Ludoscrum.com

Una prova de concepte per a la ludificació de processos SCRUM

Autora
Africa Campmany Buisan

Ludoscrum.com
Desenvolupament d'aplicacions interactives

Tutor/a de TF
Lluís Beltrà Valenzuela

Professor/a responsable de l'assignatura
Carlos Casado Martínez

Universitat Oberta de Catalunya

18 juny 2023

Agraïments

Voldria agrair-li al meu tutor Lluís Beltrà Valenzuela i al professor d'aquesta assignatura Carlos Casado Martínez pel seu suport. La seva ajuda ha estat vital per a la consecució d'aquest projecte.
També vull expressar el meu agraïment a la meva família per donar-me suport durant tota la carrera.

Llicència

Índex

Índex de figures

Resum

El treball en equip i la col·laboració són crucials per a l'èxit en qualsevol projecte, especialment en el desenvolupament de programari. No obstant això, motivar als membres de l'equip i mantenir el seu compromís al llarg del projecte pot representar un desafiament.

En aquest context, la ludificació sorgeix com una tècnica poderosa per a fomentar la motivació i el compromís dels participants, transformant les tasques en una experiència emocionant. Amb això en ment, la recerca proposada s'enfoca en el desenvolupament d'una prova de concepte per a una Progressive Web App (PWA) que ludifica processos dins d'un entorn SCRUM. SCRUM és una metodologia àgil i flexible per a gestionar projectes de programari, que es basa en equips multidisciplinaris i autoorganitzats.

El propòsit d'aquesta recerca és demostrar la viabilitat i el potencial del concepte proposat mitjançant la PWA creada, que té com a meta millorar la cooperació, la participació i la gestió de conflictes en els equips de desenvolupament.

A través d'aquest estudi, es pretén contribuir a la comprensió de com la ludificació pot millorar la col·laboració i promoure la motivació en aquests equips. Les troballes i coneixements adquirits poden informar futurs desenvolupaments i aplicacions de la ludificació en contextos similars.

Paraules clau: Ludificació, Progressive Web App (PWA), SCRUM

Abstract

Teamwork and collaboration are crucial for success in any project, particularly in software development. However, motivating team members and maintaining their commitment throughout the project can pose a challenge.

In this context, gamification emerges as a powerful technique to foster motivation and engagement among participants, transforming tasks into an exciting experience. With this in mind, the proposed research focuses on the development of a proof of concept for a Progressive Web App (PWA) that gamifies processes within a SCRUM environment. SCRUM is an agile and flexible methodology for managing software projects, relying on multidisciplinary and self-organized teams.

The purpose of this research is to demonstrate the feasibility and potential of the proposed concept through the created PWA, which aims to improve cooperation, participation, and conflict management within development teams.

Through this study, we aim to contribute to the understanding of how gamification can enhance collaboration and promote motivation in these teams. The findings and knowledge acquired can inform future developments and applications of gamification in similar contexts.

Keywords: Gamification, Progressive Web App (PWA), SCRUM

Introducció

1.1. Contextualització

1.1.2. Marc Teòric

Gamificació/Ludificació

La ludificació és una tècnica que consisteix a aplicar mecàniques de joc en contextos no lúdics amb l'objectiu d'augmentar la motivació i la participació dels usuaris. Aquesta tècnica es basa en el fet que els éssers humans tenen una tendència natural a jugar i a competir, i que l'aplicació d'elements de joc pot fer que les tasques siguin més atractives i divertides.

Encara que la ludificació ha guanyat popularitat en les últimes dècades, el seu ús es remunta a la dècada de 1890. Durant aquest temps, la companyia de targetes de cigarrets American Tobacco Company va incloure cromos col·leccionables en els seus productes per a promoure les vendes i augmentar el compromís del consumidor.[1]
A mitjan segle XX, la gamificació va començar a utilitzar-se en contextos educatius, on es van utilitzar jocs i recompenses per a fomentar l'aprenentatge i el rendiment dels estudiants. En la dècada de 1980, els jocs educatius van començar a guanyar popularitat amb l'aparició dels primers jocs de computadora per a nens.
Amb l'arribada de l'era digital en la dècada de 2000, la ludificació es va convertir en una tècnica popular en molts camps, incloent-hi el màrqueting, la publicitat, la salut i el desenvolupament personal. Empreses com Nike i Foursquare van començar a usar elements de joc per a motivar als consumidors a adoptar comportaments saludables i comprar productes. Al mateix temps, les xarxes socials com Facebook i Twitter van començar a emprar tècniques de gamificació, com els "m'agrada" i les recompenses virtuals, per a millorar la interacció i el compromís dels usuaris.
En l'actualitat, la ludificació s'ha convertit en una tècnica popular en molts camps, i es fa servir en entorns tan diversos com l'educació, la capacitació corporativa, el màrqueting digital i la salut.

En el context dels equips de treball, la ludificació es presenta com una tècnica efectiva per fomentar un enfoc cooperatiu entre companys. Quant es plantegen les tasques

com una experiència més gratificant, es pot augmentar la motivació dels membres de l'equip i, per tant, millorar la qualitat i eficiència del treball.

Desenvolupament amb metodologia àgil i SCRUM

La metodologia àgil de desenvolupament de programari té les seves arrels en el Manifest Àgil[2], un document que va ser redactat i publicat en 2001 per un grup d'experts en desenvolupament de programari. Aquest grup incloïa programadors, consultors i autors de llibres que es van reunir en Snowbird, Utah, per a discutir els desafiaments i limitacions dels mètodes tradicionals de desenvolupament de programari.

El Manifest Àgil estableix quatre valors fonamentals: "Individus i interaccions sobre processos i eines; programari funcionant sobre documentació exhaustiva; col·laboració amb el client sobre negociació contractual; i resposta al canvi sobre seguir un pla". Aquests valors destaquen la importància de la comunicació, la col·laboració i la capacitat d'adaptació en el desenvolupament de programari.

Des de la publicació del Manifest Àgil, s'han desenvolupat diverses metodologies àgils populars, com Scrum[3], Kanban, extremi Programming (XP) i llegeixin. Cadascuna d'aquestes metodologies té les seves pròpies pràctiques i principis, però totes comparteixen l'enfocament en el lliurament primerenc i continu de programari funcional, la comunicació oberta i col·laborativa, i la capacitat d'adaptació als canvis de requisits i de l'entorn.
Al llarg dels anys, la metodologia àgil s'ha convertit en una forma popular i efectiva de desenvolupament de programari en molts entorns, i ha influït en el pensament i la pràctica en moltes altres àrees de treball, incloent-hi la gestió de projectes i la innovació empresarial.

Si parlem de les particularitats de Scrum. En aquesta metodologia, el treball es divideix en iteracions curtes anomenades "sprints", que solen durar d'una a quatre setmanes. Cada sprint té un objectiu clar i es divideix en tres fases: planificació, desenvolupament i revisió. Durant la planificació, l'equip defineix les tasques i objectius per al sprint i tria quins elements del backlog (la llista de tasques pendents) s'abordaran durant l'sprint. Durant el desenvolupament, l'equip treballa en les tasques definides per al sprint i fa reunions diàries per a revisar el progrés i resoldre qualsevol problema. Al final del sprint, es duu a terme una revisió en la qual l'equip presenta el seu treball i rep comentaris i retroalimentació del client i altres interessats.

Scrum també inclou la figura del Scrum Màster, el paper del qual és facilitar i guiar a l'equip en l'ús de la metodologia Scrum, i el Product Owner, qui és responsable de definir els requisits del producte i mantenir el backlog actualitzat.

Tècniques de ludificació col·laboratives i entorns SCRUM

L'SCRUM és una metodologia àgil de gestió de projectes de programari que es basa en equips multidisciplinaris i auto organitzats. En aquest entorn, es treballa en cicles curts anomenats sprints, en els quals s'estableixen objectius i es defineixen les tasques a realitzar en cada cicle.

La gamificació pot aplicar-se en l'entorn SCRUM per a millorar la motivació i el rendiment en cada sprint. Es poden atorgar punts per cada tasca completada o establir un sistema de recompenses per a l'equip que aconsegueixi complir amb els objectius en el menor temps possible.

A més, la ludificació també pot ser útil per a resoldre o prevenir conflictes entre companys.
Un exemple és el "Pòquer de Planificació". El "Pòquer de Planificació" és una activitat d'estimació del treball que involucra a tots. En aquesta activitat, cada participant tria una carta que representa l'esforç necessari per a completar una tasca específica. En revelar les cartes al mateix temps, es poden discutir i resoldre diferències d'opinió sobre la complexitat de la tasca. Aquesta tècnica fomenta la col·laboració i el compromís en involucrar a tots en el procés d'estimació, la qual cosa pot millorar la qualitat de les estimacions i l'efectivitat de l'equip en general.

1.1.3. Mercat i competència

Si fem una ullada, a vista d'ocell, de la indústria de la ludificació en el marc internacional. Observarem, que aquesta ha experimentat un creixement significatiu en els últims anys i s'espera que continuï creixent en el futur. La gamificació ha demostrat ser una eina efectiva per a involucrar i motivar als usuaris, ja sigui en l'àmbit empresarial, educatiu o d'entreteniment. Segons un informe de Research and
Markets (Gamification Global Market Report 2022: Ukraine-Russia War Impact), s'espera que el mercat global de gamificació creixi a una taxa composta anual del 24,9% entre 2020 i 2026.

En termes de regions, Amèrica del Nord ha estat líder en l'adopció de ludificació a causa de la seva avançada infraestructura tecnològica i una àmplia base d'usuaris en línia. No obstant això, s'espera que el mercat de gamificació a Àsia Pacífic experimenti el creixement més gran en el futur a causa de la creixent penetració d'internet i l'augment de la consciència sobre la ludificació.

Podem assumir llavors, que la indústria de la gamificació és una indústria en creixement i s'espera que continuï creixent en els pròxims anys. La ludificació s'està convertint en una eina important per a empreses, educadors i desenvolupadors de jocs, i s'espera que el mercat de gamificació continuï evolucionant per a satisfer les necessitats i demandes dels consumidors.

Hi ha moltes empreses internacionals dedicades a crear programari de ludificació, alguns exemples són:

- **Badgeville**: Aquesta empresa s'enfoca en la ludificació d'aplicacions empresarials i en la creació de programes d'incentius i recompenses per a empleats i clients.
- **Bunchball**: Bunchball és una empresa que s'especialitza en la gamificació de l'experiència del client en línia. Ofereix solucions de ludificació a empreses de diferents sectors, des de la salut i l'educació fins a la tecnologia i l'entreteniment.
- **Kahoot!:** Kahoot! És una plataforma d'aprenentatge en línia que utilitza jocs per a involucrar i motivar als estudiants. Els seus jocs educatius són populars a tot el món i s'utilitzen en escoles, universitats i empreses.
- **Classcraft**: Classcraft és una plataforma de gamificació per a l'educació. Ajuda als mestres a crear un entorn d'aprenentatge interactiu i atractiu per als seus estudiants a través de la ludificació.

- **MindTickle**: MindTickle és una plataforma de ludificació empresarial que ajuda les empreses a millorar l'acompliment dels empleats i l'eficiència del negoci a través de la gamificació i la ludificació basada en la intel·ligència artificial.

A Espanya, també hi ha diverses empreses que es dediquen a crear programari de gamificació. Aquestes són algunes de les més destacades:

- **Nubba**: Situada a València, Nubba és una empresa espanyola que ofereix solucions de ludificació empresarial per a millorar el rendiment i la motivació dels empleats.
- **Trivu**: Amb seu a Madrid, Trivu és una plataforma de gamificació empresarial que ajuda les empreses a millorar l'aprenentatge i la retenció d'informació a través de jocs i desafiaments interactius.
- **Gamelearn**: Situada també a Madrid, Gamelearn s'enfoca en la ludificació per al desenvolupament del talent i la formació empresarial. Ofereix una àmplia gamma de jocs i cursos de gamificació per a millorar el rendiment i la productivitat dels empleats.
- **Playmotiv**: Amb base en Barcelona, Playmotiv és una empresa espanyola que s'especialitza en la ludificació per a la motivació dels empleats. Ofereix solucions de gamificació basades en la psicologia i la neurociència per a millorar la satisfacció laboral i la retenció d'empleats.
- **e-Strategia**: Situada a Sevilla, e-Strategia és una empresa espanyola que ofereix solucions de ludificació per a la millora de l'aprenentatge i la motivació en l'entorn empresarial. Les seves solucions de gamificació se centren a millorar el rendiment i la productivitat dels empleats.

"Gamification Global Market Report 2022: Ukraine-Russia War Impact" [4]

En aquest informe s'aborda el mercat de la gamificació i el seu creixement esperat en els pròxims anys, així com les tendències i factors que impulsen aquest creixement. També s'esmenten les diferents regions geogràfiques en les quals s'espera que creixi el mercat i les verticals d'usuaris finals que l'utilitzen. A més, es destaca la importància dels avanços tecnològics i la penetració dels telèfons intel·ligents en el creixement del mercat de la ludificació.
https://www.researchandmarkets.com/reports/5700430/gamification-global-market-report-2022-ukraine#src-pos-3

“Spain E-Learning Market Overview, 2027” [5]

El tema central d'aquest informe és el mercat d'aprenentatge electrònic a Espanya i com està experimentant un creixement destacat a causa de diversos factors, com l'estalvi de costos, la gestió del temps, la flexibilitat i la comoditat que ofereix, així com l'augment de l'ús de tecnologies d'aprenentatge electrònic i dispositius mòbils. També s'esmenten diverses tendències en aquest mercat, com l'ús de la intel·ligència artificial i l'aprenentatge automàtic, la ludificació i l'enfocament en l'educació centrada en l'estudiant. I, es descriu com l'aprenentatge electrònic s'està utilitzant en diferents sectors, com l'agricultura, atenció mèdica, educació i capacitació corporativa.

1.1.4. Motivació i oportunitats

En el context actual, superada la pandèmia de la COVID-19, moltes empreses continuen mantenint el teletreball. Això ha portat a un augment en la quantitat d'equips de desenvolupament que treballen de manera remota i ha plantejat nous desafiaments per a la col·laboració i la cohesió d'aquests.

La cultura col·laborativa és fonamental perquè els equips puguin treballar junts de manera efectiva i aconseguir els seus objectius en un entorn remot. Necessiten sentir-se connectats i compromesos amb els objectius del projecte. Per a assolir això, és important fomentar la comunicació positiva. I a més, és valuós que els membres se sentin part d'una comunitat unida.

La naturalesa complexa i multifacètica dels projectes de programari requereix una interacció real constant i un intercanvi d'informació entre companys. Avui dia hi ha moltes eines de cooperació, i gestió de projectes de programari disponible per a ajudar a treballar de manera més eficient. Aquestes eines, com Jira, Github, Slack o Zeplin, ofereixen una àmplia gamma de funcionalitats per a facilitar el flux d'informació, la gestió de projectes i la col·laboració en temps real.

Però per a crear una comunitat i una cultura d'empresa, fa falta alguna cosa més. Tenir bons canals d'informació i mètodes de treball optimitzats són elements crítics per a l'èxit d'un equip de desenvolupament, però per a crear una cultura empresarial sòlida i una veritable comunitat, és necessari humanitzar aquesta infraestructura i enfocar-se en la psicologia humana.

Per a aconseguir això, és fonamental entendre que els membres de l'equip són éssers humans amb necessitats emocionals i psicològiques que han de ser ateses. Per exemple, el treball remot pot portar a la sensació d'aïllament i desconnexió social, per la qual cosa és crucial promoure una bona comunicació que vagi més enllà de la tasca i que se centri en la construcció de relacions interpersonals entre els companys.

En conclusió, una aplicació de ludificació tindria l'oportunitat de ser una eina útil per a impulsar la interacció i el compromís entre els membres de l'equip. Es podrien crear jocs i desafiaments que involucrin a tots, i que els incentivin a interactuar i col·laborar entre si. Els jocs podrien ser dissenyats per a recompensar comportaments valuosos, com la comunicació socioafectiva positiva i la cooperació, la qual cosa ajudaria a fomentar una cultura més positiva en l'equip.

1.2. Objectius

L'objectiu d'aquest treball és crear una prova de concepte per a una Progressive Web App (PWA) que ludifiqui processos dins d'entorns àgils de desenvolupament de programari, com SCRUM. Es tracta d'una primera proposta experimental que té com a finalitat provar noves estratègies dins de la indústria de la ludificació.

Com a prova de concepte (PoC), serà una versió primerenca i limitada de l'aplicació, amb la finalitat de validar i demostrar que la idea és factible i viable. La prova de concepte és una fase important en el desenvolupament de programari, ja que permet avaluar el funcionament del producte i el seu possible èxit en el mercat abans d'invertir temps i recursos en el seu desenvolupament complet. En aquesta PoC, s'implementa només una petita part, prou per a demostrar la idea central. Aquesta versió primerenca pot no tenir totes les característiques i funcionalitats planejades, però és suficient per a validar la viabilitat tècnica. A més, aquesta prova de concepte també ajuda a identificar possibles problemes i desafiaments que puguin sorgir durant el desenvolupament complet de l'aplicació.

Amb aquest producte es pretén impulsar la cooperació i la comunicació interpersonal entre companys, en entorns SCRUM de desenvolupament de programari, a través d'elements de ludificació dissenyats per a aquest fi. Per a aconseguir això, s'identificaran els processos dins de l'entorn àgil que es puguin ludificar i que compleixin aquests criteris.

Es dissenyaran jocs i desafiaments que involucrin als companys de treball i que fomentin interaccions positives entre ells. És important que aquests elements de ludificació no siguin artificiosos ni complexos i tinguin una clara raó de ser. Aquests jocs estaran dissenyats per a ajudar a alinear als membres de l'equip per a afrontar els reptes que es presentin, esquivant divisions internes. Això ajudarà a fomentar la companyonia.

En definitiva, es buscarà explorar noves possibilitats per a millorar l'eficiència i la satisfacció en el treball, en entorns de desenvolupament de programari, a través de la ludificació.

1.2.1. Objectius específics

- Identificar els processos dins de l'entorn àgil que es puguin ludificar i que fomentin la cooperació i la comunicació
- Buscar elements de ludificació que no siguin artificiosos ni complexos
- Dissenyar jocs i desafiaments que involucrin als companys de treball i que fomentin interaccions positives
- Dissenyar tots els elements que es ludifiquin de manera que s'evitin els estereotips de gènere i es promogui la igualtat d'oportunitats
- El disseny ha de ser accessible per a totes les persones, independentment de les seves habilitats o discapacitats
- El 50% dels jocs estaran dissenyats per a privilegiar alguna limitació concreta
- Ha de ser adaptable a diferents dispositius i resolucions de pantalla
- Avaluar la usabilitat, ha de ser fàcil d'usar i intuïtiva
- L'aplicació ha de ser escalable i poder manejar un nombre creixent d'usuaris i dades

1.3. Impacte en aspectes de sostenibilitat, ètica i diversitat

Aspectes positius i/o negatius del TFG en les tres dimensions de la CCEG

1.3.1. Dimensió sostenibilitat

Com tot producte de programari, el resultat final d'aquest treball, s'allotjarà en el núvol, amb la qual cosa tindrà un impacte mediambiental. Perquè encara que la naturalesa intangible dels productes de programari pugui fer la sensació que no tenen cap impacte, cal tenir molt present que tot el procés de desenvolupament, manteniment i ús de l'aplicació consumeix recursos físics i energia, la qual cosa al seu torn pot generar emissions de gasos d'efecte d'hivernacle i altres impactes ambientals.

En primer lloc, l'ús de servidors en el núvol per a allotjar l'aplicació consumeix energia elèctrica. Els servidors requereixen energia constant per a funcionar i mantenir una temperatura adequada, la qual cosa pot generar emissions de diòxid de carboni si l'electricitat es genera a partir de fonts d'energia no renovable.

A més, l'ús de l'aplicació en si també pot requerir una quantitat significativa d'energia, depenent de la intensitat dels gràfics i la quantitat de processament necessari perquè l'aplicació funcioni correctament. Això és pel fet que els usuaris de l'aplicació necessiten un dispositiu que consumeixi energia, com un ordinador o un telèfon mòbil, la qual cosa al seu torn també pot suposar emissions.

Tot aparell electrònic involucrat en el procés tindrà una petjada ambiental. La producció i eliminació d'aquests equips també genera emissions i la generació de residus electrònics.

Es tindrà en compte l'impacte ambiental per a reduir la seva petjada ecològica. Per a això s'intentarà adoptar pràctiques de desenvolupament sostenibles i la implementació d'estratègies d'eficiència energètica en els servidors i dispositius utilitzats per a l'aplicació, en la mesura que sigui possible.

1.3.2. Dimensió comportament ètic i de responsabilitat social

Des de la perspectiva de gènere i diversitat

No s'esmenta explícitament el concepte d'igualtat de gènere en el projecte, però això no significa que no sigui un factor important. La ludificació pot ser una eina útil per a fomentar la inclusió i la igualtat de gènere en el lloc de treball a crear un entorn més col·laboratiu i equitatiu per a tots, independentment del seu gènere.

Un dels requisits d'aquest treball de fi de grau, serà que tots els elements que es ludifiquin, seran dissenyats de manera que s'evitin els estereotips de gènere i es promogui la igualtat d'oportunitats. És crucial fer accions concretes que empenyin a la societat cap a l'equitat. La ludificació pot ser una eina útil per a fomentar un entorn més inclusiu per a tots.

La primera acció que es durà a terme en aquest sentit, serà l'aplicació d'estratègies de llenguatge inclusiu, com l'ús de llenguatge no sexista, la utilització de pronoms inclusius i, especialment, l'alternança de gènere. Faig especial èmfasi en l'alternança de gènere, perquè és una fórmula més efectiva i senzilla per a promoure la igualtat de gènere en el discurs. A diferència d'altres estratègies, l'alternança de gènere no requereix la creació de noves paraules o estructures gramaticals complexes. En utilitzar aquesta fórmula, es poden construir frases i oracions de manera més fluida i natural, la qual cosa facilita la comunicació. En lloc de crear una barrera lingüística, contribueix a crear un ambient més inclusiu i respectuós.

