREATE INDEX idx\_accommodation\_reviews\_accommodation\_waypoint\_id ON public.accommodation\_reviews(accommodation\_waypoint\_id);

CREATE INDEX idx\_accommodation\_reviews\_profile\_id ON public.accommodation\_reviews(profile\_id);

CREATE INDEX idx\_accommodation\_reviews\_language\_code ON public.accommodation\_reviews(language\_code);

CREATE INDEX idx\_accommodation\_reviews\_moderation\_status ON public.accommodation\_reviews(moderation\_status);

CREATE INDEX idx\_accommodation\_reviews\_is\_publicly\_visible ON public.accommodation\_reviews(is\_publicly\_visible) WHERE is\_publicly\_visible = true;

CREATE INDEX idx\_accommodation\_reviews\_stay\_date ON public.accommodation\_reviews(stay\_date) WHERE stay\_date IS NOT NULL;

CREATE INDEX idx\_accommodation\_reviews\_moderated\_by\_profile\_id ON public.accommodation\_reviews(moderated\_by\_profile\_id) WHERE moderated\_by\_profile\_id IS NOT NULL;

CREATE INDEX idx\_accommodation\_reviews\_updated\_by\_profile\_id ON public.accommodation\_reviews(updated\_by\_profile\_id) WHERE updated\_by\_profile\_id IS NOT NULL;

CREATE INDEX idx\_accommodation\_reviews\_deleted\_at ON public.accommodation\_reviews(deleted\_at) WHERE deleted\_at IS NOT NULL;

-- Trigger for updated\_at

CREATE TRIGGER trigger\_accommodation\_reviews\_set\_updated\_at

BEFORE UPDATE ON public.accommodation\_reviews

FOR EACH ROW

EXECUTE FUNCTION public.set\_current\_timestamp\_updated\_at(); -- Or extensions.moddatetime

COMMENT ON TRIGGER trigger\_accommodation\_reviews\_set\_updated\_at ON public.accommodation\_reviews IS 'Trigger to automatically update updated\_at timestamp on row modification.';

-- Trigger for orphan translation cleanup

CREATE OR REPLACE FUNCTION public.cleanup\_accommodation\_reviews\_translations()

RETURNS TRIGGER AS $$

BEGIN

IF TG\_OP = 'DELETE' THEN

DELETE FROM public.translations

WHERE table\_identifier = 'accommodation\_reviews'

AND row\_foreign\_key = OLD.id::TEXT; -- Using BIGINT PK for translation key

END IF;

RETURN OLD;

END;

$$ LANGUAGE plpgsql SECURITY DEFINER;

CREATE TRIGGER trigger\_cleanup\_accommodation\_reviews\_translations

AFTER DELETE ON public.accommodation\_reviews

FOR EACH ROW

EXECUTE FUNCTION public.cleanup\_accommodation\_reviews\_translations();

COMMENT ON TRIGGER trigger\_cleanup\_accommodation\_reviews\_translations ON public.accommodation\_reviews IS 'Cleans up orphaned translations from public.translations when an accommodation\_reviews record is deleted.';

### **4. JSON Schema Mirror**

JSON

{

"title": "accommodation\_review",

"description": "Stores user-submitted reviews for accommodations. Version 1.0",

"type": "object",

"properties": {

"id": {

"type": "integer",

"format": "int64",

"description": "Unique identifier for the review. PK.",

"readOnly": true

},

"accommodation\_waypoint\_id": {

"type": "integer",

"format": "int64",

"description": "FK to accommodations.waypoint\_id."

},

"profile\_id": {

"type": "string",

"format": "uuid",

"description": "FK to profiles.id. Author of the review."

},

"review\_title": {

"type": ["string", "null"],

"maxLength": 255,

"description": "Optional title for the review. (Translatable)"

},

"review\_body": {

"type": "string",

"minLength": 1,

"maxLength": 2000,

"description": "The main content of the review. (Translatable)"

},

"language\_code": {

"type": "string",

"description": "FK to languages\_master.language\_code. Language review was written in."

},

"stay\_date": {

"type": ["string", "null"],

"format": "date",

"description": "Optional: Date of stay."

