Analysis: Pass The Ball

By: Simeon Markov

Institution: Fontys UAS

Course/Class: ICT/EN08

Date: 2025-10-13

Introduction

The goal of this document is to present the user requirements, mentioning possible user stories, going through functional, non-functional and technical requirements. It also contains the MoSCoW requirements prioritization method, used to ensure that core requirements are met, while less important ones (secondary) could be later realized (postponed).

Targeted users

Primary users

- *The Casual Enthusiast:* Wants to find running partners, or casual cycling groups.
- *The Amateur Competitor:* Seeks regular training partners and league play. Needs skill-matching and reliable scheduling.
- *The Social Connector:* Joins for community and friendship. Needs robust social features, group chats, and event feeds.

Secondary users

- *Sports Coaches & Trainers:* Use the platform to find clients, organize sessions, and promote their services.
- *Club administrators:* Need a platform for managing teams, schedules.

Wants and needs

User Type	Wants	Needs
Casual enthusiast	"Find a soccer game tonight."	Reduce the tension/anxiety of joining a new group;
Amateur Competitor	"Find players at my skill level."	Reliable, committed partners/players;
Social Connector	"Meet new people who love hiking."	A sense of belonging and a friendly environment.
Coach/Trainer	"Get more clients."	A professional profile, credibility
Club administrator	“Manage clubs easily”	Automation of repetitive tasks (scheduling, reminders) and clear communication channels.

Table 1: User stories

Functional requirements

User management

- *Registration with email verification.*
- *Autogenerate unique username during registration, which should be possible to be updated.*
- *Each user will have its own profile page with several tabs.*

- *The page should show how many followers the user has. Registration with email verification.*
- *Autogenerate unique username during registration, which should be possible to be updated.*
- *Each user will have its own profile page with several tabs.*
- *The page should show how many followers the user has.*

Activity/Event

- *Create, edit, and delete activities (sport type, time, location, max participants, skill level).*
- *RSVP system with waitlist.1*
- *In-app calendar sync.*
- *Location-based discovery using maps.*

Social networking

- *Follow other users, join groups/clubs.*
- *A personalized activity feed showing events from friends/groups.*
- *Direct and group messaging.*
- *Commenting and liking on events and posts.*

Notifications

- *Push and email notifications for invites, reminders, messages, and feed updates.*

Content sharing

- *Upload photos/videos to events and personal feeds.*

Non-functional requirements

Performance

- The app must load pages in < 3 seconds. Search for nearby events should return results in < 2 seconds.

Scalability

- The architecture must handle a rapid increase in users (up to 10k) without performance degradation.

Usability

- Intuitive interface. A new user should be able to find and join an event within 5 minutes of opening the app. Must be accessible.

Availability

- 99.9% uptime, ensuring reliability during peak hours.

Security

- User data must be encrypted. Protection against common vulnerabilities.

Cross-platform compatibility

- Consistent experience on Web and Android.

Technical requirements

Frontend (Designing & Tools)

- *Mobile:* Laravel, Vue.js.
- *Web:* Laravel, Vue.js, Inertia, JavaScript, HTML, TailwindCSS.

Backend (Software & Infrastructure)

- *Languages:* PHP (Laravel) & JavaScript, TypeScript.
- *Database:* MariaDB for relational data.
- *Search Database:* Elasticsearch for fast, complex searches.

Reasons for choosing the techstack

Frontend Architecture

- Leveraging Vue.js reusable components alongside Laravel packages helps in boosting the frontend, ensuring clean architecture.
- Laravel's reactive data binding system allows for real-time updates, useful for user feeds, notification counters, and social interactions without requiring full page reloads.

- Vue.js has proven particularly effective for social media applications, supporting features such as dynamic feeds, user profiles, and real-time chat functionality.
- Inertia.js serves as a bridge between Laravel's server-side capabilities and Vue.js frontend framework, eliminating the need for a separate API while maintaining SPA-like functionality. The approach benefits from simplified data flow, enhanced performance.

Backend Infrastructure

- PHP offers several advantages for social network development, including excellent scalability, fast loading times, and strong database connectivity. The language's ability to handle high-traffic websites makes it particularly suitable for social platforms that may experience rapid user growth.
- Laravel's design patterns, including Factory, Observer, and Strategy patterns, provide a solid architectural foundation for complex social network features.
- The combination of JavaScript and TypeScript in the techstack provides flexibility for both client-side interactivity and server-side operations.

Database Architecture

- MariaDB was chosen due to its superior performance compared to MySQL, particularly in scenarios requiring high concurrency and complex queries.

Feature	*MySQL*	*MariaDB*
Real-time analytics	Limited	Limited
Scalability	Moderate (manual tuning needed)	Moderate
Data type support	Primarily structured	Structured + semi-structured
Vector search	No	No
Performance	Good for small datasets	Good for small to medium datasets

Table 2: Comparison between MariaDB & MySQL

Elasticsearch for Advanced Search

- *Real-time Search:* Instantaneous search results across user profiles, posts, and content.
- *Social Search Features:* Advanced filtering and ranking based on user connections and social graphs.
- *Scalable Architecture:* Distributed processing for handling large-scale social media data.
- *Complex Query Support:* Boolean queries, aggregations, and relevance scoring for sophisticated social search features.

Prioritization

For the requirements prioritization, the MoSCoW technique was used.

MoSCoW technique

Must have	Registration with email verification, Autogenerating unique name, each user could post tweets, follow other users, upload profile photo, cover and comment on posts.
Should have	Creating/updating/deleting groups, users should be able to join groups, receive invitations, admins of groups should have the option to accept or reject a user.
Could have	Group admin could remove users, where the user receives a notification, post page that could be shared with others.
Will/Wish have	Generate posts with generative AI, implementing global search, adding hashtags, searching by hashtags, dark mode visibility.

Table 3: MoSCoW requirements prioritization

References

- BM Coder. (2025, August 22). *Building a social networking platform with Laravel*. **https://www.bmcoder.com/blog/building-a-social-networking-platform-with-laravel**
- WitQuick. (n.d.). *Pros and cons of the VueJS*. **https://witquick.hashnode.dev/pros-and-cons-of-the-vuejs**
- Pattem Digital. (n.d.). *Vue JS Application: Benefits & Industry Use Cases*. **https://pattemdigital.com/insight/vue-js-applications-and-case-study/**
- SingleStore. (2024, September 18). MariaDB vs. MySQL. **https://www.singlestore.com/blog/mariadb-vs-mysql/**
- Apiumhub. (n.d.). Elastic Search; advantages, case studies & books. **https://apiumhub.com/tech-blog-barcelona/elastic-search-advantages-books/**
- AI transperancy: PerplexityAI for extensive in-depth research, summarizing articles, papers.