

TIPPR

Projektsummering & Teknisk Specifikation

Version 1.0 | 2025-12-14

Innehållsförteckning

1. Executive Summary
2. Projektöversikt
3. Målgrupp & Syfte
4. Funktionella Krav
5. Användarflöde
6. Poängsystem & Spelregler
7. Teknisk Arkitektur & Stack
8. Skalbarhet & Infrastructure
9. Deployment & Drift
10. Roadmap & Framtida Funktioner

1. Executive Summary

Tippr är en modern webbapplikation för sporttippning som revolutionerar hur vänner, familjer och kollegor tävlar tillsammans under stora sportevenemang. Första versionen lanseras inför Fotbolls-VM 2026 och fokuserar på mästerskapstippning med möjlighet att skapa privata ligor, anpassade spelregler och realtidsuppdateringar av resultat och placeringar.

Projektet drivs av behovet att ersätta komplexa Excel-ark med en användarvänlig, skalbar och automatiserad lösning som hanterar allt från resultatinmatning till poängberäkning och notifikationer.

2. Projektöversikt

2.1 Vad är Tippr?

Tippr är en tippningsplattform för sportevenemang med följande egenskaper:

- **Version 1.0:** Fotbolls-VM 2026 (mästerskapstippning)
- **Version 2.0:** Ligaspel (Premier League, La Liga, etc.) och fler mästerskap (Champions League, Europa League)
- **Version 3.0+:** Fler sporter (ishockey, handboll, etc.)

2.2 Varför Tippr?

I dagsläget saknas en dedikerad plattform för sporttippning med vänner över flera evenemang. Befintliga lösningar är ofta:

- Excel-baserade med mycket manuellt administrativt arbete
- Tidskrävande för både arrangörer och deltagare
- Saknar realtidsuppdateringar och moderna funktioner
- Inte byggda för community-funktioner som chat och notifikationer

3. Målgrupp & Syfte

3.1 Målgrupp

Tippr riktar sig till:

- **Primär:** Fotbollsentusiaster som tippar med vänner och familj
- **Sekundär:** Arbetslag och kollegor som vill ha en rolig aktivitet under stora evenemang
- **Långsiktig:** Sportfantaster generellt (alla sporter)

3.2 Utvecklingssyfte

Detta projekt är inte bara en applikation, utan även ett lärandemål. Som systemutvecklare med inriktning .NET är målet att:

- Bygga ett komplett enterprise-system från grunden
- Lära sig modern molninfrastruktur och skalbarhet
- Implementera best practices för arkitektur och design patterns
- Utveckla en applikation som kan hantera tusentals samtidiga användare
- Få erfarenhet av hela utvecklingsprocessen från idé till lansering

4. Funktionella Krav

4.1 Kärnfunktionalitet

4.1.1 Användarhantering

- Registrering och inloggning med säker autentisering
- Användarprofilhantering
- Social login (Google, Facebook - framtida)

4.1.2 Liga/Grupphantering

- Skapa privata ligor med anpassade inställningar
- Gå med i flera ligor samtidigt
- Global liga där alla användare kan tävla
- Bjud in medlemmar via länk eller e-post

4.1.3 Tippingssystem

- Tippa alla matcher i mästerskapet
- Tippa bonusfrågor (skyttekung, vinnare, etc.)
- Anpassningsbara deadlines per liga
- Tre tippningslägen:
 - **Allt-i-ett:** Alla matcher måste tippas före första matchen
 - **Etappvis:** Tipping per turnering-etapp (grupp, slutspel, etc.)
 - **Match-för-match:** Individuella deadlines per match

4.1.4 Realtidsuppdateringar

- Live matchresultat via API-FOOTBALL
- Automatisk poängberäkning efter varje match
- Live-uppdatering av ligatabeller
- Realtidsnotifikationer för placeringsändringar

