



This is the **published version** of the bachelor thesis:

Hernández Cortacans, Àlex. *Implementació de bases de dades biomèdiques amb GitHub (III)*. Treball de Final de Grau (Universitat Autònoma de Barcelona), 2026 (Enginyeria Informàtica)

This version is available at <https://ddd.uab.cat/record/334442>

under the terms of the  license.

Implementació de bases de dades biomèdiques amb GitHub (III)

Alex Hernandez Cortacans

30 de juny de 2026

Resum

Aquest treball continua el desenvolupament de ReCollecTF, una aplicació web sense servidor que recrea la base de dades CollecTF per evitar que les dades de recerca desapareguin per manca de finançament, utilitzant GitHub Pages com a plataforma d'allotjament gratuïta i sostenible.

L'objectiu principal ha estat deixar l'aplicació en un estat operatiu que permeti la consulta íntegra de les dades per part de serveis externs com UniProt, mitjançant la generació de pàgines estàtiques. Paral·lelament, s'ha treballat per completar i corregir el procés de curació de dades, resolent errors preexistents com la generació incorrecta de taxonomies i les inconsistències en el format de les dades inserides.

Finalment, s'ha analitzat la dependència de serveis externs per millorar la robustesa i la sostenibilitat del projecte a llarg termini, avaluant els riscos de seguretat associats i reduint dependències innecessàries que podien comprometre el funcionament de l'aplicació.

Paraules clau

sense servidor, gratuït, curació, emmagatzematge, base de dades, aplicació web, minimitzar dependències, consulta de dades, pàgines estàtiques, factors de transcripció.

Abstract

This work continues the development of ReCollecTF, a serverless web application that replicates the CollecTF database to prevent research data from disappearing due to lack of funding, using GitHub Pages as a free and sustainable hosting platform.

The main objective has been to leave the application in an operational state that allows external services such as UniProt to fully query the data through the generation of static pages. Concurrently, work has been done to complete and correct the data curation pipeline, resolving pre-existing errors such as incorrect taxonomy generation and inconsistencies in the format of inserted data.

Finally, the dependency on external services has been analysed to improve the robustness and long-term sustainability of the project, assessing associated security risks and reducing unnecessary dependencies that could compromise the application's functionality.

Keywords serverless, GitHub Pages, biological databases, transcription factors, curation pipeline, static web application, data preservation, interoperability, GitHub Actions.



1 INTRODUCCIÓ - CONTEXT DEL TREBALL

MÚLTIPLES grups de recerca genòmica produeixen bases de dades fruit de la investigació que porten a terme. L'accés a aquestes depèn de finançament per poder mantenir-les, això provoca que, un cop acabat el finançament del projecte deixin de ser accessibles ja que no es pot pagar el preu de manteniment dels servidors que les

- E-mail de contacte: alex_cortacans@hotmail.com
- Menció realitzada: Tecnologies de la Informació
- Treball tutoritzat per: Ivan Erill Sagales (Departament d'Enginyeria de la Informació i de les Comunicacions) i Maria Carmen de Toro Valdivia (Departament d'Enginyeria de la Informació i de les Comunicacions)
- Curs 2025/26

allotgen. Aquest fenomen es conegut com l'efecte *tomba de dades*. Segons un estudi realitzat per H. J. Imker [1] només un 25% de les bases de dades creades en els darrers 25 anys continuen actives, com es pot veure a la Figura 1 realitzada per T. K. Attwood [2].



Fig. 1: Estat de 326 bases de dades bioinformatiques 18 mesos després de la seva creació.

Una d'aquestes bases de dades es CollecTF [3], desenvolupada a l'ErillLab, a la universitat de Maryland [4] amb el propòsit d'emmagatzemar dades genètiques i de donar un servei de consulta d'aquestes dades a moltes pàgines bioquímiques com UniProt [5], Meme [6]... entre d'altres, CollecTF té problemes per mantenir les dades accessibles degut a la manca de finançament del projecte.

A causa d'aquestes limitacions econòmiques, es van començar a buscar alternatives que permetessin mantenir l'accés lliure tant a les dades com al projecte. Per trobar la millor solució es va dur a terme una recerca exhaustiva d'alternatives de desplegament sense cost, de la qual [7] [8] es va trobar que GitHub Pages era la plataforma més adequada per allotjar el projecte, emmagatzemar la base de dades i desplegar un portal web per accedir i manipular aquestes dades de manera gratuïta i sostenible en el temps.

Producte d'aquesta recerca es decideix recrear la base de dades d'emmagatzemament de llocs d'unió de factors de transcripció bacterians anomenada CollecTF[3] utilitzant aquestes tecnologies amb el nou nom de ReCollecTF. L'aplicació resultant es troba accessible a <https://collectf.org/>

En el Treball de Fi de Grau de l'Eloi Milego [7], es va prendre la decisió tecnològica d'utilitzar GitHub Pages i GitHub Actions com a infraestructura de l'aplicació, demostrant que era possible implementar mecanismes de lectura i escriptura aprofitant les eines proporcionades per GitHub i el seu ecosistema. En aquest treball es va estudiar i validar que GitHub Pages podia utilitzar-se per mantenir una aplicació web que permetés accedir a la base de dades, emmagatzemant-la dins del mateix repositori de GitHub com a fitxer SQLite.

Els resultats obtinguts van demostrar que aquesta aproximació era viable, permetent la migració de la base de dades de CollecTF [3] al sistema gestor SQLite[9] i el seu emmagatzematge com a fitxer dins de GitHub. A més, es va desenvolupar una aplicació web amb funcionalitats de cerca i consulta sobre les dades obtingudes de la migració,

demostrant que era possible oferir accés funcional a la informació de CollecTF mitjançant les eines proporcionades per l'ecosistema de GitHub.

Al 2026, l'estudiant Miguel Gallego continua aquest projecte[8] utilitzant les conclusions i l'aplicació web realitzada per l'Eloi Milego amb l'objectiu de crear un *pipeline* ordenat per a la introducció de dades dins d'aquesta base de dades seguint les especificacions establertes pel procés de curació de les dades.

Aquest tercer treball continua la feina de l'Eloi Milego i el Miguel Gallego amb l'objectiu d'ampliar la funcionalitat de l'aplicació per permetre la integració amb diferents serveis de bases de dades bioquímiques, com UniProt[5], EBI[10] i PubMed[11], facilitant així l'accés i consulta de dades emmagatzemades a ReCollecTF. Paral·lelament, es busca millorar la robustesa del sistema resolent errors i limitacions actuals, així com analitzar la dependència de serveis externs per reduir possibles riscos i optimitzar la seguretat i mantenibilitat de l'aplicació, mantenint el conjunt tecnològic ja definit al projecte de l'Eloi Milego

Aquest document s'estructura de la següent manera. La Secció 2 presenta els objectius del treball. La Secció 3 descriu l'estat de l'art i el context tecnològic heretat. La Secció 4 exposa la metodologia i la planificació seguides. La Secció 5 detalla el desenvolupament realitzat, mentre que la Secció 6 presenta els resultats obtinguts. Finalment, la Secció 7 recull les conclusions finals.

2 OBJECTIUS

Aquest TFG continua la feina feta pels anteriors TFG's[7][8] i té com objectius:

- Deixar ReCollecTF en un estat operatiu que permeti la consulta íntegra de les dades de curació per part dels múltiples serveis que actualment utilitzen CollecTF com a font d'informació, suplint així la funcionalitat original mitjançant la generació de pàgines estàtiques que contenen aquesta informació.
- Solucionar problemes de funcionalitats en l'aplicació ja existents a l'inici del projecte:
 - Errors intermitents de CORS durant la curació.
 - Introducció de dades a la base de dades incorrecta o inconsistent en acabar una curació.
 - Incorrecta generació de l'arbre de taxonomies al fer una curació.
- Fer un anàlisi dels riscos que comporten les dependències en l'estat inicial del projecte proposant solucions a aquests.

