WorkShop 2

Presentado por:
Andres Felipe Alarcon Pulido - analarconp@unal.edu.co
Cristian Javier Medina Barrios - cremedinab@unal.edu.co
Cristian Daniel Montanez- cmontanez@unal.edu.co
Sara Isabel Ospina Valderrama - saospinav@unal.edu.co
Justin Brad Rodriguez Sanchez - jusrodriguez@unal.edu.co
Sergio Alejandro Ruiz Hurtado - seruizh@unal.edu.co

Profesor:
Liliana Marcela Olarte Mesa
lmolartem@unal.edu.co

7/11/2025

Universidad Nacional de Colombia
Facultad de Ingeniería
Ingeniería de Software
Departamento de Ingeniería de Sistemas

1. CRC Cards — FindItUNAL System

The following CRC cards describe the main classes identified for the FindItUNAL backend system.Each card specifies the core responsibilities of the class and the other classes it collaborates with within the architecture.

Class Name: **User**	
Responsibilities:	**Collaborators:**
Authenticate with institutional email and password	AuthenticationService
Register and manage account profile	Report
Create, edit, and delete posts for lost/found items	Category
Search and filter posts by category, location, and date	Location
Initiate and participate in conversations about items	Conversation
Submit complaints about inappropriate content	Message
Receive and manage notifications	Complaint
Configure two-factor authentication	Notification

Class Name: **Report**	
Responsibilities:	**Collaborators:**
Store item details (title, description, date found/lost)	User
Track item status (lost or found)	Category
Associate with category and campus location	Location
Link to multiple images for visual reference	Image
Allow editing by owner and deletion by owner/admin	Conversation
Support searchable and filterable queries	Complaint
Serve as basis for user conversations	SearchService
Be subject to user complaints	

Class Name: **Location**	
Responsibilities:	**Collaborators:**
Store campus buildings and area information	Report
Enable location-based search functionality	SearchService
Provide geographic context for items	
Store location names (e.g., Building 404, Central Library)	

Class Name: **Notification**	
Responsibilities:	**Collaborators:**
Notify users of new messages	User
Alert users about admin actions on their content	Message
Inform about complaint resolutions	Report
Track notification status	Admin
Store notification type and related entity references	
Provide notification title and detailed message	

Class Name: **AuthenticationService**	
Responsibilities:	**Collaborators:**
Validate user credentials	User
Handle Google OAuth login integration	
Manage user sessions and tokens	
Verify two-factor authentication codes	
Process password reset requests	
Verify institutional email domain (@unal.edu.co)	

Class Name: **SearchService**	
Responsibilities:	**Collaborators:**
Execute search queries across posts	Report
Filter by category, location, and date range	Category
Filter by status (lost/found)	Location
Sort results by relevance or date	
Provide advanced search capabilities	

Class Name: **Admin**	
Responsibilities:	**Collaborators:**
Inherit all User capabilities	User
Review and resolve user complaints	Complaint
Suspend or permanently delete user accounts	Report
Delete inappropriate posts directly	AdminAction
Access dashboard with system statistics	DashboardService
View audit logs of all administrative actions	Notification
Send system-wide notifications to users	
Manage user suspensions with documented reasons	

Class Name: **Category**	
Responsibilities:	**Collaborators:**
Classify items into predefined types	Report
Provide standardized classification system	SearchService
Support filtering in search queries	
Store category names (e.g., Electronics, Documents, Keys)	

Class Name: **Conversation**	
Responsibilities:	**Collaborators:**
Group messages between two users	User
Link to the original post being discussed	Report
Track conversation participants (user1 and user2)	Message
Maintain complete conversation history	
Track last activity timestamp for sorting	
Enable efficient message retrieval	

Class Name: **Message**	
Responsibilities:	**Collaborators:**
Store individual message content and text	Conversation
Track sender information	User
Mark messages as read or unread	Notification
Trigger notifications when message is sent	
Store precise message timestamp	
Support chronological message history	

Class Name: **AdminAction**	
Responsibilities:	**Collaborators:**
Log all administrative actions for audit trail	Admin
Record action type (suspend, delete, resolve, etc.)	User
Track target entity type (user, post, complaint)	Report
Store reason and justification for each action	Complaint
Maintain precise timestamp of action execution	
Enable accountability and admin activity monitoring	

Class Name: **DashboardService**	
Responsibilities:	**Collaborators:**
Calculate total registered users	User
Calculate total published posts	Report
Calculate active complaints count	Complaint
Calculate recovered items statistics	Admin
Display recent system activity	
Provide admin analytics and insights	

2. Mockups.

The mockups can be found in the following Figma link:
https://www.figma.com/design/GeShcSJNjrFG3cii4tXnVS/Flujo-FindItUnal?node-id=0-1&p=f&t=JELSXXVwcr4pSuih-0
All the screens and their corresponding purposes are detailed within the Figma project.

