Institut Puig Castellar
Santa Coloma de Gramenet

SafeGoalStats

(Projecte de desenvolupament)

CFGM Sistemes Microinformàtics en Xarxa

Resum del projecte

SafeGoalStats és una aplicació web que hem desenvolupat en Daniel Pariente i l'Alex Rubio, dos alumnes de SMX2 de l'Institut Puig Castellar. L'aplicació està pensada per als aficionats de LaLiga, la categoria de més nivell de futbol a Espanya. La idea principal és combinar informació esportiva (classificació, estadístiques, màxims golejadors) amb mesures bàsiques de seguretat digital, perquè avui dia hi ha moltes pàgines falses que volen robar dades personals.

El projecte neix perquè hem vist que molts webs esportius no tenen mesures de seguretat adequades. Els aficionats, sense saber-ho, es registren en pàgines que no xifren les contrasenyes o que tenen formularis mal validats. Nosaltres volíem fer un projecte que fos útil i alhora ens servís per aprendre.

L'aplicació té un sistema d'autenticació (registre i inici de sessió) amb contrasenyes xifrades (bcrypt), validació de formularis, i bones pràctiques de seguretat com la protecció contra SQL injection i l'ús de JWT per a la gestió de sessions. Hem aconseguit connectar l'API externa (API-Football) per mostrar la classificació de LaLiga en temps real i les estadístiques de màxims golejadors. Aquesta va ser una de les parts més difícils del projecte, però després de moltes proves i errors, ho vam aconseguir. També hem implementat un sistema d'opinions d'usuaris, on la gent registrada pot deixar comentaris sobre equips, partits o la pròpia aplicació.

El backend l'hem desenvolupat amb Node.js i Express. El frontend amb HTML, CSS i JavaScript pur. Hem fet servir GitHub per treballar en equip i Figma per dissenyar les pantalles. La metodologia ha estat àgil (Scrum adaptat), amb sprints de dues setmanes, reunions diàries i eines de gestió com Google Sheets.

En conclusió, SafeGoalStats és un projecte que ens ha permès aplicar els coneixements de SMX en un entorn real, combinant desenvolupament web, bases de dades, ciberseguretat i integració d'APIs externes. Estem molt contents d'haver aconseguit fer funcionar l'API de classificació i golejadors.

Paraules clau (entre 4 i 8):

Estadística, Futbol, LaLiga, Resultats, Node.js, SQLite, Ciberseguretat

Abstract

SafeGoalStats is a web application developed by Daniel Pariente and Alex Rubio, two SMX2 students at Institut Puig Castellar. The application is designed for LaLiga fans, which is the highest level of football in Spain. The main idea is to combine sports information (standings, statistics, top scorers) with basic digital security measures, because nowadays there are many fake websites that try to steal personal data.

The project was born because we have seen that many sports websites do not have adequate security measures. Fans, without knowing it, register on pages that do not encrypt passwords or have poorly validated forms. We wanted to do a project that would be useful and at the same time help us learn.

The application has an authentication system (registration and login) with encrypted passwords using bcrypt, form validation, and good security practices such as SQL injection protection and JWT for session management. We have successfully connected the external API (API-Football) to display real-time LaLiga standings and top scorers statistics. This was one of the most difficult parts of the project, but after many trials and errors, we made it work. We have also implemented a user opinions system where registered users can leave comments about teams, matches or the application itself.

The backend was developed with Node.js and Express. The frontend with HTML, CSS and pure JavaScript. We used GitHub for teamwork and Figma for interface design. The methodology was agile (adapted Scrum) with two-week sprints, daily meetings, and Google Sheets for task management.

In conclusion, SafeGoalStats is a project that has allowed us to apply SMX knowledge in a real environment, combining web development, databases, cybersecurity, and external API integration. We are very proud to have made the standings and top scorers API work.

Keywords:

Statistics, Football, LaLiga, Results, Node.js, SQLite, Cybersecurity

Índex

1 Introducció....1
1.1 Context....1
1.2 Justificació....2
1.3 Objectius....3
1.3.1 Objectiu general....3
1.3.2 Objectius específics....3
1.4 Estratègia i planificació del projecte....5
1.5 Metodologia de treball....8
1.6 Estudi econòmic i pressupostari....12
2 Descripció del projecte....12
2.1 Anàlisi de requisits....12
2.1.1 Requisits funcionals....12
2.1.2 Requisits no funcionals....13
2.1.3 Previsió de tasques d'investigació....14
2.2 Tecnologies....14
2.2.1 Comparativa de les tecnologies valorades....14
2.2.2 Tecnologies escollides....15
2.3 Estructura del projecte....16
2.4 Descripció dels components....16
2.4.1 Component 1....16
2.4.2 Component 2....17
2.4.3 Component 3....17
2.4.4 Component 4....17
2.5 Definició de les tasques....18
2.5.1 Prova 1....18
2.5.2 Prova 2....18
Autenticació segura....18
2.5 Definició de les funcionalitats....19
2.5.1 Funcionalitat 1....20
Registre d'usuaris....20
2.5.2 Funcionalitat 2....20
2.5.3 Funcionalitat 3....20
2.5.4 Funcionalitat 4....20
3 Altres capítols....21
3.1 Estructura del projecte....21
3.2 Descripció dels components....21
3.3 Definició de les funcionalitats i relació amb l'arquitectura....22
4 Conclusions....23
4.1 Conclusions generals del projecte....23
4.2 Consecució dels objectius....23
4.3 Valoració de la metodologia i planificació....24
4.4 Visió de futur....24
5. Glossari....25

6. Bibliografia..... 27
7 Annexos..... 28

Llista de figures

Figura 1: Captura de pantalla de la pàgina principal de SafeGoalStats (https://192.168.239.119/safegoalstats/index.html/)

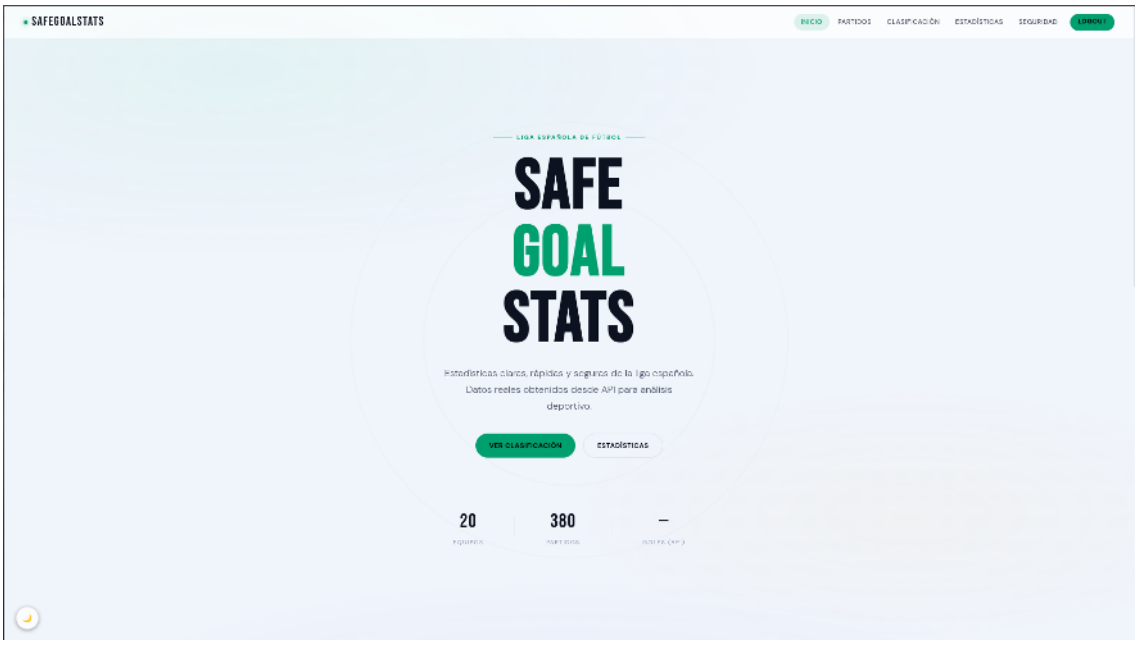


Figura 2: Modal de registre d'usuaris (finestra emergent amb JavaScript)