},

"overall\_vote": {

"type": ["string", "null"],

"description": "Optional: Overall vote. Uses public.vote\_type\_enum (e.g., 'up', 'down')."

},

"moderation\_status": {

"type": "string",

"default": "pending\_approval",

"description": "Moderation status. Uses public.content\_moderation\_status\_enum."

},

"is\_publicly\_visible": {

"type": "boolean",

"description": "Auto-calculated: TRUE if moderation\_status is 'approved\_visible' AND deleted\_at IS NULL.",

"readOnly": true

},

"moderated\_by\_profile\_id": {

"type": ["string", "null"],

"format": "uuid",

"description": "FK to profiles.id. Admin/moderator who actioned this review."

},

"moderation\_timestamp": {

"type": ["string", "null"],

"format": "date-time",

"description": "Timestamp of moderation action."

},

"moderation\_notes\_internal": {

"type": ["string", "null"],

"description": "Internal notes for moderation."

},

"created\_at": {

"type": "string",

"format": "date-time",

"readOnly": true

},

"updated\_at": {

"type": "string",

"format": "date-time",

"readOnly": true

},

"updated\_by\_profile\_id": {

"type": ["string", "null"],

"format": "uuid",

"description": "Profile ID of the user/admin who last updated this review row."

},

"deleted\_at": {

"type": ["string", "null"],

"format": "date-time",

"description": "Timestamp for soft deletion."

}

},

"required": [

"accommodation\_waypoint\_id",

"profile\_id",

"review\_body",

"language\_code",

"moderation\_status",

"is\_publicly\_visible"

]

}

### **5. Relationships & Integrity**

* Primary Key: id (BIGINT).
* Foreign Keys:
  + accommodation\_waypoint\_id REFERENCES public.accommodations(waypoint\_id) ON DELETE CASCADE.
  + profile\_id REFERENCES public.profiles(id) ON DELETE CASCADE (if a user's profile is deleted, their reviews are also deleted).
  + language\_code REFERENCES public.languages\_master(language\_code) ON DELETE RESTRICT.
  + moderated\_by\_profile\_id REFERENCES public.profiles(id) ON DELETE SET NULL.
  + updated\_by\_profile\_id REFERENCES public.profiles(id) ON DELETE SET NULL.
* Generated Column: is\_publicly\_visible depends on moderation\_status and deleted\_at.
* ENUMs: Relies on public.vote\_type\_enum and public.content\_moderation\_status\_enum (assumed to be defined globally or in Module 6).

### **6. Multilingual Strategy**

* review\_title and review\_body are translatable. The original language is stored in language\_code.
* Translations are managed in the public.translations table.
  + table\_identifier: 'accommodation\_reviews'
  + column\_identifier: 'review\_title' or 'review\_body'
  + row\_foreign\_key: The id (BIGINT) of the accommodation\_reviews record.
* An AFTER DELETE trigger (trigger\_cleanup\_accommodation\_reviews\_translations) is implemented.

### **7. Role-Based Workflow & RLS Notes**

* **Public Users (Anonymous/Authenticated without specific roles):**
  + Can SELECT reviews where is\_publicly\_visible = true.
* **Authenticated Users (Pilgrim Role):**
  + Can INSERT new reviews (defaulting to moderation\_status = 'pending\_approval', profile\_id = auth.uid()).
  + Can UPDATE their own reviews if moderation\_status is 'pending\_approval' OR if an edit window is allowed post-approval (requires careful WITH CHECK and potentially versioning/re-moderation logic - for V1, keep simpler: edit only if pending). updated\_by\_profile\_id would be auth.uid().
  + Can UPDATE their own reviews to set deleted\_at (soft delete). updated\_by\_profile\_id would be auth.uid().
* **Moderator/Admin Roles:**
  + Can SELECT all reviews, regardless of is\_publicly\_visible.
  + Can UPDATE any review's moderation\_status, moderated\_by\_profile\_id, moderation\_timestamp, moderation\_notes\_internal, deleted\_at. updated\_by\_profile\_id would be auth.uid().
  + Can DELETE records (hard delete, if necessary, though soft delete via deleted\_at is preferred).
* **RLS Policies Example Snippets:**
* SQL

-- Public Read

CREATE POLICY "Public can read approved and visible reviews"