3. Business Model Processes

Core Process: Publication and Recovery of Lost/Found Objects

Process Description:

This process documents the complete flow from when a user publishes a report (either found or lost object) on campus until both parties successfully connect and recover the object. It is the core business process of FindItUNAL, as it represents the main functionality that connects users who have information about lost/found objects.

Role in the Application:

This process is fundamental because:

- It facilitates the return of lost belongings within the university community
- It enables secure communication between users without exposing personal information
- It maintains an auditable record of all reports and claims
- It allows both finders and owners to actively report and search for objects

Actors Involved:

- **User A (Reporter):** Publishes a report (found or lost object) with detailed information
- **User B (Interested Party):** Searches for and contacts regarding an object
- **FindItUNAL System:** Validates information and maintains report status

Process Steps:

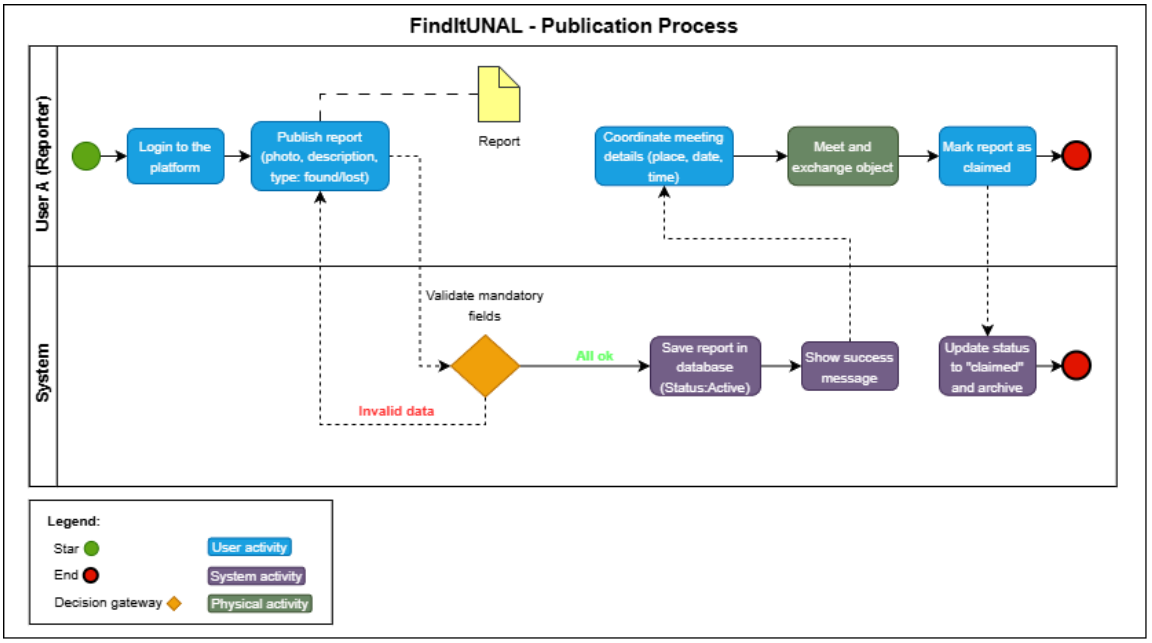
1. **Start:** User finds an object OR loses an object on campus
2. **Publish Report:** Authenticated user creates a report with:
    - Photograph of the object
    - Detailed description (minimum 20 characters)
    - Object category
    - Report type (found/lost)
    - Specific location
    - Date
3. **System Validation:** The system verifies that all mandatory fields are complete
4. **Successful Publication:** The report is published and visible to all users
5. **Search by Interested User:** Another user searches using filters (category, location, date, keywords)
6. **Object Identification:** The interested user finds a matching report
7. **Internal Contact:** The interested user initiates a conversation through the internal chat