És important destacar que l'estratègia de l'alternança, no és només una qüestió de gramàtica o lèxic, sinó que té implicacions més àmplies en la lluita per la igualtat de gènere. En fer servir aquesta fórmula, s'està contribuint a la visibilització de totes les identitats i perspectives, trencant amb els estereotips de gènere. És una eina poderosa per a crear un canvi positiu en la nostra societat i en el nostre llenguatge.[6]

Des de la perspectiva de l'accessibilitat

En qualsevol desenvolupament de programari cal tenir en compte l'accessibilitat perquè l'experiència d'ús sigui inclusiva i satisfactòria per a totes les persones, independentment de les seves capacitats. En aquest sentit, existeixen diversos punts a tenir en compte, com el disseny inclusiu o l'ús de tecnologia d'assistència. És essencial assegurar-se que l'aplicació sigui compatible amb aquestes últimes, com a lectors de

pantalla i teclats alternatius, per a garantir que les persones amb discapacitat puguin interactuar amb l'aplicació de manera efectiva.

Però a més a més, quant ens enfoquem a la ludificació i els usuaris amb discapacitat, podem descobrir que aquestes persones sovint posseeixen habilitats especials que les fan destacar en unes certes situacions. És crucial considerar aquestes habilitats en el disseny dels jocs i reptes, de manera que puguin exercir un paper actiu i destacat en el procés. Es poden dissenyar jocs ludificats inclusius, que empoderin a persones amb capacitats especials, fomentant una major comprensió i respecte per les seves habilitats úniques.

En definitiva, la ludificació inclusiva pot ajudar a crear un ambient de feina més just i equitatiu per a totes les persones, independentment de les seves capacitats.

En el TFG treballaré sobre aquesta matèria en dues etapes, el disseny de les ludificacions i l'accessibilitat del codi.

En primer lloc. El disseny dels jocs i reptes, haurien de ser avaluats en termes d'accessibilitat. Però, a més estableixo que el 50% dels jocs estaran dissenyats per a privilegiar alguna limitació concreta, podent ser des d'una dificultat lleu, com la dislèxia, fins a un repte major, com una discapacitat visual.

D'altra banda, quan parlem de l'accessibilitat del codi, ens referim a la facilitat amb la qual una persona pot interactuar amb el joc utilitzant tecnologies de suport com a lectors de pantalla i dispositius d'entrada alternatius. Per a garantir l'accessibilitat del codi, hauré de seguir les pautes i estàndards internacionals en matèria d'accessibilitat web, com les pautes WCAG (Web Content Accessibility Guidelines).[7]

Des de la perspectiva de l'equitat social

Reconec la necessitat d'abordar les desigualtats i els biaixos que poden sorgir a causa de la raça o l'origen ètnic, així com els contrastos associats a la classe social. Des de la meva perspectiva, pretenc que aquest projecte fomenti un ambient inclusiu i just que permeti l'accés igualitari a oportunitats, independentment del seu origen ètnic o posició socioeconòmica.

Des d'una eina de ludificació, es pot afrontar aquest punt des de dos vessants. En primer lloc, fomentant la companyonia, animant als membres de l'equip a relacionar-se

com a iguals, i, d'altra banda, pot crear un entorn segur per a la resolució i prevenció de conflictes per motius de raça, religió o posició socioeconòmica.

1.4. Planificació

1.4.1. La planificació per etapes

- **Disseny conceptual del joc:**

  En aquesta fase es defineix la visió general del joc, la seva temàtica, objectius, regles i mecàniques. S'identifiquen i documenten els requisits funcionals i no funcionals que ha de complir el joc i es modela la interfície i la interacció de l'usuari amb aquest. També es modela el domini de l'aplicació.

  - Definició d'objectius, regles, mecàniques i temàtica.
  - Identificació i documentació de requisits funcionals i no funcionals del joc.
  - Modelització de la interfície i la interacció de l'usuari amb el joc.
  - Modelatge del domini de l'aplicació.

- **Avaluació i selecció de tecnologies:**

  En aquesta fase s'avaluen les tecnologies disponibles per a implementar el joc i se selecciona la més adequada per al projecte. S'han de considerar factors com l'escalabilitat, la disponibilitat de recursos i el cost.

  - Avaluació de tecnologies disponibles per a implementar el joc.
  - Selecció de la tecnologia més adequada per al projecte.

- **Implementació i desenvolupament del joc:**

  En aquesta fase es crea una versió inicial del joc amb funcionalitats bàsiques, coneguda com a versió. Es dissenya la usabilitat i els tests unitaris i es revisa el disseny del programari. S'implementen connexions amb serveis de tercers si són necessaris. Després es crea una versió avançada del joc per a provar i millorar l'aplicació, coneguda com a versió beta. Finalment, es dissenya la versió final del joc.

- Creació d'una versió inicial del joc amb funcionalitats bàsiques (versió alfa).
- Disseny de la usabilitat i dels tests unitaris.
- Revisió del disseny del programari.
- Implementació de connexions amb serveis de tercers (si són necessaris).
- Creació d'una versió avançada del joc per a provar i millorar l'aplicació (versió beta).
- Disseny final del joc.

- **Llançament i manteniment del joc:**

  En aquesta fase es revisa el compliment de tots els requisits del projecte i es llança el joc en producció perquè els usuaris puguin accedir i gaudir d'aquest. S'avaluen opcions d'allotjament i se selecciona la millor opció per a l'aplicació. Es manté i actualitza el joc per a millorar la seva qualitat i corregir errors. Aquesta fase també inclou l'avaluació del rendiment i l'escalabilitat del joc, així com l'atenció al client i la resolució de problemes tècnics.

  - Revisió del compliment de tots els requisits del projecte.
  - Llançament del joc en producció perquè els usuaris puguin accedir i gaudir del joc.
  - Avaluació d'opcions d'allotjament i selecció de la millor opció per a l'aplicació.
  - Manteniment i actualització del joc per a millorar la seva qualitat i corregir errors.

1.4.2. Temporització

He creat un diagrama de Gantt amb les tasques planificades.
Aquest és l'enllaç:
https://view.monday.com/1192616034-b24fdca1467d9999df0f34d887da399f?r=euc1

He triat usar un diagrama de Gantt perquè considero que és una eina molt útil per a la gestió de projectes que permet visualitzar de manera clara i estructurada les tasques, terminis i dependències. Em permet veure i mostrar la planificació temporal de les activitats, i em facilita el seguiment del progrés.

Amb aquesta mena de diagrames, és possible identificar fàcilment les dates d'inici i finalització de cada tasca, així com les interdependències entre elles. Això em proporciona una visió global del projecte, permetent-me la detecció de possibles colls d'ampolla i m'ajuda a prendre decisions informades per a complir amb els objectius que m'he fixat.

2. Implementació

2.1. Abast de la prova de concepte (PoC)

En aquesta PoC, l'objectiu és validar la viabilitat d'un projecte de ludificació, centrant-se en el stack tecnològic, proves i sostenibilitat, inclusió en entorns SCRUM i accessibilitat. L'enfocament principal és desenvolupar el primer joc amb totes les funcionalitats bàsiques, centrant-se en els següents casos d'ús específics per a un joc de "Pedra, paper, tisora" amb integració en Slack:

- Obrir una nova partida:
  - Els usuaris poden obrir una partida i assignar-li un nom.
  - Aquest cas d'ús implica la creació d'una partida nova en l'aplicació.
- Registrar-se en la partida:
  - Els usuaris poden registrar-se en una partida existent.
  - Això permet als usuaris unir-se a una partida en curs i participar en el joc.
- Desafiar a un altre usuari:
  - Els usuaris poden desafiar a un altre usuari a través de Slack.
  - El desafiament s'enviarà a través de Slack com una notificació o missatge directe a l'usuari objectiu.
- Joc de "Pedra, paper, tisora":
  - Els usuaris participen en el joc clàssic de pedra, paper i tisores contra altres jugadors.
  - Cada jugador selecciona la seva opció (pedra, paper o tisores) i espera al fet que l'oponent faci el mateix.
  - Després que tots dos jugadors hagin seleccionat la seva opció, es mostra el resultat del joc i es determina al guanyador.

No es considera el cas d'ús de registre, ja que no forma part del joc en si. Seguint l'exemple de Planningpokeronline.com, el registre és opcional i no és un requisit per a participar en el joc.

Àmbit de proves i sostenibilitat

En la part de proves i sostenibilitat, es realitzaran proves unitàries per a garantir la qualitat i estabilitat del codi. Aquestes proves s'enfocaran a verificar el funcionament individual dels components i mòduls del joc, assegurant el seu correcte comportament i detecció precoç de possibles errors.

A més, s'implementaran bones pràctiques de sostenibilitat, com seguir estàndards de codificació, utilitzar un sistema de control de versions per a un seguiment adequat dels canvis i aplicar principis de disseny net i modularitat per a facilitar futures modificacions i actualitzacions.

L'objectiu és establir una base sòlida per al desenvolupament del projecte, assegurant la qualitat del codi i facilitant el seu manteniment a llarg termini.

Àmbit d'accessibilitat

Pal que fa a l'accessibilitat, s'aplicaran diverses estratègies per a garantir que el joc sigui accessible per a tots els usuaris, incloent-hi aquells amb discapacitats visuals o de mobilitat. Algunes de les estratègies que s'implementaran són:

- Ús d'etiquetes semàntiques: S'utilitzaran les etiquetes HTML semàntiques per a estructurar correctament el contingut i facilitar la comprensió i navegació per als usuaris que utilitzin lectors de pantalla.
- Ús d'atributs ÀRIA: S'agregaran atributs ÀRIA (Accessible Rich Internet Applications) en elements interactius per a proporcionar informació addicional i millorar l'accessibilitat.
- Contrast adequat: Es prestarà especial atenció al contrast de colors emprats en el disseny de la interfície. S'assegurarà que existeixi un contrast suficient entre el text i el fons per a facilitar la llegibilitat, complint amb les pautes d'accessibilitat.
- Ús de text alternatiu en imatges: Es proporcionarà text alternatiu descriptiu en imatges perquè els usuaris amb discapacitat visual puguin comprendre el contingut de les imatges a través de lectors de pantalla.

2.2. Disseny i Justificació de l'Arquitectura

plantuml

He utilitzat PlantUML per representar els diagrames UML[8]. PlantUML és una eina de codi obert que permet generar diagrames UML utilitzant una sintaxi senzilla i llegible en forma de text. Aquests diagrames es poden convertir en imatges que representen visualment l'estructura i les relacions entre les classes, els objectes i els components d'un sistema.

Amb PlantUML, es pot generar un diagrama UML en diversos formats, com PNG, SVG o fins i tot ASCII art, cosa que permet la seva visualització en diferents entorns i eines.

Casos d'ús crítics

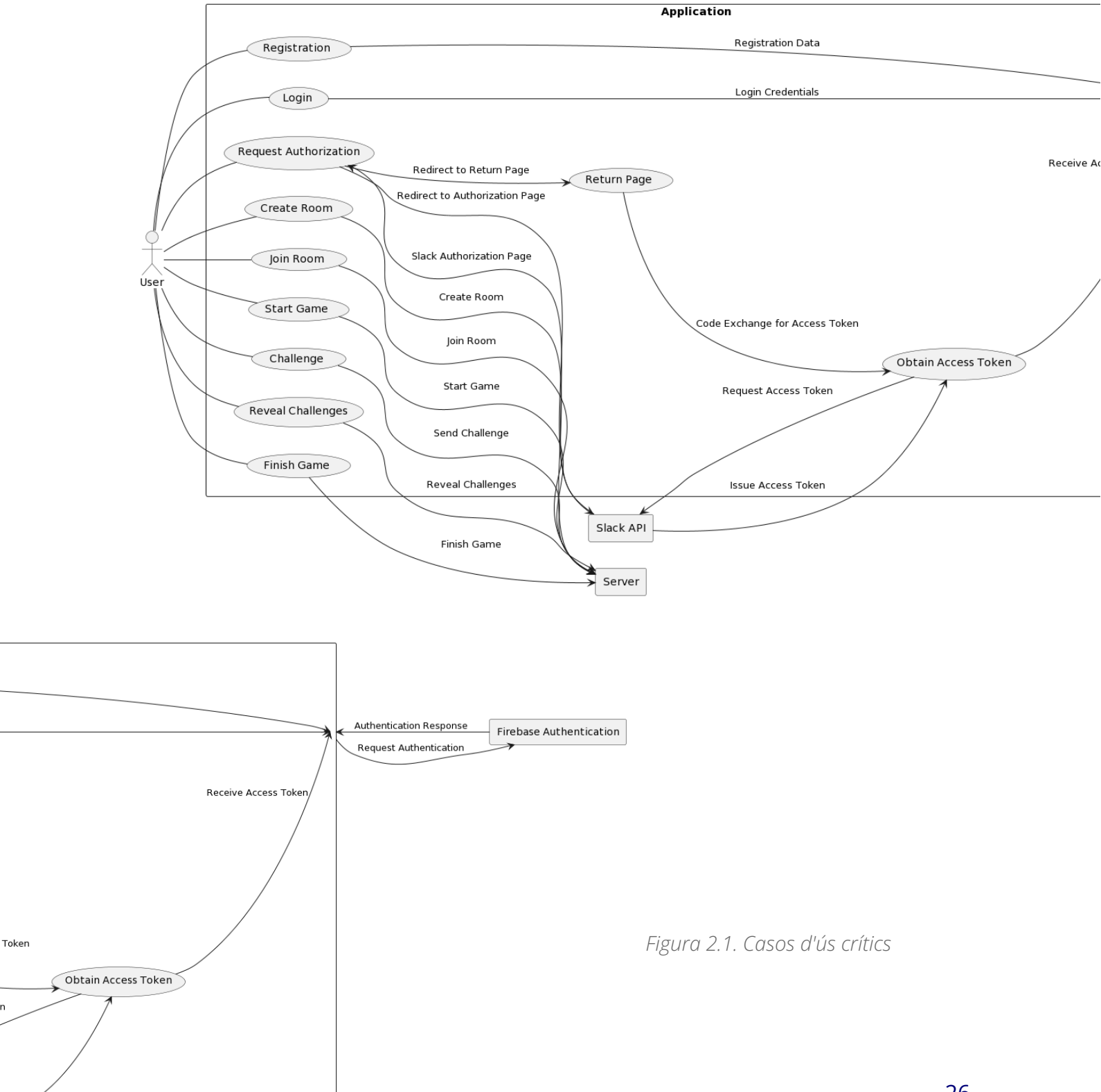


Figura 2.1. Casos d'ús crítics

El diagrama il·lustra la interacció d'un usuari amb l'aplicació a través de diversos casos d'ús.
Consta de diverses funcionalitats:

Accions que requereixen interacció amb Firebase:

- **Registre d'usuari:** Permet als usuaris registrar-se a l'aplicació proporcionant les vostres dades personals. Aquesta acció implica interactuar amb Firebase per emmagatzemar la informació de l'usuari a la base de dades i permetre'n l'autenticació posterior.
- **Inici de sessió d'usuari:** Permet als usuaris iniciar sessió a l'aplicació utilitzant les credencials. El Firebase s'utilitza per verificar la identitat de l'usuari i proporcionar accés segur a les funcionalitats de l'aplicació.
- **Autenticació del Firebase:** Aquesta acció implica interactuar amb Firebase per autenticar i validar les credencials dels usuaris. Firebase proporciona un sistema segur d'autenticació i gestió de sessions per garantir la privacitat i la seguretat de les dades.

Accions que requereixen interacció amb Slack:

- **Sol·licitud d'autorització:** Es fa una sol·licitud a Slack per obtenir permisos d'autorització. Això pot implicar redirigir l'usuari a la pàgina d'autorització de Slack, on se us demana que atorgueu permisos específics a l'aplicació.
- **Autorització de Slack:** Després que l'usuari hagi atorgat els permisos sol·licitats, Slack torna una autorització que indica que l'aplicació té accés autoritzat a certes funcionalitats de Slack.
- **Obtenir el token d'accés:** Un cop s'ha obtingut l'autorització de l'Slack, s'intercanvia un codi per un token d'accés vàlid. Aquest Token d'accés s'utilitzarà posteriorment per realitzar accions en nom de l'usuari a Slack.

Altres accions:

- **Iniciar Joc:** Inicia el joc i estableix l'estat del joc com a "en progrés".
- **Crear Desafiament:** Permet crear un desafiament dins del joc.
- Unir-se a Desafiament: Permet a un jugador unir-se a un desafiament existent al joc.
- **Jugada:** Permet a un jugador fer una jugada o respondre a un desafiament dins del joc.
- **Revelar Resultat:** Mostra els resultats d'un desafiament o una jugada específica dins del joc.
- **Finalitzar Joc:** Acaba el joc i mostra els resultats finals.

Amb aquestes accions, el diagrama mostra el flux d'accions i interaccions entre l'usuari, l'aplicació, els serveis externs com l'API de Slack i l'Autenticació de Firebase, i el component del servidor. Representa el procés dalt nivell i el flux de comunicació de la funcionalitat de l'aplicació.

Més informació a l'annex: ***Casos d'ús crítics***

Entitats i les seves relacions

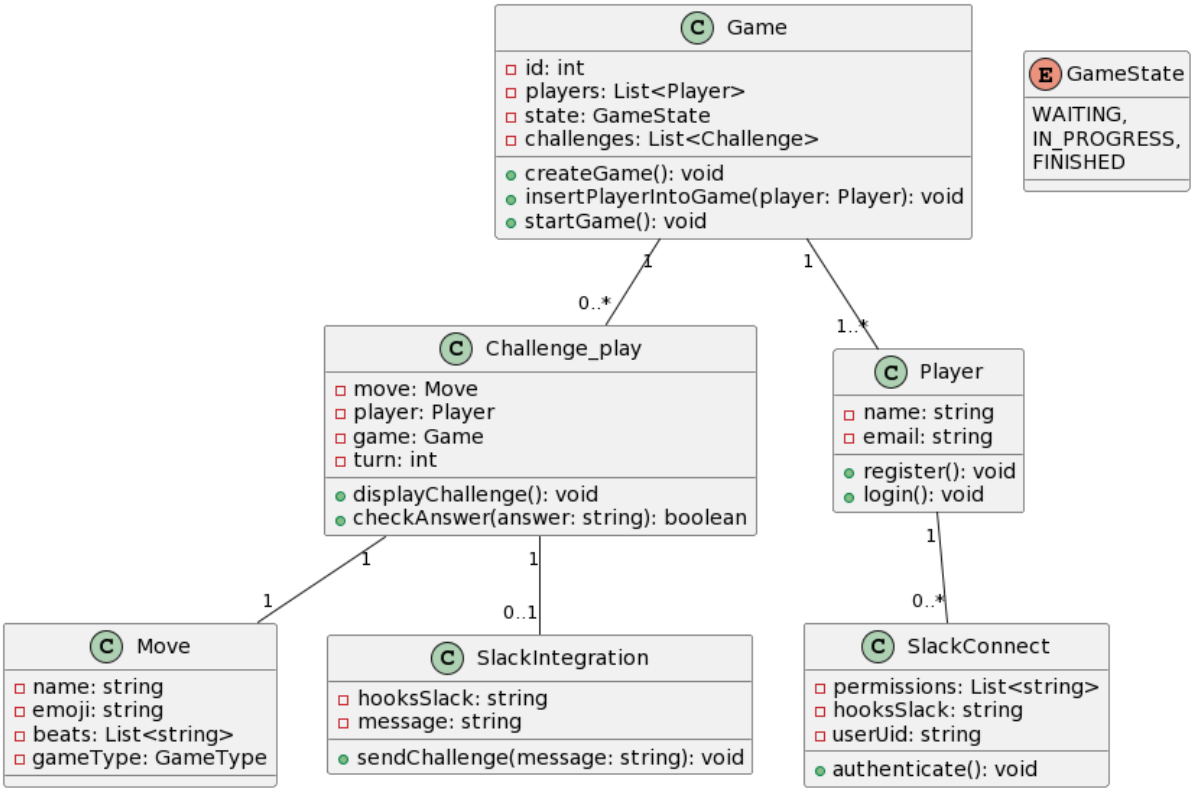


Figura 2.2. Entitats i les seves relacions

Al diagrama, represento l'estructura del joc els elements que vertebren la integració amb Slack.

Les classes representades i les seves responsabilitats:

- **Classe Game:** Representa un joc i té atributs com id, una llista de jugadors (Player), estat del joc (GameState) i desafiaments (Challenge). Té mètodes per crear un joc, inserir jugadors al joc i començar el joc.
- **Classe Challenge_play:** Representa un desafiament al joc. Té atributs com a moviment (Move), jugador (Player), joc (Game) i torn. Teniu mètodes per mostrar el desafiament i verificar una resposta.

- **Classe Move:** Representa un moviment al joc. Té atributs com a nom, emoji, llista de moviments superats (beats) i tipus de joc (gameType).
- **Classe Player:** Representa un jugador al joc. Té atributs com a nom i correu electrònic. Teniu mètodes per registrar-vos i iniciar sessió.
- **Classe SlackIntegration:** Representa la integració amb Slack. Té atributs com hooksSlack (connexions amb Slack) i missatge. Té un mètode per enviar un desafiament a través de Slack.
- **Classe SlackConnect:** Representa la connexió amb Slack. Té atributs com a permisos, hooksSlack i userUid (identificador de l'usuari). Té un mètode per autenticar-se.
- **Enum GameState:** Representa els estats possibles del joc, com WAITING (esperant), IN_PROGRESS (en progrés) i FINISHED (finalitzat).

Les relacions entre les classes:

- **Game:** té una relació d'un a molts amb jugadors (Player).
- **Game:** té una relació opcional d'un a molts amb reptes (Challenge_play).
- **Challenge_play:** té una relació d'un a un amb un moviment (Move).
- **Challenge_play:** té una relació opcional d'un a zero o una amb SlackIntegration.
- **Player:** té una relació d'un a molts amb connexions de Slack (SlackConnect).

Aquest diagrama proporciona una representació visual de l'estructura i les relacions entre les classes en el context del joc i la integració amb Slack.

Més informació a l'annex: ***Entitats i les seves relacions***

Avaluació de dissenys i patrons

A la PoC s'ha utilitzat un disseny inicial basat en el patró d'arquitectura MVC (Model-Vista-Controlador)[9] com a punt de partida per a l'aplicació. Tot i això, es reconeix la necessitat d'explorar i adoptar patrons de disseny més apropiats en futures etapes del projecte, a mesura que s'adquireixi un major coneixement dels desafiaments específics que enfronta.