ON public.accommodation\_reviews FOR SELECT USING (is\_publicly\_visible = true);

-- Authenticated users can insert their own reviews

CREATE POLICY "Authenticated users can insert their own reviews"

ON public.accommodation\_reviews FOR INSERT WITH CHECK (profile\_id = auth.uid());

-- Users can update their own pending reviews or soft delete them

CREATE POLICY "Users can update their own reviews if pending or soft delete"

ON public.accommodation\_reviews FOR UPDATE USING (profile\_id = auth.uid())

WITH CHECK (

profile\_id = auth.uid() AND

(

(OLD.moderation\_status = 'pending\_approval' AND OLD.deleted\_at IS NULL) OR -- Can edit most fields if pending

(NEW.deleted\_at IS NOT NULL AND OLD.deleted\_at IS NULL) -- Can soft delete

-- Add more complex logic here if edits after approval are allowed

)

);

-- Moderators/Admins can manage all reviews

CREATE POLICY "Moderators can manage all reviews"

ON public.accommodation\_reviews FOR ALL

USING (public.has\_role\_on\_profile(auth.uid(), 'moderator') OR public.has\_role\_on\_profile(auth.uid(), 'admin\_platform')) -- Example role check

WITH CHECK (public.has\_role\_on\_profile(auth.uid(), 'moderator') OR public.has\_role\_on\_profile(auth.uid(), 'admin\_platform'));

* *Enable RLS on the table after policies are defined.*

### **8. ENUM vs Lookup Discussion**

* Relies on vote\_type\_enum and content\_moderation\_status\_enum. These are ENUMs as per existing patterns for similar concepts in the project.

### **9. UI/UX Enablement**

* Display of reviews (title, body, author, stay date, overall vote) on accommodation pages.
* Form for submitting reviews.
* Moderation interface for admins/moderators.
* Potentially, filtering or sorting reviews by stay date or overall vote.

### **10. Key Considerations & Definitions**

* **ENUM Definitions**: vote\_type\_enum (e.g., 'up', 'down', or perhaps a 1-5 star rating if changed from proposal) and content\_moderation\_status\_enum (e.g., 'pending\_approval', 'approved\_visible', 'rejected\_hidden') must be created and managed. The proposal used 'pending\_approval', 'approved\_visible'.
* **Edit Policy**: The RLS example allows editing only for 'pending\_approval' status. If reviews can be edited after approval, this needs careful consideration regarding re-moderation workflows.
* overall\_vote **Integration**: For V1, this field is a simple attribute of the review. It does not automatically update aggregated votes on the waypoints table or link to user\_waypoint\_votes. Such integration is a V2+ consideration.
* **Anonymity**: The current design links directly to profile\_id. If anonymous reviews (or reviews displayed with less identifiable user info) are needed, this requires further design. For V1, linking to profile\_id is assumed.

### **11. Scalability & Future-Proofing**

* The table can grow large. Proper indexing is crucial.
* Partitioning by accommodation\_waypoint\_id or created\_at (e.g., yearly) could be a V2+ consideration if volume becomes extremely high.
* Adding features like "Was this review helpful?" votes, or host replies, would be future extensions.

### **12. Next-Action Checklist**

* 🔴 **Define/Verify ENUM Types**: Ensure public.vote\_type\_enum and public.content\_moderation\_status\_enum are defined according to project standards.
* 🔴 **Create Table**: Execute the DDL for public.accommodation\_reviews (Version 1.0).
* 🔴 **Apply Triggers**: Ensure set\_current\_timestamp\_updated\_at and trigger\_cleanup\_accommodation\_reviews\_translations are created and applied.
* 🔴 **Create Indexes**: Implement all necessary indexes as defined in the DDL.
* 🟠 **RLS Policies**: Implement and thoroughly test the RLS policies, including helper functions like has\_role\_on\_profile.
* 🟠 **Review Edit Workflow**: Confirm the business rules for when a user can edit their review and adjust RLS UPDATE policy if needed.
* 🟢 **Translation Entries**: Plan for how original review text will be flagged for translation if automated translation is a feature, or how user-submitted translations would be handled (V2+). For V1, language\_code captures the submission language.
* 🟢 **Moderation Interface**: Plan the UI/UX for the review moderation workflow.