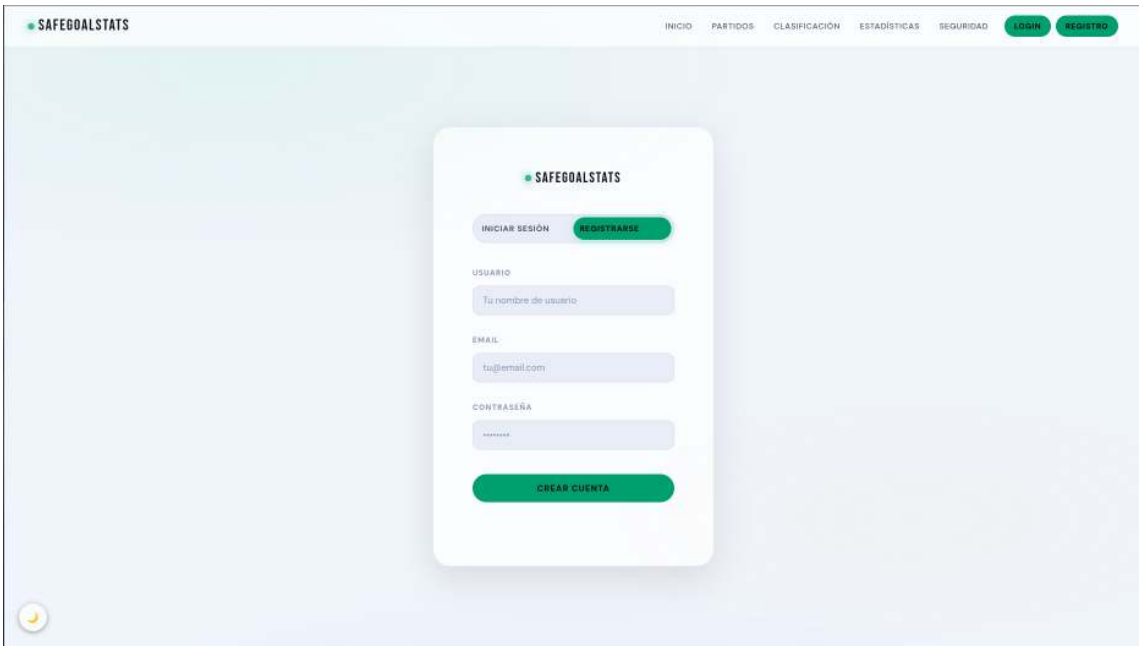


Figura 3: Modal d'inici de sessió (login) amb validació de formularis

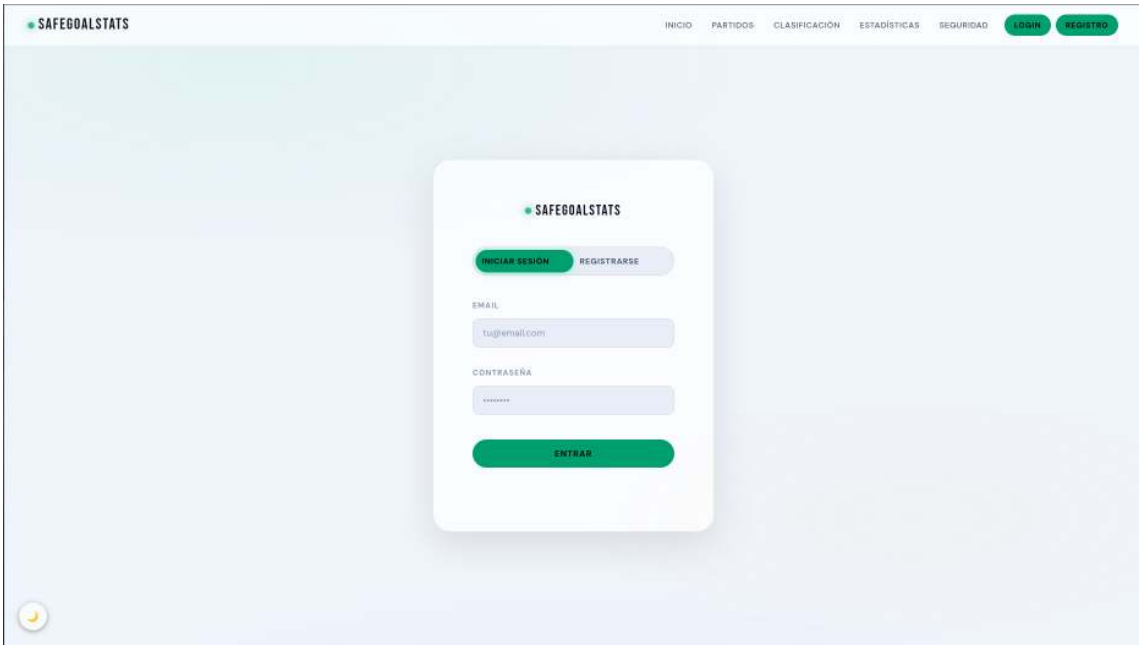


Figura 4: Menú de navegació de la pàgina (Inici, Registre, Login, Estadístiques, Seguretat)