4.1.5 Community-funktioner

- Chat-funktion i varje liga
- Notifikationssystem för:
 - Kommande deadlines
 - Matchresultat och poängfördelning
 - Placeringsändringar i ligatabellen
 - Nya meddelanden i chatten

5. Användarflöde

5.1 Grundläggande Flow

Steg	Beskrivning
1	Användare registrerar sig & loggar in
2	Skapar eller går med i en liga/grupp
3	Bjuder in vänner till ligan
4	Tippar alla matcher enligt ligans regler
5	Tippar bonusfrågor (skyttekung, vinnare, etc.)
6	Matcher spelas och resultat uppdateras live
7	Användare ser sin placering uppdateras i realtid
8	Evenemanget avslutas och vinnaren koras

6. Poängsystem & Spelregler

6.1 Standard Poängsystem (Global Liga)

Baserat på klassiska Excel-tippningar med möjlighet för administratörer att anpassa:

Kategori	Poäng	Beskrivning
Rätt resultat (ett lag)	2p	Korrekt antal mål för ett av lagen
Rätt utkomst	3p	Rätt vinnare/oavgjort
Rätt resultat (båda lagen)	7p	Exakt korrekt slutresultat
Lag i åttondel	2p	Per korrekt lag som går vidare
Lag i kvartsfinal	4p	Per korrekt lag som går vidare
Lag i semifinal	6p	Per korrekt lag som går vidare
Lag i final	8p	Per korrekt lag som går vidare
Skyttekung	20p	Rätt skyttekung
Mästare	20p	Rätt vinnare av turneringen
Flest mål (grupp)	10p	Lag med flest mål i grupp-spelet
Flest insläppta (grupp)	10p	Lag med flest insläppta i grupp-spelet

6.2 Anpassningsbara Regler för Privata Ligor

Ligaadministratörer kan konfigurera:

- **Poängvärden:** Anpassa alla poängvärden (t.ex. skyttekung kan ge 30p istället för 20p)
- **Tippningsläge:** Välj mellan allt-i-ett, etappvis eller match-för-match
- **Deadline:** Anpassa tid innan match (t.ex. 3 timmar, 1 timme, 30 min)
- **Bonusfrågor:** Aktivera/avaktivera olika bonusfrågor

7. Teknisk Arkitektur & Stack

7.1 Rekommenderad Tech Stack

Komponent	Teknologi	Motivering
Backend	.NET 8 + C#	Modern, performant och starkt typat. Utmärkt för enterprise-applikationer.
Arkitektur	Clean Architecture + CQRS (Med Simple)	Separation of concerns, testbarhet och skalbarhet. CQRS för komplex business logic.
Frontend	React + TypeScript	Stort ekosystem, komponentbaserat och starkt typat. Bättre för komplex UI.
UI Framework	Tailwind CSS + shadcn/ui	Modern styling, responsiv design och färdiga komponenter.
Database	Supabase (PostgreSQL)	Gratis tier med 500MB databas, inbyggd auth, realtime och storage. Perfekt för MVP.
ORM	Entity Framework Core	Integrerat med .NET, migrations och code-first approach.
Caching	In-Memory / Upstash Redis	Start med .NET IMemoryCache, migrera till Upstash Redis (gratis tier) vid behov.
Realtime	Supabase Realtime	Inbyggt i Supabase för live-uppdateringar. SignalR för chat-funktioner.
API Data	API-FOOTBALL	Gratis tier: 100 requests/dag. Perfekt för utveckling och MVP!
Autentisering	Supabase Auth + JWT	Inbyggd auth med social logins, JWT tokens och refresh tokens. Helt gratis.
File Storage	Supabase Storage	Gratis 1GB storage för profilbilder och uploads.
Cloud Platform	Vercel (Frontend) + Railway/Render (Backend)	Gratis (PaaS) för hobby-projekt. Vercel för React, Railway/Render för .NET.
CI/CD	GitHub Actions	Gratis för public repos, automatiserad build och deploy.
Monitoring	Sentry (gratis tier)	Error tracking och performance monitoring. 5k events/månad gratis.