Aquests objectius es volen assolir tenint en ment que posteriorment es pugui modificar el projecte en posteriors TFG's, per tant, es dissenyarà sempre les funcionalitats posant per davant la modularitat i fàcil lectura del codi per a ser fàcilment entès i d'extendre a futurs TFG's.

3 ESTAT DE L'ART

Tota la base tecnològica i l'arquitectura de la qual parteix aquest projecte ve donada pels Treballs de Fi de Grau de l'Eloi Milego [7] i el Miguel Gallego [8]. S'ha demostrat amb els TFG's anteriors que l'arquitectura és eficient i encaixa amb les necessitats del projecte, per tant, es manté en aquest TFG aquesta arquitectura. En aquest estat de l'art s'exposarà quina és aquesta base tecnològica i d'arquitectura heretada, així com els problemes preexistents detectats a l'inici d'aquest treball.

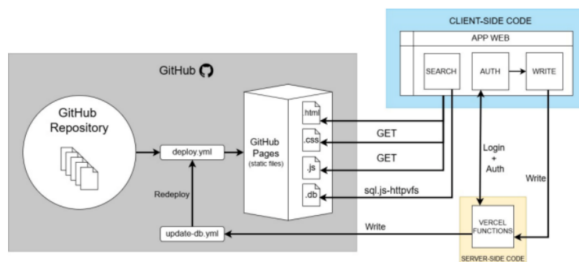


Fig. 2: Arquitectura del projecte definida per l'Eloi Milego.

L'arquitectura del projecte es basa en tres components principals que interactuen entre ells, com es pot veure a la Figura 2: un repositori de GitHub, *server functions* de Vercel [12] i una aplicació web. GitHub Pages s'utilitza per servir l'aplicació web ReCollecTF, la base de dades SQLite i els fitxers estàtics amb dades genòmiques accessibles des de tercers. Dins del repositori s'emmagatzema el fitxer de la base de dades SQLite, el codi font de l'aplicació web, les pàgines HTML generades per a cada factor de transcripció i els arxius CSS i JavaScript que componen el front-end. GitHub Actions s'encarrega d'orquestrar les operacions sobre el repositori mitjançant dos *workflows*:

- *deploy.yml*: Compila el projecte i el publica a GitHub Pages quan es detecta un canvi al repositori.
- *update-db-and-create-static-page.yml*: Executa les sentències SQL sobre la base de dades quan s'introdueix una nova curació i genera la pàgina estàtica HTML corresponent al factor de transcripció curat.

El *frontEnd* s'executa íntegrament al navegador de l'usuari i es comunica amb GitHub Pages mitjançant peticions GET per obtenir els fitxers estàtics i la base de dades. Aquesta base de dades es carrega localment al navegador a través de la biblioteca *sql.js-httpvifs*, que permet fer consultes SQL directament al client sense necessitat d'un servidor de base de dades. Per a les operacions d'escriptura, el front-end es comunica amb les *server functions* de Vercel:

- *logincallback.ts*: Gestiona l'autenticació dels usuaris mitjançant OAuth de GitHub [13].
- *create-pending-sql.ts*: Rep les dades d'una nova curació, crea un fitxer SQL temporal al repositori i dispara el workflow *update-db-and-create-static-page.yml*.
- *update-db-and-create-static-page.ts*: Rep el contingut HTML de la nova pàgina estàtica i dispara el workflow

update-db-and-create-static-page.yml per publicar-la al repositori.

El projecte fa servir una arquitectura de Single Page Application mitjançant el framework React [14] i TypeScript [15] per al front-end, NodeJS [16] per a la creació de pàgines estàtiques en mode manual, Vite [17] per al servidor local de desenvolupament, SQLite [9] per a la base de dades, TypeScript per les *server functions* de Vercel [12] i YAML [18] amb bash per les GitHub Actions. En tot moment s'ha prioritzat el fàcil manteniment i la llegibilitat del codi per facilitar futures modificacions al projecte.

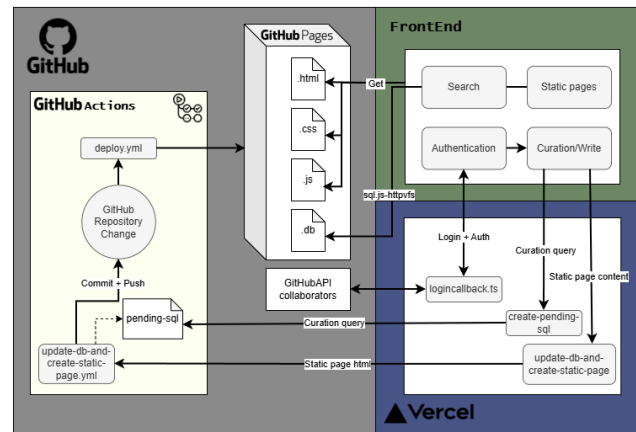


Fig. 3: Arquitectura del projecte actual.

4 METODOLOGIA I PLANIFICACIÓ

Per al desenvolupament d'aquest treball s'ha utilitzat una metodologia àgil basada en l'assignació de tasques concretes i ben definides, amb reunions periòdiques de seguiment. Tot i no seguir estrictament cap marc de treball específic, s'han utilitzat els principis de desenvolupament de *software Agile*, fent cicles iteratius de treball amb presentacions al client amb els avanços i recolecció de *feedback*.

La planificació s'ha estructurat en tasques independents que permetien avançar de manera paral·lela quan era possible i adaptar-se als canvis que anaven sorgint durant el desenvolupament.

4.1 Planificació de tasques

El projecte s'ha dividit en set tasques principals, cadascuna amb una data d'inici i una data de finalització estimada. La Taula 1 mostra la distribució temporal de les tasques realitzades.

A continuació es descriu breument el contingut de cada una de les tasques planificades:

- **Habitució al projecte (3/04/26 - 22/04/26)**: Comprensió de l'arquitectura del projecte, familiarització amb la base de codi existent i anàlisi de l'estat inicial de l'aplicació per identificar les necessitats i problemes a resoldre.
- **Anàlisi de seguretat de Vercel (23/04/26 - 1/05/26)**: Estudi de la dependència amb Vercel per a l'autenticació d'usuaris, avaluació de riscos de seguretat associats i determinació de la viabilitat de mantenir o eliminar aquesta dependència externa.

- **Eliminació de proxies CORS (4/05/26 - 12/05/26):** Investigació de l'ús de proxies de tercers en les consultes a APIs externes, desenvolupament d'una alternativa que realitzés consultes directes i eliminació de la dependència sobre serveis de proxy.
- **Generació de pàgines estàtiques (13/05/26 - 2/06/26):** Implementació del sistema de generació de pàgines HTML estàtiques per a factors de transcripció, tant en mode manual com en mode automàtic, per permetre la consulta des de serveis externs.
- **Correcció de la taxonomia (3/06/26 - 11/06/26):** Correcció de l'error que impedia generar correctament l'arbre taxonòmic complet durant el procés de curació, modificant el codi per emmagatzemar tota la cadena jeràrquica des del node arrel fins a l'espècie.
- **Integritat del pipeline de curació (12/06/26 - 22/06/26):** Correcció de les inconsistències en el format de les dades inserides, normalització dels camps afectats i proves exhaustives del flux complet de curació.
- **Reestructuració del codi i optimització de workflows (23/06/26 - 26/06/26):** Refactorització de l'estructura del projecte per integrar el pipeline de curació dins de l'aplicació principal, eliminació de configuracions duplicades i optimització de les *server functions* de Vercel per evitar limitacions de mida en els inputs.