Durant la fase de prova de concepte, s'ha desenvolupat una arquitectura MEVN (MongoDB, Express.js, Vue.js, Node.js)[10], que calia adaptar a una estructura MVC al codi. Aquesta adaptació del patró MVC a l'arquitectura MEVN s'ha abordat de la manera següent:

- **Model:** La capa de model continua sent responsable de gestionar les dades i la lògica de negoci. En aquest cas, MongoDB s'ha utilitzat com a base de dades per

emmagatzemar i recuperar les dades relacionades amb usuaris, jocs, desafiaments, entre d'altres. S'ha utilitzat Objecte-Relational Mapping (ORM) Mongoose per definir els models de dades i facilitar la interacció amb la base de dades des de l'aplicació Node.js.

- **Vista:** A l'arquitectura MEVN, la capa de vista està representada pels components de Vue.js. Aquests components són responsables de la presentació de dades i de la interacció amb l'usuari. S'han creat components Vue.js que mostren formularis de registre, pantalles d'inici de sessió, interfícies de joc, entre d'altres. Aquests components interactuen amb el controlador per enviar sol·licituds i rebre respostes.

- **Controlador:** La capa de controlador està representada per les rutes i controladors d'Express.js. Express.js actua com el framework web al costat del servidor i controla les rutes i les sol·licituds HTTP. S'han definit rutes que corresponen a les accions de l'usuari, com crear desafiaments, unir-se a un joc, registrar connexions Slack, entre d'altres. Els controladors són responsables de gestionar aquestes rutes i accedir als mètodes apropiats del model.

A més del disseny MVC, s'han identificat alguns patrons de comportament i estructurals que podrien ser aplicables a versions futures del projecte:[11]

Patrons de Comportament:

- **Factory Method:** El mètode createGame() a la classe Game podria representar el patró de disseny Factory Method, on es delega la creació d'objectes a subclasses. Això permetria que les subclasses específiques de Game implementin la lògica per crear diferents tipus de jocs.

- **Template Method:** El mètode startGame() a la classe Game podria representar el patró de disseny Template Method, on es defineix un esquelet d'algorisme en una classe base i es permet a les subclasses implementar passos específics. La classe base Game definiria el flux general de l'inici d'un joc i les subclasses concretes proporcionarien la implementació específica dels passos necessaris per iniciar un joc en particular.

- **Singleton:** Es podria implementar el patró Singleton a la classe Game per garantir que només existeixi una instància de Game a tot el sistema. Això seria útil si cal assegurar que només hi hagi un joc en execució alhora.

- **Adapter:** Es podria implementar el patró Adapter per permetre que la classe SlackIntegration es comuniqui amb altres objectes que utilitzin una interfície diferent. Per exemple, si SlackIntegration necessita enviar notificacions mitjançant un servei de correu electrònic. Com, per enviar la invitació a registrar-se a tots els usuaris de l'espai de treball. Es podria crear un adaptador que tradueixi les trucades de SlackIntegration al format esperat pel servei de correu electrònic.

- **Observer:** El patró Observer es pot aplicar per mantenir actualitzats i sincronitzats els objectes que depenen de l'estat del joc. Els objectes Challenge_play, SlackIntegration i Player actuarien com a observadors que se subscriuen als canvis d'estat de l'objecte Game. Quan es produeix un canvi a l'estat del joc, es notificaria als observadors perquè realitzin les accions corresponents.

Patró d'Estructura:

- **Composite:** La relació de composició entre les classes Game i Challenge, on Game té una col·lecció de Challenge, podria representar el patró de disseny Composite. En aquest patró, es tracta un grup d'objectes de la mateixa manera que un objecte individual. En aquest cas, la classe Game actuaria com a objecte compost que conté una col·lecció de desafiaments (Challenge). Això permetria tractar tant un joc com els seus desafiaments individuals de manera uniforme, ja que tots dos implementen una interfície comuna.

La incorporació d'aquests patrons de disseny en versions posteriors del projecte permetria millorar l'estructura, la modularitat i la flexibilitat del codi.

2.3. Disseny de la Interfície i Wireframes

Identitat visual de l'aplicació

En abordar el disseny de la identitat visual, tenia molt present que havia de considerar no només l'estètica, sinó també la usabilitat i l'accessibilitat. Vaig centrar-me a crear elements visuals que fossin intuïtius i fàcils d'entendre per als usuaris.

Per aconseguir una bona usabilitat i l'accessibilitat, em vaig assegurar de seleccionar tipografies llegibles i clares, amb mides adequades per facilitar-ne la lectura. A més, vaig

buscar colors que no només fossin visualment atractius, sinó que mantinguessin un nivell de contrast adequat.

També vaig prestar atenció a l'organització i la jerarquia de la informació en els elements visuals. Vaig treballar amb la idea de crear dissenys equilibrats i estructurats, de manera que els usuaris poguessin trobar fàcilment la informació rellevant i navegar sense dificultats. Això va implicar ajustar elements visuals, simplificar interfícies i optimitzar la interacció general.

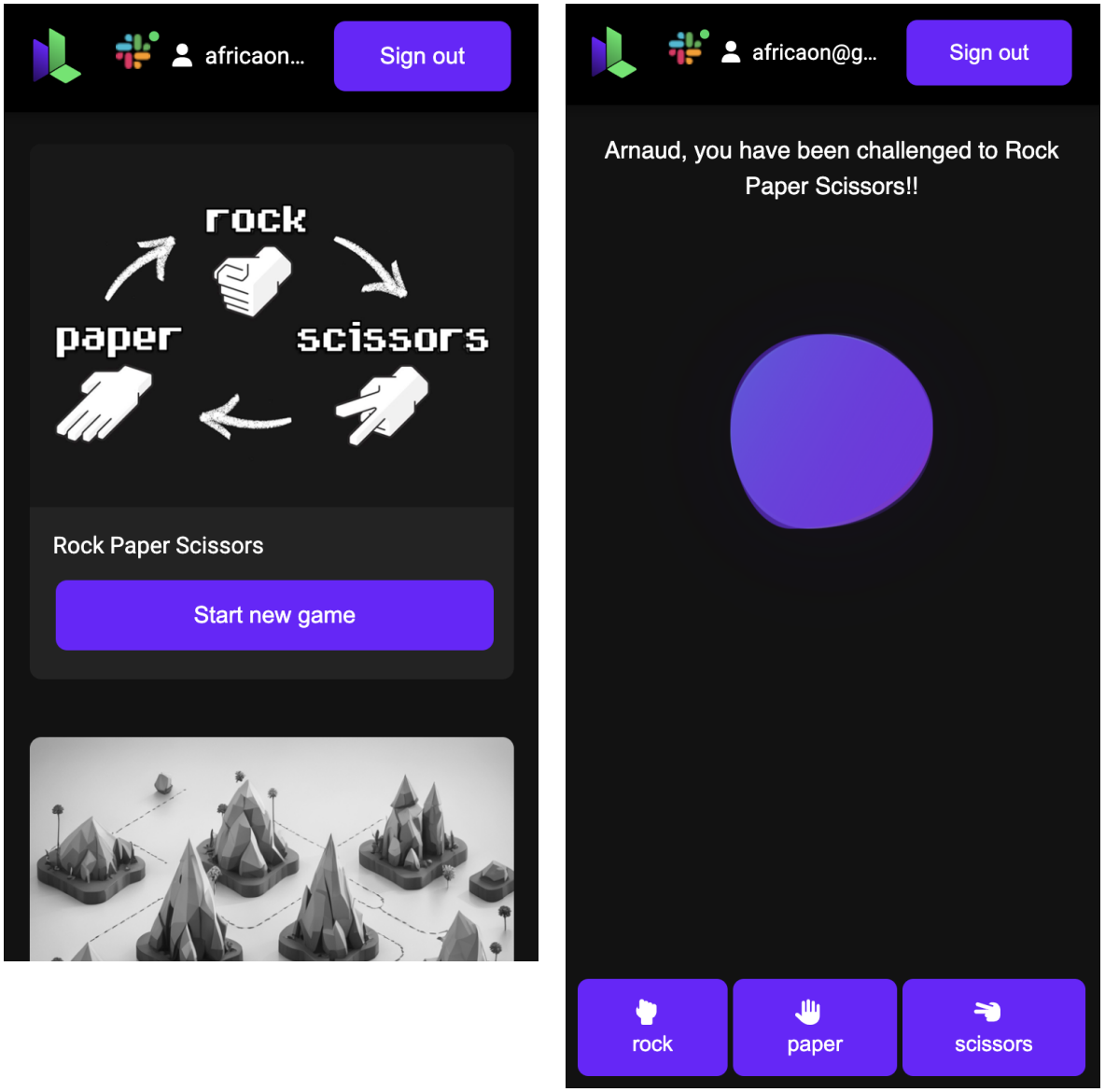


Figura 2.3. Captura de pantalla de la home i la sala del joc en versió Mobile.

Paleta de colors

La identitat de marca de Ludoscrum es basa en una paleta de tons foscos, amb el negre i diversos grisos foscos com a protagonistes. Aquests tons foscos han estat seleccionats estratègicament per crear un ambient modern i d'acord amb les preferències dels professionals del desenvolupament. L'elecció de tons foscos també té en compte la comoditat visual, evitant el cansament ocasionat per una il·luminació excessiva en pantalles, i proporciona un fons ideal per ressaltar elements visuals, com ara textos, gràfics i colors més vibrants, aconseguint un contrast efectiu i una llegibilitat òptima.

Si ens centrem en els colors, el color principal utilitzat és el #6E1FFF, un porpra fosc que destaca sobre els grisos foscos, però amb un contrast moderat. Aquest color s'utilitza per ressaltar elements clau i atraure l'atenció del públic objectiu. Com a color secundari, es fa servir el #3AE762, un verd clar que aporta frescor i energia. Aquest to complementa perfectament el color principal, creant un equilibri visual atractiu.

Figura 2.4. *Paleta de colors Ludoscrum*

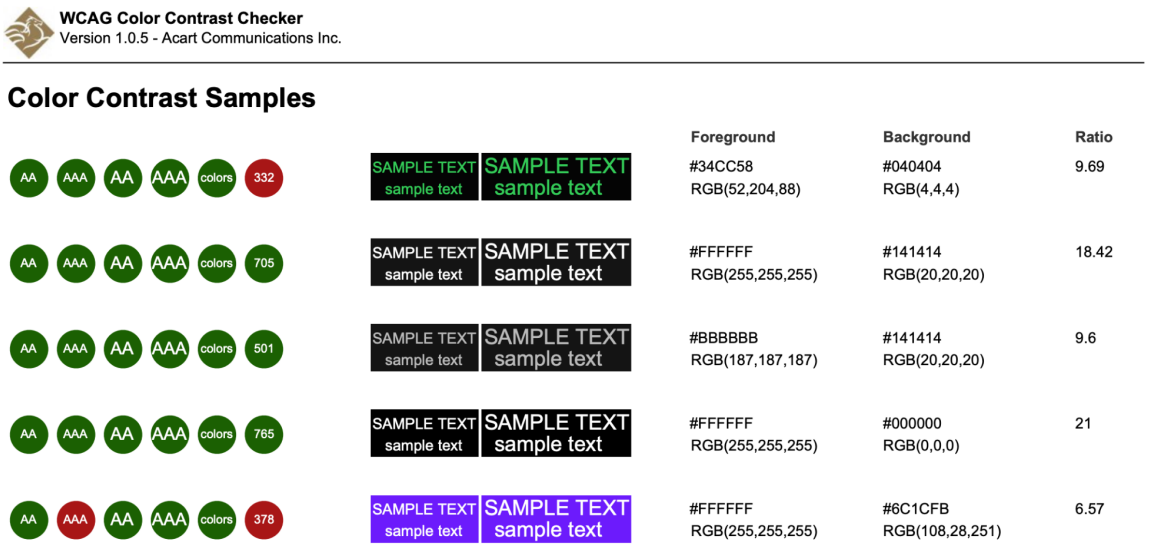
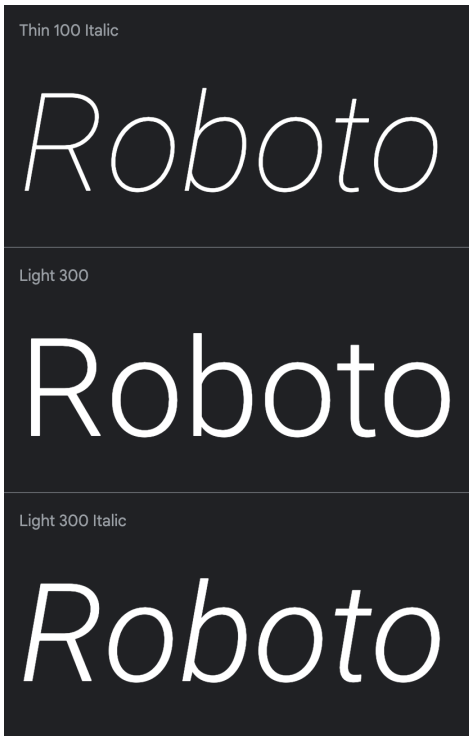


Figura 2.5. Captura de pantalla d'anàlisis de contrast [12].

Tipografia



A l'aplicació, va utilitzar la tipografia Roboto. Roboto és una font sans-serif àmpliament reconeguda i usada en disseny gràfic i desenvolupament web. Va ser creada per Christian Robertson i desenvolupada per Google com una font de codi obert. Aquesta tipografia es caracteritza pel seu disseny modern, net i versàtil, sense rematades a les terminacions de les lletres. El seu objectiu principal és proporcionar una llegibilitat excel·lent en diferents mides i plataformes, la qual cosa la fa especialment adequada per al seu ús en interfícies digitals. Amb una àmplia varietat de pesos i estils disponibles, Roboto permet crear jerarquies visuals i transmetre diferents tons o èmfasi al contingut de l'aplicació.[13]

Figura 2.7. Captura de pantalla parcial de google fonts, amb la tipografia Roboto.

Logotip

Figura 2.6.Logo Ludoscrum

El logotip de Ludoscrum ha estat dissenyat per capturar l'essència de la marca.

La forma principal del logotip és una lletra "L", que representa el nom de l'empresa. Aquesta “L” es presenta en una configuració tridimensional, evocant un espai contenidor obert que simbolitza l'obertura, la creativitat i les àmplies possibilitats que ofereix Ludoscrum.

El color verd, més brillant #3AE762, s'utilitza per a la lletra "L", ressaltant-ne la forma i destacant la identitat lúdica de la marca. A més, el color principal, #6E1FFF, s'utilitza en elements addicionals del logotip.

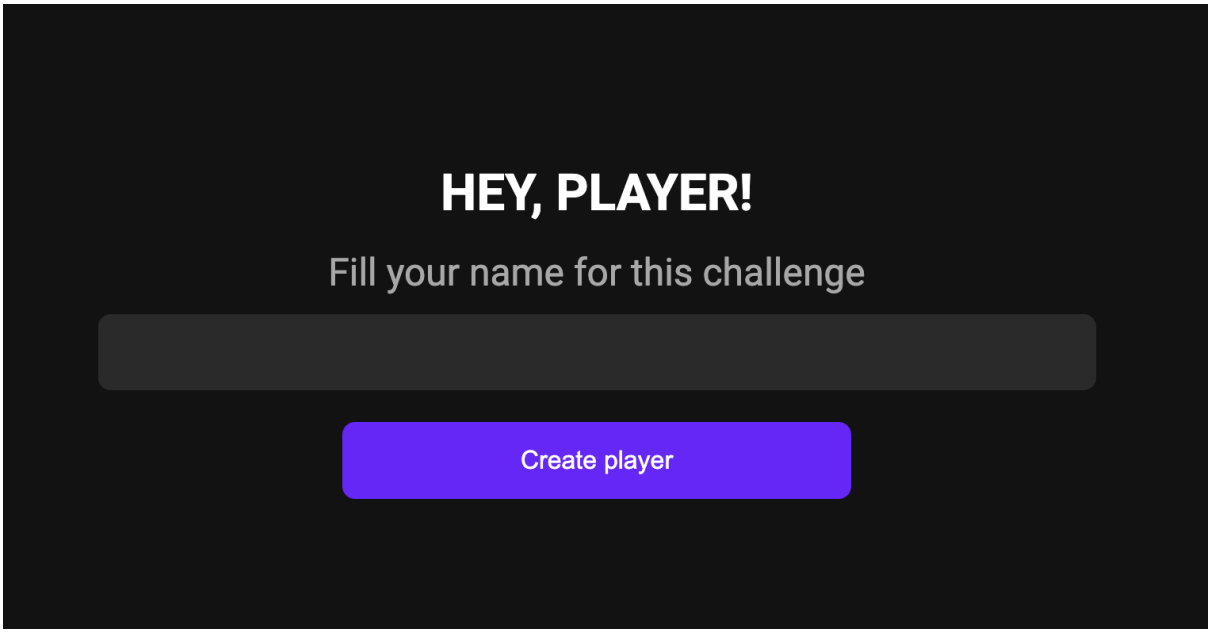


Figura 2.8. Captura de pantalla d'ingrés d'usuari, a l'aplicació de Ludoscrum

Navegació i usabilitat[14]

A l'encapçalament de l'aplicació, s'han inclòs botons de navegació bàsics com ara "Iniciar sessió" i "Registrar-se", que estan estratègicament ubicats per facilitar l'accés ràpid i senzill a les funcionalitats de l'aplicació. Aquests botons estan dissenyats pensant en la usabilitat, cosa que ajuda els usuaris a reconèixer-los fàcilment i a familiaritzar-se amb la seva ubicació i comportament. Cosa que promou l'eficiència i redueix la corba d'aprenentatge per als usuaris recurrents.

Quan l'usuari ha iniciat sessió a l'aplicació, es troben disponibles els botons d'autenticació de Slack, que ofereixen la possibilitat de connectar el compte de l'aplicació amb Slack. Aquesta integració agilitza la comunicació i la col·laboració, ja que els desafiaments es poden enviar directament a l'espai de treball de Slack de l'usuari. Aquest enfocament millora l'eficiència i l'accessibilitat en permetre que els usuaris accedeixin a les funcionalitats de l'aplicació des de la plataforma de treball preferida.

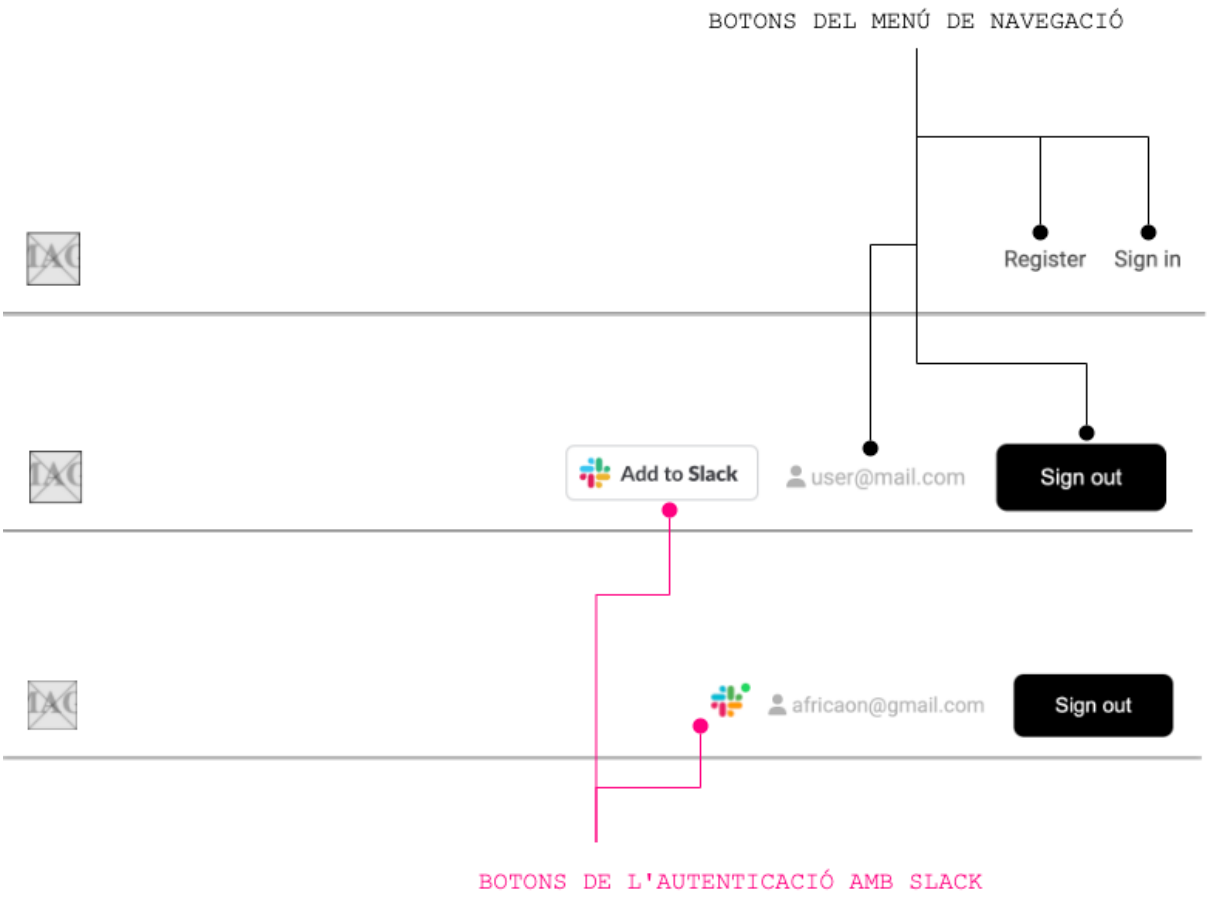


Figura 2.9. Disseny comparatiu entre els diferent estat de a la capçalera.

