# Validation document: Pass The Ball

**By**: Simeon Markov

**Institution Name**: Fontys UAS

**Course/Class**: ICT/EN08

**Date:** 2025-10-09

## Introduction

This document presents the last stage of development of the project. It includes the sections test plan (doing validation from the perspective of the user on different features mentioned in the analysis document under 'functional requirements') and explaining part of the project's testing (unit tests).

## Validation

This section presents the test plan, where is evaluated if certain features have passed the validation.

| Feature | Brief Description | Passed |
|---|---|---|
| **Registration with Email Verification** | Users can register for an account with email verification to ensure valid accounts. | Passed |
| **Autogenerate Unique Username** | System automatically generates a unique username during registration, which can be updated by the user. | Passed |
| **User Profile Page** | Each user has a dedicated profile page with multiple tabs displaying user information. | Passed |
| **Follower Count Display** | Profile page shows the number of followers the user has. | Passed |
| **Create Activities/Events** | Users can create activities with details. | Passed |
| **Edit Activities/Events** | Users can modify existing activities they have created. | Passed |
| **Delete Activities/Events** | Users can remove activities they have created from the platform. | Passed |
| **RSVP System with Waitlist** | Users can RSVP to events within groups with a waitlist feature for admin approval. | Passed |
| **Follow Other Users** | Users can follow other users to stay updated on their activities. | Passed |
| **Join Groups/Clubs** | Users can join existing groups or clubs based on interests or sports. | Passed |
| **Personalized Activity Feed** | Users see a customized feed showing events and posts from friends and groups they follow. | Passed |
| **Group Messaging** | Users can participate in group conversations with multiple participants. | Passed |
| **Commenting on Events/Posts** | Users can add comments to events and posts shared by others. | Passed |
| **Liking Events/Posts** | Users can like events and posts to show appreciation or interest. | Passed |

| | | |
|---|---|---|
| **Push Notifications** | Users receive real-time push notifications for invites, reminders, messages, and feed updates. | Passed |
| **Email Notifications** | Users receive email notifications for important events, invites, and updates. | Passed |
| **Upload Photos/Videos** | Users can upload photos and videos to events and their personal feeds. | Passed |

*Table 1: Features table validation*

## Unit tests

***Overview***

- **Framework:** Pest PHP wraps PHPUnit; assertions and helpers follow Pest syntax inside the tests directory.
- **Location:** Targeted unit specs live in tests/Unit; shared factories and helpers stay in tests/Pest.php.
- **Execution:** Command php artisan test --testsuite=Unit (or vendor\bin\pest --group=unit on Windows) to execute only unit coverage.
- **Purpose:** Each unit test isolates a single class or pure function, mocking external services and database calls to keep feedback fast.
- **Conventions:** Files are named with the subject under test (e.g., UserServiceTest.php).

**Example – ImageOptimizationServiceTest:**

The suite boots Storage::fake('public'), uploads a synthetic JPEG, and pushes it through processPhoto. Assertions confirm the response exposes paths for each generated size, metadata, and basic dimensions, then verifies the fake disk actually holds those files. Follow-up expectations run deletePhoto against the returned paths to prove every variant is removed, demonstrating how the service keeps storage tidy without touching the real filesystem.

```php
// Arrange
Storage::fake('public');
$file = UploadedFile::fake()->image('test.jpg', 3840, 2160);
$path = 'photos/test';

// Act
$result = $this->service->processPhoto($file, $path);

// Assert generated variants and metadata
$this->assertArrayHasKey('thumbnail_path', $result);
$this->assertArrayHasKey('medium_path', $result);
$this->assertArrayHasKey('large_path', $result);
$this->assertArrayHasKey('original_path', $result);
$this->assertArrayHasKey('metadata', $result);
$this->assertTrue(Storage::disk('public')->exists($result['thumbnail_path']));
$this->assertTrue(Storage::disk('public')->exists($result['original_path']));

// Cleanup: all variants removed
$this->service->deletePhoto([
    $result['thumbnail_path'],
    $result['medium_path'],
    $result['large_path'],
    $result['original_path'],
]);

$this->assertFalse(Storage::disk('public')->exists($result['thumbnail_path']));
```

