MyDiary

Encrypted Online Diary Web App

CASE STUDY REPORT

Project URL: https://mydiary.gt.tc

Developed by: Yash Anil Patil

Contact: yash.patil.yp687@gmail.com

Under The Guidance of Mrs. Mrunal Jagtap Subject: DMS

Tech Stack: PHP | MySQL | HTML/CSS/JS | AES-256-CBC

Hosted on: InfinityFree.net

Prepared: 2025

1. Executive Summary

MyDiary is a full-stack, secure personal online diary web application built using PHP, MySQL, HTML/CSS, and JavaScript. Hosted at https://mydiary.gt.tc, the application enables users to write, store, and manage private diary entries with end-to-end AES-256-CBC encryption. Every piece of content — including entry text, mood tags, weather data, energy levels, and uploaded images — is encrypted before being persisted to the database, ensuring no plain-text personal data is ever stored at rest.

The application was developed by Yash Anil Patil as an educational and personal project, deployed on InfinityFree.net with a MySQL backend. Beyond simple journaling, MyDiary features a rich Home Dashboard displaying writing streaks, peak moods, active goals, and live weather, a Collage Builder for visual memory creation from uploaded photos, and a Profile page with persistent personal metadata such as an auspicious day and a monthly-throttled life aim tracker.

This case study provides a detailed technical examination of the system architecture, database design, encryption implementation, feature scope, security posture, user experience flow, and the planned future roadmap for the application.

2. Project Overview

2.1 Project Background

Personal diary applications are among the oldest and most trusted categories of personal productivity software. However, many existing solutions either store data in plain text, rely on opaque cloud providers, or lack a meaningful security model. MyDiary was conceived to address this gap — providing a lightweight, self-hosted-capable diary that encrypts all personal content while still delivering a rich user experience.

The project is built using a classic LAMP-adjacent stack (Linux, Apache, MySQL, PHP) deployed on InfinityFree free hosting, making it accessible and reproducible without requiring paid infrastructure. The choice of PHP procedural code keeps the backend simple and auditable, while AES-256-CBC encryption (the same standard used by financial institutions) protects all user data.

2.2 Project Goals

- Provide a secure, encrypted space for users to record daily thoughts, moods, and memories.
- Implement AES-256-CBC end-to-end encryption on all stored content, including metadata.
- Deliver a fully responsive UI that works across desktop, tablet, and mobile browsers.
- Offer image upload support with 1–4 photos per entry for rich memory documentation.

- Build a meaningful home dashboard with writing streaks, mood tracking, and goal setting.
- Host the application on free infrastructure to demonstrate production readiness at zero cost.

2.3 Scope and Limitations

The current version (v1.0) covers authentication, encrypted diary creation, image uploads, collage building, a home dashboard, and a profile page. Out of scope for this version are native mobile applications, push notifications, analytics dashboards, AI-powered features, and multi-user sharing. These are tracked as planned features in the roadmap (see Section 10).

3. System Architecture

3.1 Architecture Overview

MyDiary follows a traditional three-tier web architecture: a Presentation Layer (HTML/CSS/JS running in the browser), an Application Layer (PHP scripts executing on the server), and a Data Layer (MySQL database). There is no modern JavaScript framework or REST API; all page rendering is server-side via PHP, producing complete HTML documents for each request.

This approach prioritises simplicity and portability. Any shared hosting provider with PHP and MySQL support can run the application. The lack of a Node.js runtime, containerisation, or build toolchain means that a developer can clone the repository, update db.php with credentials, import the SQL file, and have a running instance in under five minutes.

3.2 Component Breakdown

- db.php — Database connection singleton; holds master encryption key as a PHP constant.
- auth.php — Handles registration (bcrypt hashing) and login (session initiation).
- entry.php — Handles diary entry creation, encryption, and storage.
- view_entries.php — Retrieves and decrypts entries for the authenticated user.
- images.php — Manages image upload, base64 encoding, and encrypted storage.
- collage.php — Renders the visual image collage from all uploaded diary photos.
- dashboard.php — Aggregates streak, mood, weather, and goal data for the home view.
- profile.php — Displays and updates user profile data including aim and auspicious day.

3.3 Request–Response Lifecycle

1. User submits an HTML form (e.g., new diary entry) via HTTP POST.
2. PHP validates the session token; unauthenticated requests redirect to login.
3. Application code sanitises and validates all input fields.
4. Encryption module generates a unique 16-byte IV and runs AES-256-CBC on each field.
5. Ciphertexts and IV are inserted into MySQL via a prepared statement.
6. On read, the row is fetched, the IV and master key are used to decrypt all fields.
7. Decrypted plain-text is escaped with htmlspecialchars() and rendered in the HTML template.