4.2 Seguiment i comunicació

Per garantir un avanç adequat del projecte i rebre orientació sobre les decisions tècniques, s'han realitzat reunions de seguiment setmanals amb els tutors del Treball de Fi de Grau, Ivan Erill Sagales i Maria Carmen de Toro Valdivia. En aquestes reunions es presentava l'estat d'avanç de les tasques, es discutien les necessitats del projecte, es resolien dubtes sobre l'arquitectura i el desenvolupament, i es rebia *feedback* per ajustar la planificació o redirigir l'enfocament de les tasques pendents quan era necessari.

Aquesta dinàmica de treball ha permès mantenir un ritme constant de desenvolupament, adaptant-se a les dificultats tècniques que anaven sorgint i assegurant que les decisions preses estiguessin alineades amb els objectius generals del projecte.

5 DESENVOLUPAMENT

5.1 Anàlisi de requisits

Abans de començar a implementar cap canvi, es va dur a terme una anàlisi exhaustiu de l'estat de l'aplicació per comprendre com estava construïda i detectar les necessitats i problemes existents.

L'aplicació heretada constava d'un *pipeline* de curació de set passos on l'usuari introdueix progressivament les dades d'un factor de transcripció: la publicació associada, el TF i el seu organisme, les tècniques experimentals utilitzades, els llocs d'unió al genoma, l'anotació de cada lloc, els gens regulats i finalment la informació de curació. En completar

el procés, el sistema genera les sentències SQL corresponents i les envia a les *server functions* de Vercel, que actualitzen la base de dades i generen la pàgina estàtica del factor de transcripció al repositori.

Durant l'anàlisi es van identificar els següents problemes: errors de CORS en les consultes a PubMed resolts amb proxies de tercers, taxonomies que només emmagatzemaven el node terminal sense la cadena jeràrquica completa, inconsistències en el format de les dades inserides, un subprojecte independent amb configuracions duplicades dins del repositori principal, i una limitació de mida en els inputs de les *server functions* de Vercel que impedia enviar en una sola crida el contingut SQL i l'HTML d'una curació completa.

5.2 Proxies

Una de les contribucions realitzades ha estat l'eliminació de l'ús de proxies en les consultes a APIs externes al pas 1 del *pipeline* de curació. El codi original realitzava les peticions a través de serveis de proxy intermediaris quan la consulta directa a l'API fallava a causa d'errors de CORS. Un error de CORS (Cross-Origin Resource Sharing) es produeix quan un navegador bloqueja una petició HTTP realitzada des d'un origen diferent al del servidor que hostatja el recurs, a no ser que aquest servidor ho autoritzi explícitament mitjançant les capçaleres CORS adequades.

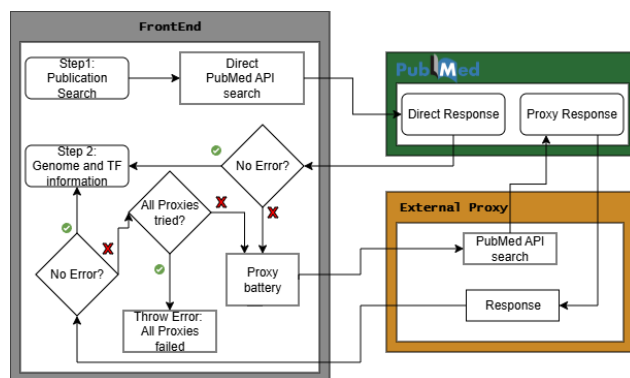


Fig. 4: Diagrama de Flux de l'antic *fetch*.

En el context d'aquest projecte, les consultes s'executen des del navegador de l'usuari, atès que l'aplicació es desplega a GitHub Pages i el codi JavaScript s'executa en el client. Per tant, les restriccions de CORS són aplicables i el codi original gestionava aquestes situacions emprant proxies de tercers per a evitar el bloqueig del navegador. Aquesta aproximació, però, introduïa una dependència externa sobre serveis de proxy que no estaven sota el control del projecte, reduint la resistència de l'aplicació davant de canvis o interrupcions en aquests serveis que podien inutilitzar la funcionalitat de cerca de publicacions. L'eliminació d'aquesta dependència s'ha abordat mitjançant la creació d'una nova funció de consulta la API de PubMed[11], crida directament la API sense passar per tercers o la de CrossRef si no ha retornat cap resultat.

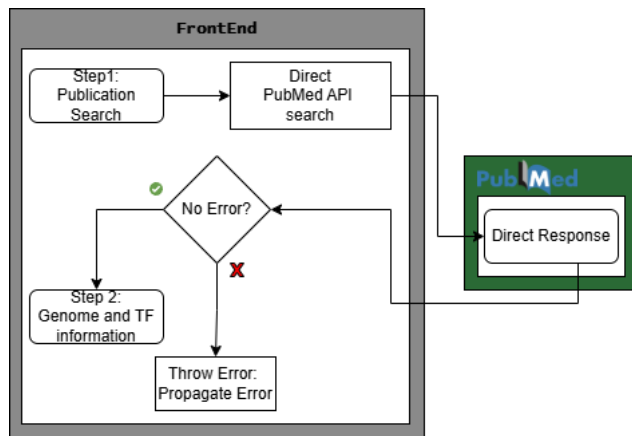


Fig. 5: Diagrama de Flux del nou fetch.

Per verificar que les consultes directes funcionaven correctament sense necessitat de proxies, es va desenvolupar una petita aplicació de prova desplegada a GitHub Pages que realitzava consultes directes a l'API de PubMed. Mitjançant una bateria de deu consultes consecutives automatitzades, es va comprovar que totes retornaven els resultats correctament en tots els casos, sense necessitat d'intermediaris. Un cop confirmada la hipòtesi, es va substituir el codi original que gestionava les consultes a APIs externes amb proxies de tercers per una crida directe a la API de PubMed[11], eliminant així la dependència sobre serveis externs i fent l'aplicació més autònoma i robusta davant possibles canvis futurs en serveis de tercers.

5.3 Anàlisi de seguretat de Vercel

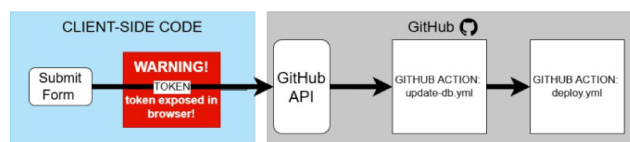


Fig. 6: Seguretat de Vercel.

Es va dur a terme una anàlisi de seguretat de la dependència amb Vercel[12] per a l'autenticació amb GitHub, amb el propòsit d'avaluar els riscos associats i determinar si era possible prescindir d'aquesta dependència externa. L'anàlisi es va centrar principalment en la cookie de sessió que transporta el token d'autenticació amb GitHub, avaluant el seu temps de vida i les seves característiques de seguretat per determinar els vectors d'atac reals de l'aplicació.

Pel que fa a les característiques de seguretat de la implementació, aquesta cookie té un temps de vida de vuit hores, cosa que limita la finestra d'oportunitat per a un possible atac. A més, la cookie és de tipus *HttpOnly*, la qual cosa impedeix que pugui ser llegida directament per codi JavaScript al navegador. Finalment, GitHub força l'ús de HTTPS en el procés d'autenticació, de manera que el contingut de les capçaleres viatja xifrat a nivell de transport i no pot ser interceptat en text pla.