*Table 2: Snapshot of the ImageOptimizationServiceTest*

**Validation of non-functional requirements**

**Performance Requirements**

**Requirement**

- **Page Load Time:** < 3 seconds
- **Group Search:** < 2 seconds

**Current State Analysis**

⚠ **PARTIALLY MET**

**Already Implemented (☑):**

- ☑ **MariaDB Database**: Production-grade RDBMS with multi-connection support
- ☑ **Image Optimization**: ImageOptimizationService generates multiple sizes (thumbnail 300x300, medium 800x600, large 1920x1080) with 85% JPEG compression
- ☑ **Lazy Loading**: AttachmentPreview component has enableLazyLoad prop for post attachments
- ☑ **Infinite Scroll**: Paginated posts API reduces initial load
- ☑ **Eager Loading**: Controllers use with() to prevent N+1 queries
- ☑ **Queue System**: Heavy tasks (image processing, notifications) run asynchronously

**Missing for < 3s Load Time:**

- ❌ **No APM**: No performance monitoring (Laravel Telescope, New Relic, DataDog)
- ❌ **No CDN**: Static assets served directly from Laravel
- ❌ **Cache Driver**: Using database cache (should be Redis for production)
- ❌ **No Database Indexes**: No documented indexing strategy for high-traffic columns
- ❌ **No Performance Budgets**: Build size not tracked

**Recommended actions:**

- Installing Laravel Telescope for query monitoring
- Switching to Redis for cache/sessions
- Adding indexes on posts.user_id, posts.created_at, comments.post_id
- Configuring CDN (CloudFlare/AWS CloudFront)
- Implementing performance budgets in Vite config

**2. Scalability Requirements**

**Requirement**

- Handle rapid increase in users (up to 10k) without performance degradation

**Current State Analysis**

⚠ **PARTIALLY MET**

**Already Implemented (☑):**

- ☑ **MariaDB**: Can handle 10k+ concurrent users with proper tuning
- ☑ **Queue System**: QUEUE_CONNECTION=database offloads heavy tasks
- ☑ **Stateless Architecture**: Inertia.js enables horizontal scaling
- ☑ **Pagination**: Infinite scroll prevents large dataset issues

**Missing for 10k Users:**

- ❌ **Session Storage**: SESSION_DRIVER=database (not distributed-system friendly)
- ❌ **Cache Storage**: CACHE_STORE=database (not distributed)
- ❌ **Broadcasting**: BROADCAST_CONNECTION=log (not production-ready). Could handle up to 200000 connections per day.
- ❌ **No Rate Limiting**: Only 3 throttle rules (auth routes only)
- ❌ **No Load Balancing**: Single-server deployment
- ❌ **No Load Testing**: Performance under 10k users not verified

**Recommended actions:**

- Migrate to Redis for sessions, cache, and queues
- Configure Pusher or Laravel WebSockets for broadcasting (for production ready)
- Add rate limiting: 100 req/min per user on API endpoints
- Set up load balancer (Nginx/AWS ALB)
- Perform load testing with Apache JMeter or k6
- Implement Laravel Horizon for queue monitoring

---

**3. Usability Requirements**

**Requirement**

- Intuitive interface
- New user can find and join a group within 5 minutes
- Must be accessible

**Current State Analysis**

⚠ **PARTIALLY MET**

**Already Implemented (☑):**

- ☑ **Accessible UI Components**: Reka UI (Radix Vue) provides WCAG-compliant primitives
- ☑ **Responsive Design**: Mobile-friendly with sidebar collapse at 768px
- ☑ **Type-Safe Navigation**: TypeScript + Wayfinder prevents routing errors
- ☑ **Groups Feature**: /groups page with discovery and join functionality
- ☑ **Flash Messages**: Clear user feedback system

**Missing for 5-Min Onboarding:**

- ❌ **No Accessibility Audit**: WCAG 2.1 compliance not certified
- ❌ **No Global Search**: Only user search exists (/api/users/search)
- ❌ **Missing ARIA Labels**: Icon-only buttons lack screen reader support
- ❌ **No Keyboard Shortcuts**: No documented keyboard navigation

**Recommended actions:**

- Conduct WCAG 2.1 AA audit with axe DevTools
- Add onboarding tour (Shepherd.js or Intro.js)
- Implement global search (posts, groups, users) with Meilisearch
- Add aria-label to all icon buttons
- Test with NVDA/JAWS screen readers

