IBAN Number Validation Web Application

Name: Danushka Dissanayaka
Submitted Date: 22/12/2024

1. Project Overview

The application enables users to validate IBAN numbers while ensuring secure user management for both regular users and administrators.

2. Features Implemented

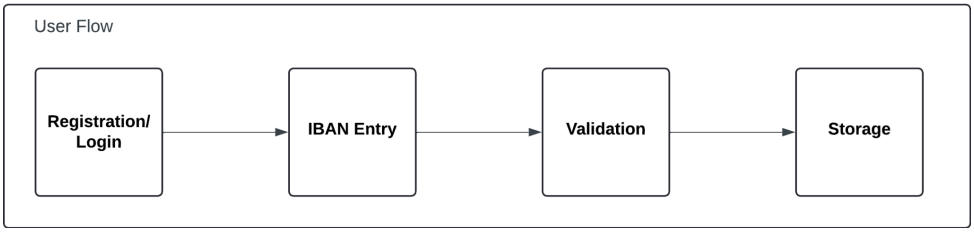
- User Registration - Allow users to sign up with the application to use the IBAN validator.
- User Login and IBAN Entry- Directs logged-in users to a page where they can enter an IBAN.
- IBAN Validation - Implements custom logic to validate IBAN numbers without third-party libraries.
- Data Storage - Stores valid IBAN numbers in the database with encryption.
- Admin Login - Enables admins to securely log in using their credentials.
- Paginated List of IBAN Numbers - Admins can view a paginated list of all submitted IBAN numbers and search iban number form user name.

3. Technologies Stack Used

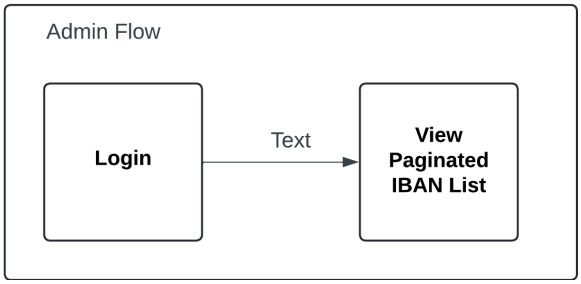
Frontend	● Vue.js ● Node.js 22.9.0 ● fortawesome/fontawesome-svg-core 6.7.2 ● element-plus 2.9.1 ● bootstrap 4
Backend	● laravel 10 ● php 8.2 ● tymon/jwt-auth 2.1
Database	● MySQL

4. Application Architecture

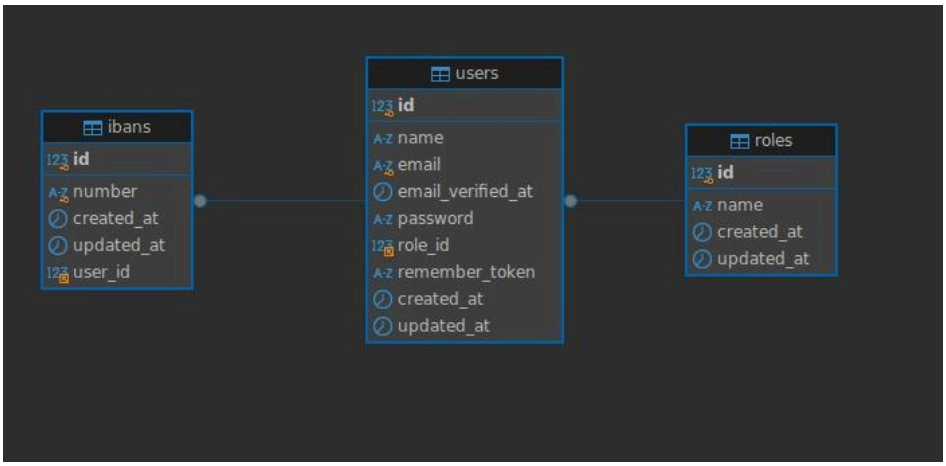
- Frontend
    - Built with Vue.js to create a dynamic and responsive UI.
    - Integrated a RESTful API for seamless communication with the backend.
    - Utilized Vue.js for its reactive components and efficient state management.
- Backend
    - Implemented a Laravel-based three-layered architecture for better maintainability.
    - Employed Laravel's dependency injection to decouple concrete code from abstract interfaces.
    - Developed the backend to meet requirements without relying on third-party libraries.
- Database
    - User and admin credentials are securely stored with hashed passwords.
    - Database normalization was applied to structure IBAN entries along with metadata such as timestamps.
    - IBAN numbers are securely handled in the database by encrypting their values.
- Error Handling Approach
    - Frontend: Global error handlers provide user-friendly notifications.
    - Backend: Validates inputs, sanitizes data, and returns appropriate HTTP status codes.
- User Flow
    - Registration/Login -> IBAN Entry -> Validation -> Storage.



    - Admin Flow Login -> View Paginated IBAN List.



ER Diagram



1. The Users Table stores information about application users, including their unique id, name, email, hashed password, and role_id linking to the roles table. It tracks email verification with email_verified_at, session management with remember_token, and includes created_at and updated_at timestamps.

2. The Roles Table defines user roles with a unique id, name (e.g., "Admin" or "User"), and created_at and updated_at timestamps.

3. The IBANs Table stores user-entered IBANs with a unique id, the IBAN number, a user_id linking to the users table, and created_at and updated_at timestamps.

Key Design Decisions

1. Role Management
    - By maintaining a separate roles table, the application can easily scale to support more roles in the future (e.g., "Super Admin," "Guest").
    - This design also makes it easier to enforce role-based access control (RBAC).
2. IBAN Tracking
    - Associating IBANs with users (user_id foreign key) ensures traceability, making it possible to view all IBAN submissions made by a specific user.
    - Timestamp fields (created_at, updated_at) provide a historical record of when IBANs were submitted or modified.
3. Scalability
    - The use of relationships (e.g., linking users to roles and ibans) ensures a normalized database structure, reducing data redundancy and improving maintainability.

5. IBAN Account Number Validation Algorithm

Validate IBAN Account Number XX2501101250000000069512300

Step 1: Rearrange the Characters

Move the first four characters of the IBAN to the end of the number.

- Original IBAN: XX2501101250000000069512300
- Rearranged: 011012500000000069512300XX25

Step 2: Translate Letters to Numbers

Convert each letter into its corresponding numerical value using the provided alphabet-to-number translation table.

Alphabet to Number Translation Table:

Letter	Value	Letter	Value	Letter	Value	Letter	Value
A	10	G	16	M	22	S	28
B	11	H	17	N	23	T	29
C	12	I	18	O	24	U	30
D	13	J	19	P	25	V	31
E	14	K	20	Q	26	W	32
F	15	L	21	R	27	X	33
Y	34	Z	35				

Conversion:

- Rearranged IBAN: 011012500000000069512300XX25
- Convert letters:
    - X → 33
    - X → 33
- Result: 011012500000000069512300333325

Step 3: Divide by 97

Divide the converted number by 97 to calculate the modulo

- Converted number: 011012500000000069512300333325
- Division: 011012500000000069512300333325 ÷ 97
- Modulo (remainder): 19

Step 4: Check the Modulo

If the modulo equals 1 the IBAN is valid. Otherwise, it is invalid.

- Modulo: 19
- Result: The IBAN XX25011012500000000069512300 is **invalid** as the modulo is not 1.

– END –