Analysis – Project 2

Movey – Sports Community app

By Rareș Nagy

Table of Contents

Introduction

In the years following the COVID-19 pandemic, global studies have shown a noticeable decline in physical activity across all age groups. University students, in particular, experienced prolonged periods of isolation, remote learning, and disrupted routines that made regular sports participation difficult. Although restrictions have lifted and facilities have reopened, sports engagement has not fully recovered.

One of the most significant barriers to getting active again is not a lack of motivation or access—but a lack of social connection. Many students report wanting to exercise or play sports but struggle to find peers who share their interests or are available at the same time. Traditional sports clubs and university teams often require long-term commitment or cater only to advanced players, leaving casual athletes without an easy way to connect.

This analysis explores the development of a sports-focused social application designed to bridge that gap. By making it simple for users to discover nearby activities, form teams spontaneously, or invite others to join informal sessions, the app aims to remove the social friction that prevents people from staying active. In doing so, it supports both physical well-being and community building in the post-pandemic university environment.

Objective of the project

The goal is to create a functional web application within 4 weeks that:

- Allows users to connect through a social app
- Incentivizes users to create and join different sport events
- Gets users over the fear of attending a sports activity alone

These functions are based on specific requirements that the web application must meet (see
next chapter).'

Requirements

What falls within scope:

- A database to store both user and event data
- A frontend to interact with
- A backend, to connect the database to the frontend

What does not fall within scope:

- **Any sort of messaging functionality**
- **Comments on events**

MoSCoW analysis:

- The App *must*:
  - Have a **secure** user login feature
  - Allow users to create sport events
  - Allow users to join sport events
- The App Should
  - Allow users to share sport events
  - Allow users to view other user's profiles
- The App could
  - Allow users to view sports events they have joined
  - Allow users
- The app won't
  - Have an event search feature
  - Track the user's sports activities (through smart watches, pedometers etc.)

High level overview

The web app will have a feed of sport events that users can express their desire of attending. This is so they can live a healthy, active life without the fear of being alone on a hike for example.

Competitors

The main competitor to the sports social media market is Strava, an app

where athletes can keep track of their activities using fun adventures and new

routes. Others include:

- Garmin Connect – Deep integration with Garmin devices and robust performance analytics out of the box.

- MapMyRun / MapMyRide – Simple and accessible with a large route library and community challenges
- TrainingPeaks – Favored by serious athletes and coaches for structured training plans and detailed load tracking.
- Komoot – Exceptional for discovering and planning scenic cycling, hiking, and gravel routes with turn-by-turn navigation.
- Nike Run Club – Motivational audio coaching and gamified experience aimed at casual and intermediate runners.

Tech stack

Frontend

The market of Frontend JS frameworks is vast, it feels like there's a new one every day. Since my project will need to integrate the frontend with the backend, the choice of what JavaScript framework will be used (if any) is an important one. I used this video by Fireship and my own personal experience to create a decision matrix and find out the answer.

Criteria/Framework	Familiarity	Ease of use	Declarative	Overkill	State management	Popularity	Total
Vanilla	7	3	2	0	2	9	238
React	3	2	6	3	8	10	218
Svelte	8	9	10	2	10	5	380
Vue	7	8	10	2	9	7	365
Importance Multiplier	8	9	10	10	4	3	N/A

I chose Svelte, not just because it scored the best in the table, but also because I had a bias in choosing it. I just like using it.

Backend

Because of the applications relatively small scope, the backend choice is not that impactful. Speed is not a huge worry either, so it doesn't need to be fast necessarily. I chose the **Go** programming language, as I've used it to build backends before, and it has great support for REST APIs. Its great performance is more of a bonus than an advantage for this project.