### **Key Conceptual API Endpoints**

1. **Endpoint: List Active Warnings for a Trail Segment**
   * **Purpose**: To retrieve all currently active and publicly published warnings for a specific trail segment.
   * **Path Pattern**: GET /segments/{segment\_id}/warnings
   * **Query Params / Filters / Options**:
     + lang (optional, string, e.g., it): ISO language code to request translations for display names and descriptions.
     + include\_image\_variants (optional, boolean, default false): If true, attempts to include pre-defined variants for the primary image from media.image\_variants\_json.
     + limit (optional, integer, default 20): Number of warnings to return.
     + offset (optional, integer, default 0): Offset for pagination.
2. **Endpoint: Search Active Warnings by Geographical Area**
   * **Purpose**: To find and display all currently active and publicly published warnings within a given geographical bounding box, typically for map displays.
   * **Path Pattern**: GET /warnings/search/geo
   * **Query Params / Filters / Options**:
     + bbox (required, string, e.g., minLng,minLat,maxLng,maxLat): The bounding box coordinates (WGS84).
     + lang (optional, string, e.g., it): ISO language code for translations.
     + include\_image\_variants (optional, boolean, default false).
     + limit (optional, integer, default 50): Maximum number of warnings to return.
     + offset (optional, integer, default 0): Offset for pagination.
     + min\_severity\_code (optional, string, e.g., caution\_advised): Filter warnings by minimum severity level (inclusive), based on warning\_severities\_master.sort\_order.

&lt;hr/>

### **Example JSON Responses**

1. GET /segments/{segment\_id}/warnings?lang=it&include\_image\_variants=true
2. JSON

[

{

"warning\_id": 101,

"segment\_id": 42,

"title": "Frana Minore", // Assuming 'it' translation was found

"description\_message": "Piccola frana sul sentiero, prestare attenzione.", // 'it' translation

"location\_on\_segment\_description": "Circa al km 2.5, vicino alla vecchia quercia.", // 'it' translation

"location\_on\_segment\_km\_approx": 2.5,

"location\_on\_segment\_geom": { /\* GeoJSON PointZ object \*/ },

"detour\_information\_url": "https://example.com/detour/101",

"detour\_description\_notes": "Seguire la deviazione segnalata.", // 'it' translation

"date\_warning\_reported": "2025-05-15T10:00:00Z",

"date\_warning\_effective\_from": "2025-05-15T12:00:00Z",

"date\_warning\_expected\_resolution": "2025-05-25T17:00:00Z",

"warning\_type": {

"code": "hazard\_natural\_landslide",

"name": "Pericolo Naturale - Frana", // 'it' translation

"icon\_identifier": "fas fa-mountain-city"

},

"warning\_severity": {

"code": "caution\_advised",

"name": "Cautela Consigliata", // 'it' translation

"ui\_color\_hex": "#F1C40F"

},

"warning\_source\_type\_name": "Gestore Regionale", // 'it' translation

"primary\_image\_media\_id": "a1b2c3d4-e5f6-7890-1234-567890abcdef",

"primary\_image\_variants": { // if include\_image\_variants=true and media record has variants

"thumbnail\_small": "/media/variants/a1b2c3d4\_thumb\_s.webp",

"display\_medium": "/media/variants/a1b2c3d4\_display\_m.jpg"

},

"translations\_available": ["en", "de"] // Conceptual: languages for which full translation exists for this warning

}

]

1. GET /warnings/search/geo?bbox=12.0,43.0,12.5,43.5&limit=1
2. JSON

[

{

"warning\_id": 105,

"segment\_id": 77,

"title": "Bridge Damaged", // Base language (English) as no 'lang' param specified

"description\_message": "The small wooden bridge over the creek is damaged. Crossing is risky.",

"location\_on\_segment\_description": "At the crossing of Fosso della Volpe.",

"location\_on\_segment\_km\_approx": 1.2,

"location\_on\_segment\_geom": { /\* GeoJSON PointZ object \*/ },

"detour\_information\_url": null,

"detour\_description\_notes": "Consider fording the creek upstream if water levels are low, otherwise backtrack.",

"date\_warning\_reported": "2025-05-18T09:30:00Z",

"date\_warning\_effective\_from": "2025-05-18T09:30:00Z",

"date\_warning\_expected\_resolution": null,

"warning\_type": {

"code": "trail\_damage\_bridge",

"name": "Trail Damage - Bridge",

"icon\_identifier": "icon-bridge-damage"

},

"warning\_severity": {

"code": "hazard\_significant",

"name": "Significant Hazard / Consider Alternative",

"ui\_color\_hex": "#E67E22"

},

"warning\_source\_type\_name": "Official Authority",

"primary\_image\_media\_id": "b2c3d4e5-f6g7-8901-2345-67890abcdef0",

"primary\_image\_variants": null // if include\_image\_variants=false or no variants

}

]

&lt;hr/>

### **Database-Support Analysis**

For both conceptual endpoints, the primary data source would ideally be the public.public\_active\_segment\_warnings\_view.