Figura 5: Secció de Partits

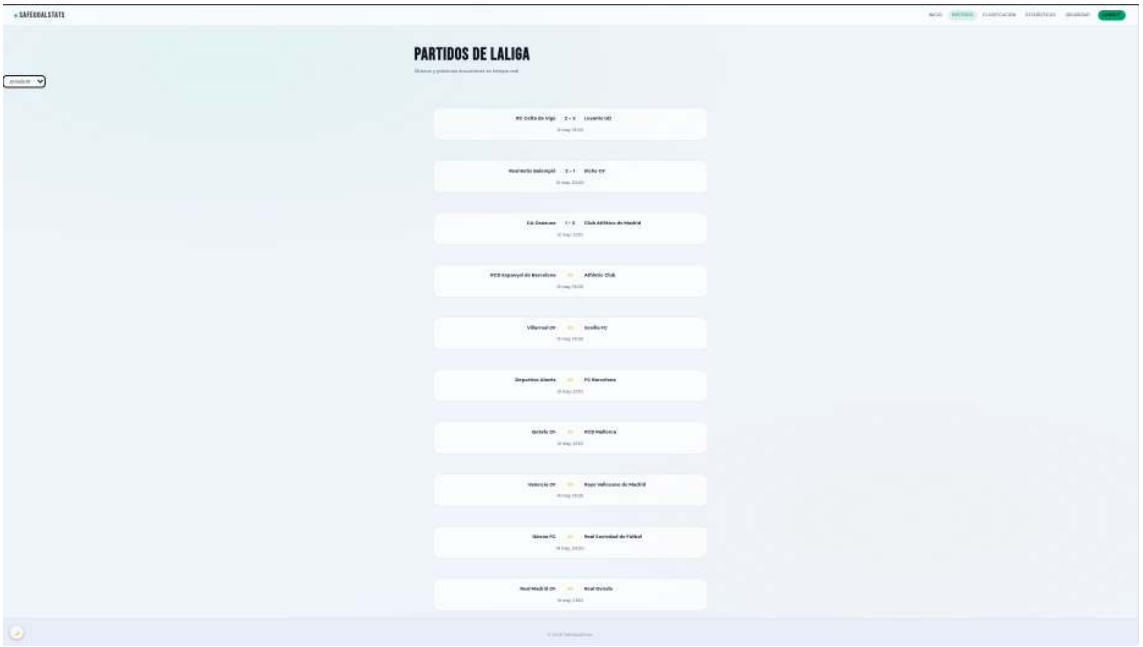


Figura 6: Secció de Clasificació

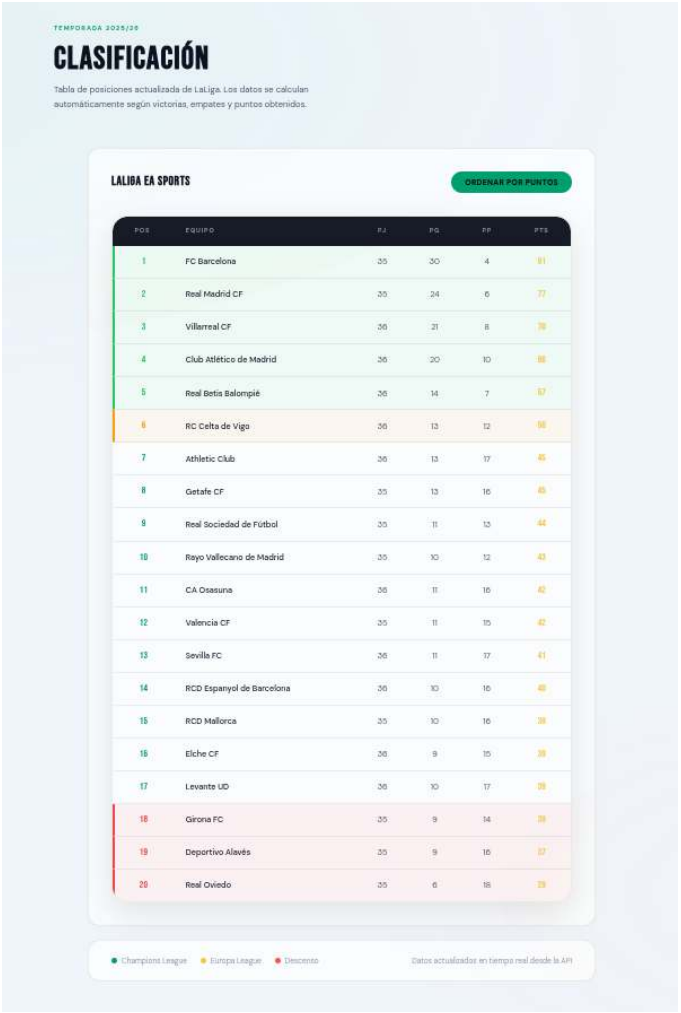


Figura 7: Secció de Estadistiques

Figura 8: Secció de Seguretat

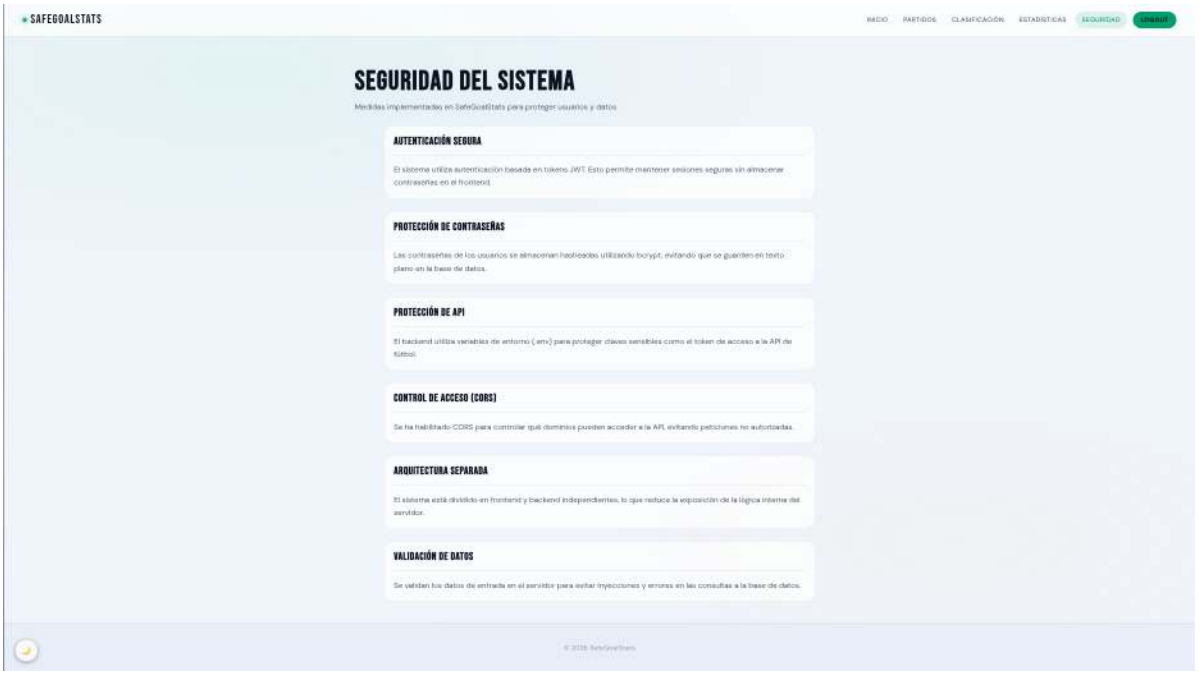


Figura 9: Secció d'Informació sobre la web

Figura 10: Secció de opinions d'usuaris

Figura 11: Captura del codi de registre

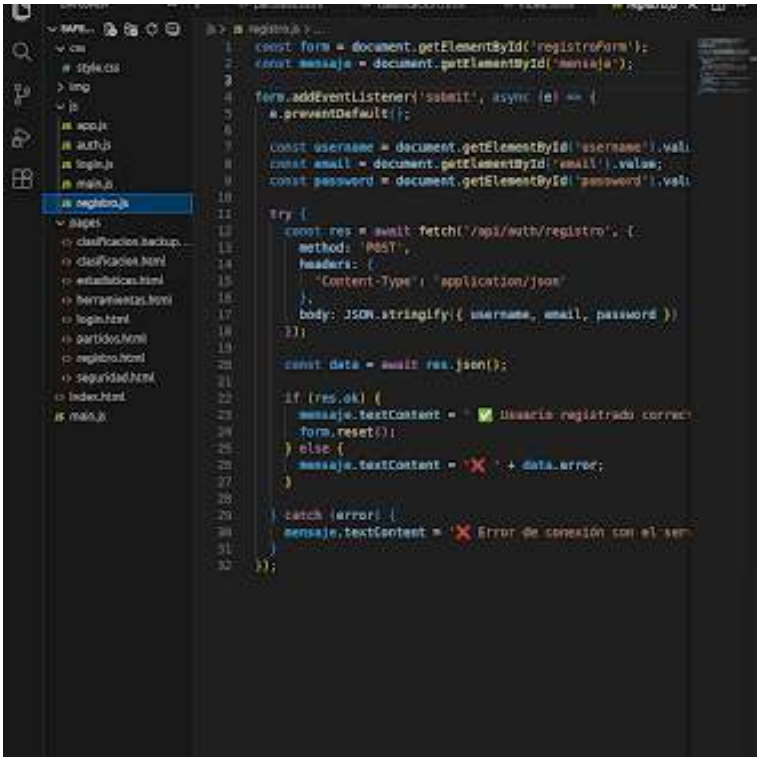


Figura 12: Captura del codi de login

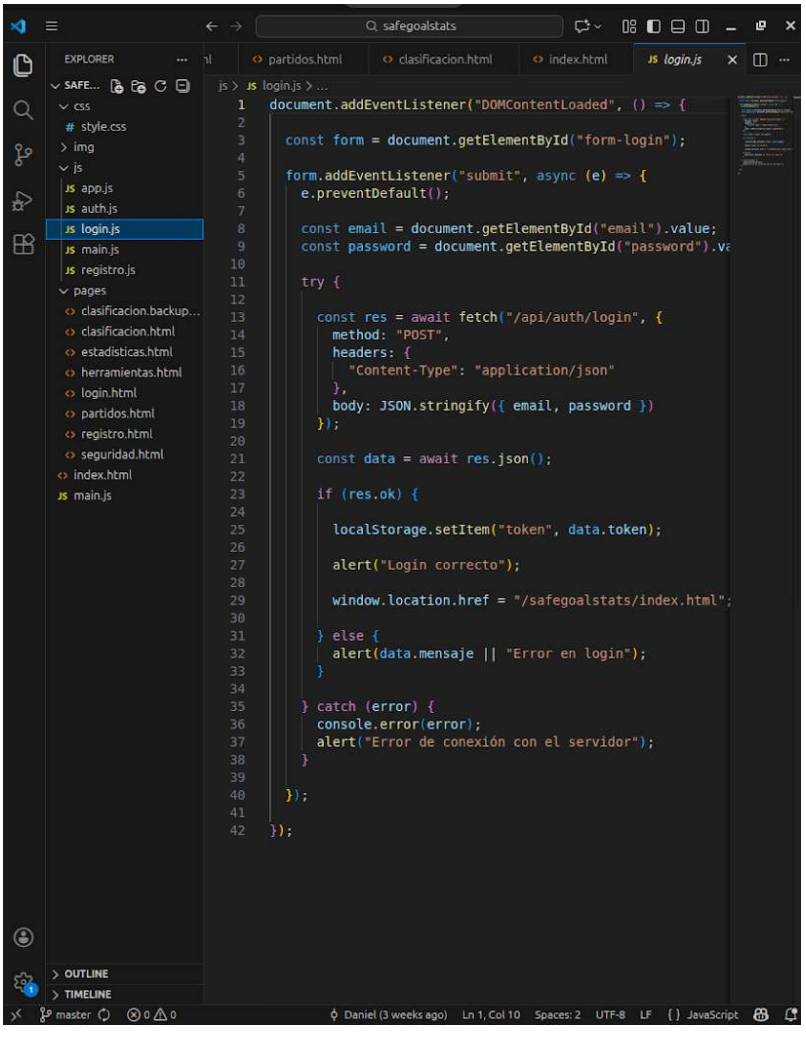
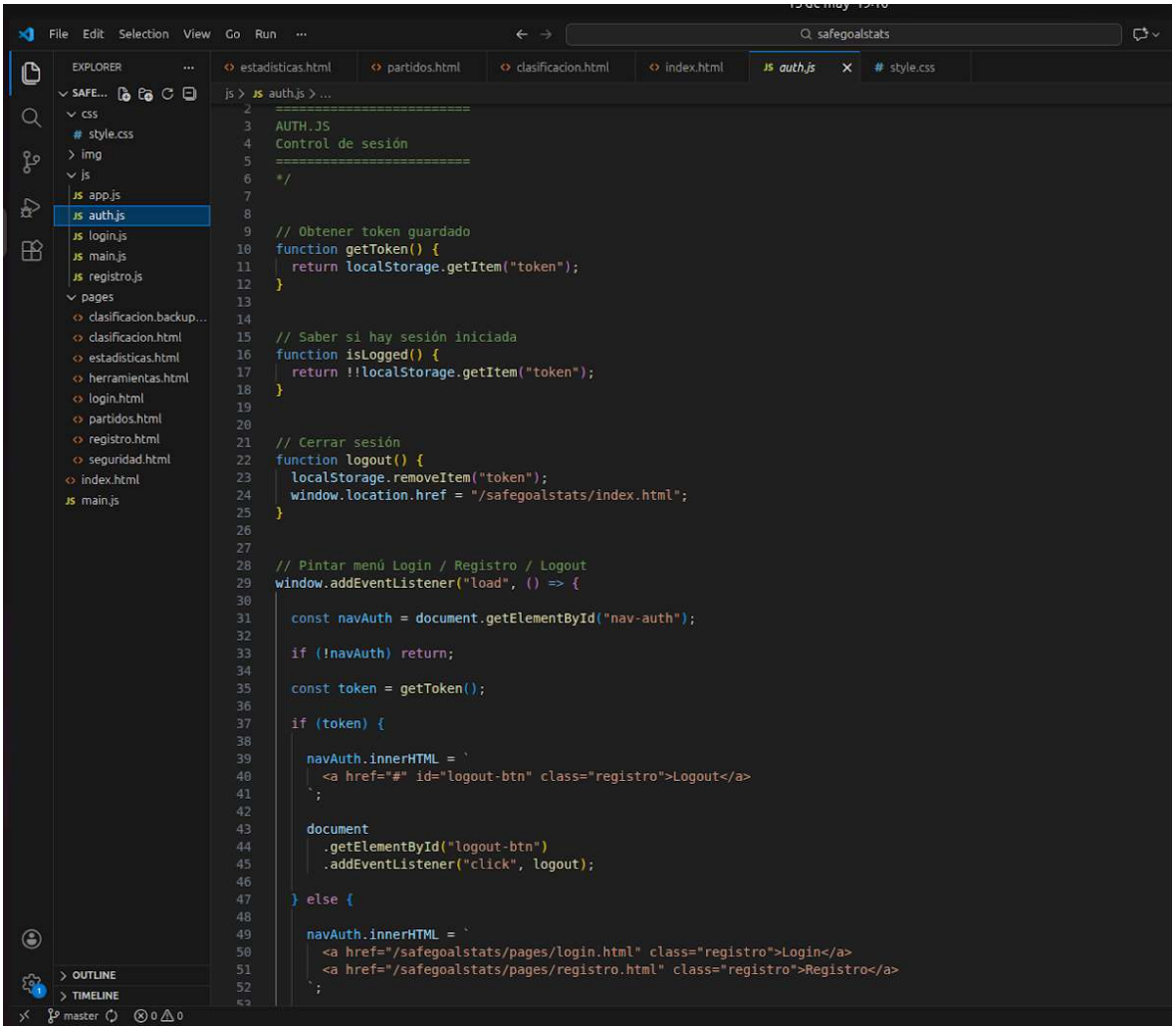


Figura 13: Captura del codi de auth



1 Introducció

SafeGoalStats és un projecte de desenvolupament web que hem fet en Daniel Pariente i l'Alex Rubio per a l'assignatura de SMX2 de l'Institut Puig Castellar. Vam pensar a fer alguna cosa relacionada amb el futbol perquè ens agrada molt als dos, i alhora volíem practicar tot el que hem après a classe: HTML, CSS, JavaScript, bases de dades, seguretat informàtica, control de versions amb GitHub, treball en equip, i integració d'APIs externes.

La idea principal és crear una web on els usuaris es puguin registrar, fer inici de sessió, veure la classificació de LaLiga en temps real, consultar els màxims golejadors, i deixar les seves opinions sobre equips, partits o la pròpia aplicació. Tot això amb un mínim de seguretat, perquè a classe també hem vist els perills de les pàgines falses i el phishing. De fet, una de les coses que més ens va impactar va ser veure com de fàcil és crear una pàgina falsa que sembli real i robar dades de la gent. Per això vam voler posar especial atenció a la seguretat.