No obstant això, s'han identificat tres vectors d'atac possibles. El primer és un atac de tipus *Cross-Site Request*

Forgery (CSRF), en el qual un atacant extern podria crear una pàgina web maliciosa que realitzés peticions autenticades als endpoints de Vercel en nom de l'usuari. A diferència d'un atac XSS, l'atacant no necessita llegir la cookie directament, ja que el navegador l'adjunta automàticament a qualsevol petició dirigida al domini de Vercel independentment del seu origen. Un atac CSRF exitós podria disparar els *workflows* de GitHub Actions o modificar el contingut del repositori sense el coneixement de l'usuari, durant la finestra de vuit hores de vida de la cookie. La protecció estàndard contra aquest tipus d'atac consisteix a verificar la capçalera *Origin* o *Referer* a les *server functions*, o bé a incorporar un token CSRF addicional a cada petició. El segon és un atac de tipus *man-in-the-middle*, en el qual un atacant amb accés a la xarxa podria intentar desxifrar els paquets transmesos per HTTPS per obtenir la cookie; aquest atac es considera poc viable donat el temps computacional necessari per desxifrar el trànsit i el curt temps de vida de la cookie. El tercer vector és la presència de *malware* al navegador de l'usuari, com ara extensions malicioses que podrien llegir i exfiltrar la cookie de sessió.

El risc principal en cas d'èxit d'un d'aquests atacs és la compromissió total del compte de GitHub de l'usuari autenticat, atès que l'atacant podria tenir accés complet per llegir, modificar, crear i eliminar qualsevol contingut del compte. Com a possible solució, tot i que el vector d'atac és reduït i la finestra temporal molt limitada, es recomana que els usuaris de la plataforma utilitzin comptes de GitHub no vinculats a projectes professionals o personals d'importància, de manera que, en el supòsit que un atacant aconseguís accedir a la cookie durant el temps d'oportunitat, l'impacte real sobre el patrimoni digital de l'usuari quedés completament limitat.

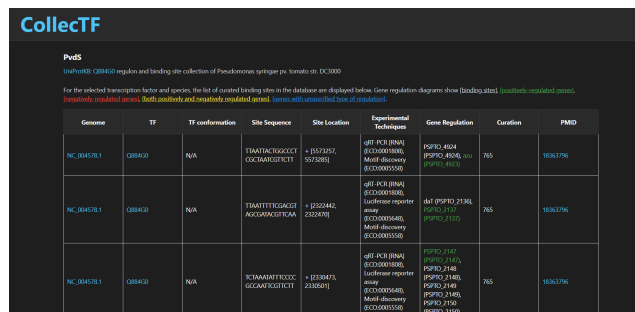
5.4 Generació de pàgines estàtiques

S'ha desenvolupat un sistema de generació de pàgines estàtiques amb informació sobre factors de transcripció específics llistats en el fitxer *dbxref.txt*, fitxer que té una entrada per cada identificador intern anomenat *expressionId* que representa un factor de transcripció amb el seu identificador corresponent a UniProt[5]. Aquest sistema pot ser cridat a partir de dos modalitats:

- Manual - Cridat manualment per un desenvolupador del projecte amb el propòsit d'actualitzar totes les pàgines estàtiques llistades al fitxer d'equivalències entre els identificadors interns de la base de dades i els que utilitza UniProt per identificar factors de transcripció. Crea una pàgina estàtica per cada entrada d'aquest fitxer. Per pujar els canvis al projecte s'ha de fer un commit i push al repositori de GitHub
- Automàtic - Cridat pel *pipeline* de curació en acabar una entrada de dades. Crea una sola pàgina depenent de si la informació ja està llistada al arxiu de equivalències i si es experimentalment verificada. Un cop executat el workflow correctament els canvis es pugen automàticament al projecte.

5.4.1 Contingut de les pàgines estàtiques

La funció principal rep l'identificador públic anomenat *expressionId*, consulta la base de dades local per obtenir les dades del factor de transcripció i els seus llocs d'unió, després genera el contingut HTML de la pàgina estàtica que conté Fig. 7:



Gene	TF	TF confirmation	Site Sequence	Site Location	Experimental Techniques	Gene Regulation	Curation	PMID
NC_044738.1	CBF10	N/A	TAAATCTGGCTT GGATGCTCTT	+ (327355, 307586)	ChIP-Seq (RNAi) ECOD0001895, Motif discovery ECOD0001908	PSF0_004 PSF0_004, 001 PSF0_001, 002	765	1683796
NC_044738.1	CBF10	N/A	TAAATCTGGCTT ACCAAACTTAA	+ (327346, 323878)	ChIP-Seq (RNAi) ECOD0001895, Lifesense reporter assay ECOD0001948, Motif discovery ECOD0001908	def (PSF0_2146) PSF0_2147 PSF0_2135	765	1683796
NC_044738.1	CBF10	N/A	CTCAATTTCTCC GGAAATCTCTT	+ (323847, 323849)	ChIP-Seq (RNAi) ECOD0001895, Lifesense reporter assay ECOD0001948, Motif discovery ECOD0001908	PSF0_2147 PSF0_2145 PSF0_2146 PSF0_2148 PSF0_2149 PSF0_2150 PSF0_2158	765	1683796

Fig. 7: Exemple de pàgina estàtica generada.

Una taula que inclou el genoma, el factor de transcripció, la conformació, la seqüència del lloc, la localització, les tècniques experimentals, la regulació gènica, l'identificador de curació i el PMID[11]. La regulació gènica es codifica amb colors segons el tipus de regulació: verd per a regulació positiva (ACT), vermell per a regulació negativa (REP), groc per a regulació dual (DUAL) i blau per a regulació no especificada (N/A).

Les pàgines generades s'emmagatzemen al directori *public/{expressionId}/index.html* del repositori i es serveixen com a pàgines estàtiques a través de GitHub Pages.

5.4.2 API serverless a Vercel

Seguint l'arquitectura proposada al TFG d'Eloi Milego, per implementar les operacions que requereixen l'execució de lògica de servidor, s'han creat *server functions* desplegades a Vercel.

Aquestes funcions actuen com a capa intermèdia entre el client, GitHub i la resta de serveis externs. El seu principal objectiu és executar operacions sensibles, com ara la validació de l'autenticació dels usuaris, la gestió de credencials o la invocació dels *workflows* de GitHub Actions, evitant que informació confidencial, com ara els *Personal Access Tokens* de GitHub, quedi exposada al navegador.

Per controlar les accions que cada usuari pot realitzar sobre l'aplicació, les funcions utilitzen un mecanisme d'autenticació basat en *JSON Web Tokens* (JWT[19]). Un JWT[19] és un testimoni signat digitalment que conté informació sobre la identitat de l'usuari i que permet verificar-ne l'autenticitat sense necessitat de consultar una base de dades en cada petició. Aquest token s'emmagatzema en una *cookie HTTP-only*, de manera que només és accessible des de les funcions desplegades a Vercel[12] i no des del codi JavaScript executat al navegador, reduint així el risc de robatori del token.

Cada vegada que el client sol·licita una operació que modifica el contingut de ReCollectTF, la corresponent *server function* valida el JWT[19], comprova que l'usuari

disposa dels permisos necessaris i, únicament en cas afirmatiu, executa les accions corresponents sobre el repositori, com ara disparar els *workflows* encarregats d'actualitzar la base de dades o generar noves pàgines estàtiques. D'aquesta manera, Vercel[12] no només proporciona la infraestructura d'execució de les funcions, sinó que també constitueix el punt de control encarregat de limitar les operacions que els usuaris autenticats poden realitzar sobre l'aplicació.

Aquesta aproximació permet mantenir l'arquitectura principal de ReCollectTF completament estàtica, evitant la necessitat de desplegar i administrar un servidor dedicat, alhora que proporciona un entorn segur per a les operacions que no poden executar-se directament des del client. Les funcions implementades són les següents:

- *create-pending-sql.ts* - Funció disparada pel pas final de la curació que rep la query SQL amb la informació a afegir a la base de dades i el string *sqlPath* que conté el nom de l'arxiu a crear. El propòsit de la funció és crear un arxiu *sqlPath.sql* al repositori amb la informació corresponent a la curació realitzada.
- *update-db-and-create-static-page.ts* - Funció disparada pel pas final de curació, aquesta funció rep:
 - *htmlContent*: Contingut HTML generat pel *pipeline* de curació per posar a la pàgina estàtica a generar.
 - *uniprotAccession*: Identificador d'UniProt[5] del factor de transcripció.
 - *expressionId*: Identificador públic del factor de transcripció.
 - *expressionInfo*: Booleà que indica si hi ha informació experimentalment verificada o no a la curació.