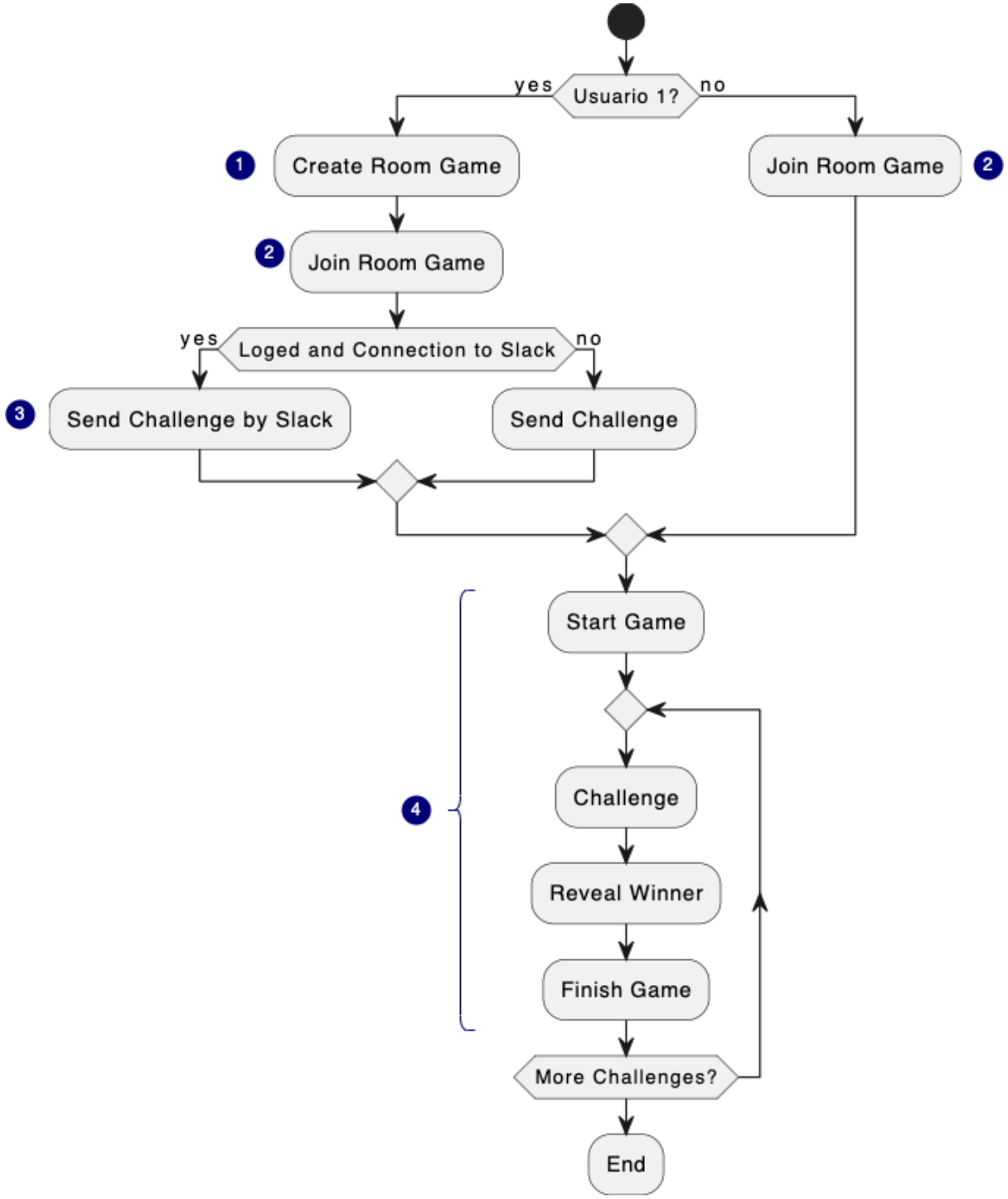


Figura 2.10. Gràfic UML del flux principal de Ludoscrum

El gràfic representa el flux d'accions que es fan a l'aplicació, per jugar:

- Comença el joc.
- Si sou l'usuari principal (usuari 1), se us dóna l'opció de crear una sala de joc, si no, vol dir que és l'usuari desafiat, i només teniu l'opció d'unir-vos a una existent.

- Si l'usuari principal està registrat i té una connexió a Slack, pot enviar desafiaments a través de Slack.
- Si l'usuari no està registrat, encara pot enviar un desafiament (URL) pel canal que preferiu.
- El joc comença i es repeteixen les accions següents fins que no hi hagi més desafiaments:
    - Es presenta un desafiament.
    - Es revela el guanyador del desafiament.
    - S'acaba el joc actual.
- Finalment, el joc arriba al final.

Aquest gràfic mostra el flux general del joc, on els usuaris poden crear o unir-se a un joc, enviar desafiaments i jugar diverses rondes fins que es completin tots els desafiaments. Tot i que la possibilitat de jugar diverses rondes en una mateixa sala de joc, no està implementada completament, si està dissenyada i desenvolupada en gran part.

Al diagrama assenyalo diferents passos, numerats de l'1 al 4, que corresponen a les diferents pàgines de l'aplicació, de les quals presento els seus dissenys a continuació, amb l'ajuda de wireframes.

1. Crear una sala de joc

En aquest pas se sol·licita a l'usuari principal un nom per a la sala del joc.

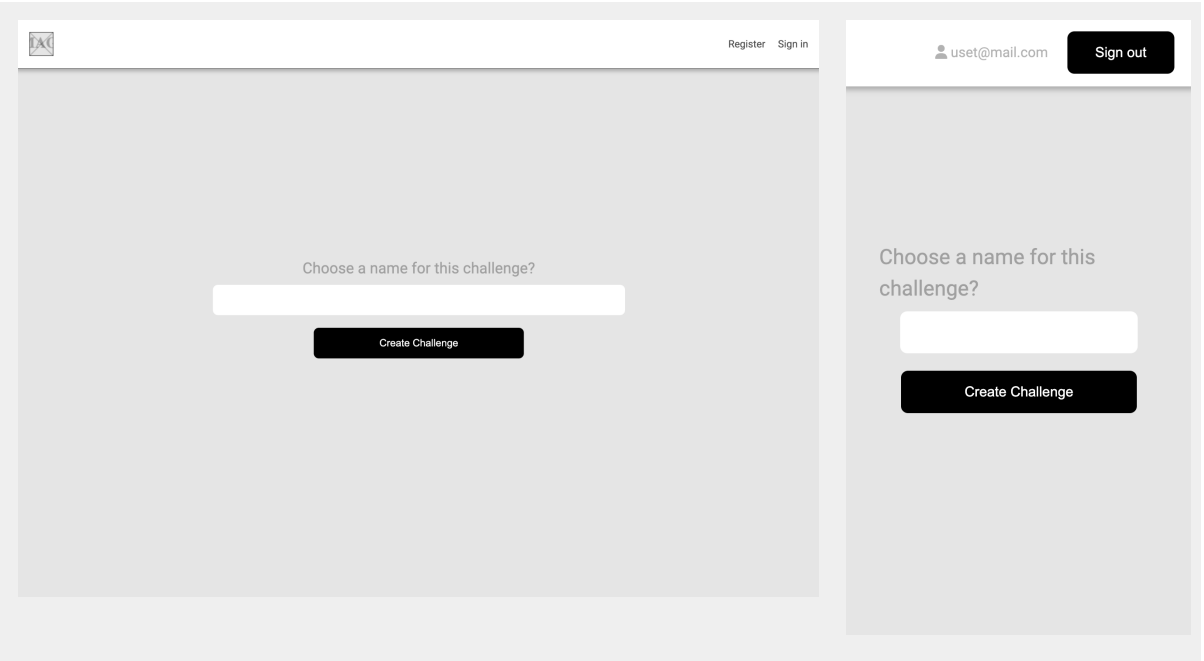


Figura 2.11. Wireframes de la pantalla "Crear una sala de joc" de Ludoscrum.

2. Unir-se al joc

Tots dos usuaris accedeixen a aquesta pàgina, on se'ls sol·licita l'àlies que volen fer servir en aquesta ocasió.

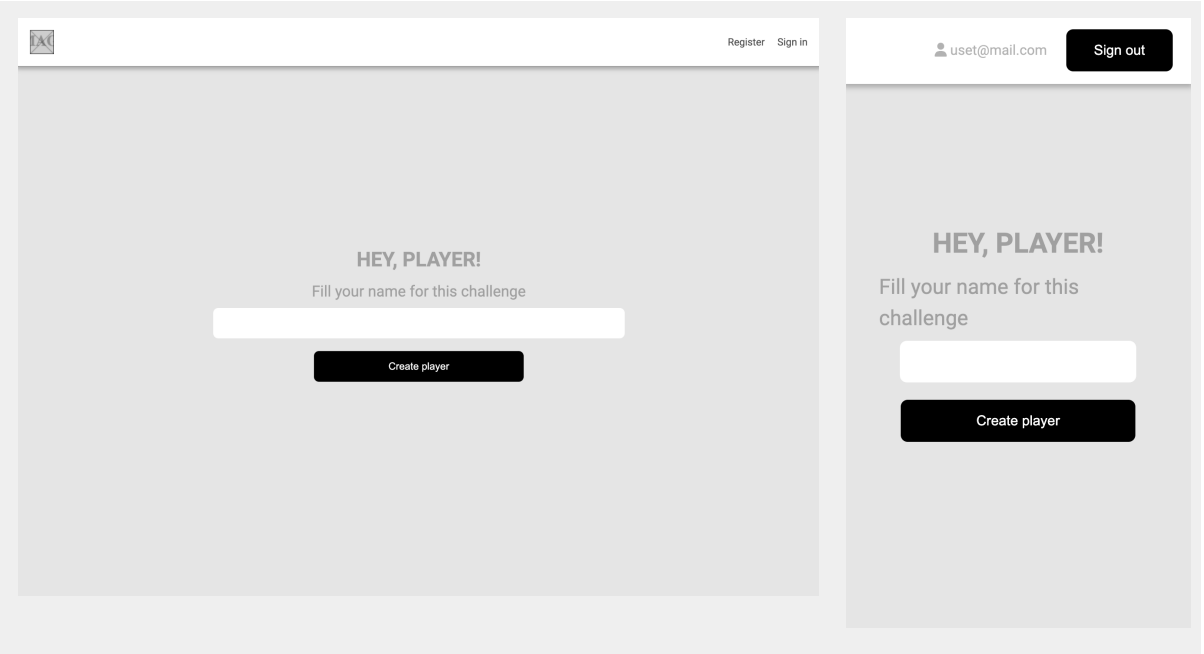


Figura 2.12. Wireframes de la pantalla "Unir-se al joc" de Ludoscrum.

3. Manar desafiament

Aquesta és una acció que realitzarà l'usuari principal. Si està allotjat i tenen la connexió amb Slack, podrà enviar el desafiament per aquest canal, si no se li proporciona un enllaç que haurà de fer arribar al seu oponent.

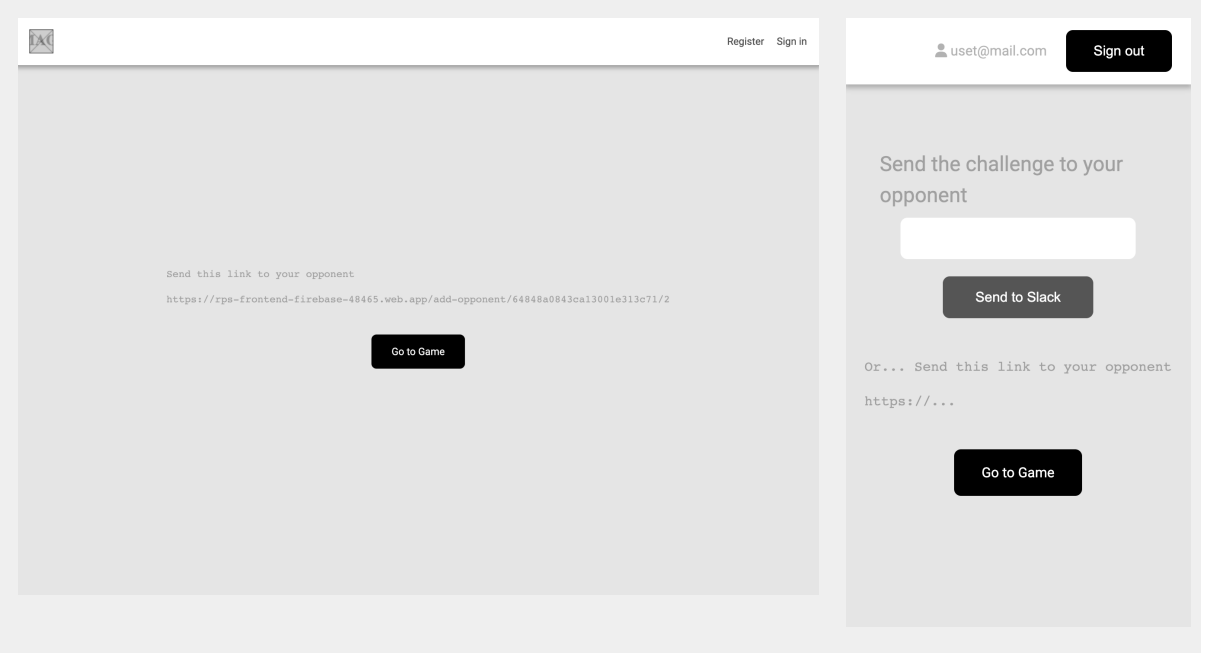


Figura 2.13. Wireframes de la pantalla "Manar desafiament" de Ludoscrum.

4. Finalment arriben a la pantalla del joc

En aquesta pantalla tots dos jugadors faran la seva jugada, i veuran el resultat de l'enfrontament

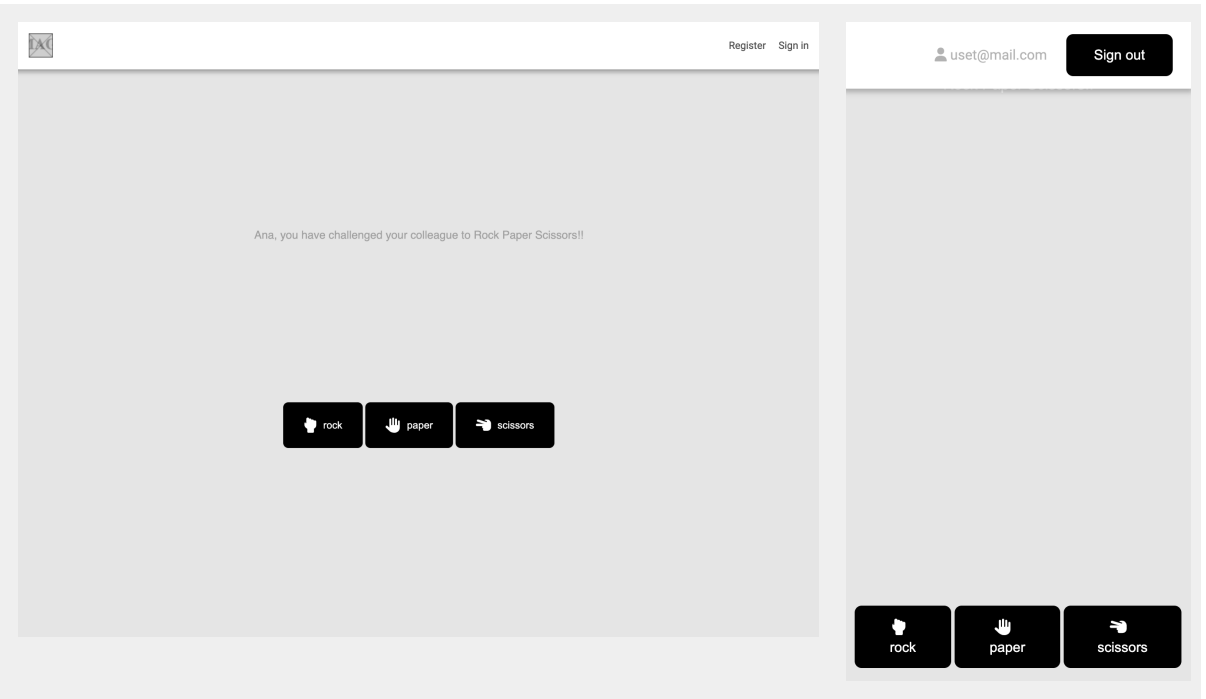


Figura 2.14. Wireframes de la "pantalla del joc" de Ludoscrum.

Aquest flux d'accions està basat en el que àmpliament s'utilitza per al poker de planificació, molt comú com a element de ludificació en entorn scrum. Ho he triat per aquest motiu i perquè, així, s'obtenen beneficis significatius en termes d'usabilitat, com ara:

- Reducció de la corba d'aprenentatge: Els usuaris no hauran d'aprendre un sistema o procés complex nou, ja que estaran familiaritzats amb el flux d'accions utilitzat. Això permet començar a fer ús de l'aplicació de manera ràpida i efectiva.

- Major eficiència: La familiaritat amb el flux d'accions facilita als usuaris fer les tasques de manera eficient, sense haver de pensar massa en com navegar per l'aplicació o què fer a continuació.

Aquests factors contribueixen a una millor experiència d'usuari i a una acceptació més gran de l'aplicació.

La Home

Mencció a part mereix la pàgina d'inici (home) de l'aplicació s'ha dissenyat específicament com la porta d'entrada principal per als usuaris. El seu objectiu és presentar un menú de jocs seleccionables, oferint als usuaris l'opció de triar el joc amb què volen interactuar. L'estructura de la maqueta s'ajusta a diferents mides de pantalla. A les pantalles grans, es mostra en tres columnes, mentre que en dispositius mòbils es presenta en una columna. Aquesta adaptabilitat millora la usabilitat i accessibilitat, permetent als usuaris explorar i seleccionar jocs de manera intuïtiva, sense importar el dispositiu utilitzat.

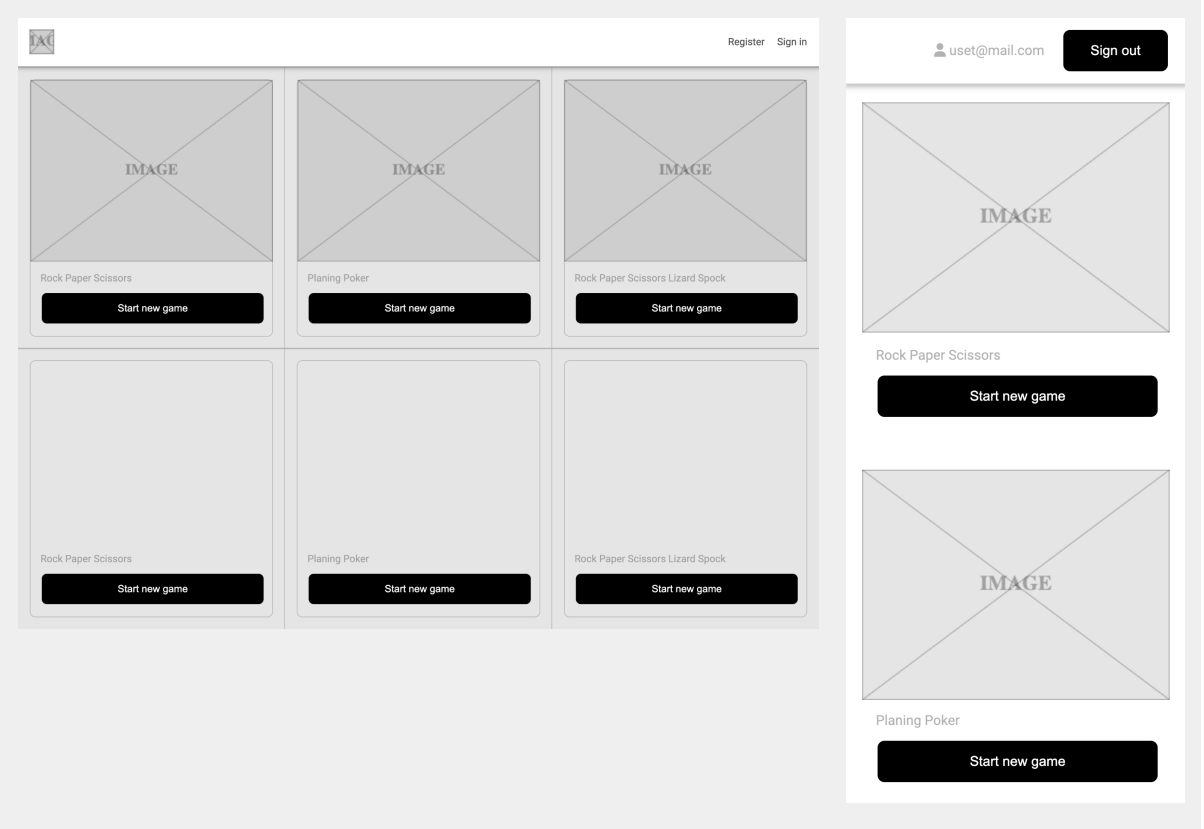


Figura 2.15. Wireframes de la "La Home" de Ludoscrum.

2.4. Stack de tecnologies

He triat construir el meu PoC amb una arquitectura MEVN, utilitzant MongoDB com a base de dades per a la persistència de dades, Node.js com a entorn de temps d'execució de JavaScript del costat del servidor, Express.js com framework de servidor web, i Vuejs com framework de frontend per a construir la interfície d'usuari.

Amb aquesta combinació de tecnologies, la meva aplicació serà capaç de manejar sol·licituds de manera eficient i flexible, proporcionant una experiència d'usuari interactiva i d'alta qualitat.

MongoDB és una base de dades NoSQL molt popular i escalable, la qual cosa és essencial per a l'èxit de qualsevol aplicació moderna. Node.s és molt ràpid, la qual cosa ho converteix en una excel·lent opció per entorn de temps d'execució del costat del servidor. Express.js és un framework de servidor web flexible i fàcil d'usar per a crear una API RESTful de manera senzilla. Finalment, Vue.js és un framework de frontend molt

potent i flexible que em permetrà crear una experiència d'usuari interactiva i d'alta qualitat.

MongoDB

MongoDB és una base de dades NoSQL[15] orientada a documents, que permet l'emmagatzematge de dades en format JSON. Es va començar a fer servir al 2009 i és una de les bases de dades NoSQL més populars i àmpliament utilitzades en l'actualitat.

Les principals característiques de MongoDB.

- És escalable: MongoDB està dissenyat per a manejar grans quantitats de dades i es pot escalar horitzontalment agregant més servidors a mesura que augmenta la càrrega de treball.
- És flexible: MongoDB fa servir una estructura de dades flexible i dinàmica que permet emmagatzemar diferents tipus de dades i canviar l'esquema de la base de dades sense interrompre l'aplicació.
- És ràpid: MongoDB és coneguda per la seva velocitat i rendiment. Les consultes són processades ràpidament i els resultats es poden retornar en mil·lisegons.
- Suporta operacions de lectura i escriptura complexes: MongoDB admet una àmplia gamma d'operacions de lectura i escriptura, incloent-hi operacions d'agregació, cerca geoespacial i text complet.
- Ofereix alta disponibilitat: MongoDB proporciona alta disponibilitat mitjançant la replicació de dades en diversos servidors, cosa que significa que si un servidor falla, les dades encara estan disponibles en altres servidors.
- És de codi obert: MongoDB és una base de dades de codi obert i està disponible de manera gratuïta sota la Llicència Pública de GNU.