7.2 Varför React istället för Blazor?

För Tippr rekommenderas React över Blazor av följande skäl:

- **Ekosystem & Community:** Enormt bibliotek av komponenter och lösningar
- **Prestanda:** Lättare att optimera för tusentals samtidiga användare
- **Mobile-first:** React Native möjliggör enkel mobil-expansion
- **Kompetens:** Lättare att hitta utvecklare och resurser
- **Realtime:** Bättre integreringar för WebSockets och live-updates

7.3 Varför Supabase?

Supabase är det perfekta valet för ett studentprojekt som Tippr:

- **Gratis Tier:** 500MB PostgreSQL databas, helt gratis!
- **Inbyggd Auth:** JWT, email/password, social logins (Google, Facebook) - ingen extra kostnad
- **Realtime:** WebSocket-baserad realtime för live-uppdateringar - gratis!
- **Storage:** 1GB gratis fillagring för profilbilder etc.
- **Auto-genererad REST API:** Sparar utvecklingstid

- **Skalbarhet:** Lätt att uppgradera till Pro (\$25/månad) när användarbas växer
- **PostgreSQL:** Full PostgreSQL-kompatibilitet, kan alltid migrera till self-hosted

Kostnadsjämförelse (per månad):

- **Supabase Gratis:** \$0 (upp till 500MB databas, 2GB bandwidth)
- **Azure SQL:** ~\$5-15 för basic tier
- **Supabase Pro:** \$25 (8GB databas, 50GB bandwidth) - när du behöver skala

7.4 Budget-Medveten Arkitektur

Som student är det viktigt att hålla kostnaderna nere. Här är den rekommenderade budget-stacken:

Service	Provider	Kostnad	Användning
Databas + Auth	Supabase	\$0/mån	500MB PostgreSQL, Auth, Realtime, Storage
Backend Hosting	Railway / Render	\$0/mån	Gratis tier för .NET apps (500h/mån)
Frontend Hosting	Vercel	\$0/mån	Unlimited React deployments, global CDN
Caching	.NET IMemoryCache	\$0/mån	Inbyggt i .NET, ingen extra kostnad
API Data	API-FOOTBALL	\$0/mån	Gratis tier: 100 requests/dag
CI/CD	GitHub Actions	\$0/mån	Gratis för public repos
Monitoring	Sentry	\$0/mån	Gratis tier: 5k events/månad
Email	Resend / Brevo	\$0/mån	Gratis tier: 300 emails/dag

Total månadskostnad: \$0 ■

7.5 Skalningsväg (när användarbas växer):

När Tippr växer och behöver mer resurser:

- **Steg 1 (0-1000 användare):** Gratis tier klarar det perfekt
- **Steg 2 (1000-10000 användare):** Uppgradera Supabase till Pro (\$25/mån)
- **Steg 3 (10000-50000 användare):** Railway/Render Pro (~\$20/mån) + Upstash Redis (~\$10/mån)
- **Steg 4 (50000+ användare):** Migrera till Azure/AWS med dedikerade resurser

7.6 Arkitekturmönster & Design Patterns

7.6.1 Clean Architecture

Projektstruktur:

- **Domain Layer:** Entities, Value Objects, Domain Events
- **Application Layer:** Use Cases, Commands/Queries (CQRS), DTOs
- **Infrastructure Layer:** EF Core, External APIs, SignalR
- **Presentation Layer:** REST API, Controllers

7.6.2 CQRS med MediatR

Separera läsning och skrivning:

- **Commands:** CreateLeague, SubmitPrediction, UpdateScore
- **Queries:** GetLeagueStandings, GetUserPredictions, GetLiveMatches
- **Fördelar:** Skalbarhet, testbarhet, tydlig business logic

7.6.3 Repository Pattern (valfritt)

Med modern EF Core är Repository Pattern inte alltid nödvändigt. Rekommendation:

