# 📋 Test Plan & Report – Quiz Application (Laravel)

## 1. Scope & Objectives

| **Application** | Web‑based quiz platform (guest users + admin panel) |
| --- | --- |
| **Tech stack** | Laravel 11, MySQL, Blade, REST API, PHPUnit/Pest, Dusk |
| **Focus of testing** | Functional behaviour, edge‑case robustness, basic security posture & role segregation |
| **Out of scope** | UI/look‑and‑feel polishing, performance benchmarking |

## 2. Test Strategy

| Layer | Tooling | Approach |
| --- | --- | --- |
| **Unit / feature** | **PHPUnit + Pest** | Cover model relations, controllers & API endpoints |
| **E2E / acceptance** | **Laravel Dusk** | Simulate browser flows (guest quiz, admin CRUD) |
| **Security sanity** | Manual + Laravel guards | CSRF, auth, file‑upload, SQL‑injection probes |

## 3. Test Data & Seeders

Create a dedicated seeder so functional & E2E runs don’t rely on production data.

// database/seeders/TestDataSeeder.php  
class TestDataSeeder extends Seeder  
{  
 public function run(): void  
 {  
 // 3 quizzes x 3 questions x 3 answers each  
 Quiz::factory(3)->create()->each(function ($quiz) {  
 Question::factory(3)->create(['quiz\_id' => $quiz->id])->each(function ($question) {  
 Answer::factory()->count(3)->sequence(  
 ['correct' => 1], // first correct  
 ['correct' => 0],  
 ['correct' => 0]  
 )->create(['question\_id' => $question->id]);  
 });  
 });  
 // admin user  
 User::factory()->create([  
 'name' => 'Admin QA',  
 'email' => 'qa\_admin@example.com',  
 'password' => bcrypt('securePass!'),  
 'role' => 'admin',  
 ]);  
 }  
}

**Run**: php artisan migrate:fresh --seed --class=TestDataSeeder

## 

## 4. Functional Test Suite (happy‑path)

| ID | Scenario | Route | Expected |
| --- | --- | --- | --- |
| F‑01 | Guest can register name & reach quiz list | POST /start | Redirect → /selectquiz ; record in users table |
| F‑02 | Guest can answer full quiz & see score | /quiz/{id} … /results/{id} | Percentage & N/M correct displayed |
| F‑03 | Admin login accepts valid creds | /admin/login | 302 → /admin/selectquiz |
| F‑04 | Admin adds question w/ image | POST /admin/add‑question… | DB rows + file in storage |
| F‑05 | Admin edits answers (all‑at‑once) | PUT /admin/edit‑answers/{question} | Answers updated; exactly one marked correct |
| F‑06 | First question deletion blocked | DELETE /admin/delete‑question/{id} where order=1 | Error flash; record remains |

## 5. Edge‑Case Tests

| ID | Edge Input | Expected Guard |
| --- | --- | --- |
| E‑01 | Empty quiz (0 questions) | Front‑end blocks start ; API returns 404/422 |
| E‑02 | Image >2 MB or non‑image | 422 validation error; file not stored |
| E‑03 | Two answers both correct=1 | Validation fails (only one allowed) |
| E‑04 | Guest manipulates querystring answers length ≠ questions | Controller ignores extra / pads missing; no crash |
| E‑05 | Delete question w/ existing results | Foreign‑key cascade or protected with error message |

## 6. Security Checks

| ID | Risk | Test | Expected |
| --- | --- | --- | --- |
| S‑01 | **CSRF** | Submit admin form without token | 419 error |
| S‑02 | **AuthZ** | Access /admin/editquiz as guest | Redirect to admin login |
| S‑03 | **File upload** | Upload .php disguised as image | Validation fail (mime check) |
| S‑04 | **SQLi** | ' OR 1=1 -- in search params | Query builder param‑bind prevents injection |
| S‑05 | **Directory traversal** | ../../evil.jpg as filename | Storage façade sanitises path |

## 7. Execution Instructions

1. git checkout <feature‑branch>
2. composer install && npm i && npm run build *(if assets)*
3. php artisan migrate:fresh --seed --class=TestDataSeeder
4. **Run unit/feature:** php artisan test
5. **Run Dusk (headless):** php artisan dusk --browse=chrome (ensure chromedriver running)

## 8. Test Report Template (Markdown)

### Test Cycle: 2025‑07‑xx  
| Case ID | Result | Notes |  
|---------|--------|--------------------------|  
| F‑01 | Pass | ‑ |  
| F‑02 | Fail | Score % mis‑calc when 0/0 |

## 9. Acceptance Test – QA & PO

1. Log in as [**qa\_admin@example.com**](mailto:qa_admin@example.com)\*\* / securePass!\*\*
2. Create a new quiz, add question with image, verify preview.
3. Log out ➜ register as guest ➜ complete quiz ➜ confirm score.
4. Verify results & result\_details rows in DB.
5. Confirm image displays in guest flow.

If all steps pass → **PO sign‑off**.

## 10. Register Findings in GitHub

# shortcut using GitHub CLI  
gh issue create -t "[Bug] F-02 Fail – Score calc" -b "Steps to reproduce: ..." -l bug

Tag each issue: bug / enhancement / security and link commit / test ID.

**Next steps:**

* Add more unit cases around edge validations.
* Configure CI (GitHub Actions) to run php artisan test + Dusk container.

## 11. Test Summary

|CaseID| Description | Result | Notes |

|---------|---------------------------------------------------------|------------|-------------------------------------------|

| F‑01 | Guest registration + quiz list | ✅ Pass | |

| F‑02 | Guest completes quiz and sees result | ✅ Pass | |

| F‑03 | Admin login with valid credentials | ✅ Pass | |

| F‑04 | Admin adds question with image | ✅ Pass | Image displays via public disk |

| F‑05 | Admin edits answers for a question | ✅ Pass | |

| F‑06 | Prevent delete of first question | ✅ Pass | |

| E‑01 | Prevent empty quiz | ✅ Pass | |

| E‑02 | Invalid image upload | ✅ Pass | 422 validation |

| E‑03 | Two correct answers set | ✅ Pass | Server rejects |

| E‑04 | Mismatch answer count in results | ✅ Pass | Handles gracefully |

| E‑05 | Delete question with results exists | ✅ Pass | Soft delete or foreign key handled|

| S‑01 | CSRF protection on admin | ✅ Pass | 419 on no token |

| S‑02 | Guest blocked from admin panel | ✅ Pass | Redirects |

| S‑03 | Malicious file upload | ✅ Pass | Rejected by MIME check |

| S‑04 | SQL injection attempt | ✅ Pass | Escaped |

| S‑05 | Directory traversal prevention | ✅ Pass | Laravel storage abstraction |

|---------|---------------------------------------------------------|------------|-------------------------------------------|