4. Database Design

4.1 Entity–Relationship Summary

The database contains three core tables: users, diary_entries, and entry_images. The relationships are straightforward: a user can have many diary entries, and each diary entry can have many associated images. The username field serves as the primary foreign-key linkage because it is the unique, human-readable identifier chosen at registration.

Parent Table	Child Table	Cardinality	Join Key
users	diary_entries	One-to-Many	username (FK)
users	entry_images	One-to-Many	username (FK)
diary_entries	entry_images	One-to-Many	entry_id (FK)

Note: Using username as a foreign key rather than a numeric user ID is an intentional design choice that keeps joins readable and debugging straightforward. A future enhancement could migrate to integer foreign keys for better indexing performance at scale.

4.2 Table: diary_entries

This is the central table of the application. Every diary entry record stores encrypted text for all user-authored fields plus the unique IV needed for decryption. No plain-text content is stored.

Column	Data Type	Constraint	Description
id	INT (PK, AI)	NOT NULL	Auto-increment primary key
username	VARCHAR(100)	NOT NULL	Owner of the diary entry
entry_date	DATE	NOT NULL	Date of the diary entry
entry_time	TIME	NOT NULL	Time the entry was created
encrypted_entry	LONGTEXT	NOT NULL	AES-256-CBC encrypted diary body
encrypted_title	VARCHAR(512)	NOT NULL	Encrypted entry title
encrypted_mood	VARCHAR(512)	NOT NULL	Encrypted mood tag
encrypted_weather	VARCHAR(512)	NOT NULL	Encrypted weather data
encrypted_energy_level	VARCHAR(512)	NOT NULL	Encrypted energy level (1-5)

Column	Data Type	Constraint	Description
encrypted_social_interaction	VARCHAR(512)	NOT NULL	Encrypted social interaction tag
iv	VARCHAR(64)	NOT NULL	Unique IV per entry for AES-CBC

4.3 Table: entry_images

Images are stored as AES-encrypted Base64 strings linked to a diary entry via the entry_id foreign key. Each image has its own IV, meaning images are independently encrypted from their parent entry.

Column	Data Type	Constraint	Description
id	INT (PK, AI)	NOT NULL	Auto-increment primary key
entry_id	INT (FK)	NOT NULL	References diary_entries.id
username	VARCHAR(100)	NOT NULL	Owner of the image
encrypted_image	LONGTEXT	NOT NULL	Base64-encoded, AES encrypted image blob
encrypted_caption	VARCHAR(512)	NULLABLE	Encrypted caption for the image
iv	VARCHAR(64)	NOT NULL	Unique IV for this image encryption
uploaded_at	DATETIME	NOT NULL	Timestamp of upload

4.4 Table: users

The users table stores all account-level data. Sensitive fields such as passwords are hashed (not encrypted) because they do not need to be retrieved in plain-text. The auspicious_day_info field is set once and made read-only through application logic, not a database constraint.

Column	Data Type	Constraint	Description
id	INT (PK, AI)	NOT NULL	Auto-increment primary key
username	VARCHAR(100)	UNIQUE NN	Unique username (login identifier)
email	VARCHAR(255)	UNIQUE NN	User email address
profile_picture	LONGTEXT	NULLABLE	Base64 or file path of profile picture
auspicious_day_info	VARCHAR(512)	NULLABLE	Set-once auspicious day; read-only after save
current_aim	VARCHAR(512)	NULLABLE	User's current aim/goal text

Column	Data Type	Constraint	Description
aim_last_set_date	DATE	NULLABLE	Date aim was last updated (monthly throttle)
password	VARCHAR(255)	NOT NULL	Hashed password (bcrypt)
created_at	DATETIME	NOT NULL	Signup timestamp
last_login	DATETIME	NULLABLE	Timestamp of most recent login
last_entry	DATETIME	NULLABLE	Timestamp of most recent diary entry

5. Encryption Architecture

5.1 Algorithm Choice: AES-256-CBC

AES (Advanced Encryption Standard) with a 256-bit key in CBC (Cipher Block Chaining) mode is the encryption algorithm used throughout MyDiary. AES-256 is the same standard used by governments and financial institutions worldwide for protecting classified and sensitive data. CBC mode with a unique IV per record ensures that identical plaintext inputs produce completely different ciphertexts, defeating frequency analysis attacks.