- Använd EF Core DbContext direkt i Infrastructure Layer
- Lägg till Repository endast om verklig abstraktion krävs
- Fokusera istället på Unit of Work-pattern för transaktioner

8. Skalbarhet & Infrastructure

8.1 Skalbarhetsstrategi

För att hantera tusentals samtidiga användare:

8.1.1 Horisontell Skalning

- **Load Balancing:** Azure Load Balancer för fördelning av trafik
- **Stateless API:** JWT-tokens gör att vilken server som helst kan hantera requests
- **Auto-scaling:** Azure App Service auto-scaling baserat på CPU/minne

8.1.2 Caching-strategi

- **Redis Cache:** Ligatabeller, användarsessions, match-data
- **Response Caching:** API-endpoints med sällan uppdaterad data
- **CDN:** Azure CDN för statiska assets (bilder, JS, CSS)

8.1.3 Database Optimization

- **Indexering:** På foreign keys, user_id, league_id, match_id
- **Read Replicas:** Separata databaser för läsningar (Azure SQL Read Replicas)
- **Connection Pooling:** EF Core connection pooling för effektiv databasanvändning

8.1.4 Asynkron Bearbetning

- **Background Jobs:** Poängberäkningar efter matcher (Hangfire eller Quartz.NET)
- **Budget Option:** Använd .NET BackgroundService för enkla scheduled tasks
- **Batch Processing:** Uppdatering av ligatabeller i bulk

8.2 Realtidshantering

Supabase Realtime används för:

- **Live Match Updates:** Databas-triggers när matchresultat uppdateras
- **Ligatabeller:** Live-uppdatering av placeringar via Supabase subscriptions
- **Notifikationer:** Push-notiser för resultat och deadlines

SignalR används för:

- **Chat:** Realtidsmeddelanden i ligor (WebSocket via .NET backend)
- **Hybrid Approach:** Supabase Realtime för data-synk, SignalR för chat

8.3 Budget-Medveten Skalning

För att hålla kostnaderna låga under tillväxt:

- **Cache Aggressivt:** Använd .NET IMemoryCache för att minska databas-queries
- **Batch Updates:** Uppdatera ligatabeller var 5:e minut istället för realtid
- **API Rate Limiting:** Cacha API-FOOTBALL svar i databasen (1 request/match räcker)
- **Lazy Loading:** Ladda endast data som användaren aktivt tittar på
- **Database Indexing:** Optimera queries med indexes på foreign keys och vanliga filter

9. Deployment & Drift

9.1 Budget-Vänlig Infrastruktur (Gratis Tier)

Service	Användning	Gratis Tier Limits
Supabase	PostgreSQL databas, Auth, Realtime, Storage	500MB databas, 2GB bandwidth/mån, 50MB storage
Vercel	React frontend hosting med CDN	Unlimited deploys, 100GB bandwidth/mån
Railway / Render	.NET backend hosting	500 timmar/mån, 512MB RAM, sleep efter inaktivitet
GitHub Actions	CI/CD pipeline	2000 minuter/mån för privata repos (unlimited för public)
Sentry	Error tracking & monitoring	5000 events/månad, 1 projekt
Resend / Brevo	Email-utskick (verifiering, notiser)	100-300 emails/dag

9.2 Viktiga Överväganden för Gratis Tier

- **Railway/Render Sleep:** Backend somnar efter 15 min inaktivitet (tar ~30 sek att vakna)
- **Workaround:** Använd cron-job (GitHub Actions) för att pinga backend var 10:e minut
- **Databas Limit:** 500MB räcker för MVP (tusentals användare, hundratusentals tips)
- **API Requests:** 100 requests/dag till API-FOOTBALL räcker under utveckling
- **Strategi:** Cacha matchdata aggressivt, synka endast några gånger per dag

9.3 Alternativ: Azure for Students

Som student har du tillgång till **Azure for Students** med \$100 gratis kredit/år:

- Azure App Service (B1 tier): ~\$13/månad
- Azure SQL Database (Basic): ~\$5/månad
- Azure Storage: ~\$0.50/månad
- **Total:** ~\$18-20/månad (täcks av gratis kredit i 5+ månader)

Rekommendation: Starta med helt gratis tier (Supabase + Vercel + Railway). Migrera till Azure när du vill ha 'alltid-på' backend eller när gratis tier inte räcker.