A pesar que MongoDB és una base de dades NoSQL molt popular, existeixen alguns inconvenients que han de tenir-se en compte en considerar el seu ús en un projecte.

- Escalabilitat vertical limitada: A diferència d'altres bases de dades NoSQL, MongoDB té una capacitat d'escalabilitat vertical limitada, cosa que significa que pot haver-hi un límit en la quantitat de dades que pot manejar un sol servidor.
- Ús intensiu de recursos: MongoDB utilitza una gran quantitat de recursos del sistema, la qual cosa pot fer que sigui més costós en termes de recursos que altres bases de dades NoSQL.

- Requereix experiència tècnica: MongoDB pot ser més difícil d'usar que altres bases de dades NoSQL i pot requerir més experiència tècnica per a configurar i mantenir.
- No és adequat per a transaccions complexes: MongoDB no és adequat per a transaccions complexes i transaccions que involucrin diverses col·leccions.
- Falten algunes característiques importants: A MongoDB li manquen d'algunes característiques importants que es troben en altres bases de dades relacionals, com la integritat referencial.
- Risc de corrupció de dades: A causa del seu model d'emmagatzematge de documents, és possible que es produeixi una corrupció de dades en MongoDB, la qual cosa pot afectar la integritat de les dades.

Node.js

Node.js és un entorn de temps d'execució de JavaScript que s'utilitza per a construir aplicacions de servidor escalables i d'alt rendiment. Va ser llançat per primera vegada en 2009 i es basa en el motor V8 de Google Chrome. Algunes de les principals característiques de Node.js inclouen:

- És asincrònic: Node.js fa ús d'un model de E/S asincrònic que permet manejar múltiples sol·licituds simultàniament sense bloquejar el procés d'execució.
- És escalable: Igual que MongoDB, Node.js és altament escalable i està dissenyat per a manejar grans quantitats de sol·licituds simultànies.
- És ràpid: Node.js és conegut per la seva velocitat i rendiment. Fa servir una arquitectura de processament sense bloqueig i la capacitat de manejar sol·licituds asincròniques per a assegurar un alt rendiment.
- És fàcil d'aprendre: Node.js usa JavaScript com el seu llenguatge de programació, cosa que significa que els desenvolupadors que ja coneixen JavaScript poden aprendre i començar a usar Node.js ràpidament.
- Gran quantitat de paquets: Node.js té una àmplia comunitat de desenvolupadors que han creat una gran quantitat de paquets i biblioteques que es poden fer servir per a agregar funcionalitat addicional a les aplicacions Node.js.
- És de codi obert: Igual que MongoDB, Node.js és de codi obert i està disponible de manera gratuïta sota la Llicència MIT.

En considerar l'ús de Node.js en un projecte, és important tenir en compte les limitacions i desavantatges.

- Massa complexitat en projectes grans: Si bé Node.js és fàcil d'aprendre i usar en projectes més petits, pot tornar-se més complex en projectes grans i complexos que requereixen una major organització i estructuració del codi.
- Menys suport per a bases de dades relacionals: Encara que Node.js funciona bé amb bases de dades NoSQL com MongoDB, pot haver-hi menys suport per a bases de dades relacionals més tradicionals.
- Falta de biblioteques madures: Si bé hi ha moltes biblioteques disponibles per a Node.js, algunes d'elles poden no ser tan madures o estables com les disponibles en altres llenguatges de programació.
- Major ús de recursos: Node.js utilitza un model de E/S no bloquejant que permet manejar múltiples sol·licituds simultàniament, però això també significa que pot consumir més recursos del sistema que altres entorns de temps d'execució.
- Necessitat d'administrar dependències: Atès que Node.js fa ús de moltes biblioteques i dependències, pot ser necessari administrar i actualitzar aquestes dependències regularment per a mantenir la seguretat i el rendiment de l'aplicació.
- Falta de suport per a algunes característiques: Algunes característiques que es troben en altres llenguatges de programació poden no estar disponibles en Node.js, la qual cosa pot limitar la funcionalitat d'algunes aplicacions.

Express

Express, és un popular framework de Node.js utilitzat per a construir aplicacions web i API. És conegut per ser minimalista i flexible, la qual cosa permet crear aplicacions web i API de manera ràpida i senzilla.

- És fàcil d'aprendre: Express és un framework senzill i fàcil d'aprendre per als desenvolupadors que ja estan familiaritzats amb Node.js.
- És altament personalitzable: Express permet als desenvolupadors crear aplicacions web i API personalitzades que s'adaptin a les seves necessitats específiques.
- Té una gran quantitat de middleware: Express té una gran quantitat de middleware disponible que permet als desenvolupadors agregar fàcilment funcionalitats com a autenticació, compressió i gestió de cookies a les seves aplicacions web i API.
- És compatible amb una àmplia gamma de mòduls: Express és compatible amb una àmplia gamma de mòduls de Node.js, la qual cosa permet als desenvolupadors integrar fàcilment diferents tecnologies i serveis en les seves aplicacions web i API.

- És escalable: Express és fàcilment escalable i pot manejar grans quantitats de trànsit i sol·licituds simultànies.
- És adequat per a la construcció d'API RESTful: Express és especialment adequat per a la construcció d'API RESTful, ja que proporciona un marc de treball eficient per a la creació de rutes i gestió de sol·licituds i respostes HTTP.

Encara que Express és un framework de Node.js àmpliament utilitzat, també existeixen alguns inconvenients.

- No és adequat per a aplicacions de gran grandària: Encara que Express és escalable, no és adequat per a aplicacions de gran grandària o complexos que poden requerir un framework més robust i estructurat.
- Pot haver-hi problemes de seguretat: A l'ésser un framework minimalista, Express no proporciona totes les funcionalitats de seguretat que es necessiten per a protegir completament una aplicació web o API.
- Els desenvolupadors han de ser conscients d'això i prendre mesures per a protegir la seva aplicació.
- Requereix més codi: Pel fet que Express és un framework minimalista, pot requerir que els desenvolupadors escriguin més codi que si usessin un framework més complet.

Vue

Vue és un framework de JavaScript progressiu utilitzat per a construir interfícies d'usuari web i aplicacions d'una sola pàgina (SPA). Va ser llançat en 2014 i s'ha convertit en un dels frameworks més populars en l'actualitat a causa de la seva facilitat d'ús i flexibilitat.

Les principals característiques de Vue són les següents:

- Components reutilitzables: Vue utilitza components per a dividir la interfície d'usuari en peces reutilitzables, la qual cosa facilita la construcció d'interfícies d'usuari complexes.
- Directives: Vue utilitza directives per a enllaçar dades amb elements del DOM, la qual cosa permet una actualització automàtica de la interfície d'usuari quan canvien les dades.
- Renderitzat declaratiu: Vue utilitza un enfocament declaratiu per al renderitzat de la interfície d'usuari, cosa que significa que els desenvolupadors només han d'especificar com hauria de veure's la interfície d'usuari en un estat determinat.

- Reactivitat: Vue utilitza un sistema de reactivitat per a rastrejar els canvis en les dades i actualitzar automàticament la interfície d'usuari en conseqüència.
- Vue CLI: Vue CLI és una interfície de línia de comandos que proporciona una configuració fàcil i ràpida del projecte, així com eines per a proves, empaquetat i implementació.
- Flexibilitat: Vue és un framework molt flexible que es pot integrar fàcilment amb altres biblioteques i marcs, la qual cosa ho fa ideal per a projectes de qualsevol grandària.

Encara que Vue és una eina habitual per al desenvolupament web, cal tenir en compte alguns inconvenients

- Mida del paquet: La grandària del paquet de Vue pot ser més gran que el d'uns altres frameworks, la qual cosa pot afectar el temps de càrrega de l'aplicació.
- Corba d'aprenentatge: Encara que Vue és relativament fàcil d'aprendre en comparació amb uns altres frameworks, encara pot tenir una corba d'aprenentatge per als desenvolupadors nous en la tecnologia.
- Suport de la comunitat: Encara que Vue té una comunitat activa i creixent, pot tenir menys recursos i suport que alguns dels frameworks més establerts.

2.5. Justificació de l'Slack tecnològic triat

He triat MongoDB, Vue.js i Node.js (MEVN Stack) per al projecte basen-me en les fortaleses d'aquesta arquitectura.
MEVN Stack ofereix avantatges com una base de dades flexible i escalable, una interfície d'usuari reactiva i components reutilitzables, i processos eficients. A més, l'ús de JavaScript en tots dos costats (frontend i backend) simplifica els processos i fomenta la coherència. La combinació d'aquestes tecnologies enforteix el desenvolupament, permetent compartir codi i lògica entre el frontend i el backend, mentre es garanteix l'escalabilitat, flexibilitat i una experiència d'usuari dinàmica.

Com pretenc dividir la plataforma de ludificació en jocs independents, m'asseguro que el projecte en conjunt no se sobredimensioni. Aquesta divisió en jocs individuals permet enfocar-se a desenvolupar i llançar iterativa-ment cada joc de manera independent, evitant l'acumulació de funcionalitats complexes i optimitzant el temps i els recursos dedicats a cada joc en particular. En adoptar aquest enfocament, s'aconsegueix una major agilitat i flexibilitat en el desenvolupament, permetent el lliurament primerenc de

productes mínimament viables (MVP) i l'obtenció de retroalimentació i validació de l'usuari de forma més ràpida. Així mateix, es facilita el manteniment i actualització de cada joc de manera individual, sense comprometre l'estabilitat i el rendiment general. Aquesta estratègia ajuda a mantenir un equilibri adequat entre la complexitat i l'abast del projecte en conjunt.

3.Resultats

3.1. Desenvolupament i desplegament del backend

El desenvolupament backend de l'aplicació es basa en un projecte de Node.js amb Express.js i Mongoose, aquest és el package.json:

`package.json`

El fitxer "package.json" és un fitxer de configuració utilitzat en projectes de Node.js per administrar les dependències, scripts i metadades del projecte.

```
{
 "name": "rps",
 "version": "1.0.0",
 "description": "",
 "main": "server.js",
 "scripts": {
   "start": "node server.js"
 },
 "author": "acampmanyb",
 "license": "ISC",
 "dependencies": {
   "axios": "^1.4.0",
   "cors": "^2.8.5",
   "dotenv": "^10.0.0",
   "express": "^4.17.1",
   "form-data": "^4.0.0",
   "mongoose": "^5.8.10"
 }
}
```

Figura 3.1. Mostra de codi de l'arxiu package.json del backend

Pel que fa a la base de dades, s'utilitza MongoDB com a sistema de gestió de bases de dades NoSQL. Específicament, s'utilitza MongoDB Atlas, que és el servei al núvol ofert per MongoDB. MongoDB Atlas simplifica l'administració i la configuració de clústers de bases de dades MongoDB al núvol, proporcionant una infraestructura escalable i segura per emmagatzemar les dades de l'aplicació.

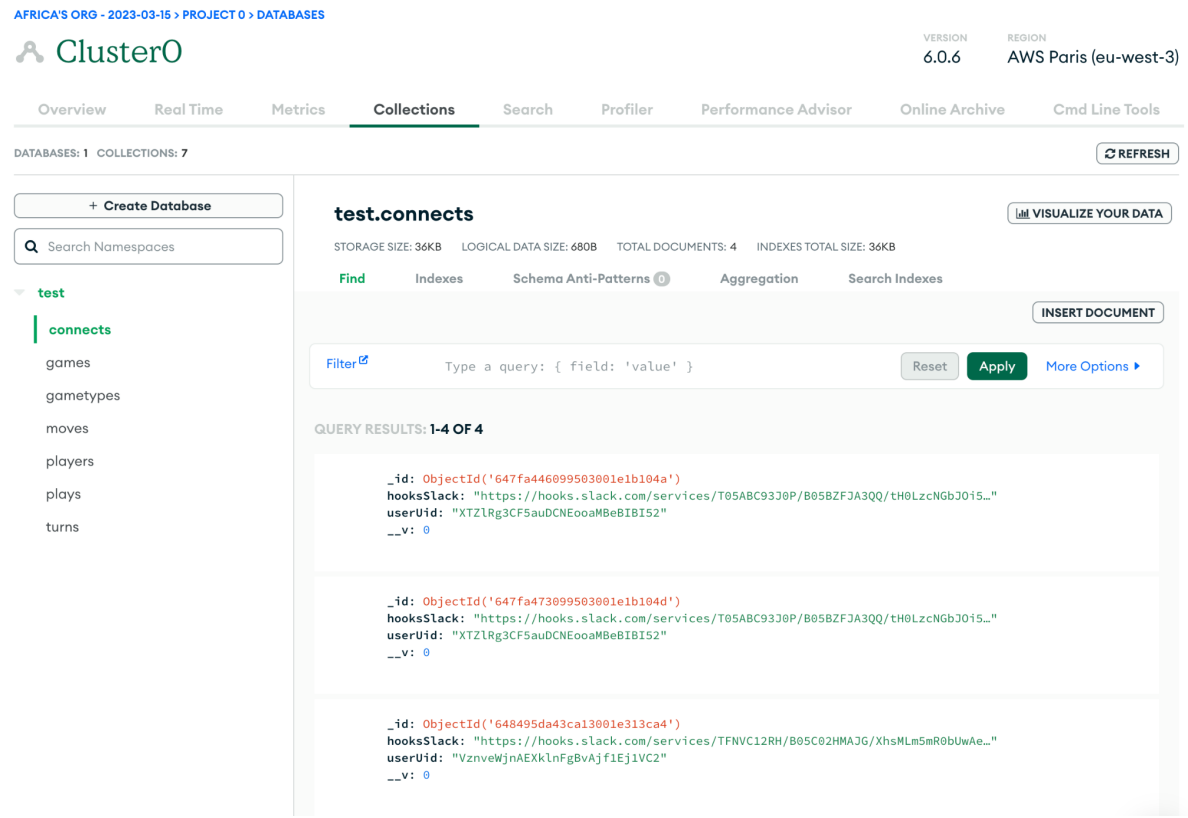


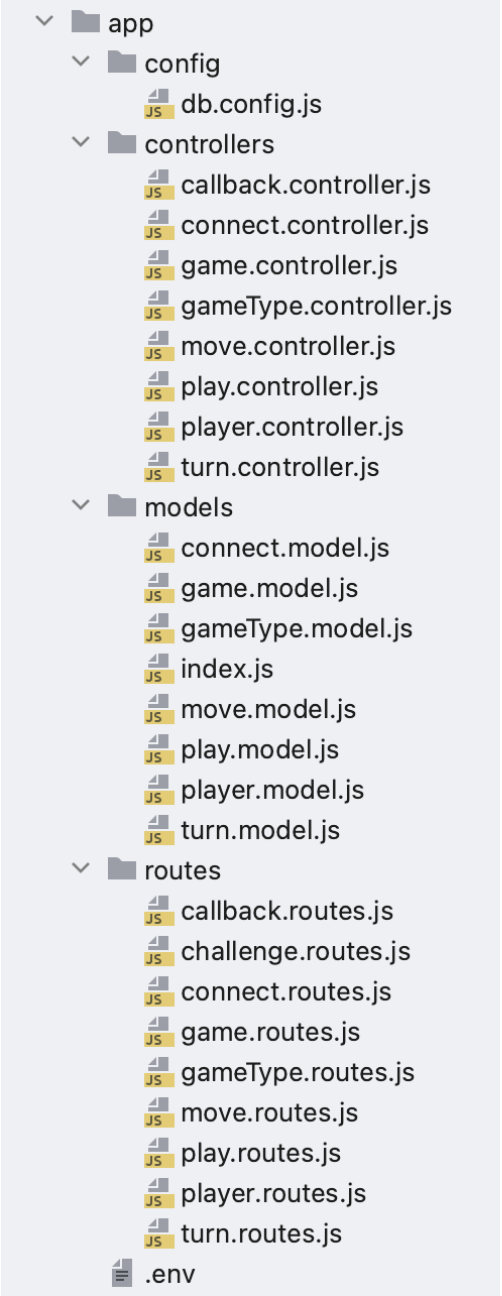
Figura 3.2. Vista del backoffice de la base de dades de test que s'utilitza a la PoC

El document de configuració de la base de dades al projecte **app/config/db.config.js**

```
const {
 DB_USER,
 DB_PASSWORD,
 DB_HOST
} = process.env;

module.exports = {
 url:
`mongodb+srv://${DB_USER}:${DB_PASSWORD}@${DB_HOST}/?retryWrites=true&w=majority`
};
```

Figura 3.3. Mostra de codi de l'arxiu db.config.js del backend, on

DB_USER, DB_PASSWORD, DB_HOST es troben definits al document .env.



Node.js amb Express.js i Mongoose tenen un paper crucial a les capes Model i Controlador del model MVC. Mongoose actua com un ODM per administrar els models de dades i facilitar la interacció amb la base de dades MongoDB. Express.js s'encarrega de manejar les rutes i les sol·licituds HTTP, coordinant la lògica de negoci als controladors.

A l'aplicació, les capes Model i Controlador es poden identificar com a directoris que agrupen els documents i scripts corresponents, permetent una estructura organitzada i modular.

Figura 3.4. Vista de carpetes del projecte backend.

server

El document server.js en aquest projecte exerceix el paper de fitxer principal o punt d'entrada del servidor. És on es configuren i s'inicien els components principals del servidor utilitzant Node.js i Express.js.
Al fitxer server.js, es defineixen i es configuren les rutes i els controladors necessaris per gestionar les sol·licituds i les respostes HTTP. També s'estableix la connexió amb la base de dades MongoDB amb Mongoose.

A més, conté configuracions addicionals, com ara la configuració del port on s'executarà el servidor, la gestió d'errors i altres paràmetres específics del projecte.

server.js

```
require("dotenv").config();
const express = require("express");
const axios = require("axios");
const cors = require("cors");

const app = express();

var corsOptions = {
 origin:
["https://ludoscrum.github.io","http://localhost:8080","https://rps
-frontend-firebase-48465.web.app"]
};

app.use(cors(corsOptions));

// parse requests of content-type - application/json
app.use(express.json());

// parse requests of content-type -
application/x-www-form-urlencoded
app.use(express.urlencoded({ extended: true }));

const db = require("./app/models");

console.log(db.url);

db.mongoose
   .connect(db.url, {
     useNewUrlParser: true,
     useUnifiedTopology: true
   })
   .then(() => {
     console.log("Connected to the database!");
   })
   .catch(err => {
     console.log("Cannot connect to the database!", err);
     process.exit();
   });

// simple route
app.get("/", (req, res) => {
 res.json({ message: "Welcome to rps application." });
```

```
});

require("./app/routes/game.routes")(app);
require("./app/routes/player.routes")(app);
require("./app/routes/move.routes")(app);
require("./app/routes/play.routes")(app);
require("./app/routes/turn.routes")(app);
require("./app/routes/gameType.routes")(app);
require("./app/routes/challenge.routes")(app);
require("./app/routes/callback.routes")(app);
require("./app/routes/connect.routes")(app);

// set port, listen for requests
const PORT = process.env.PORT || 8080;
app.listen(PORT, () => {
 console.log(`Server is running on port ${PORT}.`);
});
```

Figura 3.5. Codi de l'arxiu server.js del backend.

Exemple de rutes

`game.routes.js`

Aquesta funció configura les rutes per gestionar les operacions CRUD (crear, llegir, actualitzar i eliminar) dels jocs a l'aplicació.
Finalment, totes les rutes configurades al router es munten al path "/api/games" utilitzant app.use("/api/games", router), la qual cosa significa que les rutes estaran disponibles en aquest endpoint de l'API.

```
module.exports = app => {
 const games = require("../controllers/game.controller.js");
 var router = require("express").Router();
 // Create a new Game
 router.post("/", games.createGame);

 // Retrieve all Games
 router.get("/", games.findAllGamesByTitle);

 // Retrieve all active Games
 router.get("/active", games.findAllActiveGame);

 // Retrieve a single Game with id
 router.get("/:id", games.findOneGameById);
```

```
  // Update a Game with id
  router.put("/:id", games.updateGameById);

  // Delete a Game with id
  router.delete("/:id", games.deleteGameById);

  // Create a new Tutorial
  router.delete("/", games.deleteAllGame);

  // insert a new player into a game
  router.post("/:id/player", games.insertPlayerIntoGame);

  app.use("/api/games", router);
};
```

Figura 3.6. Codi de l'arxiu game.routes.js del backend.

Exemple de model

`game.model.js`

Aquesta funció defineix i configura l'esquema de dades per al model "Game" a l'aplicació, aquest defineix l'estructura i les propietats dels documents "Game" a la base de dades MongoDB. Inclou els camps següents:

- "title": un camp de tipus String que representa el títol del joc.
- "active": un camp de tipus Boolean que indica si el joc està actiu o no.
- "turns": un camp de tipus Number que representa el nombre de torns del joc.
- "type": un camp de tipus Array que conté referències als tipus de joc relacionats.
- "players": un camp de tipus Array que conté referències als jugadors relacionats.
- "maxPlayers": un camp de tipus Number que indica el nombre màxim de jugadors permesos al joc.

A més, s'utilitza l'objecte "timestamps: true" per afegir automàticament les propietats "createdAt" i "updatedAt" als documents, que registren la data i l'hora de creació i actualització respectivament.

Es defineix el mètode "toJSON" a l'esquema per personalitzar la representació JSON dels documents "Game", excloent-ne les propietats "__v" i "_id", i afegint la propietat "id" que és el valor de "_id".
Finalment, es crea i s'exporta el model "Game" utilitzant l'esquema definit.

```
module.exports = mongoose => {
```

```
 var schema = mongoose.Schema(
   {
       title: {
           type: String,
           trim: true
       },
       active: Boolean,
       turns: Number,
       type: [{ type: mongoose.Schema.Types.ObjectId, ref:
'gameType' }],
       players: [{ type: mongoose.Schema.Types.ObjectId, ref:
'player' }],
       maxPlayers: Number,
   },
   { timestamps: true }
 );

 schema.method("toJSON", function() {
   const { __v, _id, ...object } = this.toObject();
   object.id = _id;
   return object;
 });

 const Game = mongoose.model("game", schema);
 return Game;
};
```

Figura 3.7. Codi de l'arxiu game.model.js del backend.