1. **Endpoint:** GET /segments/{segment\_id}/warnings  
   * **Indexes**:
     + The view itself queries segment\_warnings using segment\_id. An index on segment\_warnings.segment\_id is crucial and already defined (idx\_segment\_warnings\_segment\_id).
     + The view also filters by is\_currently\_active = true and workflow\_status\_code = 'published'. The composite index idx\_segment\_warnings\_is\_currently\_active\_published is well-suited for this.
     + Joins to master tables use their PKs (id), which are indexed.
     + **Sufficient**: Yes, existing indexes on base tables and those supporting the view's WHERE clause are good.
   * **Join Complexity**:
     + The public\_active\_segment\_warnings\_view already encapsulates the necessary joins (to warning\_types\_master, warning\_severities\_master, warning\_source\_types\_master).
     + If include\_image\_variants=true, an additional LEFT JOIN to public.media on primary\_image\_media\_id would be needed if not already part of the core view. This is a simple join on an indexed FK.
     + Fetching translations would involve separate lookups to the public.translations table per translatable field, per warning, if the API handles translation assembly.
     + **Recommendation**: The view greatly simplifies this. For translations, the API backend would query public.translations using table\_identifier='segment\_warnings', row\_foreign\_key=warning\_id, column\_identifier IN (...), and the requested language\_code. This can be optimized.
   * **Performance Gotchas**:
     + RLS on segment\_warnings and master tables will be applied. If the RLS policies are complex or involve many subqueries, they could add overhead. However, the view is for public data, so RLS for an anonymous/authenticated role should be straightforward.
     + The number of warnings per segment is expected to be low, so performance should be good.
   * **Missing Data?**:
     + The concept of translations\_available in the example JSON is not directly stored. The API backend would need to determine this, perhaps by checking existing translations for a given warning\_id.
2. **Endpoint:** GET /warnings/search/geo **(using** bbox**)**
   * **Indexes**:
     + Crucially relies on the GIST index on segment\_warnings.location\_on\_segment\_geom (idx\_segment\_warnings\_location\_geom) for efficient spatial queries (e.g., using ST\_MakeEnvelope and && operator).
     + The view's filters on is\_currently\_active and workflow\_status\_code are also important, covered by idx\_segment\_warnings\_is\_currently\_active\_published.
     + **Sufficient**: Yes, provided the GIST index is correctly utilized.
   * **Join Complexity**:
     + Similar to the first endpoint, the view handles base joins.
     + The geo query itself is on segment\_warnings.location\_on\_segment\_geom.
     + Filtering by min\_severity\_code would require joining warning\_severities\_master (if not already in the view in a way that allows filtering on sort\_order associated with the code) or an efficient lookup of sort\_order from code to apply to the view's warning\_severity\_code or a derived warning\_severity\_sort\_order. The public\_active\_segment\_warnings\_view already includes warning\_severity\_code and warning\_severity\_name. Adding wsm.sort\_order AS warning\_severity\_sort\_order to the view would be beneficial for this filter.
   * **Performance Gotchas**:
     + Spatial queries can be intensive if the dataset is huge and the bbox is very large, but with proper indexing and typical warning volumes, it should be manageable.
     + Pagination (limit/offset) on geo-queries needs careful handling, often best combined with a deterministic sort order (e.g., date\_warning\_reported DESC) to ensure stable results if not sorting by distance (which is a different query type).
   * **Missing Data?**:
     + To filter by min\_severity\_code effectively based on hierarchy, the sort\_order from warning\_severities\_master should be easily accessible, ideally exposed by the view.

&lt;hr/>

### **Immediate Schema Tweaks (if any)**

1. 🟢 **Add** sort\_order **from** warning\_severities\_master **to** public\_active\_segment\_warnings\_view:  
   * **Reason**: To efficiently support the min\_severity\_code filter on the /warnings/search/geo endpoint by allowing filtering on the numerical sort\_order.
   * **Change**: Modify public\_active\_segment\_warnings\_view DDL:
   * SQL

-- In SELECT list:

wsm.sort\_order AS warning\_severity\_sort\_order

1. 🟢 **Consider adding** media.image\_variants\_json **to** public\_active\_segment\_warnings\_view:  
   * **Reason**: If include\_image\_variants=true is a common API parameter, embedding the image\_variants\_json directly into the view (if primary\_image\_media\_id is not null) could simplify API logic by avoiding an N+1 query problem or a secondary join at the API level.
   * **Change**: Modify public\_active\_segment\_warnings\_view DDL:
   * SQL

-- Add LEFT JOIN to media:

-- LEFT JOIN public.media m ON sw.primary\_image\_media\_id = m.id

-- In SELECT list:

-- m.image\_variants\_json AS primary\_image\_variants\_json

* + **Note**: This makes the view slightly heavier, but potentially more convenient for API consumers.

1. 🟠 **Standardize Translatable Object Representation in API**:  
   * **Reason**: The example JSON shows title: "Frana Minore" directly using the translated string. A more robust API might return an object for translatable fields, e.g., title: { "text": "Frana Minore", "lang": "it", "base\_text": "Minor Landslide" } or provide a clear mechanism for clients to request specific translations using the translations table structure. The current public\_active\_segment\_warnings\_view only provides base language text.
   * **Change**: This is more of an API design/backend logic decision than a direct schema tweak to *this module's tables*, but it impacts how data from the view is processed and enriched with translations from public.translations. For the database, ensuring efficient querying of public.translations is key (e.g., good indexing on table\_identifier, column\_identifier, row\_foreign\_key, language\_code).
   * No direct schema change to *this module's DDLs* from this point, but a reminder for overall API consistency.

No 🔴 must-fix items identified for the current database schema of the Dynamic Conditions Module based solely on these conceptual API endpoints, assuming the proposed view is adopted and slightly enhanced. The existing tables and the proposed view provide a strong foundation.