

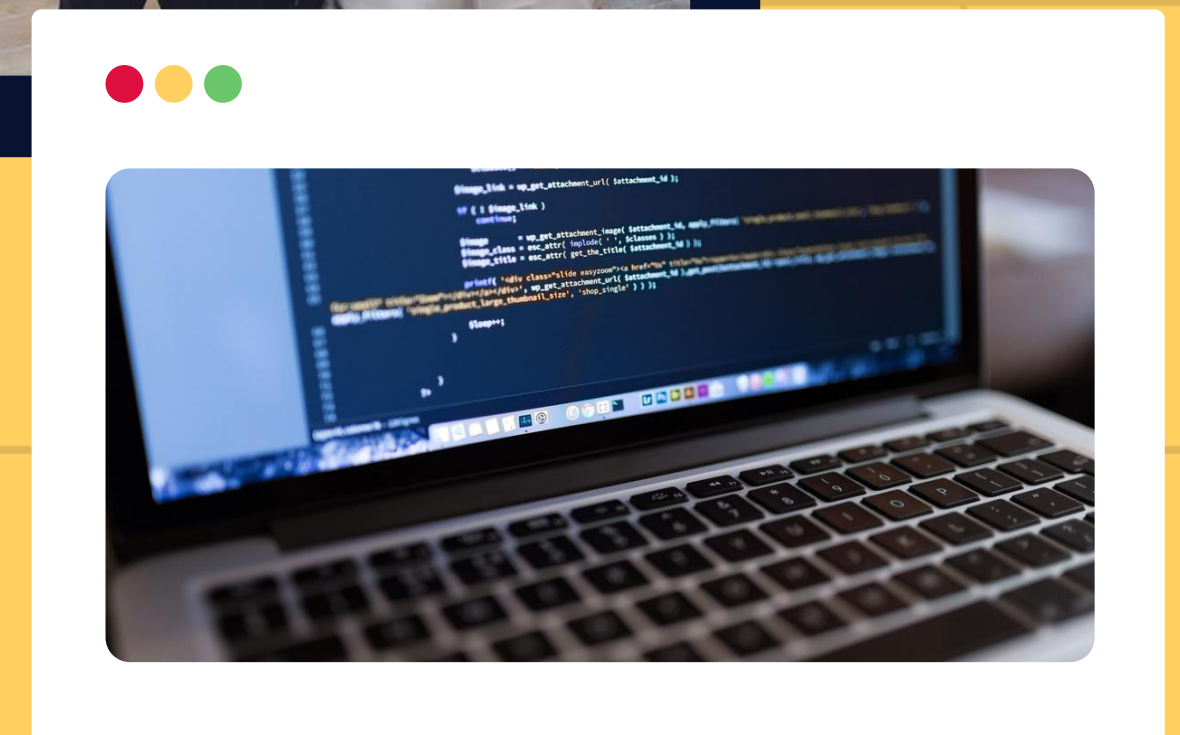
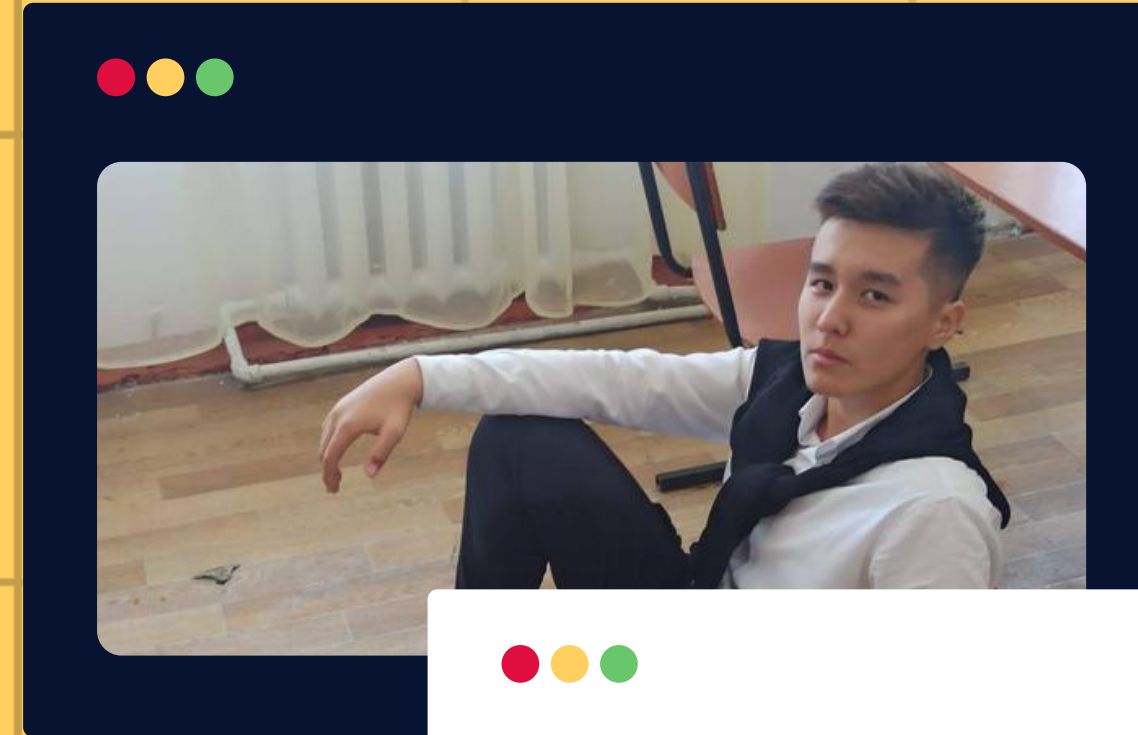
Online Quiz System - DBMS