8. **Delivery Coordination:** Both users agree on location, date, and time for the return
9. **Physical Exchange:** The object is successfully returned to its owner
10. **Marked as Claimed:** The user who created the report marks it as "object claimed/recovered"
11. **Process Closure:** The report changes its status to "claimed" and is archived

BPMN Diagram:

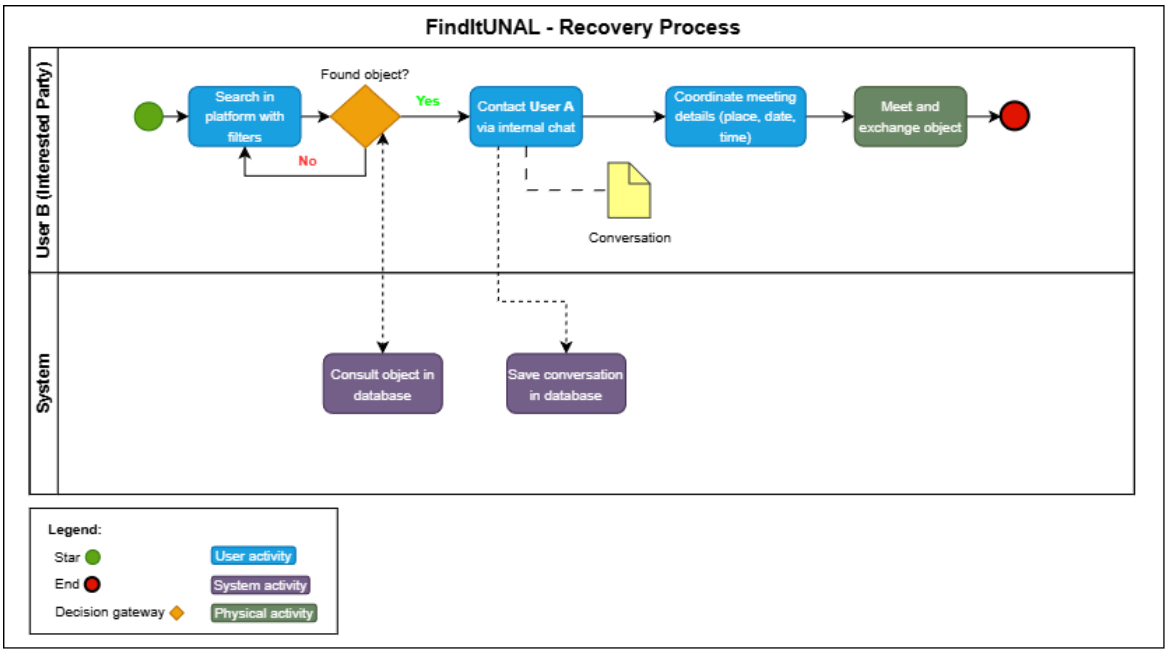
Publication Process:

To view the diagram more clearly, please visit the following link: BMP Puplication.png



Recovery Process:

To view the diagram more clearly, please visit the following link: BMP Recovery.png



Phase 1: Report Publication

- User A finds an object OR loses an object on campus
- Logs in with their institutional credentials
- Completes the form with all required information (photograph, detailed description, category, report type: found/lost, location, date)
- The system validates that all mandatory fields are complete and meet requirements (e.g., minimum description of 20 characters)
- If there are errors, validation messages are displayed and the user can correct them
- Once validated, the system saves the report in the database with "active" status
- The system displays a success message to User A

Phase 2: Search and Contact

- User B (interested party) searches on the platform using filters (category, location, date, keywords)
- If they find a report that matches their lost/found object, they proceed to contact User A
- Uses the platform's internal chat to communicate privately with User A
- Both users coordinate the delivery/meeting details (location, date, time)
- If no match is found, User B can continue searching or end the process

Phase 3: Exchange and Closure

- User A and User B meet physically to exchange the object
- After the successful delivery/recovery of the object, User A (the original reporter) marks the report as "claimed/recovered"
- The system automatically updates the report status to "claimed" and archives it
- The report remains in the history for statistical purposes
- The process concludes successfully