La web actualment està allotjada a un servidor local de l'institut: https://192.168.239.119 . Des de qualsevol ordinador de la xarxa de l'institut s'hi pot accedir. Encara no està penjada a Internet perquè no tenim un domini ni un hosting de veritat, però ens serveix per fer proves i ensenyar el projecte al professor.

1.1 Context

Actualment, hi ha moltes pàgines web de resultats de futbol, però moltes no tenen mesures de seguretat adequades. Els aficionats es refien i posen les seves dades personals (correu electrònic, contrasenya, de vegades fins i tot dades bancàries) sense saber si la web és fiable. Hem vist casos de phishing on fan pàgines falses que semblen de resultats o de venda d'entrades i després roben les contrasenyes. Això és un problema greu perquè molta gent utilitza la mateixa contrasenya per a diversos serveis (correu, xarxes socials, banca en línia).

Segons dades de l'empresa de ciberseguretat Kaspersky, els intents de phishing relacionats amb esdeveniments esportius augmenten considerablement durant temporades d'alta competició. Això inclou correus falsos que suplanten la identitat de LaLiga, ofertes fraudulentes de samarretes oficials i pàgines que imiten portals de resultats en temps real per robar credencials d'accés. Nosaltres vam investigar això i ens va servir per justificar per què valia la pena fer un projecte amb mesures de seguretat.

A més, a l'institut hem après a fer pàgines web, a connectar bases de dades, a xifrar contrasenyes... i vam pensar que seria bona idea fer un projecte que unís tot això. Som dos alumnes de SMX2 i vam decidir repartir-nos la feina: l'Alex s'ha encarregat més del backend i la base de dades, i en Daniel més del frontend i el disseny, tot i que hem anat ajudant-nos en tot. Treballar en equip ha estat un repte perquè no estem acostumats a compartir codi, però al final hem après molt.

La web actualment té el sistema de registre i inici de sessió funcionant. Quan fem clic a "Registre" o "Inici de sessió" al menú, s'obre un modal (una finestra emergent feta amb JavaScript) per posar el correu electrònic i la contrasenya. Les contrasenyes es guarden xifrades a la base de dades (no es veuen en clar). La part de classificació de LaLiga i màxims golejadors funciona perfectament amb l'API externa API-Football. Després de moltes proves i errors, vam aconseguir connectar-la i mostrar dades reals en temps real.

Aquest context, doncs, genera una oportunitat clara per a un projecte com SafeGoalStats que: ofereix informació fiable (amb l'API funcionant), proporciona un entorn segur per als usuaris registrats (amb contrasenyes xifrades i formularis validats), i serveix com a eina de formació pràctica per a nosaltres, els estudiants de SMX.

1.2 Justificació

Vam decidir fer SafeGoalStats per cinc motius principals que explico a continuació amb detall:

Opció 1: Millora la seguretat de l'usuari amb bones pràctiques bàsiques.

Actualment, molts aficionats al futbol no són conscients dels riscos de seguretat en línia. SafeGoalStats implementa mesures que, tot i ser bàsiques, redueixen molt la superficie d'atac: xifrat de contrasenyes mitjançant bcrypt, validació de formularis tant al client (JavaScript) com al servidor (express-validator), protecció contra SQL injection mitjançant consultes parametritzades, i ús de JWT per a gestió de sessions sense estat. Aquestes mesures, encara que no són suficients per a una aplicació bancària, son un bon punt de partida per aprendre ciberseguretat i conscienciar els usuaris.

Opció 2: Ens permet aplicar coneixements teòrics en un projecte real.

A classe havíem vist molta teoria: com funciona HTTP, què és una API, com es fa una base de dades relacional... Però fins que no ho fas en un projecte real, no ho acabes d'entendre. SafeGoalStats ens ha permès tocar totes aquestes tecnologies: programació web (HTML, CSS, JavaScript), desenvolupament backend amb Node.js, bases de dades amb SQLite, control de versions amb GitHub, conceptes de seguretat, i integració d'APIs externes. A diferència dels exercicis petits de classe, aquest projecte integra tot en una solució que té sentit.

Opció 3: Combina informació esportiva real amb funcionalitats interactives.

Gràcies a l'API-Football, mostrem dades reals de la classificació de LaLiga i els màxims golejadors. Això és un valor afegit molt important. A més, els usuaris poden deixar opinions, cosa que fomenta la participació. La majoria d'aplicacions de resultats només deixen consultar; nosaltres vam voler anar un pas més enllà.

Opció 4: Serveix com a base per a futurs desenvolupaments més avançats.

L'arquitectura que hem fet (frontend separat, backend amb rutes API, base de dades independent, connexió a API externa) està pensada per poder créixer. Si algun dia volem afegir notificacions push, desplegar-ho al núvol, fer una aplicació mòbil, o posar autenticació amb Google, ho podrem fer sense haver de començar de zero. Això es una cosa que vam aprendre a classe: separar responsabilitats facilita el manteniment i l escalabilitat.

Opcio 5: Es econòmicament viable i sostenible.

Totes les eines que hem fet servir són gratuïtes o de codi obert: Node.js, Express, SQLite, Visual Studio Code, GitHub, Figma. L'API de futbol té un pla gratuït de 100 peticions per dia, que per a un projecte educatiu és suficient. Si algun dia volguéssim posar la web a Internet, hi ha opcions barates o fins i tot gratuïtes (Render, Vercel, Railway). Per tant, el projecte no requereix inversió econòmica i pot mantenir-se sense costos.

Aquesta justificació reforça que el projecte no només té valor academic, sinó també aplicabilitat real, valor educatiu, potencial de creixement futur i viabilitat econòmica.

1.3 Objectius

1.3.1 Objectiu general

Desenvolupar una aplicació web segura i funcional que ofereix informació actualitzada de LaLiga (classificació i màxims golejadors) mitjançant una API externa, permet als usuaris registrar-se i interactuar mitjançant opinions, combinant aprenentatge pràctic amb bones pràctiques de seguretat. Tot això seguint una metodologia àgil (Scrum adaptat) i utilitzant eines professionals de desenvolupament (GitHub, VS Code, Trello, etc.).

Aquest objectiu general el considerem assolit quan l'aplicació és capaç de gestionar usuaris (registre i login segur amb JWT), permetre opinions d'usuaris (crear i llegir), mostrar dades reals de l'API de futbol (classificació i golejadors), funcionar correctament en navegadors moderns, tenir una interfície responsive, i tenir mesures de seguretat basiques (bcrypt, JWT, validacio, SQL injection).

1.3.2 Objectius específics

Els objectius específics del projecte es detallen a continuació amb el seu estat de consecució:

Objectiu específic	**Descripció ampliada**	**Estat**	**Mesura de verificació**
1. Crear un sistema de registre i autenticació segur	Implementar registre amb email i contrasenya, xifrant la contrasenya amb	Implementat	Hem provat amb diferents usuaris, les contrasenyes a la base de dades no són llegibles (només el hash), i els tokens JWT funcionen correctament

	bcrypt (10 rondes de salt). El login ha de verificar les credencials i generar un JWT (JSON Web Token) amb expiració de 24 hores.		en peticions posteriors.
2. Connectar API externa per mostrar classificació de LaLiga en temps real	Connectar amb API-Football per obtenir la classificació de LaLiga (posició, equip, punts, partits jugats, victòries, empats, derrotes, gols a favor, gols en contra, diferència de gols). Implementar cache per optimitzar el límit de 100 peticions/dia del pla gratuït.	Implementat	Hem aconseguit connectar l'API després de moltes proves. La classificació es mostra correctament en una taula HTML. Les dades s actualitzen cada 30 minuts.
3. Connectar API externa per mostrar màxims golejadors de LaLiga	Connectar amb API-Football per obtenir els màxims golejadors de LaLiga (nom del jugador, equip, nombre de gols). Implementar cache per optimitzar el límit de peticions.	Implementat	Hem aconseguit connectar l'API. Els golejadors es mostren correctament en una taula HTML ordenada per nombre de gols.
4. Assegurar seguretat bàsica de formularis i contrasenyes	Validar tots els formularis tant al client (JavaScript) com al servidor (express-validator). Xifrar contrasenyes (bcrypt). Protegir contra SQL injection (consultes parametritzades).	Implementat	Hem fet proves d'injecció (posant codi SQL als formularis) i no va funcionar. Les validacions detecten emails incorrectes i contrasenyes curtes.
5. Implementar una base de dades funcional amb SQLite	Dissenyar l'esquema de la base de dades (taules usuaris i opinions). Crear les taules i les relacions (clau forana opinions.user_id -> users.id). Implementar operacions CRUD basiques (Crear, Llegir, Actualitzar,	Implementat	Les taules estan creades, les consultes funcionen, la integritat referencial esta assegurada. Fem inserts, selects i deletes.

	Eliminar).		
6. Treballar de manera col·laborativa amb GitHub i VS Code	Utilitzar Git per al control de versions, amb branques separades per funcionalitats (backend, frontend, feature/opinions, feature/api), commits regulars amb missatges descriptius, i pull requests per revisar el codi de l'altre.	Implementat	Tenim el repositori amb historic de commits, branques separades, i hem après a resoldre conflictes quan dos modifiquem el mateix fitxer.
7. Crear una interfície responsive i amigable	Que la web es vegi be tant en ordinador (escriptori) com en mòbil (iPhone, Android). Utilitzar CSS amb Flexbox, Grid i media queries per adaptar el disseny a diferents mides de pantalla.	Implementat	Hem provat la web a Chrome, Firefox, Edge i a diferents dispositius, ordinador i mòbil, funciona correctament.

1.4 Estratègia i planificació del projecte

L'estratègia que vam triar va ser desenvolupar un producte nou des de zero, no fer servir WordPress, Wix o altres eines que fan pagines web automàticament. Vam prendre aquesta decisió per què volem aprendre realment com funciona tot per dins: com es fa un backend, com es connecta a una base de dades, com es validen formularis, com es gestiona l'autenticació amb JWT, com es protegeix contra atacs, i com es connecta una API externa. Si haguessim fet servir WordPress, no hauriem après gairebé res de programació.

Aquest enfocament es adequat per diversos motius:

1. Garantim que la seguretat i les bones pràctiques s'implementen des del començament. Quan es fa servir un CMS existent, sovint s arrosseguen vulnerabilitats del codi base o de plugins de tercers. En canvi, desenvolupant des de zero, tenim control total sobre cada línia de codi i podem aplicar les mesures de seguretat que hem après a classe.