9.4 CI/CD Pipeline (GitHub Actions)

Automatiserad deployment-process:

Build & Test Pipeline:

1. Trigger på push till main/develop
2. Restore dependencies & build .NET solution
3. Run unit tests & integration tests

4. Build React app (npm build)
5. Run linting & type checking (ESLint, TypeScript)

Deploy Pipeline:

1. Create Docker images för backend
2. Push till Azure Container Registry
3. Deploy till Azure App Service (staging)
4. Run smoke tests på staging
5. Deploy React app till Azure Static Web Apps
6. Manuell approval för production deployment

9.5 Monitoring & Logging (Gratis Verktyg)

- **Sentry:** Real-time error tracking och performance monitoring (5k events/mån gratis)
- **Supabase Dashboard:** Inbyggd monitoring för databas-queries och API requests
- **Vercel Analytics:** Frontend performance metrics och Web Vitals
- **Railway/Render Logs:** Built-in logging för backend
- **Structured Logging:** Serilog i .NET med console sink (gratis)

10. Roadmap & Framtida Funktioner

10.1 Fas 1: MVP - Fotbolls-VM 2026 (Q1-Q2 2026)

- Användare kan registrera sig och logga in
- Skapa/gå med i ligor med standard poängsystem
- Tippa VM-matcher och bonusfrågor
- Live-resultat från API-FOOTBALL
- Automatisk poängberäkning
- Ligatabeller med realtidsuppdateringar
- Grundläggande notifikationer

10.2 Fas 2: Community & Anpassning (Q3 2026)

- Chat-funktionalitet i ligor
- Anpassningsbara ligaregler (poäng, deadlines, lägen)
- Utökade notifikationer (placeringsändringar, chat)
- Användarstatistik och historik
- Social features (achievements, badges)

10.3 Fas 3: Expansion - Ligaspel (Q4 2026 - Q1 2027)

- Support för ligaspel (Premier League, La Liga, etc.)
- Veckovis tippning
- Säsongsbaserade ligor
- Champions League & Europa League

10.4 Fas 4: Fler Sporter & Mobil App (2027+)

- Ishockey-VM och NHL
- Handboll-VM
- React Native mobilapp
- Push notifications för mobil
- Social login (Google, Facebook)

10.5 Långsiktiga Mål

- Monetisering (premium features, annonsering)
- Partner-integrationer med sportsidor

- Machine learning för tipprekommendationer
- Global expansion med multi-språkstöd

Sammanfattning & Nästa Steg

Tippr är en ambitiös men genomförbar webbapplikation som kombinerar modern teknik med ett genuint användarbehov. Genom att använda .NET 8 för backend, React för frontend och Azure för hosting skapas en skalbar enterprise-lösning som kan växa från hundratals till tusentals användare.

Rekommenderade första steg:

1. **Projektuppsättning:** Skapa GitHub repository och grundläggande projektstruktur
2. **Database Design:** Designa ER-diagram och skapa EF Core migrations
3. **API Foundation:** Bygg autentisering, användarhantering och liga-endpoints
4. **Frontend Prototype:** Skapa wireframes och grundläggande React-komponenter
5. **API Integration:** Testa API-FOOTBALL och bygg integrationslager
6. **MVP Development:** Fokusera på kärnfunktionalitet för VM 2026
7. **Testing & Deployment:** Sätt upp CI/CD och beta-testning

Med denna tekniska grund och tydliga roadmap är Tippr redo att utvecklas från idé till verklighet. Lycka till med projektet!