Presented By
Alish Akadil

why this topic?

I chose the topic **Online Quiz System** for my database project cause I noticed that many teachers still use paper-based tests. This method is time-consuming and not efficient. By creating an online system, we can simplify the process of giving, taking, and checking quizzes. It will save time, reduce paper use, and make it easier for teachers to track students' results.



where did i start?

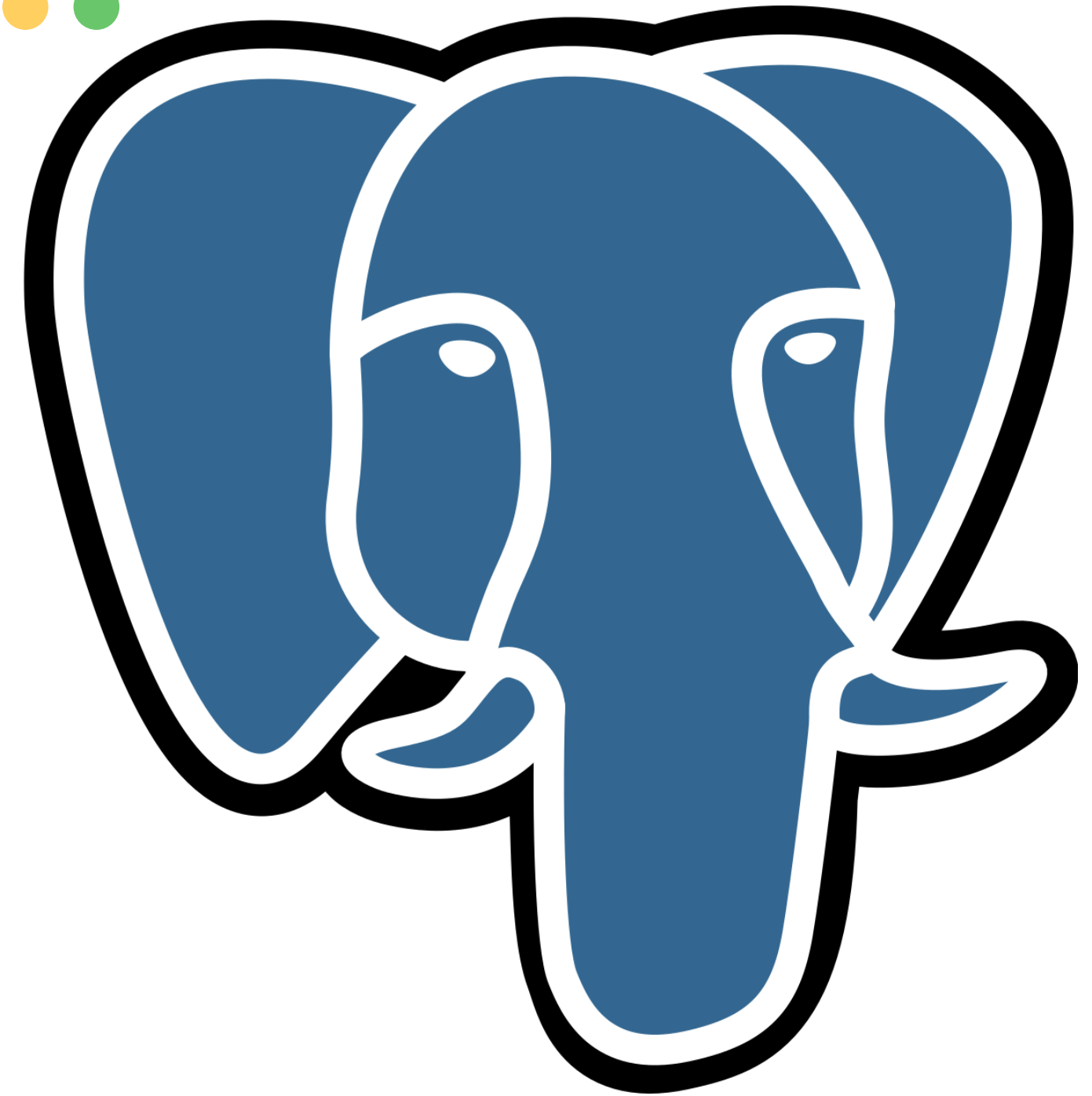
I started by creating a basic version of the **Online Quiz System** in my first mini project. It included simple tables like users, quizzes, and questions.