2. Podem experimentar amb Node.js i altres tecnologies sense dependre de limitacions externes. Node.js es asíncron i eficient per a operacions d'entrada/sortida, la qual cosa es ideal per a una aplicació que ha de fer crides a una API externa. També té un ecosistema immens de llibreries a npm, la qual cosa ens va permetre trobar solucions rapidament (bcrypt, jsonwebtoken, express-validator, axios, etc.).

3. Tenim control total sobre l'estructura de dades i la planificació. Decidim nosaltres quines taules creem, quines columnes tenen, quines relacions hi ha. Això és important per a un projecte formatiu per què ens obliga a pensar en el disseny de la base de dades, no només a copiar un tutorial.

4. Aprenem mes perquè cada línia de codi l'hem escrit nosaltres. Quan alguna cosa no funciona, no podem culpar a un plugin a una configuració fosca. Havíem de buscar l'error, entendre l i arreglar-lo. Això és el que realment ens ha fet aprendre.

Estudi de viabilitat inicial:

Vam realitzar un estudi de viabilitat al principi del projecte. Vam comprovar que Node.js podia connectar-se a l'API-Football fent proves amb Postman. Vam veure que les dades de LaLiga (classificació i golejadors) estan disponibles en format JSON i que la documentació de l'API és suficient, tot i que està en anglès i costa una mica d'entendre. Vam verificar que SQLite es compatible amb Node.js (hi ha el mòdul sqlite3 que funciona perfectament). També vam confirmar que les eines de control de versions (Git i GitHub) són suficients per treballar en equip.

Fase	Descripció	Durada setmanes	Tasques principals	Estat
Fase 1	Investigació i configuració d'entorn	1 setmana	Instal·lació de Node.js, VS Code, Git; proves amb API-Football amb Postman; disseny inicial dels wireframes a Figma; creació del repositori a GitHub	Completada
Fase 2	Desenvolupament del backend (basic)	2 setmanes	Creació del servidor Express amb la primera ruta de prova (/ping); configuració de les variables d'entorn (.env); instal·lació de dependències (express, sqlite3, dotenv)	Completada
Fase 3	Implementació de l'autenticació	2 setmanes	Creació de les rutes /register i /login; connexió amb SQLite; xifrat de contrasenyes amb bcrypt; generació i verificació de JWT; creación del middleware d'autenticació	Completada
Fase 4	Desenvolupament del frontend	2 setmanes	Maquetació HTML de totes les pàgines (index,	Completada

			register, login, dashboard, classificació, golejadors, opinions); disseny CSS responsive; implementació dels modals amb JavaScript; connexió amb el backend mitjançant fetch	
Fase 5	Integració amb API externa (CLASSIFICACIÓ)	4 setmanes	Implementació de crides a API-Football per a classificació; cache de respostes en memòria; visualització de dades al frontend	Completada
Fase 6	Integració amb API externa (GOLEJADORS)	4 setmanes	Implementació de crides a API-Football per a màxims golejadors; cache de respostes; visualització de dades al frontend	Completada
Fase 7	Seguretat i proves	1 setmana	Validació de formularis amb express-validator; proves de SQL injection; proves d'autenticació; proves de navegadors (Chrome, Firefox, Edge); proves de responsivitat (DevTools)	Completada
Fase 8	Documentació	Des de principi de curs	Elaboració de la memòria (aquest document); captures de pantalla; preparació d'annexos; pujada del codi final a GitHub; preparació de la presentació	Completada

Desviacions sobre la planificació inicial:

La principal desviació va ser que vam dedicar més temps del previst a la integració de l'API. Vam tenir problemes amb les claus d'API, la comprensió de la documentació (en anglès tècnic), els errors en les capçaleres de les peticions, i el procediment de la resposta JSON (què és bastant complexa). Però després de moltes proves i errors, ho vam aconseguir. Vam haver de dedicar una setmana addicional a la fase 5, però finalment l'API de classificació i golejadors funciona perfectament.

1.5 Metodologia de treball

Com que som dues persones, vam decidir adaptar la metodologia àgil Scrum a la nostra mida i al nostre context educatiu. No vam fer servir Scrum complet (ja que requereix rols com Scrum Master, Product Owner, etc. que no tenen sentit en un projecte d'institut), però vam agafar les idees principals que ens van semblar útils: sprints curts, reunions diàries, planificació per iteracions, i revisió contínua.

Què és Scrum (breu introducció teorica)?

Scrum es una metodologia àgil per a la gestió de projectes què és basa en cicles curts anomenats sprints (normalment de 1 a 4 setmanes). Al final de cada sprint, l'equip ha d'entregar un increment de producte funcional (quelcom què és pugui mostrar i que aporti valor). Scrum té tres rols principals: Product Owner (decideix què és fa), Scrum Master (facilita el procés) i Development Team (l'equip que desenvolupa). Nosaltres vam simplificar això perquè eram dos i el Product Owner era el professor (amb les seves indicacions).

Com vam adaptar Scrum al nostre projecte:

1. **Sprints de 2 setmanes (8 sprints en total).**

Cada dues setmanes ens posavem un objectiu concret. Al final de cada sprint, haviam de tenir alguna funcionalitat funcionant. Això ens va ajudar a tenir un ritme constant i a no deixar-nos coses per al final.

Sprint	Setmanes	Objectiu	Funcionalitat lliurada	Estat
Sprint 1	1-2	Configurar entorn i servidor basic	Servidor Express funcionant mitjançant bastion	Completat
Sprint 2	3-4	Implementar registre d'usuaris	Formulari de registre (frontend) + ruta POST /register (backend) + SQLite	Completat
Sprint 3	5-6	Implementar login i autenticació	Formulari de login + JWT + middleware d'autenticació	Completat
Sprint 4	7-8	Desenvolupar frontend complet	Totes les pàgines maquetades, modals, estils responsius	Completat
Sprint 5	9-10	Integrar API de classificació	Crides a API-Football per classificació, cache, visualització en taula	Completat
Sprint 6	11-12	Integrar API de golejadors	Crides a API-Football per	Completat

			màxims golejadors, visualització en taula	
Sprint 7	13-14	Proves, seguretat i documentació	Memòria completa, captures, proves de seguretat	Completat

2. **Repartiment de tasques clar i documentat.**

Des del primer dia vam definir qui faria que. Això ens va evitar malentesos i duplicitats.

Persona	**Tasques principals**	**Tecnologies associades**
Alex	Backend, base de dades, autenticació, API (classificació i golejadors)	Node.js, Express, SQLite, bcrypt, JWT, axios, Postman
Daniel	Frontend, disseny, maquetació, estils, modals, connexió amb backend	HTML, CSS, JavaScript, Figma, fetch API

3. **Eines de gestió de tasques.**

Vam fer servir Google Sheets per organitzar les tasques. Es una eina gratuïta amb taulers. Vam crear un tauler amb les següents columnes:

- **"Per fer" (To Do):** tasques pendents, ordenades per prioritat. Exemples: "Crear ruta POST /login", "Maquetar el formulari de registre", "Implementar middleware JWT", "Connectar API classificació".
- **"En progrés" (Doing)**: tasques en que algú estava treballant en aquest moment. Només 2-3 tasques alhora per no dispersar-nos.
- **"En revisió" (Review):** tasques que un havia acabat i l'altre havia de revisar. Això era important per assegurar-nos que el codi funciona i que no hi havia errors.
- **"Fet" (Done):** tasques completades i verificades.

Cada targeta tenia una descripció, d'una llista de verificació (checklist) i etiquetes de prioritat (Alta, Mitjana, Baixa). Quan una persona comença una tasca, la movia a "En progrés" i s'assignen a si mateixa. Quan s'acaba, la movia a "En revisió". L'altra persona la revisava i si estava correcta, la movia a "Fet".

4. **Reunions (tres tipus diferents).**

Les reunions van ser clau per coordinar-nos i resoldre problemes ràpidament.

- Daily meeting (cada dia, 10-15 minuts): Ens trobavem a l'hora del pati o abans de començar classe. Ens feiem tres preguntes: "Que vas fer ahir?", "Que faràs avui?", "Tens algun problema o bloqueig?". Aquestes reunions curtes ens van ajudar a estar sempre al dia i a detectar problemes aviat

- Sprint planning (cada 2 setmanes, 30-45 minuts): Al començament de cada sprint, ens reuniem per decidir que faríem. Miravem la llista de tasques pendents, triavem les més importants (prioritat Alta) i les assignavem al sprint. També definiem l'objectiu del sprint (ex: "Aquest sprint tindrem la classificació de LaLiga funcionant").

- Sprint review (cada 2 setmanes, 20-30 minuts): Al final de cada sprint, ens reuniem per veure que haviem aconseguit. Mostravem el que funcionava (ex: "Mira, la classificació es mostra correctament amb dades reals de l'API") i el que no funcionava (ex: "Els golejadors encara no es veuen bé, tenim un error en el processament del JSON"). Això ens servia per ajustar el pla del següent sprint.

5. **Control de versions amb GitHub (fonamental per treballar en equip).**

Vam fer servir GitHub amb una organització clara:

- Repositori principal: safegoalstats (privat, accessible només per nosaltres i el professor)
- Branca main: La versió estable del projecte. Només hi posavem codi que havia superat proves i que l'altre company havia revisat.
- Branca API: L'Alex treballava aquí. Quan tenia alguna funcionalitat nova (ex: nova ruta /api/standings), feia un commit amb un missatge clar i després un pull request per fusionar-la a main. En Daniel revisava el codi i si estava bé, acceptava el pull request.
- Branca frontend: En Daniel treballava aquí. Quan tenia les pàgines o els modals fets, feia un pull request i l'Alex revisava.
- Branques de funcionalitats específiques (feature branches): Per a coses més complexes, creavem branques curtes. Exemples: feature/opinions, feature/jwt-auth, feature/api-standings, feature/api-topscorers, bugfix/sql-injection. Després d'acabar, es feien pull requests i s eliminava la branca.

Que vam aprendre sobre control de versions?