Key Differences from Previous Version:

- **Bidirectional reporting:** Both finders and owners can publish reports
- **Self-managed closure:** The user who created the report is responsible for marking it as claimed (no automatic notifications)
- **Clearer roles:** User A (Reporter) and User B (Interested Party) instead of assuming roles
- **Physical exchange activity:** Explicitly shows the real-world meeting step

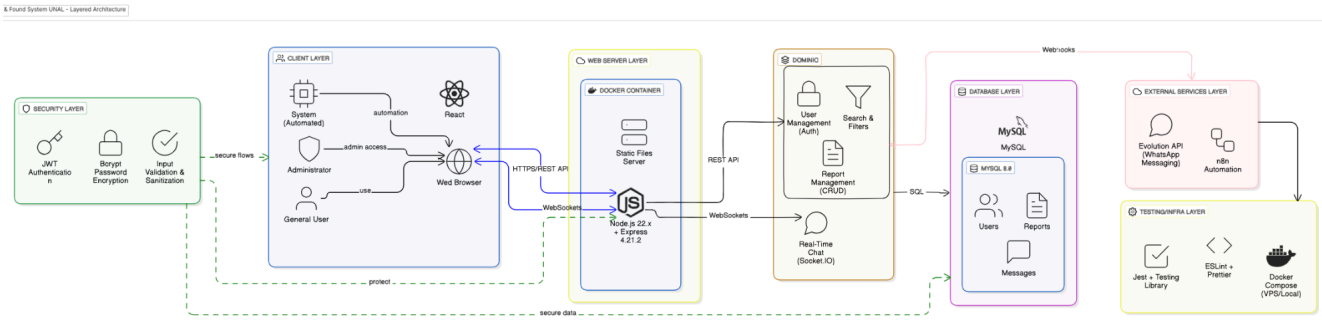
References:

- BPMN 2.0 Specification: https://www.omg.org/spec/BPMN/2.0/
- Visual Paradigm BPMN Guide: https://www.visual-paradigm.com/guide/bpmn/what-is-bpmn/

This core process is essential for FindItUNAL because it represents the complete value cycle that the platform offers: facilitating connections between community members to reunite lost objects with their owners efficiently and securely.