PHP's built-in OpenSSL extension provides the implementation via openssl_encrypt() and openssl_decrypt(). The master key is defined as a PHP constant in db.php, stored server-side and never transmitted to the browser.

5.2 Per-Field Encryption Model

A key design decision in MyDiary is that every column containing user-authored content is individually encrypted — including the title, mood, weather, energy level, and social interaction fields, not just the main diary text. This means that even if an attacker gains read access to the database, they cannot discern patterns in mood or behavioural data without the master key.

5.3 Step-by-Step Encryption Flow

#	Step	Description
1	User writes entry	Browser sends form POST with plain-text fields to PHP backend
2	Generate unique IV	PHP generates a cryptographically random 16-byte IV for this record
3	Encrypt each field	openssl_encrypt() called on title, mood, weather, energy, text separately with AES-256-CBC + Master Key + IV
4	Store ciphertext + IV	All encrypted blobs and the IV are stored in diary_entries table
5	Retrieve on read	PHP queries row, extracts IV and all encrypted columns
6	Decrypt on display	openssl_decrypt() called with Master Key + stored IV; plain-text rendered to authenticated user
7	Key never leaves server	Master key is a PHP constant in db.php; never exposed to browser or URL

5.4 Encryption Limitations and Recommendations

- The master key is stored on the same server as the database. A compromised server exposes both the key and the data simultaneously. A key management service (e.g., AWS KMS) would mitigate this in a production environment.

- The master key cannot be changed after deployment without re-encrypting all records. Users should be warned not to modify it post-launch.
- Future versions plan user-derived key encryption (PBKDF2 from user password), which would achieve true end-to-end security even from the server administrator.

6. Feature Analysis

6.1 Feature Status Matrix

The table below summarises all current and planned features, their implementation status, and supplementary notes.

Feature	Status	Notes
Secure Signup / Login	**Implemented**	bcrypt password hashing, PHP sessions
AES-256-CBC Entry Encryption	**Implemented**	Unique IV per record, master key on server
Mood / Weather / Energy Tags	**Implemented**	All metadata fields encrypted individually
Image Upload (1-4 per entry)	**Implemented**	Base64-encoded, encrypted before storage
Collage Builder Page	**Implemented**	Visual memory collage from uploaded images
Home Dashboard	**Implemented**	Streak, peak mood, goal, weather widget
Profile Page + Picture	**Implemented**	Email, signup date, aim, auspicious day
Aim (monthly throttle)	**Implemented**	Change aim restricted once per calendar month
Dark Mode	**Planned**	Scheduled for future release
Mood Analytics	**Planned**	Trend charts and mood insights dashboard
Daily Reminders	**Planned**	Push/email notification system
AI Journal Suggestions	**Planned**	AI-assisted writing prompts
Calendar View	**Planned**	Visual month-view browsing of entries

6.2 Core Feature Descriptions

6.2.1 Diary Entry Creation

The diary entry form collects six user-authored fields: entry date, entry title, main diary text, mood tag, weather condition, and energy level. Each field is individually encrypted before storage. Users can additionally upload between 1 and 4 photos per entry. The form is fully validated client-side with JavaScript and server-side in PHP.

6.2.2 Home Dashboard

The dashboard aggregates several computed metrics: a daily thought/quote, the total number of diary entries (Total Memories), the count of new entries in the current calendar month, the current writing streak (consecutive days with at least one entry), the user's peak mood (most frequent

mood tag in recent entries), live weather for the user's locale, and the active goal from the profile page.

6.2.3 Collage Builder

The Collage Builder page retrieves all images uploaded across all diary entries, decrypts them, and renders them in a responsive grid layout. Users can visually browse their photo memories as a mosaic without needing to navigate individual entries. This page decrypts images only for the authenticated session and never writes decrypted data to disk.

6.2.4 Profile Page

The profile page displays the user's email, signup date, last entry timestamp, auspicious day (set once and locked), current aim, and profile picture. The 'Change Aim' button is throttled to once per calendar month using the aim_last_set_date column and a PHP date comparison, preventing impulsive goal-abandonment while still allowing periodic updates.

7. Technology Stack

MyDiary is built on a deliberately simple, widely-supported technology stack to maximise portability and minimise hosting requirements. Every component of the stack is available on standard shared hosting providers without any server configuration.