Amb aquests *inputs*, codifica el contingut HTML en base64 i dispara el *workflow update-db-and-create-static-page.yml* de GitHub Actions enviant els *inputs* que ha rebut.

5.4.3 Integració amb GitHub Actions

S'han realitzat contribucions sobre els *workflows* de GitHub Actions del repositori. S'ha creat el *workflow update-db-and-create-page.yml*, que rep les variables que li ha enviat la funció de Vercel[12] *update-db-and-create-static-page.ts*.

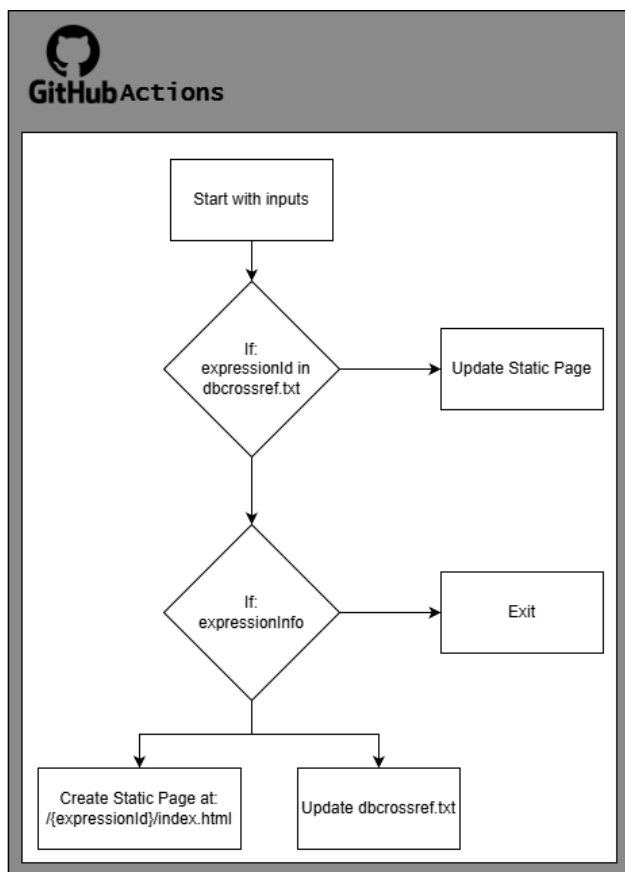


Fig. 8: Diagrama de flux de la creació de pàgines estàtiques automàtica al workflow de *update-db-and-create-static-page.yml*.

El comportament de la funció es el següent:

1. Comproba que l'arxiu *pendingSQL.sql* existeix al repositori fent *git pulls* del repositori cada 3 segons fins trobar l'arxiu o arribar als 6 minuts, on retorna error.
2. Un cop ha trobat l'arxiu realitza una actualització de l'arxiu que conté la base de dades de la aplicació, si aquest procés falla, retorna un error.
3. Si l'actualització de la base de dades ha acabat satisfactòriament, comença el procés de creació de la pàgina estàtica i actualització del fitxer amb les equivalències de identificadors interns amb els identificadors d'UniProt[5] seguint el diagrama de flux de la Figura.8

5.4.4 Generació automàtica en detall

Un cop l'usuari ha completat tots els passos de curació, el sistema genera automàticament la pàgina HTML estàtica del factor de transcripció a partir de les dades introduïdes. Per fer-ho, s'ha implementat una funció que transforma les dades del context de curació al format esperat pel sistema de generació de pàgines estàtiques, combinant les dades noves introduïdes pel curador amb les dades ja existents a la base de dades pel mateix factor de transcripció. Paral·lelament, les sentències SQL generades a partir de la curació s'envien a la *server function create-pending-sql.ts*, que crea el fitxer SQL temporal al repositori de GitHub. Un cop generat el contingut de la pàgina i obtinguda la ruta del fitxer SQL, ambdós es comuniquen a la *server function update-db-and-create-static-page.ts*, que dispara el workflow de

GitHub Actions *update-db-and-create-page.yml* per executar les sentències SQL sobre la base de dades i publicar la nova pàgina estàtica al repositori.

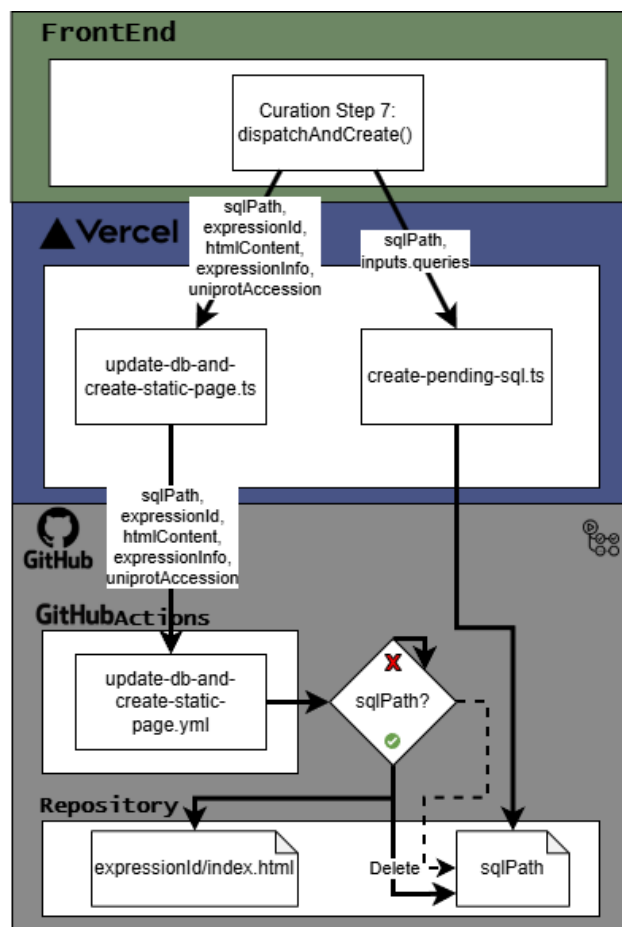


Fig. 9: Diagrama de flux de la creació de pàgines estàtiques automàtica.

5.4.5 Generació manual en detall

S'ha desenvolupat un script de generació massiva que permet crear les pàgines estàtiques HTML per a tots els factors de transcripció existents a la base de dades en una sola execució. El script obté primer la llista completa d'identificadors de factors de transcripció interns de la base de dades, converteix cadascun al format públic corresponent i crida la funció per generar el contingut del document HTML. Cada pàgina generada correctament s'emmagatzema al directori *public/{expressionId}/index.html* i l'identificador s'afegeix a un conjunt d'entrades exitoses. En cas que la generació d'una pàgina falli, l'error es registra per consola i el procés continua amb el següent factor de transcripció sense interrompre l'execució global. Un cop finalitzada la generació de totes les pàgines, el script crida *generateUniProtDbXRef* passant-li únicament el conjunt d'identificadors generats correctament, garantint que el fitxer *uniprot.dbxref.txt* només contingui entrades per a les quals existeix una pàgina estàtica vàlida al repositori.

5.5 Refactorització del pipeline de curació

Durant l'anàlisi inicial es va detectar que el pipeline de curació desenvolupat per Miguel Gallego[8] havia estat integrat al repositori principal com un subprojecte independent,

amb la seva pròpia estructura de directoris, fitxers de configuració i dependències duplicades. Es va procedir a refactoritzar la base de codi, eliminant fitxers redundants, unificant les dependències compartides i integrant el *pipeline* dins l'estructura principal de l'aplicació, simplificant així l'arquitectura i facilitant el manteniment futur.