Then, in the second mini project, I expanded the database by adding more entities and relationships. I'm using **PostgreSQL** for all development.

INNOVATION:

The innovative part of my project is that it replaces traditional paper-based quizzes with a digital system. Many teachers still use printed tests, which take time to prepare and check. My Online Quiz System offers a faster, more efficient, and environmentally friendly way to create, deliver, and grade quizzes.



Tables:



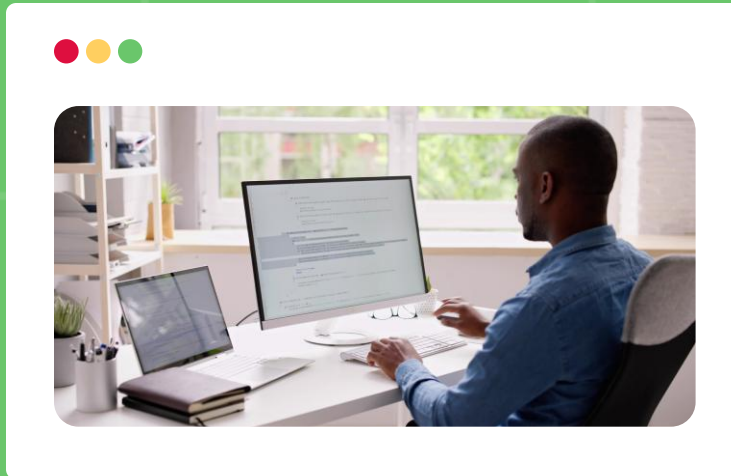
When you enter the correct PIN code, you will be redirected to the menu

You are greeted by a beautiful menu with enticing music.