Layer	Technology	Details
Frontend	HTML5 / CSS3 / JavaScript	Responsive, custom-styled UI with no external CSS frameworks
Backend	PHP (Procedural)	Server-side processing, session management, form handling
Database	MySQL	Relational storage hosted on InfinityFree MySQL service
Encryption	AES-256-CBC (OpenSSL)	All entry text and metadata encrypted before database storage
Hosting	InfinityFree.net	Free cloud hosting with MySQL, PHP 7/8 support
Auth	Session-based + bcrypt	PHP sessions; passwords hashed with bcrypt via password_hash()
Images	Base64 / File Uploads	Profile pics and diary images stored as encrypted blobs

7.1 Technology Selection Rationale

PHP was chosen for its ubiquity on shared hosting and its built-in support for OpenSSL encryption, session management, and MySQL connectivity. No framework is used — procedural PHP keeps the codebase readable and free from framework-specific conventions, making the security model transparent and auditable.

MySQL on InfinityFree provides a zero-cost relational store with sufficient performance for a single-user or small-group diary application. The schema is intentionally minimal — three tables with clear relationships — to keep queries simple and efficient.

Vanilla HTML, CSS, and JavaScript are used for the frontend. No React, Vue, or Angular is needed because all page rendering is server-side. The responsive design is achieved with custom CSS media queries, avoiding the bundle-size overhead of a CSS framework like Bootstrap.

8. Security Analysis

8.1 Threat Model and Mitigations

The following table maps identified threat vectors to the mitigations implemented in the current version of MyDiary, along with an honest assessment of residual risks.

Threat Vector	Mitigation	Details
SQL Injection	Prepared Statements / PDO	All DB queries use parameterized inputs to prevent injection attacks
Password Storage	bcrypt (cost factor 12)	Passwords never stored in plain text; salted hash via password_hash()
Entry Confidentiality	AES-256-CBC Encryption	Every entry encrypted with a unique Initialization Vector (IV)
Session Hijacking	PHP Session + HTTPS	Sessions regenerated on login; HTTPS enforced on InfinityFree hosting
Image Safety	Encrypted Blob Storage	Images stored as AES-encrypted Base64 strings, not raw files
Key Management	Server-side Master Key	Single master key stored in config file; IV unique per record
Brute-Force Login	No Rate Limit (planned)	Basic protection via session; CAPTCHA/throttle planned for v2
XSS Prevention	htmlspecialchars() on output	User-supplied content escaped before rendering in PHP templates

8.2 Encryption Key Security

The most significant security design decision in MyDiary is the placement of the master AES key. Currently, the key resides in db.php as a PHP constant on the same server as the application and database. While this is standard practice for shared-hosting projects, it does mean that a server-level compromise (e.g., a malicious hosting provider or a server breach) would expose both the key and the ciphertexts.

For the educational and personal use target audience, this risk is accepted as a reasonable trade-off for simplicity. A production deployment with higher security requirements should consider environment variable injection, a hardware security module (HSM), or a cloud key management service to physically separate key storage from data storage.

8.3 HTTPS and Transport Security

InfinityFree provides free SSL/TLS certificates for custom domains. The application is accessible over HTTPS at https://mydiary.gt.tc, ensuring all data in transit is encrypted. PHP sessions are

bound to HTTPS-only cookies to prevent session tokens from leaking over unencrypted connections.

9. User Journey & Experience

9.1 End-to-End User Flow

The following table traces a new user's complete journey through MyDiary from their first visit to a standard daily usage pattern.

Step	Action	User Does	System Response
1	**Visit Site**	Opens https://mydiary.gt.tc	Home / Login page displayed
2	**Register**	Fills signup form with username/email	Account created; redirected to dashboard
3	**Set Auspicious Day**	Enters once-in-lifetime auspicious day	Saved permanently; cannot be changed
4	**Write Entry**	Enters diary text, mood, weather, energy	Entry encrypted and stored in DB
5	**Upload Images**	Attaches 1-4 photos to entry	Images encrypted and linked via entry_id
6	**View Collage**	Opens Collage Builder page	Gallery of all uploaded images rendered
7	**Check Dashboard**	Views home page stats	Streak, peak mood, weather, goal shown
8	**Update Aim**	Clicks 'Change Aim' on profile	New aim saved; button locked for 30 days
9	**Logout**	Clicks logout button	PHP session destroyed; redirected to login

9.2 UX Design Principles

- Privacy first: the user is never asked to input data that cannot be encrypted before storage.
- Minimal friction: the entry form is a single page with clearly labelled fields; no multi-step wizards.
- Positive reinforcement: the writing streak counter rewards daily journaling habits.
- Emotional context: mood and weather fields encourage holistic self-reflection, not just event logging.
- Memory preservation: the collage builder transforms uploaded photos into a visual timeline of memories.