5.6 Taxonomia

Durant les proves d'inserció es va detectar que el *pipeline* de curació no generava correctament l'arbre taxonòmic complet dels organismes introduïts. En el pas 2 de la curació, l'usuari introdueix l'identificador del genoma de l'organisme d'estudi de NCBI[20], a partir del qual el sistema hauria de construir la jerarquia taxonòmica completa des del node arrel fins a l'espècie, és a dir, des del nivell de *phylum* fins a *species* passant per tots els nivells intermedis com *filum*, *classe*, *ordre*, *família*, *gènere* i *espècie*.

El comportament observat era que el sistema únicament emmagatzemava el node taxonòmic de l'organisme directament introduït per l'usuari, és a dir, el taxó de l'espècie concreta, sense recórrer ni emmagatzemar cap dels seus nodes pare dins de la jerarquia. Això provocava que a la base de dades quedés un node taxonòmic aïllat sense cap relació amb la resta de l'arbre, incomplet i sense el context jeràrquic necessari per fer consultes taxonòmiques correctes.

La causa del problema era que el codi del pas 2 de la curació consultava l'API d'NCBI[20] per obtenir la informació taxonòmica de l'organisme, però únicament processava i emmagatzemava el node terminal de la resposta, ignorant la cadena de nodes pare que l'API retorna com a part de la jerarquia completa. En conseqüència, el SQL generat al pas 7 del *pipeline* només contenia la sentència d'inserció per al node de l'espècie, sense les insercions corresponents als nodes superiors de la jerarquia ni les relacions de parentiu entre ells.

Per solucionar-ho es va modificar el codi encarregat de processar la resposta de l'API taxonòmica per recórrer tota la cadena de nodes retornada i generar les sentències SQL d'inserció per a cadascun d'ells, establint correctament les relacions de parentiu entre nodes consecutius de la jerarquia, de manera que la base de dades reflectís l'arbre taxonòmic complet des del node arrel fins a l'espècie de l'organisme introduït.

5.7 Inconsistències en el format de les dades

Durant les proves realitzades sobre el sistema d'introducció de taxonomies es va detectar que alguns dels camps generats pel *pipeline* de curació no seguien el format utilitzat per la base de dades. Aquest problema afectava, entre d'altres, als camps corresponents al tipus i a la funció dels factors de transcripció, tal com es mostra a la Figura 10.

A-Z TF_function ▼	A-Z TF_type ▼
DUAL	TETRAMER
REP	DIMER
ACT	MONOMER
DUAL	MONOMER
ACT	TETRAMER
dual	monomer
repressor	monomer
activator	monomer
activator	monomer

Fig. 10: Introducció de dades després del procés de curació.

Les entrades mostrades en minúscules corresponen als valors generats pel *pipeline* de curació abans de la correcció, mentre que les entrades en majúscules representen el format correcte utilitzat a la base de dades.

L'origen del problema era que el *pipeline* emprava directament els valors mostrats a la interfície d'usuari per generar les consultes d'inserció, sense transformar-los al format canònic emprat per la base de dades. Per solucionar-ho, es va incorporar una etapa de normalització que converteix aquests valors al format corresponent abans de generar les consultes SQL, garantint així la consistència de les dades introduïdes.

5.8 Optimització dels workflows

Durant el desenvolupament es va detectar que l'enviament de les dades d'una curació completa a les *server functions* de Vercel superava el límit de mida dels inputs admesos pel pla gratuït de la plataforma. Aquesta limitació impedia enviar en una sola crida tant el contingut SQL generat com el contingut HTML de la pàgina estàtica corresponent.

Per solucionar-ho, s'ha dividit la lògica en dues *server functions* independents que distribueixen la càrrega entre elles. La primera, *create-pending-sql.ts*, s'encarrega exclusivament de rebre les sentències SQL generades pel *pipeline* de curació i crear el fitxer SQL temporal al repositori de GitHub mitjançant l'API de GitHub. La segona, *update-db-and-create-static-page.ts*, rep la ruta del fitxer SQL ja creat, el contingut HTML de la pàgina estàtica i els paràmetres d'identificació del factor de transcripció, i dispara el workflow de GitHub Actions *update-db-and-create-page.yml* que executa les operacions sobre la base de dades i publica la nova pàgina estàtica al repositori. D'aquesta manera, cada funció gestiona una part del procés amb una mida d'entrada que es manté dins dels límits del pla gratuït de Vercel, mantenint la seguretat i el control d'accés en ambdues funcions mitjançant la validació del JWT[19] en cada crida.

6 RESULTATS

6.1 Manteniment de la dependència amb Vercel

Després de dur a terme l'anàlisi de seguretat descrit a la secció de desenvolupament, es va concloure que la dependència amb Vercel havia de mantenir-se. Les alternatives per prescindir-ne haurien requerit exposar informació sensible, com ara els *Personal Access Tokens* de GitHub, directament al codi del client, cosa que hauria suposat un risc de seguretat molt superior al que comporta la solució actual basada en JWT[19]. A més, Vercel proporciona una capa d'autenticació i de control d'accés que no seria possible replicar sense un servidor dedicat, fet incompatible amb l'objectiu de manteniment de cost zero del projecte. Tenint en compte que els vectors d'atac identificats presenten una finestra d'oportunitat molt limitada i que es poden mitigar amb bones pràctiques d'ús, es va determinar que mantenir Vercel[12] era la millor opció per garantir la seguretat i la sostenibilitat de l'aplicació.

6.2 Eliminació dels proxies

Com a resultat de la investigació sobre l'ús de proxies, s'ha implementat una nova funció que realitza les consultes directament a l'API de PubMed sense cap intermediari, eliminant completament la dependència sobre serveis de proxy de tercers. Aquesta nova funció crida directament l'API de PubMed i, en cas que no retorni cap resultat, recorre a CrossRef com a font alternativa. Des de la implantació d'aquesta solució l'error de CORS ha deixat de produir-se en cap dels escenaris de prova, fent l'aplicació més autònoma i robusta davant possibles canvis en serveis externs.

6.3 Generació de pàgines estàtiques

Com a resultat del sistema de generació de pàgines estàtiques implementat, s'han creat amb la modalitat manual un total de 336 pàgines estàtiques corresponents als factors de transcripció existents a la base de dades, cadascuna accessible a través d'una URL pública estable a GitHub Pages. La modalitat automàtica, que genera la pàgina corresponent en acabar una curació, ha estat testejada exhaustivament i funciona correctament en tots els escenaris provats.

6.4 Correcció de les taxonomies

Com a resultat de la correcció implementada al *pipeline* de curació, el sistema ara genera i emmagatzema correctament l'arbre taxonòmic complet per a cada organisme introduït durant una curació. Com es pot observar a la Figura 11, la diferència entre el comportament erroni a l'entrada amb el nom de *Verrucomicrobium spinosum* amb la categoria d'*strain*, on únicament s'emmagatzemava el node de l'espècie de manera aïllada, i el comportament corregit, amb l'arbre corresponent a *Bacteroides thetaiotaomicron* on es genera tota la cadena jeràrquica amb les relacions de parentiu correctes, on l'identificador del pare, corresponent a la última columna referència la primera columna dels taxons corresponents als pares.

Tot i haver generat correctament els arbres de taxonomia nous, falta provar que les dades s'introdueixen correctament en tots els casos en futures proves de funcionalitat.