---

**4. Availability Requirements**

**Requirement**

- 99.9% uptime (8.76 hours downtime/year)

**Current State Analysis**

❌ **NOT MET - No High Availability Infrastructure**

**Already Implemented (☑):**

- ☑ **Health Check Route**: /up endpoint exists

**Missing for 99.9% Uptime:**

- ❌ **No Uptime Monitoring**: No PagerDuty/UptimeRobot configured
- ❌ **No Database Replication**: Single point of failure
- ❌ **No Automated Backups**: No disaster recovery strategy
- ❌ **No Failover**: No redundant servers
- ❌ **No Load Balancer**: Single-server deployment

**Recommended actions:**

- Set up uptime monitoring (UptimeRobot + PagerDuty)
- Configure MariaDB master-slave replication
- Implement automated daily backups to S3
- Use blue-green deployments (Laravel Envoyer)
- Document disaster recovery procedures

---

**5. Security Requirements**

**Requirement**

- User data must be encrypted
- Protection against common vulnerabilities

**Current State Analysis**

⚠ **PARTIALLY MET**

**Already Implemented (☑):**

- ☑ **2FA Authentication**: Laravel Fortify with two-factor support
- ☑ **Password Security**: Bcrypt with 12 rounds
- ☑ **Input Sanitization**: HTML Purifier prevents XSS
- ☑ **CSRF Protection**: Automatic with Laravel/Inertia

- ☑ **Rate Limiting**: 6 attempts/min on login
- ☑ **HTTP-Only Cookies**: Session cookies protected
- ☑ **SQL Injection Prevention**: Eloquent ORM with prepared statements
- ☑ **AES-256 Encryption**: Strong cipher for sensitive data

**Missing for Production:**

- ❌ **HTTPS Not Enforced**: SESSION_SECURE_COOKIE not enabled
- ❌ **No CSP Headers**: Content Security Policy not configured
- ❌ **No Security Headers**: Missing X-Frame-Options, HSTS
- ❌ **No Dependency Scanning**: No automated vulnerability checks
- ❌ **Secrets in .env**: API keys not in secure vault

**Recommended actions:**

- Enable SESSION_SECURE_COOKIE=true for HTTPS-only cookies
- Add spatie/laravel-csp package
- Implement security headers middleware
- Enable GitHub Dependabot for dependency updates
- Move secrets to AWS Secrets Manager or HashiCorp Vault

---

**6. Cross-Platform Compatibility**

**Requirement**

- Consistent experience on Web and Android

**Current State Analysis**

❌ **NOT MET - Web Only**

**Already Implemented (☑):**

- ☑ **Responsive Design**: Tailwind CSS with mobile breakpoints
- ☑ **Mobile UI**: Sidebar adapts to small screens

**Missing for Android:**

- ❌ **No PWA**: No Progressive Web App capabilities

- ❌ **No Native App**: No Android application
- ❌ **No Service Worker**: No offline support
- ❌ **No App Manifest**: No install prompt

**Recommended actions:**

- **Short-term (2-4 weeks)**: Implement PWA with manifest.json and service worker
- **Long-term (3-6 months)**: Build Flutter app with Laravel API backend

---

**Summary Table**

| Requirement | Status | Priority | Effort to be fixed |
|---|---|---|---|
| **Performance (< 3s load)** | ⚠ Partial | High | 3-4 weeks |
| **Scalability (10k users)** | ⚠ Partial | Critical | 4-6 weeks |
| **Usability (5min group join)** | ⚠ Partial | Medium | 2-3 weeks |
| **Availability (99.9% uptime)** | ❌ Not Met | Critical | 6-8 weeks |
| **Security (encryption)** | ⚠ Partial | High | 2-3 weeks |
| **Cross-platform (Android)** | ❌ Not Met | Medium | 2-4 weeks (PWA) |

*Table 3: Summary table of met non-functional requirements*

**References/Sources**


- AI Transparency: Perplexity AI for extensive research, summary & Claude AI as code helper (used for generating partial code of the overall project with set instructions).
- YouTube video, inspired by: https://youtu.be/4iiEyOKhvao?si=vQARG5ZIs5uFc_jl .