INSA Cyber Talent

Database Design Document

Project : Afalgugn

Members:

1. Samuel Asmare
2. Dawit Lulie
3. Furtuna Yimer
4. Matiwos Kebede

Date : August 11 ,2025

Version 1.0

Table of Contents

page

Introduction

Purpose of the Document

This document serves to define and describe the database design for the Afalgugn full stack web project. It provides a detailed outline of the database structure, including tables, columns, data types, constraints, and relationships. The goal is to facilitate a clear understanding among developers,and database administrators, forimplementation, maintenance, and future enhancements.

Scope of the Database

The database covers all aspects of the Afalgugn web application backend, including user authentication, multilingual support, profile management, reporting of missing and found persons, chat functionality between users, subscription management, and verification code handling. It ensures data integrity and efficient data retrieval across these features.

Overview of the System the Database Supports

Afalgugn is an online platform dedicated to assisting in locating missing persons and reconnecting them with their families and friends. The system allows users to register, verify their accounts, submit detailed reports of missing or found individuals, communicate through an internal chat system, and subscribe to notifications. This database supports the storage and management of all such critical information.

Sample SQL Queries

1. Retrieve all verified users:
   ```
   SELECT * FROM users WHERE verified = TRUE;
   ```

2. Find all pending reports:
   ```
   SELECT * FROM report WHERE status = 'pending';
   ```

3. Get unread chat messages for a specific user (user_id = 5):
   ```
   SELECT * FROM chat WHERE receiver = 5 AND is_read = FALSE;
   ```

4. List all languages supported:
   ```
   SELECT * FROM languages;
   ```

5. Fetch profile details for a user by email:
   ```
   SELECT * FROM profile WHERE email=samuel.asmare@a2sv.org';
   ```

6. Get verification codes that are not expired and unused:
   ```
   SELECT * FROM verification_code WHERE expire_date > NOW()
   AND is_used = FALSE;
   ```

7. Count the number of reports per user:
   ```
   SELECT user_id, COUNT(*) AS report_count FROM report GROUP
   BY user_id;
   ```

8. List subscribers who subscribed in the last 30 days:
   ```
   SELECT * FROM subscribe WHERE subscribed_at >=
   DATE_UB(NOW(), INTERVAL 30 DAY);
   ```

4. Database Overview

The Afalgugn database is designed to efficiently store and manage all data related to user accounts, language preferences, profiles, missing and found person reports, messaging between users, subscription management, and

verification processes. It emphasizes data integrity through the use of relational constraints and supports multilingual functionality to serve a diverse user base. The database is structured to scale with the growth of the platform while maintaining fast query performance.

Technologies Used :

- Database System**:** MySQL
- Database Modeling Tool: dbdiagram.io (for visual schema design and ER diagrams)
- Backend Technology: Node.js with Express.js (interacts with the database via SQL queries)
- Other Tools:
    - MySQL Workbench (for database management and SQL scripting)

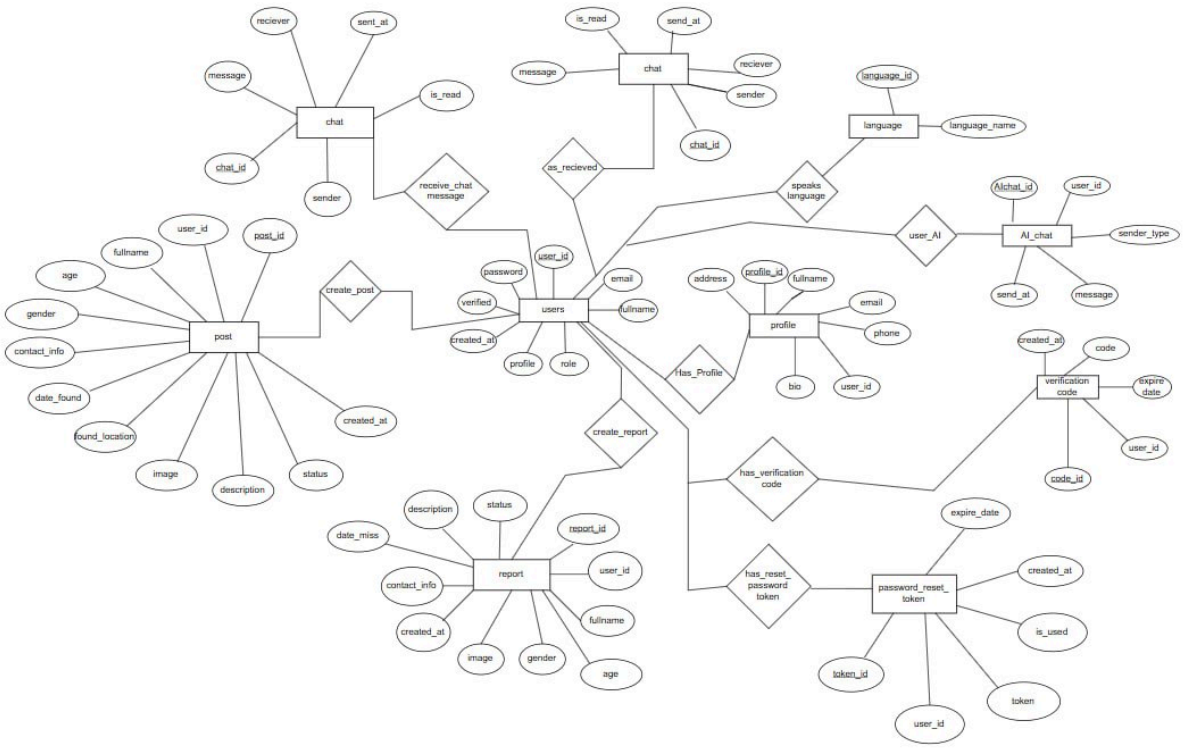
5. Entity-Relationship Diagram (ERD)

The ER diagram visually represents the core tables of the Afalgugn database and how they relate to each other. It demonstrates the structure of entities (tables), their key attributes, and the relationships enforced via foreign keys. This diagram helps both developers and stakeholders understand the data organization and constraints in the system.