Exemple de controlador

`game.controller.js`

El codi proporciona els controladors per interactuar amb el model "Game" al projecte. Aquests controladors s'encarreguen de realitzar diferents operacions CRUD (Crear, Llegir, Actualitzar, Eliminar) a la base de dades en resposta a les sol·licituds HTTP rebudes.
Aquests controladors són responsables de manejar les diferents operacions relacionades amb el model "Game" al projecte i proporcionen la lògica necessària per interactuar amb la base de dades i enviar respostes adequades a les sol·licituds HTTP rebudes.

```
const db = require("../models");
const Game = db.games;
```

```
// Create and Save a new Game
exports.createGame = (req, res) => {
 // Validate request
 if (!req.body.title) {
   res.status(400).send({ message: "Content can not be empty!" });
   return;
 }
 // Create a Game
 const game = new Game({
   title: req.body.title,
   active: req.body.active ? req.body.active : false,
   turns: req.body.turns ? req.body.turns : 1,
   type: req.body.type,
   players: req.body.players,
   maxPlayers: req.body.maxPlayers ? req.body.maxPlayers : 2
 });

 // Save Game in the database
 game.save(game).then(data => {
     res.send(data);
   })
   .catch(err => {
     res.status(500).send({
       message:
         err.message || "Some error occurred while creating the Game. (err.
5xx)"
     });
   });
};
// Find Game/s by title.
exports.findAllGamesByTitle = (req, res) => {
 const title = req.query.title;
 var condition = title ? { title: { $regex: new RegExp(title), $options: "i"
} } : {};
 Game.find(condition)
   .then(data => {
     res.send(data);
   })
   .catch(err => {
     res.status(500).send({
       message:
         err.message || "Some error occurred while retrieving games."
     });
   });
};
...
```

Figura 3.8. Mostra de codi de l'arxiu game.controller.js del backend.

Entorn local de desenvolupament

A la primera versió del backend, es va utilitzar un fitxer Docker Compose per configurar i executar un entorn de desenvolupament compost per un contenidor de MongoDB i un contenidor de l'aplicació. Aquests dos serveis es comunicaven entre si i podien ser accedits des del host a través dels ports que es van mapejar. L'objectiu era proporcionar un entorn integrat i funcional per al desenvolupament i la prova de l'aplicació, on la base de dades MongoDB i l'aplicació poguessin interactuar de manera fluida.[16]

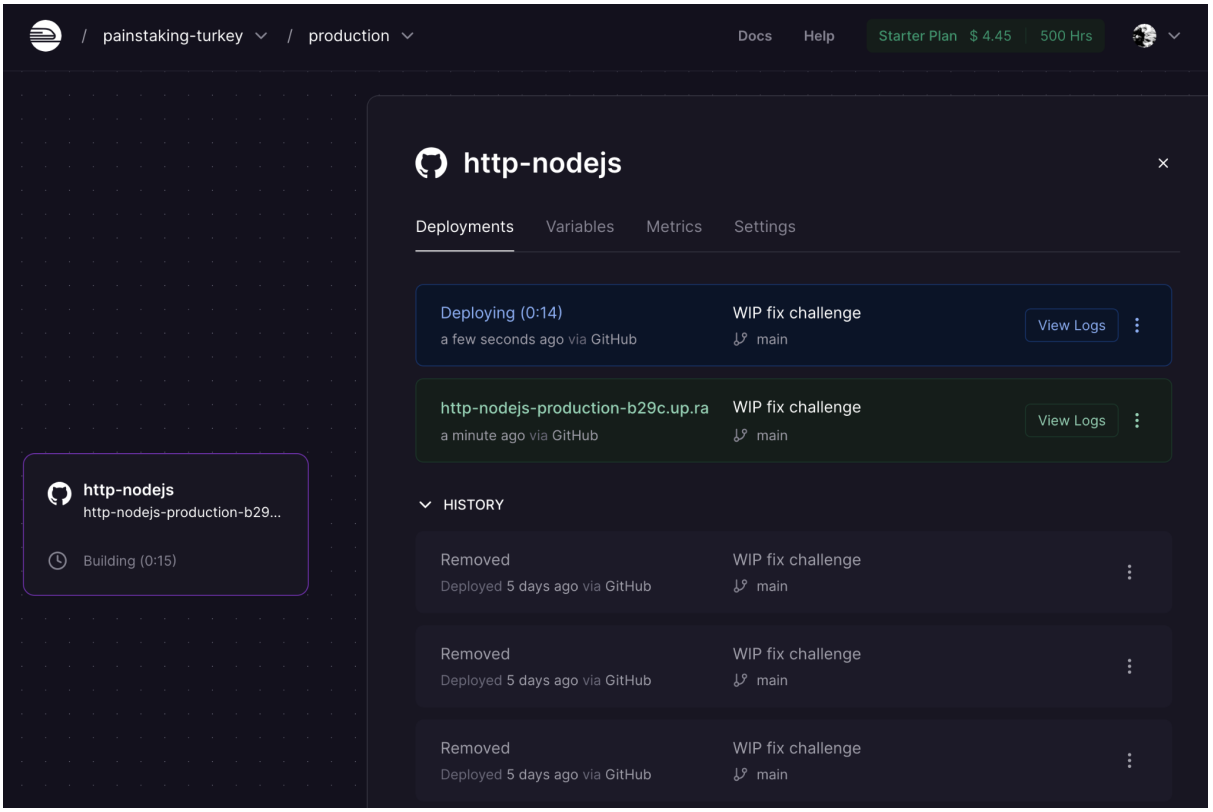
```
version: "3.8"

services:
 mongodb:
   image: mongo:5.0.2
   restart: unless-stopped
   env_file: ./.env
   environment:
     - MONGO_INITDB_ROOT_USERNAME=$MONGODB_USER
     - MONGO_INITDB_ROOT_PASSWORD=$MONGODB_PASSWORD
   ports:
     - $MONGODB_LOCAL_PORT:$MONGODB_DOCKER_PORT
   volumes:
     - db:/data/db
 app:
   depends_on:
     - mongodb
   build: ./rps-app
   restart: unless-stopped
   env_file: ./.env
   ports:
     - $NODE_LOCAL_PORT:$NODE_DOCKER_PORT
   environment:
     - DB_HOST=mongodb
     - DB_USER=$MONGODB_USER
     - DB_PASSWORD=$MONGODB_PASSWORD
     - DB_NAME=$MONGODB_DATABASE
     - DB_PORT=$MONGODB_DOCKER_PORT
   stdin_open: true
   tty: true

volumes:
 db:
```

Figura 3.8. Codi de l'arxiu docker-compose.yml.

Lamentablement, en intentar desplegar el projecte a Railway, em vaig trobar amb dificultats. En lloc de bregar amb la complexitat addicional que el Docker Compose generava, vaig prendre la decisió de simplificar el projecte. Això va implicar eliminar aquesta peça, optant per una solució més senzilla i menys problemàtica per garantir un desplegament exitós.

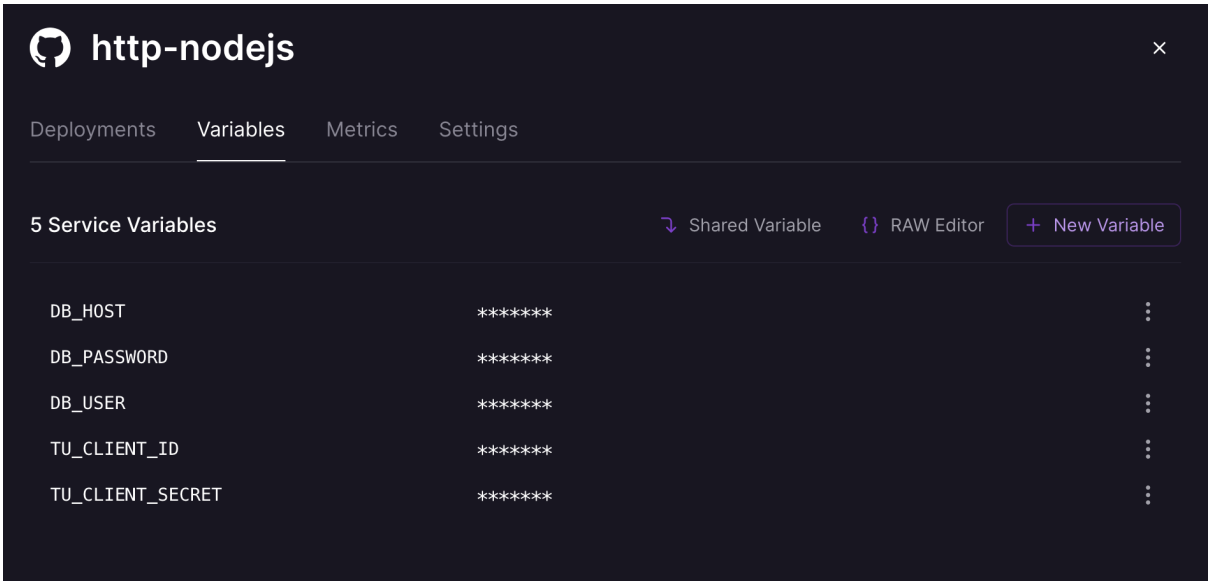
Desplegament

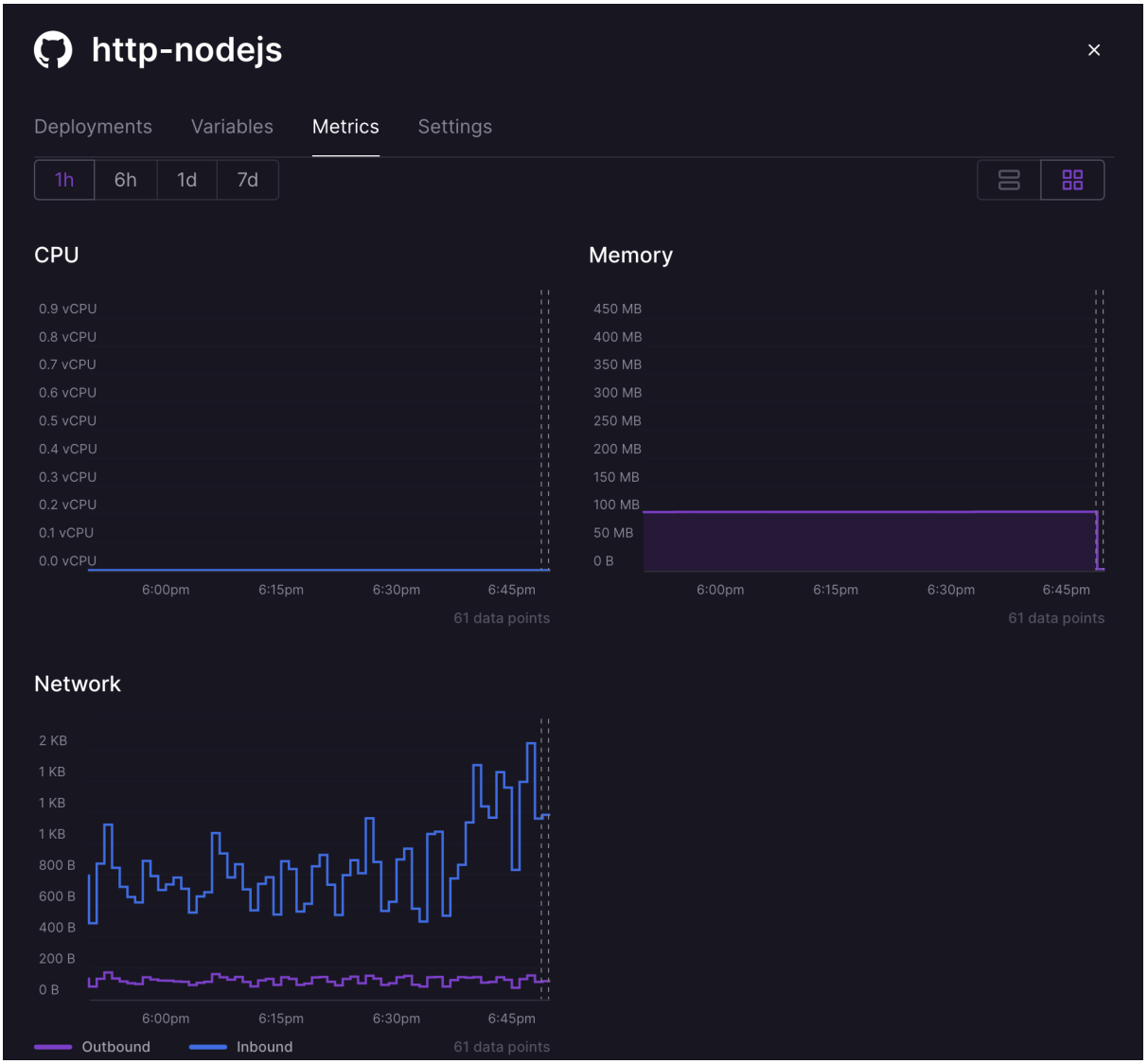
Després d'estudiar altres alternatives, finalment vaig decidir desplegar el meu projecte a Railway.
Aquesta plataforma proporciona una manera senzilla i eficient de desplegar i administrar aplicacions. No he de dedicar temps, ni recursos a la configuració de l'entorn, la gestió de sistemes o l'escalabilitat, cosa que em permet centrar-me en el desenvolupament del projecte sense preocupar-me per la infraestructura subjacent.



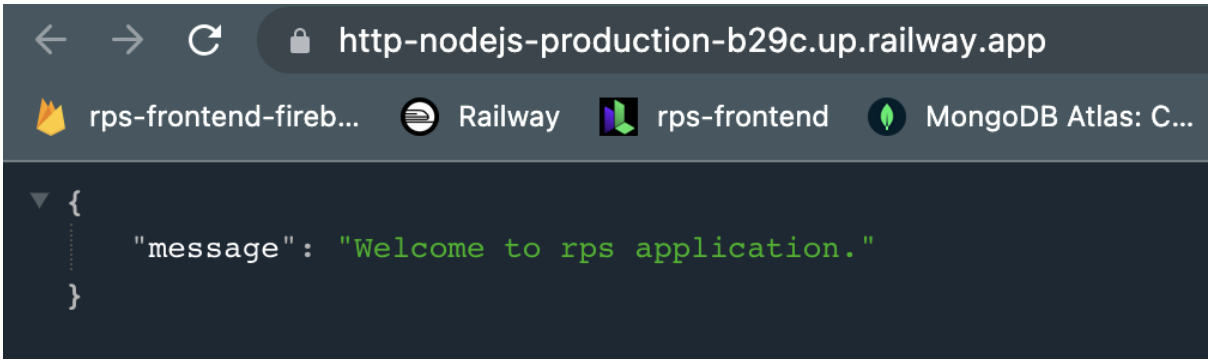
Connectant el meu repositori de GitHub amb Railway, puc realitzar desplegaments automàtics cada vegada que faig canvis al meu codi. Això facilita el procés d'implementació i em permet iterar ràpidament a l'aplicació.

A més, Railway permet configurar variables d'entorn i gestionar les meves dependències. Puc definir les variables necessàries, com ara claus d'API o credencials de base de dades, sense comprometre la seguretat de la meva aplicació.


http-nodejs
Deployments
Variables
Metrics
Settings
5 Service Variables
Shared Variable
RAW Editor
New Variable
DB_HOST
DB_PASSWORD
DB_USER
TU_CLIENT_ID
TU_CLIENT_SECRET
****


http-nodejs
Deployments
Variables
Metrics
Settings
1h
6h
1d
7d
CPU
0.9 vCPU
0.8 vCPU
0.7 vCPU
0.6 vCPU
0.5 vCPU
0.4 vCPU
0.3 vCPU
0.2 vCPU
0.1 vCPU
0.0 vCPU
6:00pm
6:15pm
6:30pm
6:45pm
61 data points
Memory
450 MB
400 MB
350 MB
300 MB
250 MB
200 MB
150 MB
100 MB
50 MB
0 B
Network
2 KB
1 KB
800 B
600 B
400 B
200 B
0 B
Outbound
Inbound

Quan el projecte es desplega a Railway, puc accedir-hi a través de l'URL proporcionada per la plataforma. Railway s'encarrega de proporcionar un entorn segur i fiable per a l'aplicació.



3.2. Desenvolupament i desplegament del frontend

El frontend s'ha generat amb Vue, concretament amb la versió 3.3.0. Vue.js és un framework de JavaScript que s'utilitza per construir interfícies d'usuari interactives i reactives.
A l'arquitectura MVC (Model-Vista-Controlador), Vue.js s'encarrega de la capa de la vista (View). Això vol dir que s'utilitza per crear la interfície d'usuari i manejar la lògica relacionada amb la presentació. En fer servir aquesta versió, puc aprofitar les característiques i millores introduïdes en aquesta. Proporciona una sintaxi declarativa i fàcil de fer servir que permet crear components reutilitzables i construir una interfície d'usuari dinàmica.

Aquest és el package.json

`package.json`

En un projecte Vue.js, el fitxer "package.json" conté metadades del projecte, scripts d'execució i les dependències necessàries per al desenvolupament i construcció de l'aplicació Vue.

```
"name": "rps-frontend",
"version": "0.1.0",
"private": true,
"scripts": {
```

```
    "serve": "vue-cli-service serve",
    "build": "vue-cli-service build",
    "test:unit": "vue-cli-service test:unit",
    "lint": "vue-cli-service lint"
  },
  "dependencies": {
    "@fortawesome/fontawesome-free": "^6.4.0",
    "@fortawesome/fontawesome-svg-core": "^6.4.0",
    "@fortawesome/free-regular-svg-icons": "^6.4.0",
    "@fortawesome/free-solid-svg-icons": "^6.4.0",
    "@fortawesome/vue-fontawesome": "^3.0.3",
    "axios": "^1.3.5",
    "core-js": "^3.8.3",
    "firebase": "^9.22.1",
    "firebase-tools": "^12.3.0",
    "pinia": "^2.1.3",
    "register-service-worker": "^1.7.2",
    "vue": "^3.3.0",
    "vue-router": "^4.0.3"
  },
  "devDependencies": {
    "@types/axios": "^0.14.0",
    "@types/jest": "^27.0.1",
    "@typescript-eslint/eslint-plugin": "^5.4.0",
    "@typescript-eslint/parser": "^5.4.0",
    "@vue/cli-plugin-babel": "^5.0.0",
    "@vue/cli-plugin-eslint": "^5.0.0",
    "@vue/cli-plugin-pwa": "^5.0.0",
    "@vue/cli-plugin-router": "^5.0.0",
    "@vue/cli-plugin-typescript": "^5.0.0",
    "@vue/cli-plugin-unit-jest": "^5.0.0",
    "@vue/cli-service": "^5.0.0",
    "@vue/eslint-config-typescript": "^9.1.0",
    "@vue/test-utils": "^2.0.0-0",
    "@vue/vue3-jest": "^27.0.0-alpha.1",
    "babel-jest": "^27.0.6",
    "eslint": "^7.32.0",
    "eslint-plugin-vue": "^8.0.3",
    "jest": "^27.5.1",
    "sass": "^1.32.7",
    "sass-loader": "^12.0.0",
    "ts-jest": "^27.1.5",
    "typescript": "~4.5.5",
    "vite-plugin-pwa": "^0.15.2"
  }
}
```

Figura 3.9. Codi de l'arxiu package.json.

De les dependències d'aquest projecte en vull destacar les següents:

Axios
És una biblioteca de client HTTP basada en promeses que és àmpliament utilitzada per comunicar-se amb servidors API i realitzar operacions CRUD (Crear, Llegir, Actualitzar, Eliminar) en aplicacions Vue.js.

En el context del meu projecte, Axios es fa servir per comunicar l'aplicació Vue.js amb l'API backend. Aquesta es pot utilitzar per fer sol·licituds HTTP a les rutes de l'API i obtenir les dades necessàries per mostrar a la interfície d'usuari, així com enviar dades actualitzades al backend per al seu processament.
A la secció de **Connexió amb l'API i altres fonts de persistència**, proporciono més informació detallada sobre com utilitzo la biblioteca Axios.

Firebase
Firebase és una plataforma de desenvolupament d'aplicacions al núvol que ofereix una àmplia gamma de serveis backend per facilitar el desenvolupament d'aplicacions web i mòbils.

En aquest projecte s'utilitza la biblioteca firebase per integrar alguns d'aquests serveis, com ara Desplegament del frontend i Login d'usuaris.

Desplegament del frontend: Firebase ofereix un servei d'allotjament web que permet desplegar aplicacions de manera senzilla. Es pot fer servir Firebase Hosting per allotjar llocs web estàtics, cosa que facilita la distribució i accés a l'aplicació a través d'Internet.

Login d'usuaris: Firebase proporciona un servei d'autenticació d'usuaris que permet gestionar l'autenticació dels usuaris de manera segura i senzilla. Es pot usar Firebase Authentication per permetre que els usuaris es registrin, iniciïn sessió i tanquin sessió a l'aplicació. Això estalvia haver d'implementar la lògica d'autenticacions des de zero i aporta característiques com l'inici de sessió amb proveïdors externs (per exemple, Google, Facebook), restabliment de contrasenyes, gestió de perfils d'usuari, entre altres.
A les seccions **Registre i login** indico com s'han utilitzat aquests serveis.

Pinia
Pinia és una biblioteca d'estat centralitzat per a Vue 3 que ofereix una alternativa al Vuex oficial. En lloc de dependre de Vuex, Pinia implementa el patró d'arquitectura

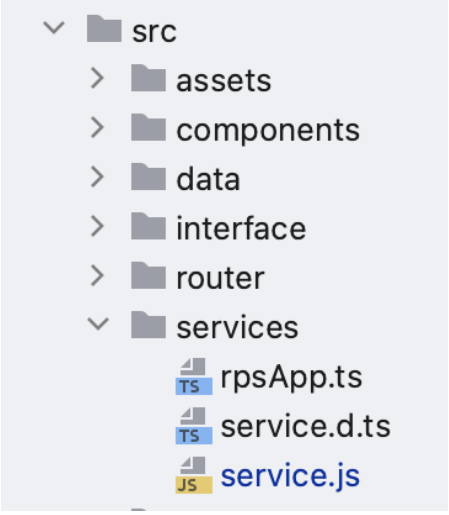
Flux, que es caracteritza per un flux de dades unidireccional i una gestió clara i predictible de l'estat de l'aplicació.
S'amplia aquest punt en **Gestió de l'estat de l'aplicació**

Fortawesome
Fa referència a la família de paquets "@fortawesome" que proporciona icones de font gratuïta i d'alta qualitat. Aquests paquets inclouen diferents conjunts d'icones, com ara "fontawesome-free", "fontawesome-svg-core", "free-regular-svg-icons" i "free-solid-svg-icons". Es poden fer servir aquestes icones a la vostra aplicació Vue.js per millorar la interfície d'usuari i proporcionar una experiència visual atractiva.