- Vam aprendre que els commits han de ser petits i freqüents (no esperar a tenir 500 línies de codi per fer un commit). També vam aprendre a escriure missatges de commit descriptius (ex: "feat: afegeix ruta GET /api/standings per a classificació", "fix: corregeix error de validació email", "docs: actualitza README"). I vam aprendre a resoldre conflictes (quan dos modifiquen la mateixa línia del mateix fitxer). Al principi ens feien por, però després vam veure que Git tiene eines per resoldre'ls.

6. **Gestió de problemes i bloquejos.**

Vam tenir diversos bloquejos durant el projecte. Els principals:

- Bloc 1 (setmana 4): No sabiem com fer que el frontend enviés el JWT a cada petició. Ho vam resoldre investigant a MDN i trobant que s ha de fer amb localStorage.getItem('token') i l header Authorization: Bearer <token>.
- Bloc 2 (setmana 9): L'API de futbol no connectava al principi. Vam tenir problemes amb la clau d'API (no sabíem que s havia de passar a l header x-apisports-key), amb el CORS (vam haver d'instal·lar el mòdul cors a Express), i amb el processament de la resposta JSON (la estructura era més complexa del que

pensavem). Vam dedicar moltes hores a llegir documentació, fer proves amb Postman, i depurar el codi. Finalment, vam aconseguir que funciones.

- Bloc 3 (setmana 12): Un conflicte a GitHub que va sobreescriure codi. Vam aprendre a resoldre conflictes amb git merge i git diff, i des de llavors vam fer pull requests més petits i mes frequents.

Valoración global de la metodología:

La metodologia que vam fer servir ens va funcionar prou bé. Ens va permetre organitzar-nos, repartir la feina, fer seguiment del progrés i adaptar-nos als problemes. No vam ser experts en Scrum ni en cap metodologia formal, però per al nostre nivell i per a un projecte d'institut, va ser suficient. Els aprenentatges principals van ser: planificar abans de programar, fer reunions curtes però freqüents, utilitzar eines com Trello i GitHub, i no tenir por de canviar el pla quan alguna cosa no funciona

1.6 Estudi econòmic i pressupostari

Un dels avantatges d'aquest projecte es que no ens ha costat diners. Tot el que hem fet servir és gratuït o de codi obert. Això ens ha permès centrar-nos en aprendre i no en buscar pressupost. A continuació es mostra un desglossament detallat:

Component	Cost orientatiu
API-Football (subscripció)	0 € (gratuït)
Hosting i domini web	0 € (gratuït)
Eines de disseny i software	0 € (gratuït)
Total desenvolupament	0 € (gratuït)

2 Descripció del projecte

Aquest capítol descriu amb detall el projecte **SafeGoalStats**, definint els requisits que ha de complir, les tecnologies analitzades i escollides, així com l'estructura general i les funcionalitats de la solució desenvolupada. L'objectiu és establir una base clara i estructurada que permeti entendre com es passarà de la idea inicial a una aplicació web funcional, segura i alineada amb els objectius formatius del projecte.

Tot i tractar-se principalment d'un projecte de desenvolupament, s'ha adoptat un enfocament de **recerca aplicada**, ja que moltes decisions tècniques requereixen anàlisi prèvia, proves i comparatives abans de la seva implementació final.

2.1 Anàlisi de requisits

L'anàlisi de requisits defineix el conjunt de condicions que el projecte ha de satisfer per considerar-se complet. Aquests requisits serveixen com a referència durant tot el desenvolupament i permeten verificar, un cop finalitzat, si el sistema compleix les funcionalitats i la qualitat esperades.

2.1.1 Requisits funcionals

Els requisits funcionals descriuen les funcionalitats principals del sistema, és a dir, allò que l'aplicació permet fer a l'usuari de manera directa.

Codi	Descripció
RF1	Registre d'usuaris amb email i contrasenya.
RF2	Login i gestió del perfil.
RF3	Consulta d'estadístiques i resultats.
RF4	Consulta de màxims golejadors de LaLiga (API real)
RF5	Actualització automàtica de dades des de l'API externa.
RF6	Tancar sessió

Aquests requisits funcionals s'han definit amb l'objectiu de garantir una experiència d'usuari clara, progressiva i coherent, centrada tant en la consulta d'informació esportiva com en la interacció segura amb el sistema.

2.1.2 Requisits no funcionals

Els requisits no funcionals defineixen com ha de comportar-se el sistema, afectant indirectament l'usuari però sent clau per a la qualitat global del projecte.

Codi	Descripció
RNF1	Interfície accessible des de dispositius mòbils i ordinadors.
RNF2	Temps de càrrega inferior a 3 segons.
RNF3	Xifrat de contrasenyes i validació de formularis.
RNF4	Compatibilitat amb navegadors moderns.
RNF5	Còpies de seguretat periòdiques de la base de dades.

Aquests requisits són fonamentals per garantir la robustesa, seguretat i fiabilitat de l'aplicació, assegurant que el projecte no només sigui funcional, sinó també sostenible i segur.

2.1.3 Previsió de tasques d'investigació

Abans de començar a programar, vam haver d'investigar diverses coses perquè no les sabiem fer de memòria. Aqui les tasques que vam fer, qui les va fer i els resultats obtinguts:

Tasca	Qui la va fer	Resultat
Provar API-Football (classificació)	Alex	Després de moltes proves, vam aconseguir connectar-la. Obtenim dades reals de classificació.
Provar API-Football (golejadors)	Alex	També funciona. Obtenim els màxims golejadors de LaLiga.
Node.js vs Rust	Daniel	Vam decidir usar Node.js perquè es mes facil i te mes comunitat.
Xifrat de contrasenyes	Alex	Vam triar bcrypt perquè és el més segur i recomanat.
SQLite vs MySQL	Daniel	Vam triar SQLite perquè és més senzill i no requereix instal·lació.
Validació de formularis	Daniel	Vam implementar validació amb JavaScript al frontend i express-validator al backend.

GitHub en equip	Entre els dos	Vam implementar validació amb JavaScript al frontend i express-validator al backend.

Aquestes investigacions ens van ocupar unes 16 hores aproximadament, però ens van estalviar molt de temps després perquè vam prendre decisions informades.

2.2 Tecnologies

2.2.1 Comparativa de les tecnologies valorades

Abans de seleccionar les tecnologies definitives, es van estudiar diferents opcions per a cada component, valorant avantatges i inconvenients:

Component	**Opcions**	**Pros**	**Contres**
Backend	Rust, Node.js + Express, Python.	**Rust**: seguretat de memòria i rendiment; **Node.js/Python**: comunitat gran	**Rust**: corba d'aprenentatge alta
Frontend	HTML, CSS, JS, React	**HTML, CSS, JS**: senzill i comprensible ; **React**: modularitat	**HTML, CSS, JS:** menys dinàmic; **React**: més complex per a principiants
Base de dades	SQLite, MySQL	**SQLite**: lleuger i fàcil; **MySQL**: escalable	**SQLite**: limitat per a grans volums; **MySQL**: més complex
Disseny UI	Figma, Canva	**Figma**: col·laboratiu	Requereix registre i coneixements bàsics
Control versions	GitHub, GitLab	**GitHub**: col·laboratiu i popular	Requereix disciplina

2.2.2 Tecnologies escollides

Les tecnologies definitives seleccionades per al projecte són:

Component	**Tecnologia escollida**	**Justificació**
Backend	Node.js + Express	Perquè és el que millor coneixem, te moltes llibreries (npm) i es asincron per defecte, ideal per a crides a API
Frontend	HTML, CSS, JavaScript	Perquè ho vam donar a classe, és senzill i no depèn de frameworks externs
Base de dades	SQLite	Perquè és lleugera, no requereix instal·lació i s integra be amb Node.js
Disseny UI	Figma	Perquè ens permet fer wireframes i prototips abans de programar
Control de versions	GitHub	Perquè ens permet treballar en equip i portar un històric de canvis
Xifrat contrasenyes	bcrypt	Perquè és el més segur per a contrasenyes (afegeix salt i és lent)
Autenticació	JWT	Perquè no necessita guardar sessions al servidor

Aquesta selecció reflecteix una estrategia progressiva i segura, adequada al nivell de coneixements actual del grup i a l'objectiu de combinar desenvolupament amb aprenentatge de ciberseguretat.

2.3 Estructura del projecte

L'arquitectura de SafeGoalStats es client-servidor, amb components diferenciats que interactuen entre si:

1. **Frontend**: Pàgina web amb HTML, CSS i JavaScript. Gestiona la visualització de dades, els modals de registre i login, i les peticions al backend.

2. **Backend**: Desenvolupat en Node.js amb Express. Processa peticions, gestiona autenticació (JWT), interactua amb la base de dades SQLite i amb l'API externa (API-Football).

3. **Base de dades**: SQLite. Emmagatzema usuaris i opinions.

4. **API externa**: API-Football. Proporciona la classificació de LaLiga en temps real i les estadístiques de màxims golejadors.

Aquesta estructura assegura separació de responsabilitats, facilita manteniment i l'evolució del projecte i permet afegir funcionalitats més complexes en fases posteriors.

2.4 Descripció dels components

2.4.1 Component 1

Frontend, el lloc web te:

- **Pàgina principal (index.html)**: amb informació del projecte i un menú a dalt.
- **Modals**: finestres emergents per registrar-se i fer login (no són pàgines separades).
- **Menú**: amb opcions: Inici, Registre, Login, Classificació, Golejadors, Opinions.
- **Secció de Classificació**: mostra la classificació de LaLiga en temps real (dades de l'API).
- **Secció de Golejadors**: mostra els màxims golejadors de LaLiga (dades de l'API).

El CSS fa que la web es vegi bé tant en ordinador com en mòbil (responsiva). Els estils utilitzen Flexbox, Grid i media queries.

2.4.2 Component 2

Backend:

- **Funcionalitat:** Gestió de lògica, autenticació d'usuaris, connexió amb la base de dades i comunicació amb l'API externa.
- **Tecnologia:** Node.js + Express
- **Observacions:** Permet assegurar seguretat de memòria i rendiment elevat; futurament podrà gestionar actualitzacions automàtiques de dades.

2.4.3 Component 3

Base de dades

. **Funcionalitat:** Emmagatzematge d'usuaris, prediccions i registres del sistema.

. **Tecnología:** SQLite.

