**Database Design Document: Testify System**

### 1. Overview

This document outlines the database design for the “Testify” platform, which enables users to take tests, submit answers, and review results. The schema supports multiple test sections, images, scoring, and user roles.

### 2. Entity-Relationship Summary

#### Users & Roles

* **users**: Stores user details including credentials, contact, and personal information.
* **roles**: Defines user roles such as admin, test creator, or student.

Relationship: - A user belongs to exactly one role (role\_id FK). - One role can have many users.

#### Tests & Sections

* **tests**: Represents a complete test. Includes metadata like time limit, score, and visibility.
* **sections**: Each test can have multiple sections. Each section can have its own time limit and question count.

Relationship: - One test has many sections (test\_id FK).

#### Questions & Answers

* **questions**: Belongs to a section. Can be of different types (MCQ, True/False, etc.).
* **answers**: Each question has multiple answers.

Relationship: - One section has many questions (section\_id FK). - One question has many answers (question\_id FK).

#### Images

* **question\_images**: Stores images related to a question.
* **answer\_images**: Stores images related to an answer.

#### Submissions

* **test\_submissions**: A record for each test that a user attempts.
* **submission\_answers**: Stores the response of a user for each question in a submission.
* **submission\_answer\_details**: Details the options selected by a user for each question.

Relationship: - A user can make many test submissions (user\_id, test\_id FK). - Each test submission can have many submission\_answers. - Each submission\_answer can have many submission\_answer\_details.

### 3. Table Definitions

#### users

* id (PK)
* email, username, password\_hash, full\_name
* avatar\_url, phone\_number, dob, gender, status
* email\_verified, last\_login\_at
* FK: role\_id
* Timestamps: created\_at, updated\_at

#### roles

* id (PK)
* name, description
* Timestamps

#### tests

* id (PK)
* title, description, instructions, total\_score
* time\_limit, is\_public, is\_active
* FK: created\_by (user ID)
* Timestamps

#### sections

* id (PK)
* FK: test\_id
* title, description, order, time\_limit, is\_randomized, number\_questions
* Timestamps

#### questions

* id (PK)
* FK: section\_id
* title, type, difficulty, score, explanation, is\_publish
* FK: created\_by
* Timestamps

#### answers

* id (PK)
* FK: question\_id
* content, is\_correct, order
* Timestamps

#### question\_images

* id (PK)
* FK: question\_id
* url, alt\_text, caption, order
* Timestamps

#### answer\_images

* id (PK)
* FK: answer\_id
* url, alt\_text, caption
* Timestamps

#### test\_submissions

* id (PK)
* FK: user\_id, test\_id
* score, started\_at, submitted\_at, duration, status, is\_passed
* created\_at

#### submission\_answers

* id (PK)
* FK: submission\_id, question\_id
* is\_correct, score, duration, status, is\_passed
* created\_at

#### submission\_answer\_details

* id (PK)
* FK: submission\_answer\_id, answer\_id
* is\_selected

### 4. Notes & Best Practices

* All tables use UUID or autoincrement PKs.
* Timestamps (created\_at, updated\_at) are included in all relevant tables.
* Foreign key constraints ensure data integrity.
* Images are stored as URLs; actual image data should be managed by a separate service.

### 5. Potential Improvements

* Normalize gender and status fields using reference tables.
* Add audit logging tables if required.
* Implement tagging or categorization for questions and tests.
* Consider caching is\_correct and score in submission\_answer\_details for performance.

**Document Version:** 1.0  
**Last Updated:** 2025-07-19