3	332	818	species	Bacteroides thetaiotaomicron	331
4	331	816	genus	Bacteroides	330
5	330	815	family	Bacteroidaceae	329
6	329	171549	order	Bacteroidales	328
7	328	200643	class	Bacteroidia	327
8	327	976	phylum	Bacteroidota	[NULL]
9	325	240016	strain	Verrucomicrobium spinosum	[NULL]

Fig. 11: Extracte de la taula de taxonomies de la base de dades.

7 ÚS DE LA IA GENERATIVA

En aquest projecte, l'ús de la IA generativa s'ha limitat exclusivament a tasques de suport com la consulta de documentació i la comprensió de conceptes complexos. No s'ha utilitzat per a la generació de codi ni per a la resolució directa de problemes específics del desenvolupament.

Per a la resolució de dubtes tècnics i la cerca d'alternatives d'implementació, s'ha prioritzat la consulta de fonts primàries i especialitzades, com ara la documentació oficial de les llibreries utilitzades i fòrums de programació, especialment Stack Overflow[21]. D'aquesta manera, es garanteix que les solucions adoptades estiguin contrastades per la comunitat i es minimitza el risc de dependre de respostes generades automàticament que puguin contenir errors o manca de context.

En definitiva, l'ús de la IA generativa ha estat testimonial i complementari, sense substituir en cap cas el criteri tècnic ni el procés de presa de decisions.

8 CONCLUSIONS

L'objectiu principal d'aquest treball ha estat deixar Re-CollecTF en un estat operatiu que permeti la consulta íntegra de les dades per part de serveis externs, corregir els errors existents al *pipeline* de curació i analitzar els riscos de les dependències externes. Els resultats obtinguts permeten afirmar que els tres objectius plantejats s'han assolit satisfactòriament.

Pel que fa al primer objectiu, s'ha implementat un sistema complet de generació de pàgines estàtiques HTML per a cada factor de transcripció de la base de dades. S'han generat 336 pàgines accessibles a través d'URLs públiques estables a GitHub Pages, i el *pipeline* de curació les publica automàticament en acabar una nova entrada, garantint que la informació es mantingui actualitzada i disponible per a serveis com UniProt.

Pel que fa al segon objectiu, s'han resolt els tres problemes de funcionalitat identificats a l'inici del projecte. S'ha eliminat la dependència sobre serveis de proxy de tercers en les consultes a l'API de PubMed[11], resolent els errors de CORS. S'ha corregit la generació incompleta de l'arbre taxonòmic, que ara emmagatzema correctament tota la cadena jeràrquica des del fílum fins a l'espècie. I s'han normalitzat les inconsistències en el format de les

dades inserides, com els valors de *TF_function* i *TF_type*, garantint la coherència entre el *pipeline* de curació i la base de dades.

Pel que fa al tercer objectiu, l'anàlisi de seguretat de la dependència amb Vercel[12] ha permès identificar els vectors d'atac possibles sobre la cookie de sessió i concloure que la dependència s'ha de mantenir, ja que les alternatives haurien requerit exposar informació sensible directament al client. Com a mesura de mitigació, es recomana que els usuaris de la plataforma utilitzin comptes de GitHub no vinculats a projectes d'importància.

Com a línies de treball futur, es proposa la incorporació d'un sistema de generació d'imatges per a diagrames de gens regulats, l'ampliació del servei de consulta a altres bases de dades com MEME, i la implementació de proves automatitzades que cobreixin la totalitat del flux de curació per garantir la integritat de les dades en tots els escenaris possibles.

9 AGRAÏMENTS

Vull expressar el meu sincer agraïment als meus tutors, Ivan Erill Sagales i Maria Carmen de Toro Valdivia, pel seu suport constant i per les nombroses reunions de seguiment que han dut a terme al llarg d'aquest projecte. La seva orientació, disponibilitat i *feedback* continu han estat fonamentals per guiar el desenvolupament del treball i assegurar que els objectius es complissin satisfactòriament.

REFERÈNCIES

- [1] H. J. Imker, "25 years of molecular biology databases: A study of proliferation, impact, and maintenance," *Frontiers in Research Metrics and Analytics*, vol. 3, 2018. [Online]. Available: <https://www.researchgate.net/publication/345650858>
- [2] T. K. Attwood, B. Agit, and L. B. M. Ellis, "Longevity of biological databases," *EMBnet.journal*, vol. 21, 2015.
- [3] S. Kiliç, E. R. White, D. M. Sagitova, J. P. Cornish, and I. Erill, "Collectf: a database of experimentally validated transcription factor-binding sites in bacteria," *Nucleic Acids Research*, vol. 42, 2013. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pubmed/24234444>
- [4] ErillLab, "Collectf project," accés: març, 2026. [Online]. Available: <https://erilllab.umbc.edu/>
- [5] UniProt Consortium, "Uniprot: the universal protein knowledgebase," accés: març, 2026. [Online]. Available: <https://www.uniprot.org>
- [6] T. L. Bailey *et al.*, "Meme suite: tools for motif discovery and searching," *Nucleic Acids Research*, accés: abril, 2026. [Online]. Available: <https://meme-suite.org>
- [7] E. M. Miralles, "Implementació de bases de dades biomèdiques amb github (i)," Master's thesis, Universitat Autònoma de Barcelona, Bellaterra, Barcelona, 2025, treball de Fi de Grau en Enginyeria Informàtica.
- [8] M. G. Cabanillas, "Implementació de bases de dades biomèdiques amb github (ii)," Master's thesis, Universitat Autònoma de Barcelona, Bellaterra, Barcelona, 2026, treball de Fi de Grau en Enginyeria Informàtica.
- [9] SQLite Consortium, "Sqlite database engine," accés: abril, 2026. [Online]. Available: <https://www.sqlite.org>
- [10] European Bioinformatics Institute, "Embl-ebi resources," accés: abril, 2026. [Online]. Available: <https://www.ebi.ac.uk>
- [11] National Center for Biotechnology Information, "Pubmed database," accés: abril, 2026. [Online]. Available: <https://pubmed.ncbi.nlm.nih.gov>
- [12] Vercel, "Vercel platform," accés: març, 2026. [Online]. Available: <https://vercel.com>
- [13] GitHub, Inc., "Authorizing oauth apps," accés: abril, 2026. [Online]. Available: <https://docs.github.com/en/apps/oauth-apps/building-oauth-apps/authorizing-oauth-apps>
- [14] Meta, "React – a javascript library for building user interfaces," accés: abril, 2026. [Online]. Available: <https://react.dev>
- [15] Microsoft, "Typescript language," accés: abril, 2026. [Online]. Available: <https://www.typescriptlang.org>
- [16] OpenJS Foundation, "Node.js documentation," accés: abril, 2026. [Online]. Available: <https://nodejs.org/docs/latest/api/>
- [17] E. You, "Vite build tool," accés: abril, 2026. [Online]. Available: <https://vitejs.dev>
- [18] YAML Language Development Team, "Yaml ain't markup language (yaml) version 1.2," accés: abril, 2026. [Online]. Available: <https://yaml.org/>
- [19] Internet Engineering Task Force (IETF), "Rfc 7519: Json web token (jwt)," 2015, accés: abril, 2026. [Online]. Available: <https://datatracker.ietf.org/doc/html/rfc7519>
- [20] National Center for Biotechnology Information, "Ncbi taxonomy," accés: abril, 2026. [Online]. Available: <https://www.ncbi.nlm.nih.gov/taxonomy>
- [21] Stack Overflow, "Stack overflow," accés: abril, 2026. [Online]. Available: <https://stackoverflow.com>
- [22] GitHub, Inc., "Token expiration and revocation," 2026, accés: abril, 2026. [Online]. Available: <https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/token-expiration-and-revocation>
- [23] Mozilla Developer Network (MDN), "Using http cookies," 2026, accés: abril, 2026. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies>

A APÈNDIX

A.1 Secció d'Apèndix

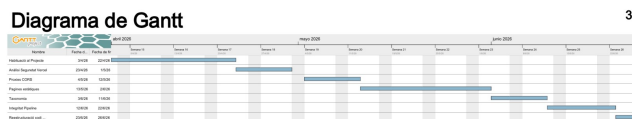


Fig. 12: Diagrama de Gantt del Projecte.