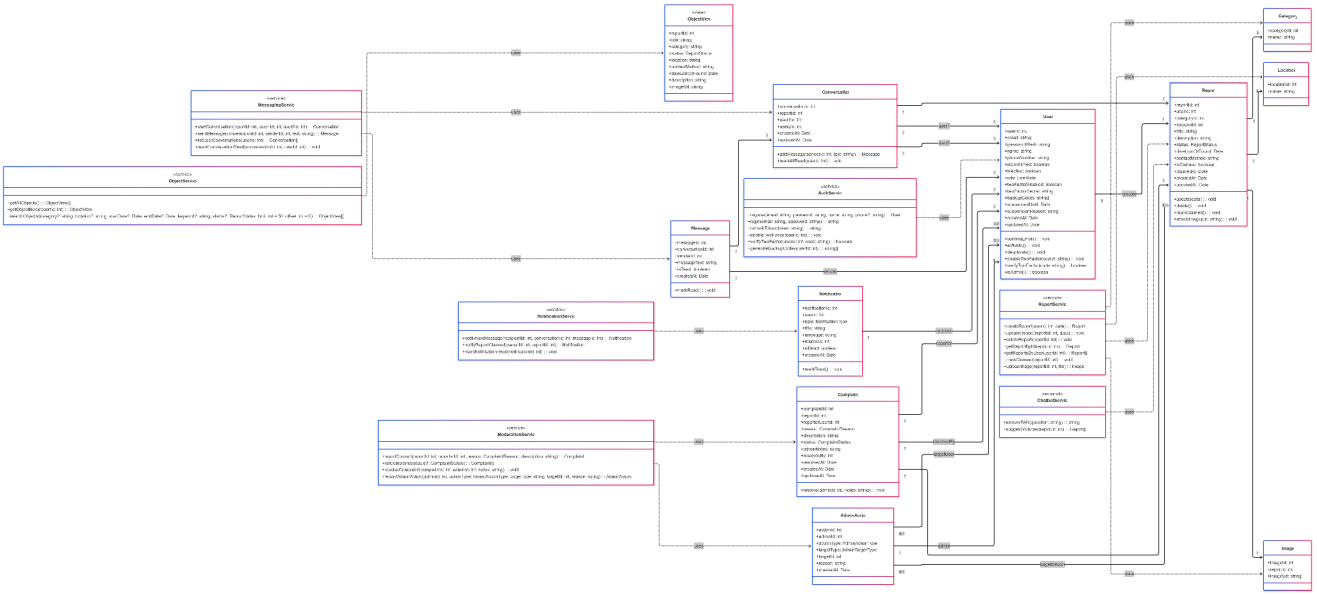
4. Architecture Diagram

To view the diagram more clearly, please visit the following link: Architecture Diagram.png



5. Class Diagram

To view the diagram more clearly, please visit the following link: Diagrama de clases.png



6. Relational Database Model

1. Users

Propósito:Manage all user accounts in the system, including regular users and administrators.

Attribute	Type	Purpose
user_id	INT (PK)	Unique user identifier
email	VARCHAR(255) UNIQUE	Institutional email (@unal.edu.co), used for login
password_hash	VARCHAR(255)	Encrypted password (never stored in plain text)
name	VARCHAR(255)	Full name of the user

phone_number	VARCHAR(20)	Optional phone number for alternative contact
is_confirmed	BOOLEAN	Indicates whether the email has been verified
is_active	BOOLEAN	Active account (TRUE) or inactive/suspended (FALSE)
role	ENUM	Defines permissions: 'user' (regular) or 'admin' (administrator)
two_factor_secret	VARCHAR(32)	Secret key for two-factor authentication (2FA)
two_factor_enabled	BOOLEAN	Indicates if the user enabled 2FA
backup_codes	JSON	Backup codes for 2FA in case of lost access
suspended_until	DATETIME	NEW: Suspension end date (NULL = not suspended)
suspension_reason	TEXT	NEW: Reason for the suspension
created_at	DATETIME	Registration date
updated_at	DATETIME	Last profile update

2. Categories

Purpose: Classify lost/found items into predefined types to improve searches.

Attribute	Type	Purpose
category_id	INT (PK)	Unique category identifier
name	VARCHAR(255) UNIQUE	Category name (e.g., "Electronics", "Keys")

3. Locations

Purpose: Register all campus buildings and locations where items can be lost or found.

Attribute	Type	Purpose
location_id	INT (PK)	Unique location identifier
name	VARCHAR(255) UNIQUE	Location name (e.g., "Building 404", "Main Library")

4. Reports

Purpose: Store all posts of lost or found items. This is the core of the system.

Attribute	Type	Purpose
report_id	INT (PK)	Unique report identifier
user_id	INT (FK)	User who created the report
category_id	INT (FK)	Item category (Electronics, Keys, etc.)
location_id	INT (FK)	Location where it happened
title	VARCHAR(255)	Short title (e.g., "Black iPhone 13 lost")
description	TEXT	Additional details and characteristics
status	ENUM	'lost' or 'found'
date_lost_or_found	DATE	Date when the item was lost or found
contact_method	VARCHAR(255)	Contact method (even though internal messaging exists now)
created_at	DATETIME	Publication date
updated_at	DATETIME	Last update

5. Images

Purpose: Store images associated with each report (a report may have multiple images).

Attribute	Type	Purpose
image_id	INT (PK)	Unique image identifier
report_id	INT (FK)	Associated report
image_url	VARCHAR(255)	Storage URL (AWS S3, Cloudinary, etc.)

6. Conversations

Purpose: Group all messages exchanged between two users about a specific report. Works like a “chat room”.

Attribute	Type	Purpose
conversation_id	INT (PK)	Unique conversation identifier

report_id	INT (FK)	Report that initiated the conversation
user1_id	INT (FK)	User who started the contact (interested party)
user2_id	INT (FK)	Owner of the report
created_at	DATETIME	When the conversation started
updated_at	DATETIME	Last activity (useful for sorting by recent conversations)

7. Messages

Purpose: Store every individual message inside a conversation.

Attribute	Type	Purpose
message_id	INT (PK)	Unique message identifier
conversation_id	INT (FK)	Associated conversation
sender_id	INT (FK)	User who sent the message
message_text	TEXT	Message content
is_read	BOOLEAN	Whether the recipient has read it
created_at	DATETIME	Sent at

8. Notifications

Purpose: Alert users about important events.