You are offered 3 games:

1. Falling Shapes, 2. Box Breaker, 3. DVD Bounce

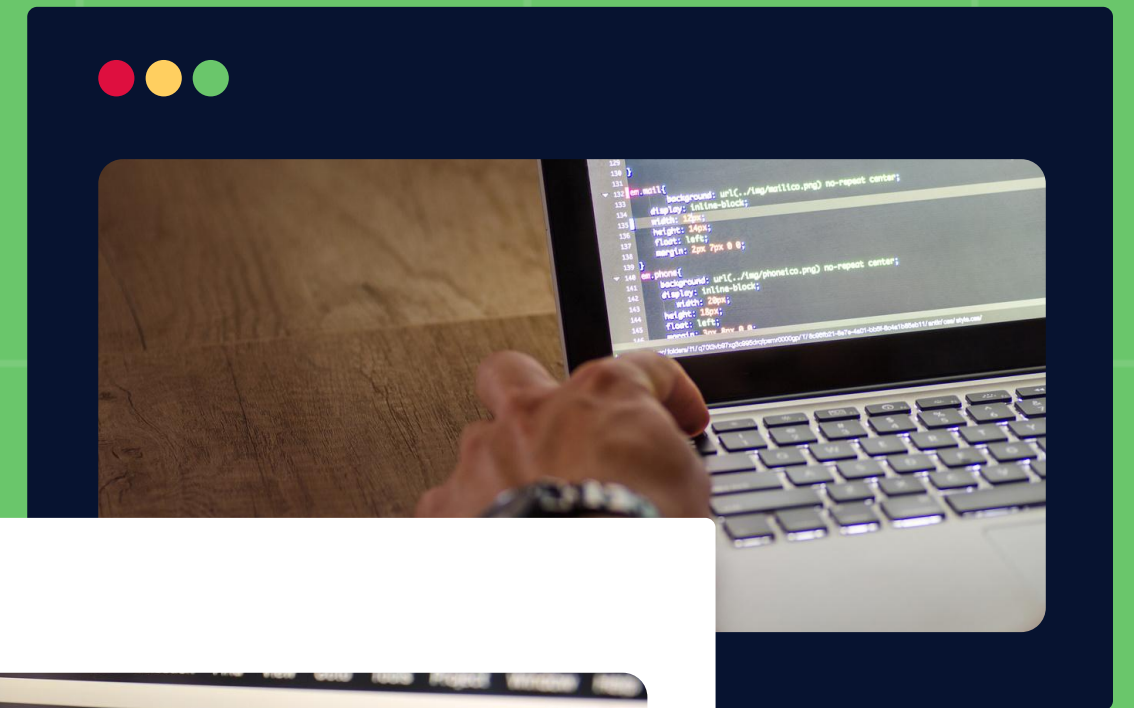
and there is a button to **exit (QUIT)**



Oops! Bugs!



Everyone makes mistakes, even programmers! Sometimes there are errors in the code, called "bugs." Programmers need to find and fix these bugs to make their programs work properly. It's like being a detective!



```
main.js  x  index.html
function hexSearch() {
  var hex = $('#hexSearch').val().replace(/#/g, '');
  if (hex.length == 0 || hex.length == 3 || hex.length == 6) {
    var re = /[0-9A-Fa-f]{6}/g;
    var re2 = /[0-9A-Fa-f]{3}/g;
    if (re.test(hex) || re2.test(hex) || hex.length == 0) {
      $('#notification').css('display', 'none');
      if (hex.length == 3) {
        hex = hex.split('');
        hex = hex[0] + hex[0] + hex[0] + hex[0] + hex[0] + hex[0];
      } else {
        $('#notificationText').html('The thing you typed into the input');
        $('#notification').css('display', 'block');
        return;
      } else {
        $('#notificationText').html('The thing you typed into the input');
        $('#notification').css('display', 'block');
        return;
      }
    }
  }
  var colors =
```

1st table: userr

1. userr

The `userr` table stores **account information** for every user in the system.

This includes personal credentials and profile data such as:

- `user_id`: Unique identifier for each user (Primary Key).
- `username`, `email`: Used for login and identification.
- `password_hash`: Encrypted password for security.
- `role`: Defines user permissions (e.g., admin, student).
- `status`, `last_login`, `created_at`: For account tracking and login history.
- `profile_picture`: Optional field for UI personalization.

💡 **Why it matters:** Every quiz, feedback, and result is tied back to a user.