TAULA 1: PLANIFICACIÓ TEMPORAL DE LES TASQUES DEL PROJECTE.

Tasca	Inici	Final
Habituar al projecte	3/04/26	22/04/26
Anàlisi de seguretat de Vercel	23/04/26	1/05/26
Eliminació de proxys CORS	4/05/26	12/05/26
Generació de pàgines estàtiques	13/05/26	2/06/26
Correcció de la taxonomia	3/06/26	11/06/26
Integritat del pipeline de curació	12/06/26	22/06/26
Reestructuració i optimització	23/06/26	26/06/26

A.2 Variables d'entorn a Vercel

Per tal de garantir el funcionament de les *server functions* de Vercel[12], l'aplicació utilitza un conjunt de variables d'entorn que emmagatzemen informació sensible per a l'autenticació i la comunicació entre serveis. Vercel[12] permet definir aquestes variables d'entorn de manera que puguin ser utilitzades per les *server functions* sense que quedin exposades al codi font o al client. Al projecte, s'utilitzen les següents variables:

- **GITHUB_CLIENT_ID**: Identificador públic de l'aplicació OAuth registrada a GitHub. Aquest valor s'obté en crear una OAuth App a la secció de desenvolupador de GitHub i s'utilitza per identificar l'aplicació durant el flux d'autenticació.
- **GITHUB_CLIENT_SECRET**: Clau secreta generada per GitHub juntament amb el *Client ID*. Aquesta clau es fa servir per obtenir un *access token* a canvi del codi d'autorització rebut durant el flux OAuth. Per motius de seguretat, aquesta clau mai no es pot exposar al codi del client.
- **JWT_SECRET**: Clau secreta utilitzada per xifrar i signar els *JSON Web Tokens* (JWT[19]) que l'aplicació utilitza per gestionar les sessions d'usuari. Aquesta clau garanteix que els tokens no puguin ser falsificats i permet a les *server functions* de Vercel[12] verificar l'autenticitat de les peticions sense necessitat de consultar una base de dades.
- **BOT_TOKEN**: Token de l'aplicació de GitHub que s'utilitza per autenticar les peticions a l'API de GitHub des de les *server functions* de Vercel. Aquest token, generat des de la configuració de l'OAuth App, permet

realitzar operacions sobre el repositori com ara crear fitxers, disparar *workflows* de GitHub Actions i gestionar el contingut del projecte en nom de l'aplicació.

- **FRONTEND_URL**: URL de l'aplicació web desplegada a GitHub Pages. S'utilitza a la *server function logincallback.ts* per redirigir l'usuari després de completar el procés d'autenticació amb GitHub, tant si l'autenticació ha estat exitosa com si ha fallat.

Per identificar quines variables d'entorn eren necessàries i quina funció complia cadascuna, va ser necessari realitzar un procés d'enginyeria inversa sobre el codi heretat dels TFG anteriors[7][8], ja que el codi no disposava de documentació sobre la configuració requerida. Es va analitzar el flux d'execució de les *server functions* i les GitHub Actions per determinar quines variables s'utilitzaven i per a què servien. Un cop identificades, es van registrar les seves funcions i es va verificar que estiguessin configurades correctament a Vercel per garantir el funcionament de l'aplicació.

A.3 Anàlisi de seguretat de Vercel

Es va dur a terme una anàlisi de seguretat de la dependència amb Vercel[12] per a l'autenticació amb GitHub, amb el propòsit d'avaluar els riscos associats i determinar si era possible prescindir d'aquesta dependència externa. L'anàlisi es va centrar principalment en la cookie de sessió que transporta el token d'autenticació amb GitHub, avaluant el seu temps de vida i les seves característiques de seguretat per determinar la superfície d'atac real de l'aplicació.

A.3.1 Característiques de seguretat de la implementació

La cookie de sessió implementada presenta les següents característiques de seguretat:

- **Temps de vida limitat**: La cookie té un temps de vida de 8 hores, cosa que redueix la finestra d'oportunitat per a un possible atac. Un cop expirada, el token perd la seva validesa i no pot ser utilitzat per un atacant.
- **Cookie HttpOnly**: La cookie és de tipus *HttpOnly*, la qual cosa impedeix que pugui ser llegida directament per codi JavaScript al navegador. Això dificulta el robatori mitjançant atacs XSS, tot i que no l'elimina completament si es produeix un segrest de sessió.
- **Xifrat HTTPS**: GitHub força l'ús de HTTPS en el procés d'autenticació, de manera que el contingut de les capçaleres viatja xifrat a nivell de transport i no pot ser interceptat en text pla. Això dificulta atacs de tipus *man-in-the-middle*.

A.3.2 Vectors d'atac possibles

Malgrat les mesures de seguretat implementades, s'han identificat tres vectors d'atac potencials:

1. **Cross-Site Request Forgery (CSRF)**: Un atacant extern podria crear una pàgina web maliciosa que realitzés peticions autenticades als endpoints de Vercel en

nom de l'usuari. A diferència d'un atac XSS, l'atacant no necessita llegir la cookie directament, ja que el navegador l'adjunta automàticament a qualsevol petició dirigida al domini de Vercel independentment del seu origen. Un atac CSRF exitós podria disparar els *workflows* de GitHub Actions o modificar el contingut del repositori sense el coneixement de l'usuari durant la finestra de vuit hores de vida de la cookie. La protecció estàndard contra aquest tipus d'atac consisteix a verificar la capçalera *Origin* o *Referer* a les *server functions*, o bé a incorporar un token CSRF addicional a cada petició.

- Documentació sobre atacs *man-in-the-middle* a Wikipedia.

2. **Man-in-the-middle (MITM):** Un atacant amb accés a la xarxa podria intentar desxifrar els paquets transmesos per HTTPS per obtenir la cookie. Aquest atac es considera poc viable donat el temps computacional necessari per desxifrar el trànsit xifrat i el curt temps de vida de la cookie, que en limita la finestra d'explo-tació.
3. **Malware al navegador:** Extensions malicioses o programari espia instal·lat al navegador de l'usuari podrien segrestar la cookie de sessió i exfiltrar-la a un servidor controlat per l'atacant. Aquest vector és especialment preocupant perquè opera fora del control de l'aplicació.

A.3.3 Riscos associats

El risc principal en cas d'èxit d'un d'aquests atacs és la compromissió total del compte de GitHub de l'usuari. Un atacant que aconsegueixi la cookie de sessió podria tenir accés complet per llegir, modificar, crear i eliminar qualsevol contingut del compte, incloent-hi repositoris, claus SSH i tokens d'accés personals.

A.3.4 Mesures de mitigació

Tot i que els vectors d'atac identificats presenten una finestra d'oportunitat molt limitada (8 hores) i requereixen condicions específiques per ser explotats, es proposa la següent mesura de mitigació:

Comptes no vinculats: Es recomana que els usuaris de la plataforma utilitzin comptes de GitHub no vinculats a projectes professionals o personals d'importància. D'aquesta manera, en el supòsit que un atacant aconseguís accedir a la cookie durant el temps d'oportunitat, l'impacte real sobre el patrimoni digital de l'usuari quedaria completament limitat al compte específic utilitzat per a ReCollecTF.

A.3.5 Fonts consultades

Per a la realització d'aquesta anàlisi s'han consultat les següents fonts:

- Documentació de GitHub sobre expiració i revocació de tokens [22]
- Documentació de Mozilla Developer Network (MDN) sobre cookies HTTP [23].
- Informació sobre segrest de sessió i atacs CSRF a Wikipedia.