Attribute	Type	Purpose
notification_id	INT (PK)	Unique identifier
user_id	INT (FK)	User receiving the notification
type	ENUM	Event type (see types below)
title	VARCHAR(255)	Short title (e.g., "New message")
message	TEXT	Detailed description
related_id	INT	Associated ID (message, report, etc.)
is_read	BOOLEAN	Whether the user has seen it
created_at	DATETIME	When it was generated

9. Complaints

Purpose: Allow users to report fraudulent or inappropriate posts so admins can review them.

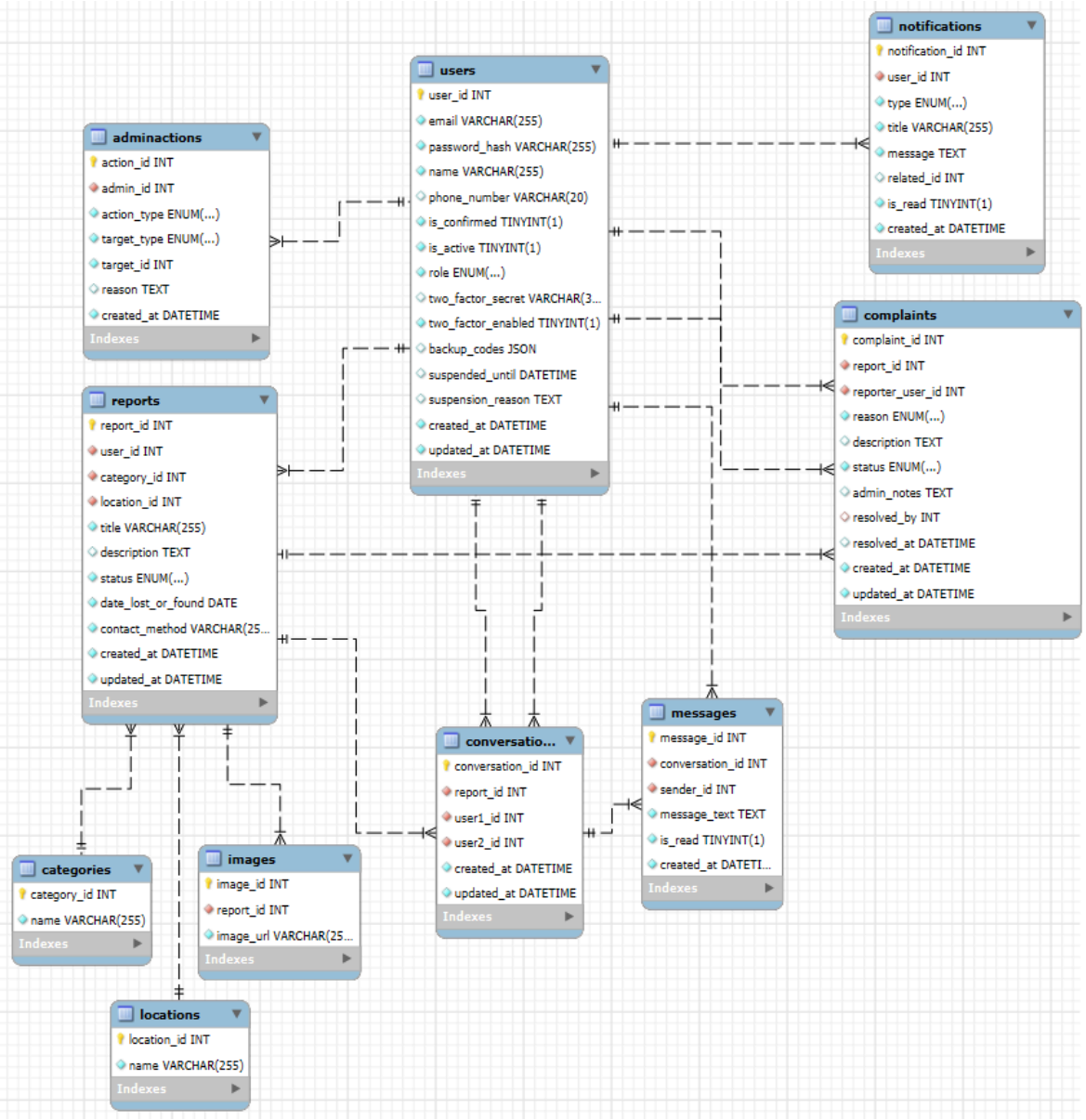
Attribute	Type	Purpose
complaint_id	INT (PK)	Unique identifier
report_id	INT (FK)	Report being complained about
reporter_user_id	INT (FK)	User who submitted the complaint
reason	ENUM	Predefined reason (see list below)
description	TEXT	Optional additional details
status	ENUM	Review status (pending/in_review/approved/rejected)
admin_notes	TEXT	Internal admin notes
resolved_by	INT (FK)	Admin who resolved the complaint
resolved_at	DATETIME	Resolution date
created_at	DATETIME	Submission date
updated_at	DATETIME	Last modification