Description of Entities and Relationships at a Glance

- **Users:** Central entity storing user account details including authentication and roles. Each user can have one profile and multiple reports and posts, chats, and verification codes.

- **Profile:** Stores extended user information like phone, address, and bio. Linked to the `users` table through a one-to-one relationship (`user_id` foreign key).



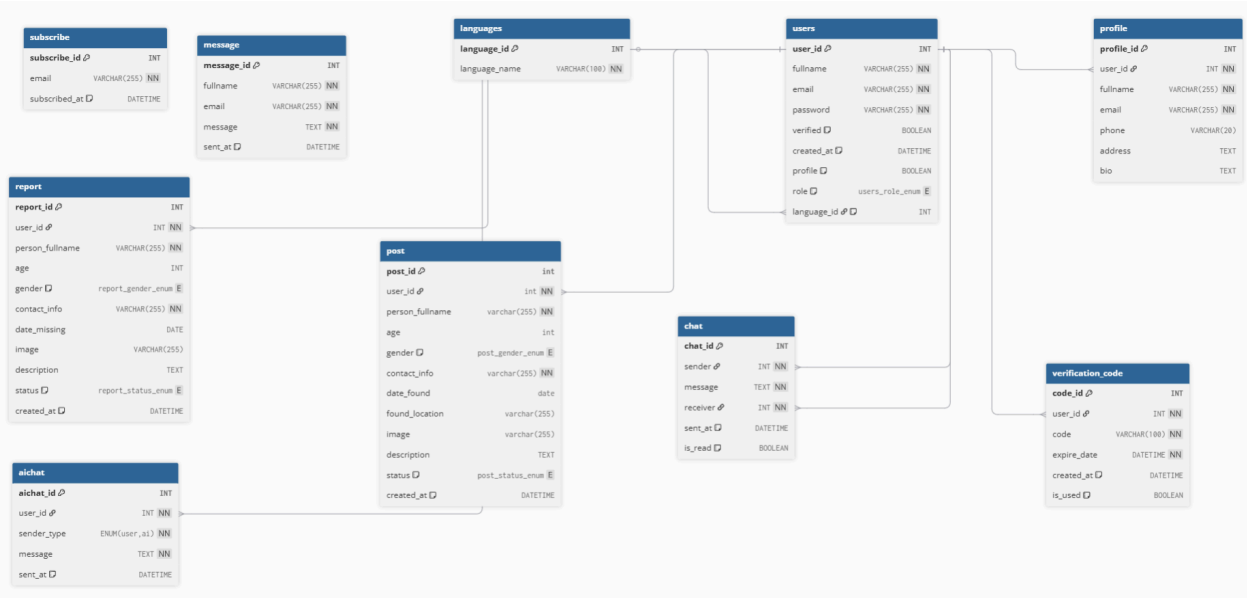
- **Report:** table for missing person reports. Each report belongs to one user (`user_id` foreign key). Reports track details such as person name, age, status, and locations.
- **Posts:** table for found person reports. Each report belongs to one user (`user_id` foreign key). Reports track details such as person name, age, status, and locations.

- **Message:** Stores messages sent through contact forms. Not linked to users, just stores sender details.

- **Chat:** Contains direct messages between users, linking sender and receiver via foreign keys to the `users` table. Tracks whether messages are read.

- **Subscribe:** Holds subscriber emails and subscription timestamps.

- **Verification_Code:** Manages codes for user verification or password resets, linked to users by `user_id`. Tracks expiration and usage status.



6. Tables Description

- The `users` table stores essential account information for every individual registered on the Afalgugn platform. Each user record contains a unique identifier (`user_id`),

which serves as the primary key. The table tracks the full name of the user under the `fullname` field and ensures each email address is unique and mandatory, enabling reliable communication and login functionality. Passwords are securely stored in hashed form within the `password` column.

- To maintain system integrity and security, the `verified` boolean indicates whether a user's email has been confirmed. The `created_at` timestamp captures the exact moment the user account was created, which is useful for auditing and analyzing user registration trends — for instance, you can query all verified users to assess the size of your active user base.

- Additional user status is recorded in the `profile` boolean to signify whether the user has completed their profile setup, while the `role` field distinguishes between regular users and administrators, supporting role-based access control within the system.

- The `language_id` links to the preferred language of the user, enabling multilingual support. This connection to the `languages` table allows the platform to tailor experiences for users by presenting content and notifications in their chosen language.

- This table is fundamental to various queries such as retrieving all verified users to manage active accounts, or linking to related tables like `verification_code` for handling email verification and password resets, and `chat` for sending and receiving messages.