9.3 Responsive Design

The application uses CSS media queries to adapt layout across screen widths. On mobile devices (< 768px), the navigation collapses to a hamburger menu, form fields stack vertically, and the dashboard grid switches from a multi-column to a single-column layout. The collage builder adapts its grid column count from 4 on desktop to 2 on tablet and 1 on mobile.

10. Future Roadmap

10.1 Planned Features

The following table outlines the planned enhancements for upcoming versions, categorised by effort level and the version milestone in which they are targeted.

Version	Feature	Effort	Description
v1.1	Dark Mode	Low	CSS variable theming + user preference stored in DB
v1.1	Calendar View	Medium	Month-grid UI showing entry dots; click to read
v1.2	Mood Analytics Dashboard	Medium	Line/bar charts of mood, energy over time
v1.2	Activity Trend Reports	Medium	Writing frequency heatmap (GitHub-style)
v1.3	AI Journal Suggestions	High	LLM-powered prompts based on past entries
v1.3	Daily Reminder Notifications	Medium	Email or push notifications to log daily entries
v2.0	End-to-End User Key Option	High	Allow user-derived key (PBKDF2) for maximum privacy
v2.0	Export / Backup	Low	Download all entries as encrypted JSON or PDF
v2.0	Mobile App (PWA)	High	Progressive Web App with offline caching

10.2 Architectural Evolution

As the application grows beyond a personal project toward a multi-user public service, several architectural improvements would be warranted. First, the username-as-foreign-key pattern should be migrated to integer user IDs for better indexing performance. Second, the monolithic PHP file structure should be organised into a simple MVC or HMVC pattern with autoloaded classes and a routing layer. Third, a caching layer (Redis or Memcached) would benefit the dashboard's aggregation queries.

For genuine end-to-end encryption where even the server administrator cannot read diary entries, a client-side encryption approach using the Web Crypto API (SubtleCrypto) with PBKDF2 key derivation from the user's password would be the gold standard. This would, however, require that the master key never leave the browser, necessitating careful key backup and recovery UX design.

11. Setup & Deployment Guide

11.1 Prerequisites

- A PHP-enabled web hosting account (PHP 7.4 or later recommended).
- A MySQL 5.7+ database with CREATE TABLE and INSERT privileges.
- FTP/SFTP access or a hosting file manager to upload project files.
- OpenSSL extension enabled in PHP (enabled by default on most shared hosts).

11.2 Installation Steps

8. Clone or download the project repository to your local machine.
9. Create a new MySQL database through your hosting control panel (e.g., cPanel or Plesk).
10. Import the provided .sql file using phpMyAdmin or the MySQL CLI to create all three tables.
11. Open db.php and update the DB_HOST, DB_USER, DB_PASS, DB_NAME, and MASTER_KEY constants with your credentials.
12. Upload all project files to the public_html (or equivalent) directory via FTP.
13. Set directory permissions to 755 for folders and 644 for PHP files.
14. Create an uploads/ directory with write permission (chmod 755) if using file-based image storage.
15. Visit your domain in a browser, register an account, and verify that the login and entry creation flows work correctly.

11.3 Security Checklist for Deployment

- Ensure MASTER_KEY is a randomly generated string of at least 32 characters.
- Do not commit db.php to a public version control repository.
- Enable HTTPS on the domain and configure PHP session cookies as HTTPS-only.
- Restrict direct web access to db.php via a .htaccess deny rule.
- Backup the MASTER_KEY securely — without it, no existing entries can be decrypted.

12. Conclusion

MyDiary represents a well-architected, privacy-focused personal journaling application that demonstrates how encryption can be integrated into a classic PHP/MySQL stack without sacrificing usability. By applying AES-256-CBC encryption to every user-authored data field, the application ensures that personal thoughts, moods, and memories remain private even if the underlying database is compromised.

The project successfully balances simplicity with security. The choice of a procedural PHP backend, vanilla JavaScript frontend, and standard MySQL schema keeps the codebase accessible to developers at all levels while providing a transparent, auditable security model. The InfinityFree deployment demonstrates that a production-grade encrypted web application can be hosted at zero cost with no compromise on core functionality.

The richness of the feature set — writing streaks, mood tracking, image uploads, collage building, and goal setting — elevates MyDiary beyond a simple text-entry application into a comprehensive personal wellness tool. The planned roadmap, which includes mood analytics, AI-assisted journaling, and a Progressive Web App, positions the project for continued growth while maintaining its foundational commitment to user privacy.