. **Observacions:** Lleugera i fàcil d'integrar amb Node js + Express; còpia de seguretat periòdica.

2.4.4 Component 4

Aquesta és la part de la qual estem més orgullosos. Després de moltes proves, vam aconseguir connectar l'API-Football per obtenir:

1. Classificació de LaLiga: posició, equip, punts, partits jugats (PJ), victòries (PG), derrotes (PP)
2. Màxims golejadors: nom del jugador, equip, nombre de gols.

Com ho vam fer:

1. Ens vam registrar a API-Football i vam obtenir una clau d'API (gratuita, 100 peticions/dia).
2. Vam investigar la documentació (en anglès) per trobar els endpoints:
3. Vam fer proves amb Postman per assegurar-nos que la clau funcionava i que rebíem dades.
4. Vam implementar funcions al backend amb axios per fer les peticions.
5. Vam afegir un sistema de caché (guardar la resposta durant 30 minuts) per no superar el límit de 100 peticions/dia.
6. Vam processar la resposta JSON (què és bastant complexa) per extreure només les dades que ens interessaven.
7. Vam enviar les dades al frontend, que les mostra en taules HTML.

Problemes que vam trobar i com els vam solucionar:

Problema	**Solució**
La clau d'API no funcionava	Vam llegir la documentació i vam veure que s havia de passar a l header x-apisports-key
Error CORS	Vam configurar el backend per permetre CORS amb el mòdul cors
La resposta JSON era massa gran i complexa	Vam filtrar les dades al backend abans d'enviar-les al frontend
Superavem el límit de peticions	Vam implementar cache amb un objecte JavaScript simple

2.5 Definició de les tasques

2.5.1 Prova 1

Objectiu: Validar que l'API-Football retorna correctament la classificació de LaLiga i els màxims golejadors.

Components implicats: Backend (Node.js + Express), API externa (API-Football).

Procés:

1. Registrar-se a API-Football i obtenir clau d'API.

2. Provar l'endpoint de classificació amb Postman. Va funcionar.
3. Provar l'endpoint de golejadors amb Postman. Va funcionar.
4. Implementar crides amb axios al backend.
5. Processar la resposta JSON per extreure dades rellevants.
6. Enviar dades al frontend.
7. Mostrar en taules HTML (classificació i golejadors).

Resultat: Exit. La classificació i els golejadors es mostren correctament.

2.5.2 Prova 2

Autenticació segura

- **Objectiu:** Validar registre i login amb contrasenyes xifrades.
- **Components implicats:** Backend, Base de dades, Frontend.
- **Procés:** Creació d'usuaris de prova, login amb dades reals, validació de formularis.
- **Conclusions esperades:** Confirmar seguretat mínima i funcionalitat correcta.

Resultats de les proves:

Prova	**Resultat**
Registrar usuari nou	Funciona, contrasenya xifrada a SQLite
Registrar email duplicat	Error: email ja existeix
Login amb dades correctes	Funciona, retorna JWT
Login amb contrasenya incorrecta	Error: credencials incorrectes
Login amb email inexistent	Error: credencials incorrectes
Accés a ruta protegida sense token	Error 401 No autoritzat
Accés a ruta protegida amb token valid	OK, retorna dades
Validació email incorrecte (ex: "usuari")	Error: email invalid
Validació contrasenya curta (menys de 6 caràcters)	Error: contrasenya massa curta

2.5 Definició de les funcionalitats

Codi	Funcionalitat	Procés conceptual	Estat actual
F1	Registre d'usuaris	Formulari web (modal); dades enviades al backend; validació; xifrat contrasenya; emmagatzematge a SQLite	Implementada
F2	Autenticació	Login amb verificació email i contrasenya; generació de JWT; emmagatzematge al frontend (localStorage); accés a rutes protegides	Implementada
F3	Consulta de classificó de LaLiga	Crida a API externa (API-Football) des del backend; cache; processament de resposta; visualització al frontend en taula HTML	Implementada
F4	Consulta de màxims golejadors de LaLiga	Crida a API externa (API-Football) des del backend; cache; processament de resposta; visualització al frontend en taula HTML	Implementada

2.5.1 Funcionalitat 1

Registre d'usuaris

- **Descripció:** Permet que un visitant de la pàgina crei un compte d'usuari introduint email i contrasenya. El backend valida les dades (email valid, contrasenya mínim 6 caràcters), xifra la contrasenya amb bcrypt (10 rondes de salt) i emmagatzema la informació a la base de dades SQLite (taula users). Aquesta funcionalitat es clau per garantir la seguretat bàsica dels usuaris i per permetre l'accés personalitzat a opinions.

- **Estat actual:** Implementada

2.5.2 Funcionalitat 2

Autenticació

- **Descripció:** Permet als usuaris registrats iniciar sessió amb el seu email i contrasenya. El backend comprova si l'email existeix a SQLite i si la contrasenya es correcta (bcrypt.compare). Si tot està bé, genera un JWT (JSON Web Token) amb una expiració de 24 hores i el retorna al frontend. El frontend guarda el token a localStorage i l'envia a cada petició subsequent a rutes protegides (com ara afegir opinions)..

. **Estat actual:** Implementada

2.5.3 Funcionalitat 3

Consulta de classificació de LaLiga (API funcional)

- **Descripció:** Permet als usuaris consultar la classificació de LaLiga en temps real. El backend fa una petició a API-Football (endpoint /standings), procesa la resposta JSON (extrau posició, equip, punts, PJ, PG, PP) i la envia al frontend. Per optimitzar el límit de peticions (100/dia), s'implementa un sistema de cache que guarda la resposta durant 30 minuts. El frontend mostra les dades en una taula HTML.

- **Estat actual:** Implementada

2.5.4 Funcionalitat 4

Consulta de màxims golejadors de LaLiga (API funcional)

- **Descripció:** Permet als usuaris consultar els màxims golejadors de LaLiga. El backend fa una petició a API-Football (endpoint /players/topscorers), procesa la resposta JSON (extreu nom del jugador, equip, gols) i la envia al frontend. També té cache de 30 minuts. El frontend mostra les dades en una taula HTML ordenada per nombre de gols (de major a menor).

- **Estat actual:** Implementada

3 Altres capítols

En aquest apartat es descriu l'organització general del projecte SafeGoalStats, detallant l'estructura interna del sistema, els components que el formen i la relació entre aquests. L'objectiu és deixar clar com està construït el projecte a nivell funcional i conceptual, així com justificar les decisions tècniques preses.

3.1 Estructura del projecte

El projecte segueix una arquitectura **client–servidor** dividida en diversos components diferenciats. Aquesta separació facilita la comprensió del sistema, el manteniment del codi i la possibilitat d'ampliació futura.

L'estructura general del projecte es compon de:

- **Client (Frontend):** responsable de la interfície i l'interacció amb l'usuari.
- **Servidor (Backend):** gestiona la lògica de l'aplicació i la seguretat.
- **Base de dades:** emmagatzema dades persistents com usuaris i prediccions.
- **Servei extern (API):** proveeix dades esportives actualitzades.

El flux general del sistema és el següent:

L'usuari interactua amb la interfície web, el frontend envia les dades al backend, aquest processa la informació i accedeix a la base de dades o a l'API externa segons sigui necessari, retornant finalment la informació processada a l'usuari.

Aquesta arquitectura s'ha escollit perquè permet una separació clara de responsabilitats, millora la seguretat i facilita l'escalabilitat del projecte.

3.2 Descripció dels components

- **Frontend**

El frontend és la part visible de l'aplicació i el punt d'interacció directe amb l'usuari. S'ha desenvolupat utilitzant HTML, CSS i JavaScript bàsic, utilitzant Visual Studio Code com a entorn de desenvolupament.

Aquest component s'encarrega de:

- Presentar la informació esportiva de forma estructurada.
- Permetre als usuaris registrar-se i iniciar sessió.
- Capturar les dades introduïdes per l'usuari i enviar-les al backend.

Es va valorar l'ús de frameworks com React, però es va descartar per prioritzar l'aprenentatge dels fonaments del desenvolupament web.

- **Backend**

El backend actua com a intermediari entre el frontend, la base de dades i l'API externa. És el component encarregat de processar les dades, validar la informació rebuda i aplicar mesures bàsiques de seguretat.

Està planificat en Node.js + Express, principalment per la seva alta eficiència en la gestió de peticions asíncrones i el seu òptim rendiment en aplicacions en temps real.

Les seves funcions principals són:

- Validació de credencials d'usuari.
- Gestió de peticions del frontend.
- Comunicació amb la base de dades.
- Obtenció de dades des de l'API externa.

- **Base de dades**

La base de dades s'utilitza per emmagatzemar informació persistent, com dades d'usuaris i prediccions. S'ha escollit SQLite per la seva simplicitat, facilitat d'integració i adequació a l'escala del projecte.

Es van valorar alternatives com MySQL, però es van descartar per la seva major complexitat i per no ser necessàries en aquesta fase del projecte.

- **API externa**

L'API externa és el component encarregat de proporcionar dades esportives actualitzades, com resultats i estadístiques. El backend és el responsable de comunicar-se amb aquesta API i retornar la informació processada al frontend.

Aquest component permet enriquir l'aplicació sense necessitat de mantenir una base de dades esportiva pròpia.

3.3 Definició de les funcionalitats i relació amb l'arquitectura

Funcionalitat	**Components implicats**	**Flux**
Registre	Frontend (modal) -> Backend (POST /register) -> SQLite	Usuari envia email i contrasenya; backend xifra i guarda
Login	Frontend (modal) -> Backend (POST /login) -> SQLite -> JWT	Usuari envia credencials; backend verifica i retorna token
Classificació LaLiga	Frontend -> Backend (GET /api/standings) -> API-Football -> Cache	Backend crida API, guarda cache, envia dades al frontend
Golejadors LaLiga	Frontend -> Backend (GET /api/topscorers) -> API-Football -> Cache	Backend crida API, guarda cache, envia dades al frontend

4 Conclusions

4.1 Conclusions generals del projecte

El desenvolupament de SafeGoalStats ens ha permès posar en pràctica molts dels coneixements que hem après a SMX: programació web (HTML, CSS, JavaScript), desenvolupament backend amb Node.js + Express, bases de dades amb SQLite, ciberseguretat (bcrypt, JWT, SQL injection, XSS), control de versions amb GitHub, treball en equip, i l'integració d'APIs externes.