1. userr

- `user_id` (*integer, PK*)
- `username` (*varchar*)
- `email` (*varchar*)
- `password_hash` (*text*)
- `created_at` (*timestamp*)
- `role` (*varchar*)
- `status` (*varchar*)
- `profile_picture` (*text*)
- `last_login` (*timestamp*)

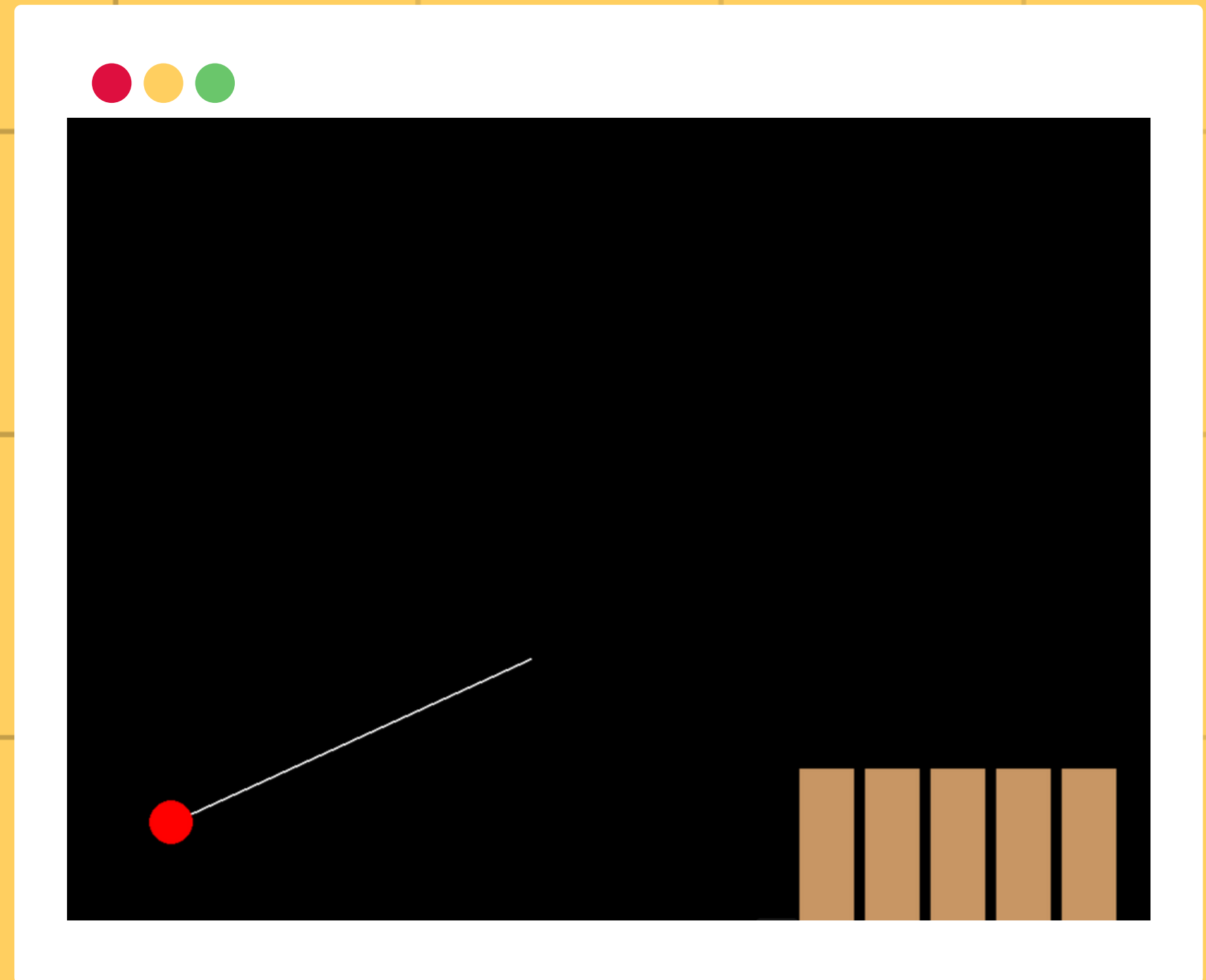
2nd table: quiz

2. quiz

This table defines the quizzes available in the system.

- `quiz_id`: Unique ID (PK).
- `creator_id`: FK to `userr` — who made the quiz.
- `title`, `description`, `category`: Textual info about the quiz.
- `difficulty_level`: easy, medium, or difficult.
- `time_limit`, `max_score`: Quiz settings.
- `visibility`: Determines if quiz is public or private.