10. AdminActions (Acciones Administrativas)

Purpose: Full audit log of all actions performed by administrators. Security logging.

Attribute	Type	Purpose
action_id	INT (PK)	Unique identifier
admin_id	INT (FK)	Admin who performed the action
action_type	ENUM	Type of action performed
target_type	ENUM	Entity type affected (user/report/complaint)
target_id	INT	Specific ID of the entity
reason	TEXT	Reason for the action
created_at	DATETIME	Timestamp

EER Diagram


notifications
notification_id INT
user_id INT
type ENUM(...)
title VARCHAR(255)
message TEXT
related_id INT
is_read TINYINT(1)
created_at DATETIME
Indexes
users
user_id INT
email VARCHAR(255)
password_hash VARCHAR(255)
name VARCHAR(255)
phone_number VARCHAR(20)
is_confirmed TINYINT(1)
is_active TINYINT(1)
role ENUM(...)
two_factor_secret VARCHAR(3...
two_factor_enabled TINYINT(1)
backup_codes JSON
suspended_until DATETIME
suspension_reason TEXT
created_at DATETIME
updated_at DATETIME
Indexes
adminactions
action_id INT
admin_id INT
action_type ENUM(...)
target_type ENUM(...)
target_id INT
reason TEXT
created_at DATETIME
Indexes
complaints
complaint_id INT
report_id INT
reporter_user_id INT
reason ENUM(...)
description TEXT
status ENUM(...)
admin_notes TEXT
resolved_by INT
resolved_at DATETIME
created_at DATETIME
updated_at DATETIME
Indexes
reports
report_id INT
user_id INT
category_id INT
location_id INT
title VARCHAR(255)
description TEXT
status ENUM(...)
date_lost_or_found DATE
contact_method VARCHAR(25...
created_at DATETIME
updated_at DATETIME
Indexes
messages
message_id INT
conversation_id INT
sender_id INT
message_text TEXT
is_read TINYINT(1)
created_at DATETI...
Indexes
conversatio...
conversation_id INT
report_id INT
user1_id INT
user2_id INT
created_at DATETIME
updated_at DATETIME
Indexes
categories
category_id INT
name VARCHAR(255)
Indexes
images
image_id INT
report_id INT
image_url VARCHAR(25...
Indexes
locations
location_id INT
name VARCHAR(255)
Indexes

References

[1] Object Management Group, "Business Process Model and Notation (BPMN) Version 2.0," Object Management Group, Ene. 2011. [En línea]. Disponible en: https://www.omg.org/spec/BPMN/2.0/

[2] Visual Paradigm, "What is BPMN?," *Visual Paradigm Guides*. Accedido: 7 de noviembre de 2025. [En línea]. Disponible en: https://www.visual-paradigm.com/guide/bpmn/what-is-bpmn/

[3] L. Bass, P. Clements, y R. Kazman, *Software Architecture in Practice*, 3.ª ed. Upper Saddle River, NJ, EE. UU.: Addison-Wesley Professional, 2012.

[4] A. Silberschatz, H. F. Korth, y S. Sudarshan, *Database System Concepts*, 7.ª ed. New York, NY, EE. UU.: McGraw-Hill, 2019.

[5] E. Gamma, R. Helm, R. Johnson, y J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, MA, EE. UU.: Addison-Wesley Professional, 1994.