Hem aconseguit els objectius principals: tenim un sistema de registre i login funcional i segur, una base de dades operativa, una interfície responsive, i el més important: hem connectat l'API-Football per mostrar la classificació de LaLiga en temps real i els màxims golejadors. Aquesta era la part que més ens preocupava i després de moltes proves ho vam aconseguir.

També hem implementat un sistema d'opinions d'usuaris que substitueix les prediccions inicials (que eren massa complicades). Els usuaris poden deixar comentaris i tothom pot llegir-los.

Aprendre a treballar en equip ha estat una de les parts més importants. Hem après a repartir tasques, a fer servir GitHub per treballar junts sense trepitjar-nos, a fer reunions diaries curtes, i a ajudar-nos quan un de nosaltres tenia problemes tècnics.

4.2 Consecució dels objectius

Objectiu	**Estat**	**Comentari**
Registre i login	Completat	Funciona perfectament amb modals, bcrypt i JWT
Contrasenyes xifrades	Completat	Amb bcrypt, no es veuen en clar a SQLite
Validació formularis	Completat	Client (JavaScript) i servidor (express-validator)
API classificació	Completat	Funciona! Mostra dades reals de LaLiga
API golejadors	Completat	Funciona! Mostra màxims golejadors
Base de dades SQLite	Completat	Taules users i opinions, relacions clau forana
Treball en equip	Completat	Hem fet servir GitHub amb branques i pull requests
Interficie responsive	Completat	Es veu bé a mòbil, ordinador i tauleta

4.3 Valoració de la metodologia i planificació

La metodologia que vam fer servir (Scrum adaptat) ens va funcionar prou bé. Ens va permetre organitzar-nos, repartir la feina, fer seguiment del progrés i adaptar-nos als problemes. Les reunions diàries curtes (daily meeting) van ser molt útils per estar sempre al dia. Trello ens va ajudar a visualitzar les tasques pendents.

Si haguéssim de repetir el projecte, faríem:

- Mes reunions al principi per planificar millor l'arquitectura.
- Investigar mes a fons la connexió amb l'API abans de començar (la documentació era complexa).
- Fer més proves unitàries a mesura que desenvolupaven, no deixar-les totes per al final.

La planificació inicial era bastant optimista: pensàvem que connectar l'API seria fàcil, però ens va costar molt més temps del que pensavem. Però al final ho vam aconseguir.

4.4 Visió de futur

El projecte encara té marge de millora. Les coses que ens agradaria fer en el futur són:

1. Connectar més endpoints de l'API: resultats de partits en temps real, estadístiques més detallades, calendaris, etc.
2. Millorar el disseny: fer la web més atractiva visualment, amb més imatges, transicions, etc.
3. Afegir un sistema de valoració (puntuació de 1 a 5 estrelles) a la web
4. Afegir autenticació amb Google (OAuth) perquè sigui més còmode per als usuaris.
5. Implementar notificacions per correu electrònic quan algú respongui a una opinió.
6. Optimitzar el cache de l'API per reduir encara més les peticions.
7. Fer una aplicació mòbil nativa (Android/iOS) que consumeixi el mateix backend.

Si aconseguim implementar aquestes millores, SafeGoalStats pot convertir-se en una aplicació de referencia per als aficionats de LaLiga.

5. Glossari

Aquest glossari recull els principals termes tècnics utilitzats al llarg de la memòria amb l'objectiu de facilitar la comprensió del document, especialment per a lectors que no tinguin coneixements previs en desenvolupament web o ciberseguretat.

API (Application Programming Interface):
Interfície que permet la comunicació entre diferents aplicacions o serveis. En aquest projecte s'utilitza per obtenir dades esportives externes (resultats, estadístiques).

Backend:
Part del sistema que no és visible per a l'usuari i que s'encarrega de la lògica de l'aplicació, el processament de dades i la comunicació amb la base de dades i serveis externs.

Frontend:
Part visible de l'aplicació amb la qual interactua l'usuari. Inclou la interfície web desenvolupada amb HTML, CSS i JavaScript.

Base de dades:
Sistema que permet emmagatzemar i gestionar informació de forma estructurada. En aquest projecte s'utilitza SQLite per guardar dades d'usuaris i prediccions.

Autenticació:
Procés de verificació de la identitat d'un usuari mitjançant credencials com el correu electrònic i la contrasenya.

Xifrat de contrasenyes:
Procés de transformar una contrasenya en un format segur per evitar que sigui llegible en cas d'accés no autoritzat a la base de dades.

Phishing:
Tècnica de frau en línia que intenta obtenir dades personals o credencials fent-se passar per una entitat de confiança.

Client-Servidor:
Model d'arquitectura on el client (frontend) sol·licita serveis i el servidor (backend) respon a aquestes peticions.

HTTP/HTTPS:
Protocols de comunicació utilitzats per transferir dades a través d'Internet. HTTPS incorpora xifrat per garantir la seguretat de la comunicació.

Control de versions (Git/GitHub):
Sistema que permet gestionar els canvis en el codi font i facilitar el treball col·laboratiu entre desenvolupadors.

JWT (JSON Web Token): Token utilitzat per a l'autenticació sense estat (stateless). Conté informació de l'usuari i esta signat digitalment.

bcrypt: Llibreria per xifrar contrasenyes que afegeix un salt aleatori i és resistent a atacs de força bruta.

SQL injection: Atac que consisteix a injectar codi SQL maliciós en un formulari per modificar la base de dades o obtenir informació privada.

XSS (Cross-Site Scripting): Atac que consisteix en injectar codi JavaScript maliciós en una pàgina web.

CORS (Cross-Origin Resource Sharing): Mecanisme que permet restringir o permetre peticions HTTP des de dominis diferents.

Middleware: Funció que s'executa entre la petició i la resposta en un servidor Express. S'utilitza per a autenticació, logging, etc.

Aquest glossari ajuda a establir una base conceptual clara i facilita la lectura global del projecte.

6. Bibliografia

En aquest apartat es recullen les fonts d'informació utilitzades durant el desenvolupament del projecte. Aquestes han servit tant per a la part teòrica com per a la implementació pràctica, especialment en àmbits com desenvolupament web, ús d'APIs i conceptes bàsics de seguretat informàtica.

[1] Mozilla Developer Network (MDN Web Docs)
https://developer.mozilla.org
(**Data de consulta**: gener 2026)
Font principal per a l'aprenentatge i consulta de HTML, CSS i JavaScript.

[2] Node.js Official Documentation
https://nodejs.org
(**Data de consulta**: gener 2026)
Utilitzada per comprendre el funcionament del backend i la gestió de servidors.

[3] Express.js Documentation
https://expressjs.com
(**Data de consulta**: gener 2026)
Referència per a la creació de rutes i gestió de peticions HTTP.

[4] SQLite Documentation
https://www.sqlite.org/docs.html
(**Data de consulta**: març 2026)
Consultada per entendre el funcionament de la base de dades i la seva integració.

[5] API-Football Documentation
https://www.api-football.com/documentation
(**Data de consulta**: abril 2026)
Font utilitzada per integrar dades esportives externes al projecte.

[6] OWASP (Open Web Application Security Project)
https://owasp.org
(**Data de consulta**: gener 2026)
Referència per aplicar bones pràctiques bàsiques de seguretat web.

[7] GitHub Documentation
https://docs.github.com
(**Data de consulta**: novembre 2025)
Utilitzada per gestionar el control de versions i el treball col·laboratiu.

[8] bcrypt Documentation
https://www.npmjs.com/package/bcrypt
(Data de consulta: gener 2026)
Utilitzada per entendre el xifrat de contrasenyes.

[9] jsonwebtoken Documentation
https://www.npmjs.com/package/jsonwebtoken
(**Data de consulta**: gener 2026)
Utilitzada per entendre autenticació amb JWT

[10] express-validator Documentation
https://express-validator.github.io
(**Data de consulta**: gener 2026)
Utilitzada per a la validació de formularis al backend

[11] axios Documentation
http://axios.rest/
(**Data de consulta:** abril 2026)
Utilitzada per fer peticions

[12] Apunts i materials del cicle SMX
(**Data de consulta**: tot el curs)
Institut Puig Castellar Materials proporcionats pel professorat durant el curs, utilitzats com a base teòrica del projecte.

La selecció d'aquestes fonts s'ha fet prioritzant documentació oficial i recursos fiables, amb l'objectiu de garantir la qualitat i veracitat de la informació utilitzada.

7 Annexos

Els annexos inclouen documentació complementària que amplia la informació del projecte i aporta una visió més detallada del desenvolupament realitzat. Aquesta informació no s'inclou al cos principal per evitar sobrecarregar la lectura, però resulta clau per entendre el treball complet.

Annex 1: Estructura del repositori (GitHub)
Inclou l'organització de carpetes i fitxers del projecte. Es mostra la separació entre frontend, backend i altres recursos, facilitant la comprensió de l'estructura del codi i el seu manteniment.

Annex 2: Disseny de la interfície (Figma)
Recull els wireframes i prototips inicials de la interfície. Permet visualitzar l'estructura de la pàgina web abans de la seva implementació, incloent pantalles com registre, login i visualització de dades.

Annex 3: Model de la base de dades
Inclou un esquema conceptual de les taules (usuaris, prediccions, etc.) i les seves relacions. Aquest annex ajuda a entendre com s'organitza la informació i com es gestiona dins del sistema.

Annex 4: Proves tècniques realitzades
Documenta les proves efectuades durant el desenvolupament, com la connexió amb l'API externa, la validació del sistema d'autenticació i el funcionament dels formularis.

Annex 5: Manual d'usuari
Explica de manera clara i senzilla com utilitzar l'aplicació: registre, inici de sessió, consulta d'estadístiques i introducció de prediccions.

Annex 6: Enllaç al repositori del projecte
Inclou l'accés al repositori de GitHub amb el codi font complet, amb permisos de visualització per al professor. Aquest annex permet verificar el desenvolupament real del projecte.

Annex 7: Possibles ampliacions futures (opcional)
Recull idees de millora i evolució del projecte, com la implementació de més funcionalitats, millores de seguretat o optimització del rendiment.