Jest
És un popular framework de proves unitàries per a JavaScript. S'utilitza normalment per escriure i executar proves unitàries en aplicacions Vue.js. Jest proporciona un conjunt complet d'eines per crear i executar proves, incloent-hi assercions, simulacions i cobertura de codi.
A l'apartat **Proves unaris** dono més detalls sobre la implantació de test unàries al projecte.

Connexió amb l'API i altres fonts de persistència

Utilitzo una estructura de fitxers composta per dos elements principals per gestionar la persistència de dades a l'aplicació, "service.js" i "rpsApp.ts".



El primer fitxer, "service.js", s'encarrega de centralitzar totes les funcions i fonts de dades. Aquest fitxer importa el fitxer "gameTypes.json" com a font d'informació estàtica i el fitxer "rpsApp.ts" per gestionar la comunicació amb l'API.

A través d'un objecte exportat, s'exposen diverses funcions que permeten interactuar amb l'API i fer operacions com inicialitzar moviments de joc, obtenir moviments, crear jocs, crear jugadors, entre d'altres.

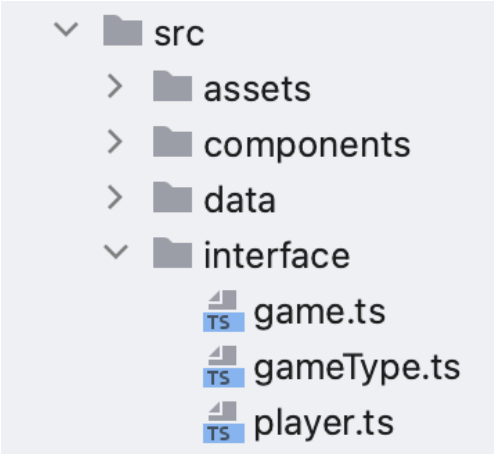
Figura 3.10. Vista de carpetes del projecte Vue. Amb el detall del contingut de "services".

El segon fitxer, rpsApp.ts, s'encarrega d'establir la comunicació amb l'API utilitzant la biblioteca Axios. Defineix una sèrie de funcions asincròniques que fan les trucades a l'API corresponents per realitzar les operacions. Aquestes funcions utilitzen l'URL de l'API i una instància personalitzada d'Axis per fer les peticions HTTP i gestionar les respostes.

```
...
async function getGamesByTitle() {
    try {
        const response = await axiosMy.get<Game>('/games');
        const game = response.data;
        return game;
    } catch (error) {
        console.log(error);
    }
}
...
```

Figura 3.11. Codi de la funció asíncrona getGamesByTitle.

Amb aquesta infraestructura s'aconsegueix una capa d'abstracció que centralitza la lògica de comunicació amb l'API i altres fonts de persistència en un sol lloc. Això es tradueix en una sèrie de beneficis que contribueixen a la gestió i el manteniment dels mecanismes d'emmagatzematge de dades.



En primer lloc, en utilitzar interfícies com Game, GameType i Player, es promou el compliment del principi SOLID de Responsabilitat Única. Aquestes interfícies defineixen les estructures d'informació utilitzades en les operacions, la qual cosa permet una separació clara de responsabilitats entre aquesta capa i les capes superiors de l'aplicació. Això facilita l'extensibilitat i la reutilització del codi, ja que les interfícies actuen com a contractes que garanteixen la consistència en l'intercanvi d'informació.

Figura 3.12. Vista de carpetes del projecte Vue. Amb el detall del contingut de "interfice".

A més, l'ús d'una capa d'abstracció centralitzada promou el principi SOLID d'Inversió de Dependències. En encapsular la lògica de comunicació amb l'API i altres fonts, evita que

les capes superiors depenguin directament dels detalls d'implementació d'aquestes fonts. En canvi, s'hi interactua mitjançant una interfície proporcionada per l'objecte exportat. Això permet modificar o canviar les fonts de persistència subjacents sense afectar el codi que utilitza aquesta capa d'abstracció, cosa que facilita la sostenibilitat i l'evolució del sistema en el temps.

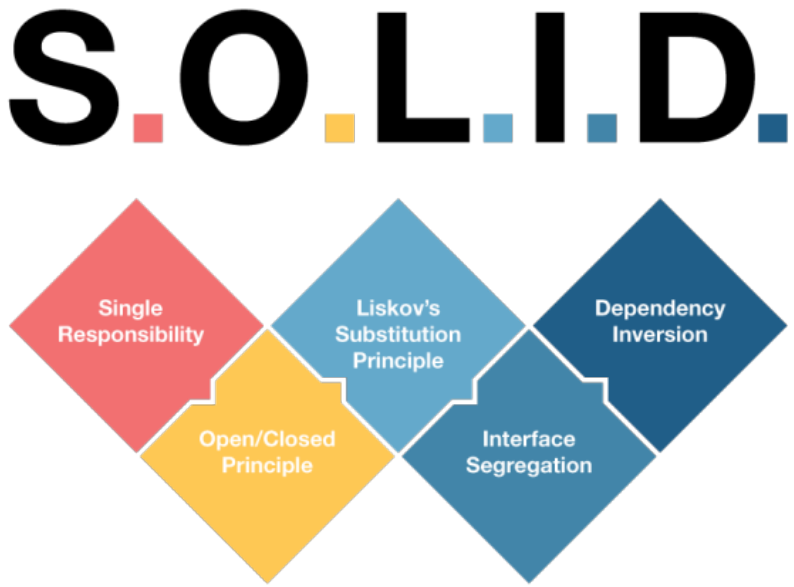


Figura 3.13. Representació gràfica dels principis SOLID.

Un altre principi SOLID que es beneficia amb aquesta capa d'abstracció és el Principi de Responsabilitat Única. En tenir un únic punt d'accés a les operacions, s'assegura que cada funció o mètode es dediqui a una única tasca específica. Per exemple, les funcions al fitxer "rpsApp.ts" s'encarreguen de fer trucades a l'API i manejar les respostes, mentre que el fitxer principal s'enfoca a l'exposició d'aquestes funcions. Aquesta separació de responsabilitats millora la cohesió i redueix la complexitat del codi.

Components

A l'arquitectura de components, l'únic punt destacable de l'aplicació és la pàgina inicial (home page). Aquesta pàgina serveix com a punt d'entrada per als usuaris i proporciona

informació general sobre l'aplicació, així com opcions per registrar-se i un llistat de jocs disponibles.

La vista Home (magenta) i els components GameList (blau) i GameType (verd) treballen en conjunt per mostrar una llista de jocs a la pàgina inicial. El component GameList s'encarrega d'organitzar i estructurar la llista, i el component gameType mostra la informació individual de cada tipus de joc.

Figura 3.14. Esquema de components de la home damunt el wireframe.

```
...
<div class="grid-item" role="game" v-for="item in gameTypeList"
:key="item.id">
 <gameType :gameType="item"></gameType>
</div>
...
```

Figura 3.15. Fracciones de codi de l'arxiu gameList.vue.

```
...
const unit = defineProps<{
 gameType: GameType,
}>()
...
<template>
 <div class="unit-game-type">
   <img :src="`${url}img/${unit.gameType.img}`" alt="gameType.name"
:class="['unit-game-type__img', unit.gameType.active ||
'unit-game-type__img--inactive']"/>
   <div class="unit-game-type__name">
     {{ unit.gameType.name }}
   </div>
   <router-link :to="`/new-game/${unit.gameType.id}`">
```

```
      <button class="button unit-game-type__button"
:disabled=!unit.gameType.active>Start new game</button>
    </router-link>
  </div>
</template>
…
```

Figura 3.16. Fracciones de codi de l'arxiu gameType.vue.

Des del component GameList, s'envia la informació de les característiques de cada joc utilitzant la interfície GameType. Mitjançant les props definides, es proporcionen els paràmetres necessaris al component "gameType", permetent així la visualització i manipulació de les dades específiques de cada joc en aquest component. D'aquesta manera, s'estableix una comunicació efectiva entre els components i s'aconsegueix una propagació adequada de la informació.

Gestió de l'estat de l'aplicació

El patró d'arquitectura Flux es basa en la premissa que l'estat de l'aplicació ha de ser gestionat de manera centralitzada i accessible des de qualsevol component. En aquest cas, Pinia s'utilitza per administrar el Global Store, que emmagatzema l'estat de la connexió a l'API de Slack a tota l'aplicació.

```
import { defineStore } from 'pinia';

export const useGlobalStore = defineStore('global', {
    state: () => ({
        slackToken: false
    }),
    // actions: {
    // },
});
```

Figura 3.17. Fracciones de codi de l'arxiu store.js.

En combinació amb el patró de disseny Observer, Pinia proporciona una forma intuïtiva i flexible de definir i utilitzar l'estat global, les accions i els getters als components de Vue.js. En fer servir Pinia, pots registrar explícitament els components com a observadors al Global Store. Això estableix una relació d'un a molts entre els components i el Global Store. Quan es produeixen canvis a l'estat, els components registrats com a observadors són notificats automàticament.

```
<script setup>
import {RouterLink} from 'vue-router';
import {onMounted, ref, watch} from "vue";
import {getAuth, onAuthStateChanged, signOut} from "firebase/auth";
import { useGlobalStore } from '@/store';
import { storeToRefs } from 'pinia'

const store = useGlobalStore()
const { slackToken } = storeToRefs(store);
…
```

Figura 3.18. Fracció de codi, com a exemple d'implantació de Pinia en un component de Vue.

Al meu codi, encara que no registre explícitament els components com a observador, utilitzo storeToRefs de Pinia per accedir i modificar l'estat del Global Store a través de la instància de store. En fer servir storeToRefs(store), obtinc referències reactives als valors de l'estat del Global Store, com ara slackToken en aquest cas. Això permet accedir a slackToken com una referència reactiva i usar-la als components. Encara que no estic registrant explícitament el component com a observador, gràcies a la naturalesa reactiva de les referències, qualsevol canvi a slackToken es reflectirà automàticament a la interfície d'usuari del component. Aquesta implementació segueix el concepte central del patró Observer, on els components poden observar i reaccionar als canvis a l'estat del Global Store, fins i tot sense un registre explícit com a observador.

Usabilitat i accessibilitat

Encara que estem en una etapa inicial del desenvolupament, he adoptat enfocaments i considerat pautes per assegurar una bona usabilitat i accessibilitat a la meva aplicació en el futur. A continuació, presento algunes de les bones pràctiques que he seguit fins ara.

Ús d'etiquetes semàntiques: S'han utilitzat etiquetes HTML semàntiques per a identificar l'estructura de la pàgina de manera clara i significativa. Per exemple, s'ha utilitzat l'etiqueta <main> per al contingut principal, <nav> per a la navegació i <section> per a agrupar contingut relacionat. Això facilita la comprensió i la navegació per als usuaris, inclosos aquells que utilitzen tecnologies d'assistència.

```
…
<nav class="navbar navbar-light">
 <div class="container">
```

```
    <div class="logo">
      <router-link to="/">
        <img src="@/assets/icon.svg" alt="Logo"/>
      </router-link>
    </div>
  </div>
...
```

Figura 3.19. Fracció de codi, amb exemple d'ús d'etiquetes semàntiques com <nav>.

Descripcions en imatges: S'ha afegit l'etiqueta alt a les imatges presents al grid. Aquesta etiqueta es fa servir per proporcionar una descripció concisa i rellevant de la imatge. En fer-ho, es garanteix que els usuaris que depenen de lectors de pantalla puguin comprendre'n el contingut visual.

```
...
<img src="@/assets/icon-slack.png" alt="Slack connected"/>
...
```

Figura 3.20. Fracció de codi, amb exemple d'ús de l'atribut "alt".

Etiquetesi aria-label a formularis: Per als formularis presents al projecte, s'han utilitzat les etiquetes per descriure cada camp d'entrada. Aquestes etiquetes estan associades amb els elements d'entrada corresponents mitjançant l'atribut for, o bé niant el camp dins de l'etiqueta . Això millora la usabilitat del formulari, ja que els usuaris poden identificar clarament quina informació s'espera a cada camp.

```
...
<form @submit.prevent="signIn">
  <div class="form-group">
    <label class="form-label" for="username">E-mail</label>
    <input
        placeholder="E-mail"
        type="email"
        id="email"
        class="form-control"
        v-model="email"
    />
...
```

Figura 3.21. Fracció de codi, amb exemple d'ús de etiquetes aria-label.

Addicionalment, l'atribut aria-label s'ha utilitzat en alguns elements per proporcionar descripcions addicionals per als usuaris que utilitzen lectors de pantalla. Aquest atribut es fa servir per comunicar informació important que no està visible al contingut, assegurant que tots els usuaris puguin accedir a aquesta informació.

```
...
<div class="grid-container" role="list" aria-label="Game list">
 <div class="grid-item" role="game" v-for="item in gameTypeList"
:key="item.id">
   <gameType :gameType="item"></gameType>
 </div>
</div>
...
```

Figura 3.22. Fracció de codi, amb exemple d'ús de etiquetes aria-label.

Registre i login

Durant el desenvolupament del sistema OAuth de l'aplicació de Slack, va sorgir la necessitat d'incloure el registre a la meva aplicació, tot i que inicialment no estava previst. Això va ser perquè per dotar l'aplicació de la capacitat de ser instal·lada en qualsevol espai de treball de Slack, era imprescindible registrar un token individual per a cada espai de treball, la qual cosa requeria emmagatzemar-lo en persistència juntament amb la identificació de l'usuari.

Pel que fa a l'elecció del sistema d'inici de sessió amb correu electrònic i contrasenya de Firebase, hi ha diversos motius que donen suport a aquesta decisió. En primer lloc, Firebase ofereix una integració senzilla amb altres serveis de la plataforma, cosa que simplifica el desenvolupament i la gestió de l'aplicació en un únic entorn. A més, aporta seguretat amb tècniques de xifratge i emmagatzematge segur de contrasenyes, així com opcions de verificació en dos passos.

Tot i ser una integració relativament senzilla, va implicar cert desenvolupament.
En primer lloc, cal configurar el Firebase. Això implica crear un projecte i habilitar el proveïdor d'inici de sessió amb correu electrònic a la consola. Aquesta configuració permet utilitzar els serveis d'autenticació.

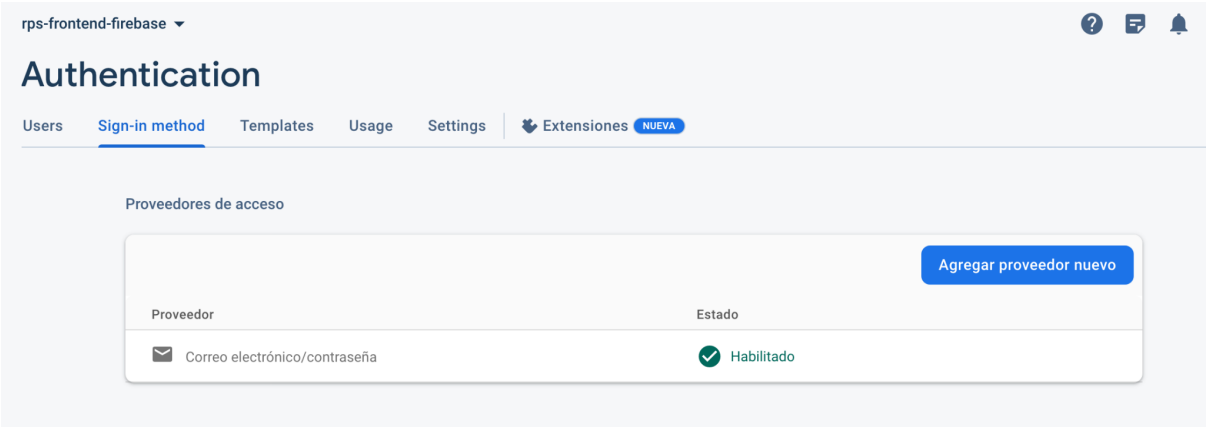


Figura 3.23. Vista de l'eina de gestió de Firebase Authentication.

Després, cal fer la integració del SDK de Firebase. Això implica afegir les dependències necessàries i configurar la connexió amb el nostre projecte. L'SDK proporciona les eines i les funcionalitats necessàries per gestionar l'autenticació de l'usuari.
Un cop configurat l'entorn, és important crear una interfície d'usuari que permeti als usuaris ingressar el vostre correu electrònic i contrasenya de manera segura.

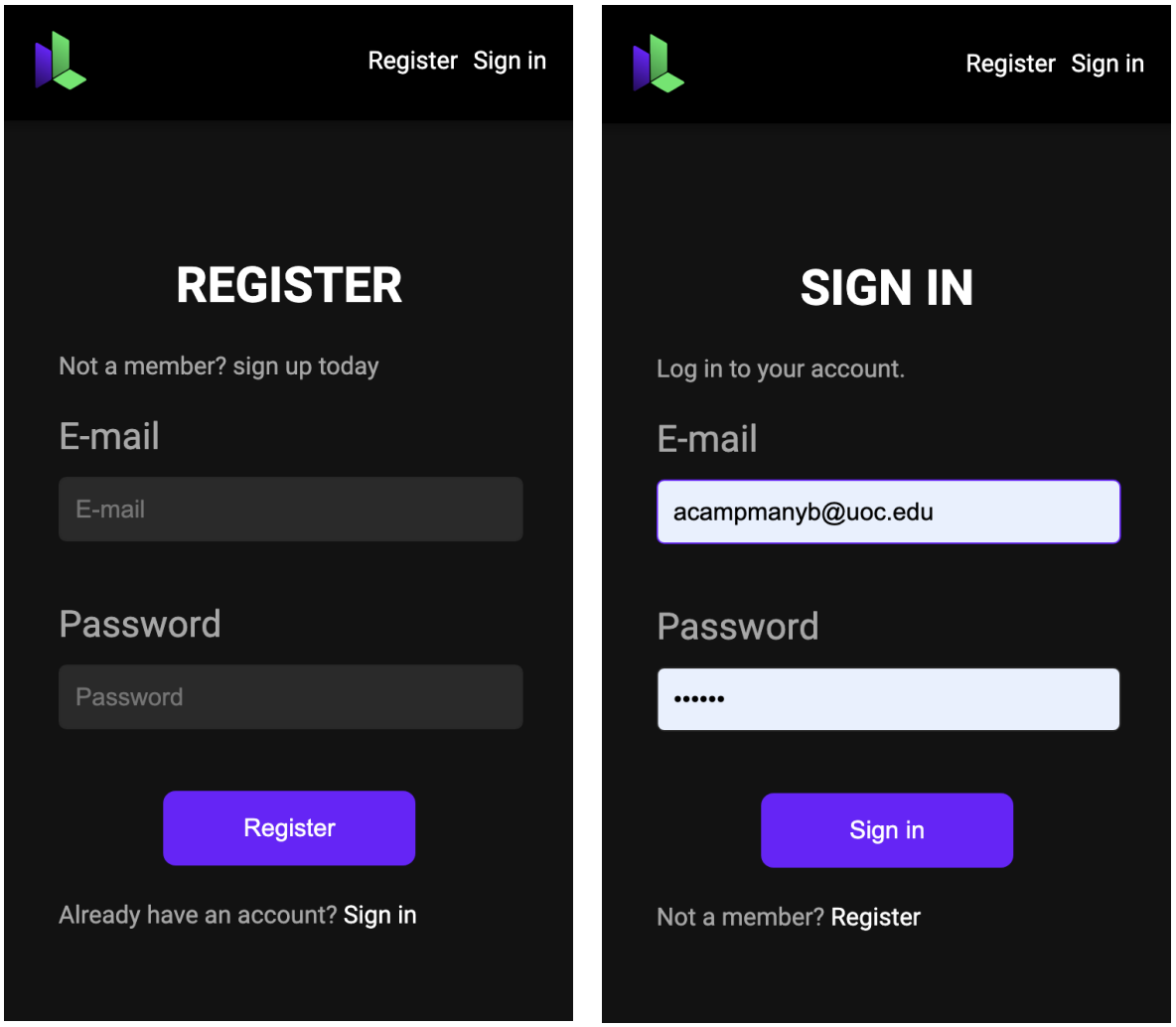


Figura 3.24. Vistes de login i el registre de l'aplicació Ludoscrum, en format Mobile.

La lògica d'inici de sessió és el pas següent. Aquí és on utilitzem les funcions proporcionades per l'SDK del Firebase per gestionar el procés d'inici de sessió.

main.ts

```
import { createApp } from 'vue'
import App from './App.vue'
import './registerServiceWorker'
import router from './router'
import { createPinia } from 'pinia';

// import './firebase'
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
    apiKey: [redacted],
    authDomain: "rps-frontend-firebase-48465.firebaseapp.com",
    projectId: "rps-frontend-firebase",
    storageBucket: "rps-frontend-firebase.appspot.com",
    messagingSenderId: [redacted],
    appId: [redacted]
};

initializeApp(firebaseConfig);
```

Figura 3.25. Vista d'una fracció de codi, de la configuració de Firebase.

Finalment, Firebase Authentication té mètodes per verificar l'estat d'inici de sessió i obtenir informació de l'usuari autenticat. Això ens permet personalitzar l'experiència de l'usuari i mostrar contingut rellevant segons la vostra autenticació.

```
...
const handleSignOut = () => {
 signOut(auth).then(() => {
   console.log('User signed out.')
   localStorage.removeItem('salckAccessToken');
   slackConnected = false;
   slackToken.value = false;
 })
} ...
```

```
...
onMounted(() => {
 auth = getAuth();
 console.log(auth)

 onAuthStateChanged(auth, (user) => {
   isLoggedIn.value = !!user;
...
```

Figura 3.26. Dues fraccions de codi, de l'arxiu d'AppHeader.vue, amb exemples dels mètodes de Firebase: signOut i onAuthStateChanged.