In summary, MyDiary is a technically sound, creatively designed, and genuinely useful application that achieves its goal of providing a safe, encrypted space for personal reflection and memory preservation. It serves as an excellent reference implementation for developers wishing to understand how to integrate application-layer encryption into a PHP web application.

— End of Case Study —

Project by Yash Anil Patil | yash.patil.yp687@gmail.com | https://mydiary.gt.tc

MyDiary — ER Diagram & Database Description

Entity Relationship Analysis | Database Schema | Attribute Classification | Relationship Cardinalities

1. Entity Relationship Diagram (ERD)

The diagram below represents the complete Entity Relationship model for the MyDiary database. Three entities are depicted — USERS (strong), DIARY_ENTRIES (strong), and ENTRY_IMAGES (weak, double-bordered) — along with their attributes (ellipses), relationships (diamonds), and cardinality labels.

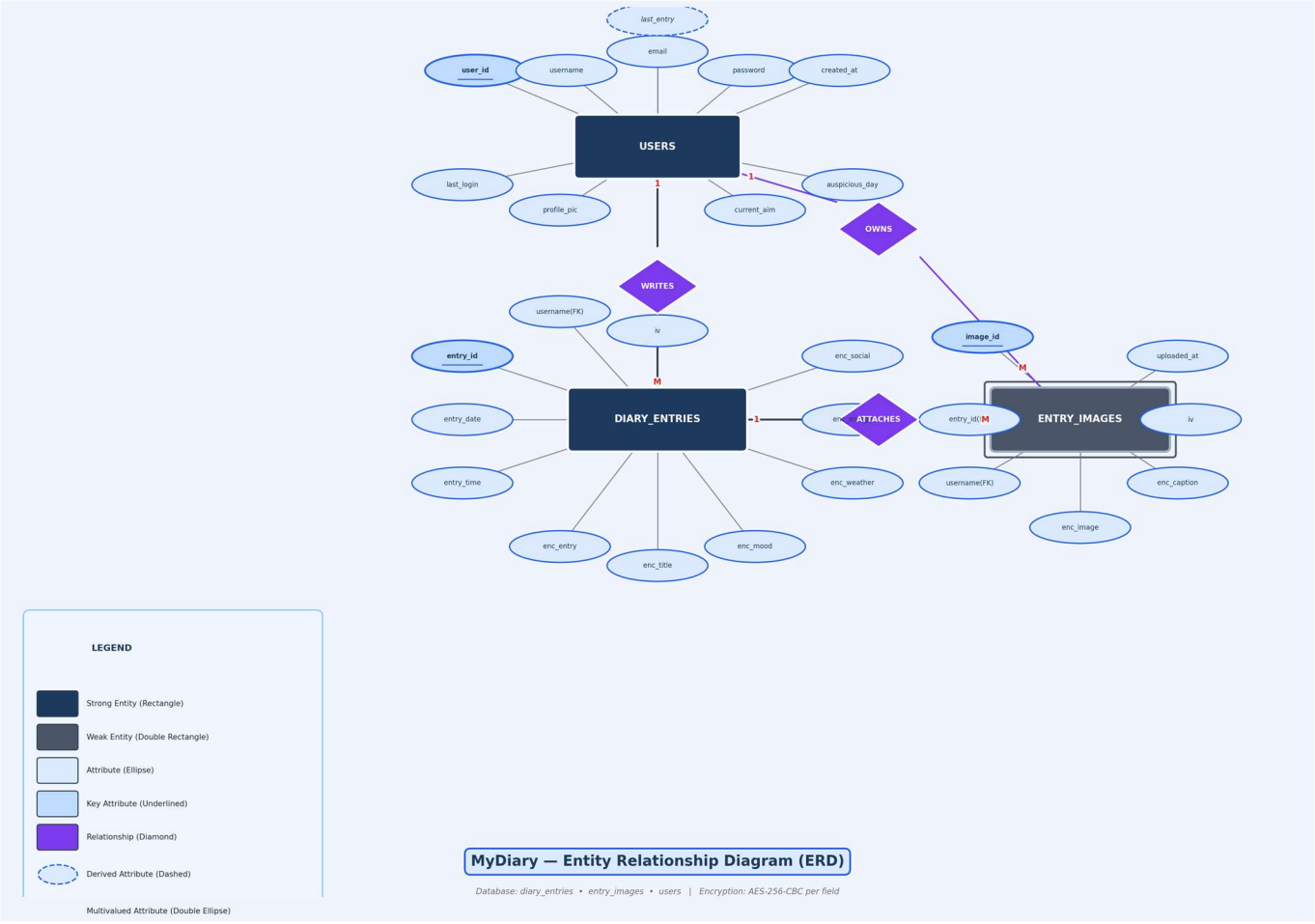

Figure 1: MyDiary Entity Relationship Diagram — USERS, DIARY_ENTRIES, ENTRY_IMAGES

2. Entities Identified

An entity is a real-world object or concept about which data is stored. MyDiary's database contains three entities, classified below as strong or weak based on their ability to exist independently.