🔗 Connected to: Questions, results, feedback.



3rd, 4th, 5th table: questions

■ 3. question_easy

Stores easy-level questions for quizzes.

- Includes `question_text`, `points`, `hint`, `question_type`.
- FK `quiz_id` links to the quiz it belongs to.

📁 Bonus fields like `image_url` and `explanation` enrich the content.

■ 4. question_medium

Same structure as `question_easy`, but for medium-difficulty questions.

■ 5. question_difficult

Same structure again — but for difficult-level questions.

This separation by difficulty allows for better quiz control and modular question pools.

📁 3. question_easy

- `question_id` (*integer, PK*)
- `quiz_id` (*FK → quiz.quiz_id*)
- `question_text` (*text*)
- `created_at` (*timestamp*)
- `points` (*integer*)
- `explanation` (*text*)
- `image_url` (*text*)
- `hint` (*text*)
- `question_type` (*varchar*)

📁 4. question_medium

- Same columns as `question_easy`

📁 5. question_difficult

- Same columns as `question_easy`

6th, 7th, 8th table: answers

■ 6. answer_easy

Stores answer options for easy questions:

- `answer_text` : The answer itself.
- `is_correct` : True/False to mark if it's correct.
- `submission_count` : How often it was selected.
- FK `question_id` links to the `question_easy` table.

■ 7. answer_medium

Same as `answer_easy`, but linked to `question_medium`.



📁 6. answer_easy

- `answer_id` (*integer, PK*)
- `question_id` (*FK → question_easy.question_id*)
- `answer_text` (*text*)
- `is_correct` (*boolean*)
- `explanation` (*text*)
- `submission_count` (*integer*)

📁 7. answer_medium

- Same columns as `answer_easy`

📁 8. answer_difficult


- Same columns as `answer_easy`

9th table: result

9. result

Keeps records of quiz attempts:

- `result_id`: Attempt ID.
- `user_id`, `quiz_id`: Foreign Keys.
- `score`, `time_taken`, `correct_answers`, `incorrect_answers`: Performance stats.
- `highest_streak`: Gamified metric.
- `quiz_feedback`: Optional free-text comment.

 Useful for tracking user performance, trends, and grading.

9. result


- `result_id` (*integer, PK*)
- `user_id` (*FK → userr.user_id*)
- `quiz_id` (*FK → quiz.quiz_id*)
- `score` (*integer*)
- `attempt_date` (*timestamp*)
- `time_taken` (*integer*)
- `correct_answers` (*integer*)
- `incorrect_answers` (*integer*)
- `highest_streak` (*integer*)
- `quiz_feedback` (*text*)

10th table: reward

10. reward

Represents virtual or real rewards users can earn:

- Tied to a `user_id`.
- Includes `reward_name`, `points_required`, and `granted_at`.
- Fields like `status` and `reward_type` allow managing redemption status or type (e.g., badge vs. certificate).

 Supports gamification or progress incentives.

10. reward

- `reward_id` (*integer, PK*)
- `user_id` (*FK → userr.user_id*)
- `reward_name` (*varchar*)
- `points_required` (*integer*)
- `granted_at` (*timestamp*)
- `reward_type` (*varchar*)
- `status` (*varchar*)

11th table: feedback

11. feedback

Collects user feedback on quizzes:

- Tied to both `user_id` and `quiz_id`.
- `rating`: Numeric score.
- `comment`, `response_from_admin`: Open-text fields.
- `submitted_at`: Timestamp for tracking.
- `feedback_category`: Organizes types like bug reports, praise, etc.

📌 Great for system improvements or user communication.

11. feedback

- `feedback_id` (*integer, PK*)
- `user_id` (*FK → userr.user_id*)
- `quiz_id` (*FK → quiz.quiz_id*)
- `rating` (*integer*)
- `comment` (*text*)
- `submitted_at` (*timestamp*)
- `response_from_admin` (*text*)
- `feedback_category` (*varchar*)

end...
?






Thank You
for ur attention!