2.1 Strong Entities

Strong entities have independent existence and possess their own primary key that uniquely identifies each record without relying on any other entity.

- USERS — Represents a registered diary user. Each user has a unique id (PK) and username. The entity can exist entirely on its own.
- DIARY_ENTRIES — Represents one diary entry written by a user on a specific date. Has its own id (PK) and exists independently, though it references a user via username (FK).

2.2 Weak Entity

A weak entity cannot exist on its own and depends on a parent entity for its identity and context.

- ENTRY_IMAGES — Represents one photograph attached to a diary entry. It cannot exist without a parent DIARY_ENTRIES record (entry_id FK) and a parent USERS record (username FK). Identified by its own id, but meaningless without the parent entry.

Reason: ENTRY_IMAGES has no independent existence — if a diary entry is deleted, its associated images lose all meaningful context and should be deleted too. The entity is shown with a double-bordered rectangle in the ER diagram to denote its weak nature.

3. Types of Attributes

Attributes describe the properties of an entity. Each attribute in MyDiary is classified below by type, with examples drawn from the three database tables.

Attribute Type	Definition	Example (MyDiary)
Simple	Cannot be divided into sub-parts	entry_date, entry_time, uploaded_at, created_at, last_login, iv
Composite	Can be subdivided into smaller sub-parts	name (USERS: username + email), address metadata (auspicious_day_info)
Multivalued	Can hold multiple values for one entity	entry_images (1–4 images per diary_entries record)
Derived	Calculated or inferred from other attributes	last_entry (derived from max entry_date in diary_entries), writing streak (computed from entry dates)
Key Attribute	Uniquely identifies each entity instance	id (USERS), id (DIARY_ENTRIES), id (ENTRY_IMAGES)

4. Strong Entities — Detail

Strong entities in the MyDiary schema are those that possess a primary key and can be stored without depending on any other table for their existence.

- USERS — Primary key: id (INT, AUTO_INCREMENT). Contains all account-level data: username, email, password (bcrypt hash), profile_picture, auspicious_day_info, current_aim, aim_last_set_date, created_at, last_login, and the derived last_entry.
- DIARY_ENTRIES — Primary key: id (INT, AUTO_INCREMENT). Stores per-entry data: entry_date, entry_time, and nine encrypted fields (enc_entry, enc_title, enc_mood, enc_weather, enc_energy_level, enc_social_interaction) plus the AES initialisation vector iv. References USERS via username (FK).

Both USERS and DIARY_ENTRIES have independent existence and their own primary keys. A USERS record can exist with zero diary entries. A DIARY_ENTRIES record can conceptually exist even if no images are attached.

5. Weak Entity — Detail

A weak entity is one that cannot be uniquely identified by its own attributes alone and depends on a parent (owner) entity for its identity and contextual meaning.

- ENTRY_IMAGES — Primary key: id (INT, AUTO_INCREMENT). However, id alone does not fully identify the record's purpose without knowing which diary entry (entry_id FK) and which user (username FK) it belongs to. The record is logically deleted when the parent DIARY_ENTRIES record is removed.

Reasons for Weak Classification:

- ENTRY_IMAGES cannot exist without a DIARY_ENTRIES record — entry_id (FK) is a mandatory NOT NULL field.
- ENTRY_IMAGES cannot exist without a USERS record — username (FK) is also mandatory.
- Both parent entities must exist first before an image can be inserted, establishing a total participation dependency.
- Images have no business meaning outside the context of their parent entry; they are subordinate records that serve only to enrich a specific diary entry.

6. Relationships in the ER Diagram

Three relationships connect the entities in MyDiary's database. Each is described below with its name, participating entities, cardinality, and a plain-language explanation — following the same convention as standard ER analysis.

1. WRITES *(USERS → DIARY_ENTRIES)*

Between: USERS → DIARY_ENTRIES
Cardinality: **1 : M**

One user can write multiple diary entries over time. Each diary entry belongs to exactly one user. A user with zero entries still exists; a diary entry without a user is not permitted (username FK is NOT NULL).

2. ATTACHES *(DIARY_ENTRIES → ENTRY_IMAGES)*

Between: DIARY_ENTRIES → ENTRY_IMAGES
Cardinality: **1 : M**

One diary entry can have multiple images attached to it (up to 4 as enforced by application logic). Each image record belongs to exactly one diary entry via the entry_id foreign key. A diary entry with no images is valid; an image without a parent entry is not.

3. OWNS *(USERS → ENTRY_IMAGES)*

Between: USERS → ENTRY_IMAGES
Cardinality: **1 : M**

One user can own multiple image records across all their diary entries. Each image record is associated with exactly one user via the username foreign key. This relationship exists in addition to the ATTACHES relationship to allow direct user-level queries of all images (e.g., the Collage Builder page) without needing to join through diary_entries.

6.1 Relationship Cardinality Summary

Relationship	From Entity	To Entity	Cardinality
WRITES	USERS	DIARY_ENTRIES	**1 : M**
ATTACHES	DIARY_ENTRIES	ENTRY_IMAGES	**1 : M**
OWNS	USERS	ENTRY_IMAGES	**1 : M**

7. Participation Constraints

Participation constraints define whether every entity instance must participate in a relationship (total participation) or only some need to (partial participation).

7.1 Total Participation (Mandatory)

- DIARY_ENTRIES in WRITES — Every diary entry must have an associated USERS record. The username (FK) column is NOT NULL; an entry cannot exist without an owner.
- ENTRY_IMAGES in ATTACHES — Every image must belong to a diary entry. The entry_id (FK) column is NOT NULL; an image cannot exist without a parent entry.
- ENTRY_IMAGES in OWNS — Every image must belong to a user. The username (FK) column is NOT NULL.

7.2 Partial Participation (Optional)

- USERS in WRITES — A user does not have to have written any diary entries. A newly registered user has zero entries; this is perfectly valid. Participation is optional (partial).
- DIARY_ENTRIES in ATTACHES — A diary entry does not have to have any attached images. Text-only entries are fully supported. Participation is optional.
- USERS in OWNS — A user who has never uploaded images has no ENTRY_IMAGES records. Participation is optional.

8. Special Design Note: Encrypted Attributes

MyDiary implements an unusual database pattern: instead of storing plain-text attribute values, every user-authored field in DIARY_ENTRIES and ENTRY_IMAGES is stored as AES-256-CBC ciphertext. From a pure database modelling perspective, these columns still function as regular VARCHAR/LONGTEXT attributes — they just contain opaque encrypted strings rather than readable text.

Each row carries its own iv (Initialisation Vector) column, which is the cryptographic nonce required to decrypt that row's ciphertext fields. The iv is unique per row, ensuring that two identical diary entries produce completely different stored ciphertexts. This means the iv is itself a critical, non-optional attribute of both DIARY_ENTRIES and ENTRY_IMAGES.

Encrypted Column	Type	Description
encrypted_entry / encrypted_image	LONGTEXT	AES-256-CBC encrypted main content
encrypted_title	VARCHAR(512)	Encrypted diary entry title
encrypted_mood	VARCHAR(512)	Encrypted mood tag (e.g. happy, sad)
encrypted_weather	VARCHAR(512)	Encrypted weather condition

Encrypted Column	Type	Description
encrypted_energy_level	VARCHAR(512)	Encrypted energy rating (1–5)
encrypted_social_interaction	VARCHAR(512)	Encrypted social context tag
encrypted_caption	VARCHAR(512)	Encrypted image caption text
iv	VARCHAR(64)	Unique AES initialisation vector for this row

9. Primary and Foreign Keys

The table below summarises all primary keys (PK) and foreign keys (FK) in the MyDiary schema, documenting how entities are identified and linked.

Table	Column	Key Type	Notes
USERS	id	PK	Auto-increment integer; uniquely identifies each user account
USERS	username	UNIQUE	Human-readable unique identifier; used as FK in other tables
DIARY_ENTRIES	id	PK	Auto-increment integer; uniquely identifies each diary entry
DIARY_ENTRIES	username	FK → USERS	Links the entry to its author; NOT NULL (total participation)
ENTRY_IMAGES	id	PK	Auto-increment integer; uniquely identifies each image record
ENTRY_IMAGES	entry_id	FK → DIARY_ENTRIES	Links image to parent entry; NOT NULL (total participation)
ENTRY_IMAGES	username	FK → USERS	Links image to owner user; NOT NULL (total participation)

— End of ER Diagram & Description —

Developed by Yash Anil Patil | yash.patil.yp687@gmail.com | https://mydiary.gt.tc