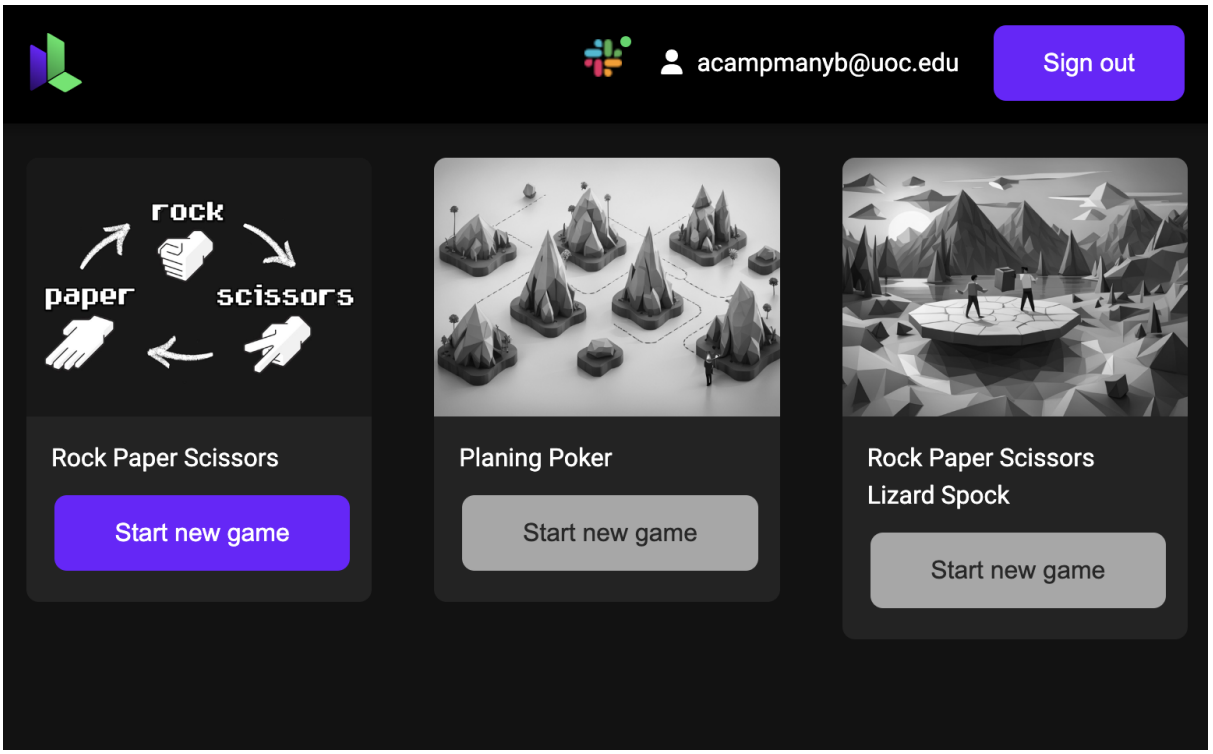


Figura 3.27. Vista de la home de Ludescrume, com a exemple de usuari amb estat de login actiu.

Desplegament

Al principi, vaig considerar utilitzar Github Pages com a opció per al desplegament del frontend de l'aplicació. No obstant això, en intentar fer el desplegament, em vaig trobar amb limitacions significatives en comparació del sistema de desplegament del Firebase. Aquesta experiència em va portar a prendre la decisió d'optar per Firebase Hosting, que m'oferia més avantatges i funcionalitats.

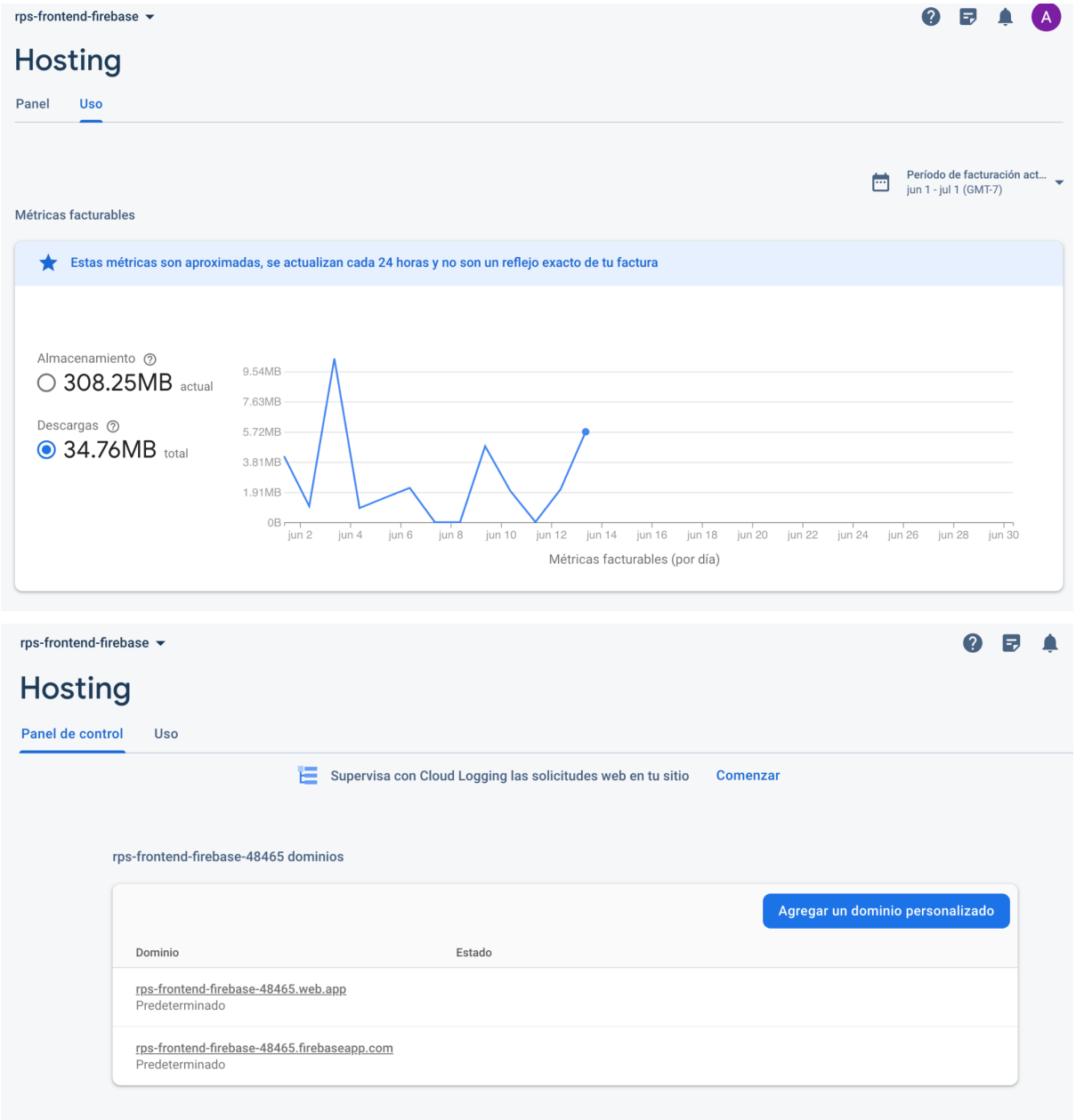


Figura 3.28. Vista de l'eina de gestió de Firebase Hosting.

Un cop configurat Firebase Hosting, es pot desplegar el frontend de l'aplicació amb tan sols unes quantes ordres.

```
→  rps-frontend git:(master) ✗ firebase deploy

=== Deploying to 'rps-frontend-firebase'...

i  deploying hosting
i  hosting[rps-frontend-firebase-48465]: beginning deploy...
i  hosting[rps-frontend-firebase-48465]: found 64 files in dist
✔  hosting[rps-frontend-firebase-48465]: file upload complete
i  hosting[rps-frontend-firebase-48465]: finalizing version...
✔  hosting[rps-frontend-firebase-48465]: version finalized
i  hosting[rps-frontend-firebase-48465]: releasing new version...
✔  hosting[rps-frontend-firebase-48465]: release complete

✔  Deploy complete!
```

Figura 3.29. Vista de la consola amb el procés de desplegament de Firebase.

Un altre aspecte interessant de Firebase Hosting és la seva capacitat per garantir un lliurament ràpid de contingut a través de la xarxa de distribució de contingut global (CDN). Aquest benefici és especialment rellevant per al meu projecte, dirigit a un sector específic però amb un abast global. La possibilitat de proporcionar una experiència de càrrega ràpida als usuaris a diferents ubicacions geogràfiques és un punt important a considerar, ja que contribueix a millorar la satisfacció de l'usuari i l'eficiència de l'aplicació en conjunt.

Firebase Hosting també ofereix xifratge SSL i la gestió de certificats. Aquestes mesures garanteixen una comunicació segura entre el navegador de l'usuari i el servidor d'allotjament. Aquest avantatge és especialment beneficiosa per a aquest projecte, pel fet que elimina la necessitat de dedicar recursos i esforços addicionals a gestionar i mantenir aquests aspectes. En lloc d'invertir-hi temps i recursos, puc enfocar-me en el desenvolupament de funcionalitats que afegeix valor. Això permet maximitzar l'eficiència i l'enfocament en el creixement i l'evolució del projecte.

3.3. Proves unaris

En aquesta etapa inicial de la PoC, com que és una versió molt bàsica de l'aplicació, no vaig considerar necessari dedicar gaire temps al desenvolupament de proves. Tot i això, per a l'element principal de l'aplicació, la sala de jocs del joc RPS, vaig decidir generar un test.

Les proves unitàries són fonamentals en el desenvolupament de programari perquè ens permeten verificar el comportament individual de cada component o funció del nostre codi. Utilitzant eines com Jest i el complement @vue/cli-plugin-unit-jest, vaig poder escriure i executar el test de manera fàcil.

A més, mitjançant el mesurament de la cobertura de codi, podem avaluar quin percentatge del nostre codi està sent provat. Això ens ajuda a identificar àrees que requereixen més atenció i ens permet assegurar-nos que estem provant adequadament totes les funcionalitats.

Encara que en aquesta etapa no s'hagi enfocat gaire a les proves, reconec que en etapes posteriors caldrà dedicar més temps i esforç a aquesta pràctica. Les proves unitàries ajuden a construir un programari més sòlid i reduir errors.

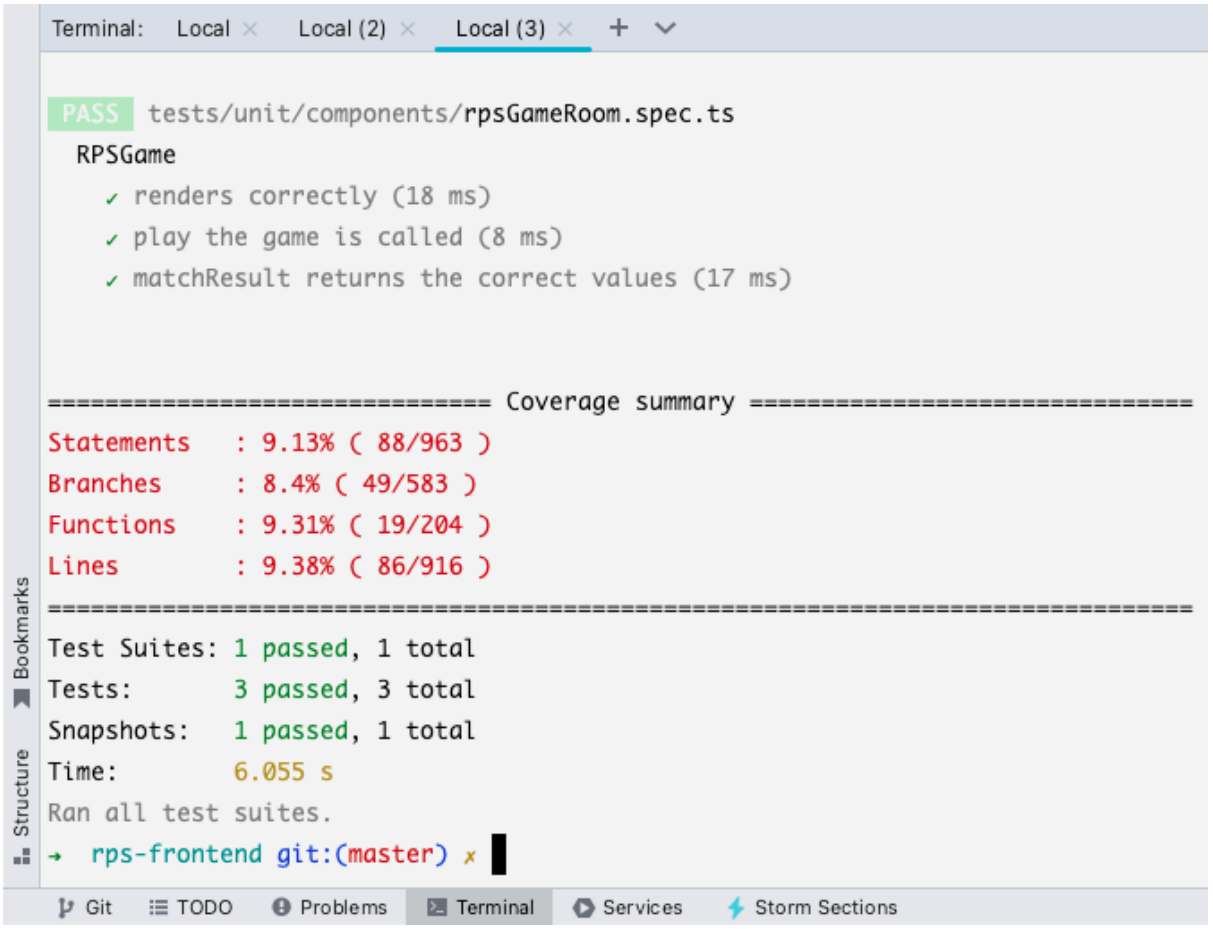


Figura 3.30. Vista de la consola amb els resultats de l'informe de Jest

Aquest informe indica que les proves unitàries han estat superades.
El component s'ha renderitzat correctament, i s'ha verificat que la funció "play" és crida correctament i la funció "matchResult" torna els valors esperats.
En termes de cobertura de codi, s'ha avaluat que es testegen el 9,13% de les declaracions (88 de 963), el 8,4% de les branques (49 de 583), el 9,31% de les funcions (19 de 204) i el 9,38% de les línies (86 de 916).

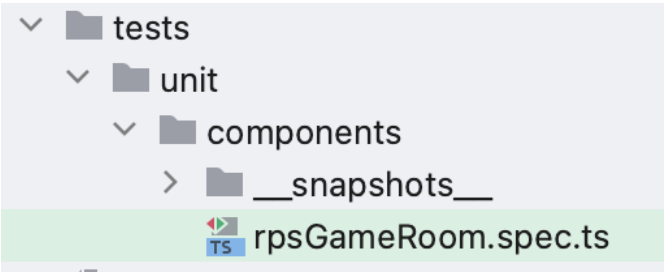


Figura 3.31. Estructura de carpetes de l'apartat tests.

```
import { shallowMount } from '@vue/test-utils';
import RPSGame from '@/components/rpsGameRoom.vue';

describe('RPSGame', () => {
   let wrapper: any;
   let playGameSpy: any;
   let getPlaysSpy: any;

   beforeEach(() => {
       const service = {
           getPlays: jest.fn().mockResolvedValue([
               { player: 'userId1', game: 'gameId1', move: 'rock' },
               { player: 'userId2', game: 'gameId1', move: 'paper' },
           ]),
       };
       wrapper = shallowMount(RPSGame, {
           mocks: {
               service,
           },
       });
       playGameSpy = jest.spyOn(wrapper.vm, 'play').mockImplementation();
       getPlaysSpy = jest.spyOn(service, 'getPlays');

       jest.clearAllMocks();
   });

   test('renders correctly', () => {
       expect(wrapper.element).toMatchSnapshot();
   });

   test('play the game is called', async () => {
       await wrapper.find('[data-testid="rock"]').trigger('click');
       expect(playGameSpy).toHaveBeenCalled();
   });

   test('matchResult returns the correct values', async () => {
       const userChoice = 'rock';
       const opponentChoice = 'paper';
       const matchResultSpy = jest.spyOn(wrapper.vm, 'matchResult');

       await wrapper.vm.matchResult(userChoice, opponentChoice);

       expect(matchResultSpy).toHaveBeenCalledWith(userChoice,
opponentChoice);
       expect(matchResultSpy.mock.calls[0][0]).toBe(userChoice);
       expect(matchResultSpy.mock.calls[0][1]).toBe(opponentChoice);
       expect(wrapper.find('[data-testid="result"]').text()).toBe('You
lose!');

       matchResultSpy.mockRestore();
```

```
    });
});
```

Figura 3.32. Codi de l'arxiu rpsGameRoom.spec.ts.

Aquests tests cobreixen diferents aspectes del component RPSGame, com ara la renderització correcta, la interacció de l'usuari en fer clic en una opció i el resultat esperat del joc. Amb aquestes proves es garanteix un nivell bàsic de cobertura i es verifica el comportament de les funcions principals del component.

En detall els tests verifiquen el següent:

- **renders correctly**: Aquest test verifica que el component RPSGame es renderitzi correctament. Es compara l'element renderitzat amb un snapshot prèviament desat, assegurant que no hi hagi canvis inesperats a l'estructura del component.

- **play the game is called**: Aquest test simula un clic a un element amb l'atribut data-testit igual a "rock". Després, es verifica que es crida a la funció. Això assegura que en fer clic a l'element "rock", s'activi correctament la lògica del joc.

- **matchResult returns the correct values**: En aquest test, es defineix l'elecció de l'usuari com a "rock" i l'elecció de l'oponent com a "paper". Es falsifica la funció service.getPlays() perquè torni un valor específic. Després, s'anomena la funció matchResult amb les eleccions definides. Es verifica que la funció hagi estat cridada amb els paràmetres correctes i es compara el resultat obtingut amb el text esperat. En aquest cas, s'espera que el resultat sigui “You lose!”.

3.4. OAuth 2.0 per a API Slack

L'app de Slack que he generat és molt senzilla, la seva funció principal és enviar missatges a un canal específic. Tot i això, perquè aquesta app pugui ser instal·lada per altres usuaris en diferents espais de treball de Slack, és necessari implementar el flux d'OAuth 2.0 [17].
Aquest protocol permet que un usuari pugui instal·lar l'aplicació en qualsevol espai de treball de Slack. En finalitzar el procés OAuth, l'aplicació obté un token d'accés.

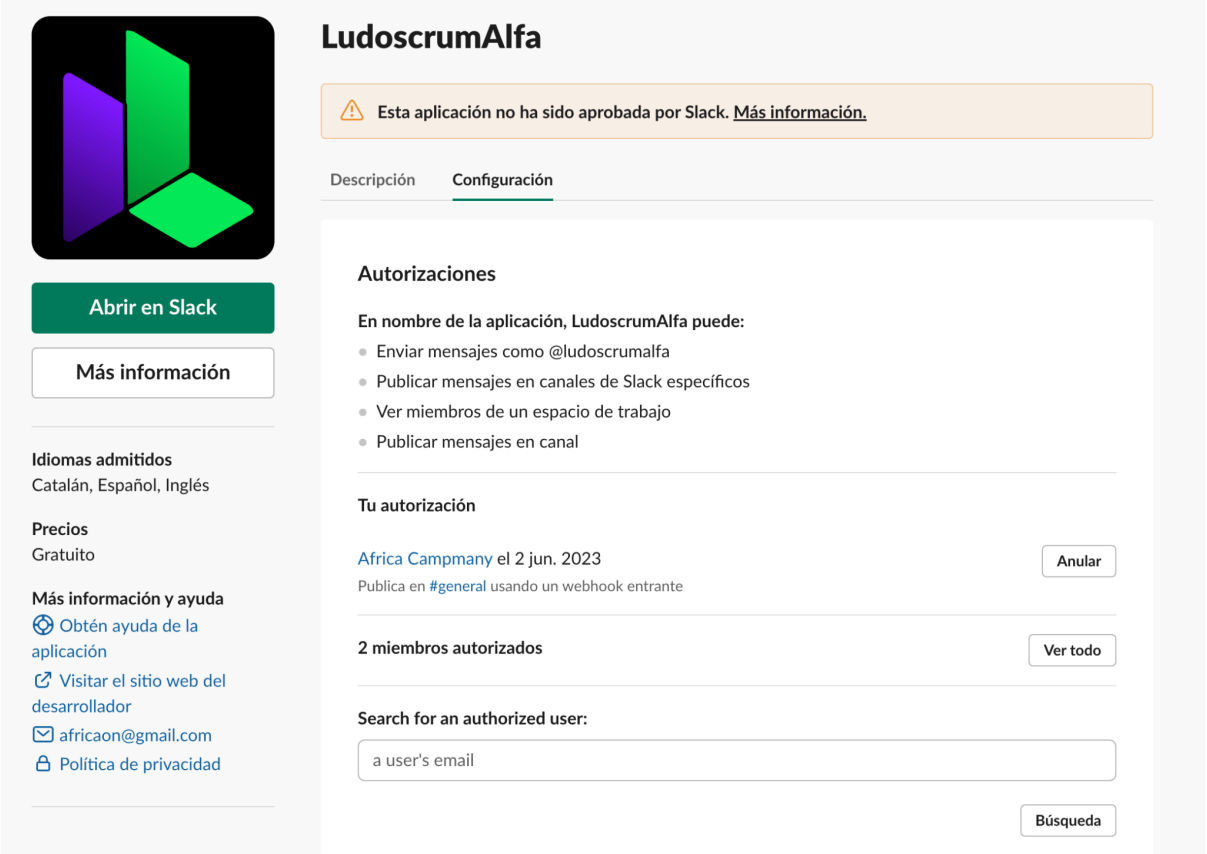


Figura 3.33. Vista de la fitxa de l'aplicació de Slack "LudoscrumAlfa", a l'apartat Manage Apps, de Slack app directory

Per obtenir els tokens d'accés amb OAuth a Slack, vaig haver de seguir tres passos.

Primer, vaig sol·licitar els permisos necessaris per a la meva aplicació. Durant el desenvolupament, vaig determinar els permisos mínims que la meva aplicació requeria.

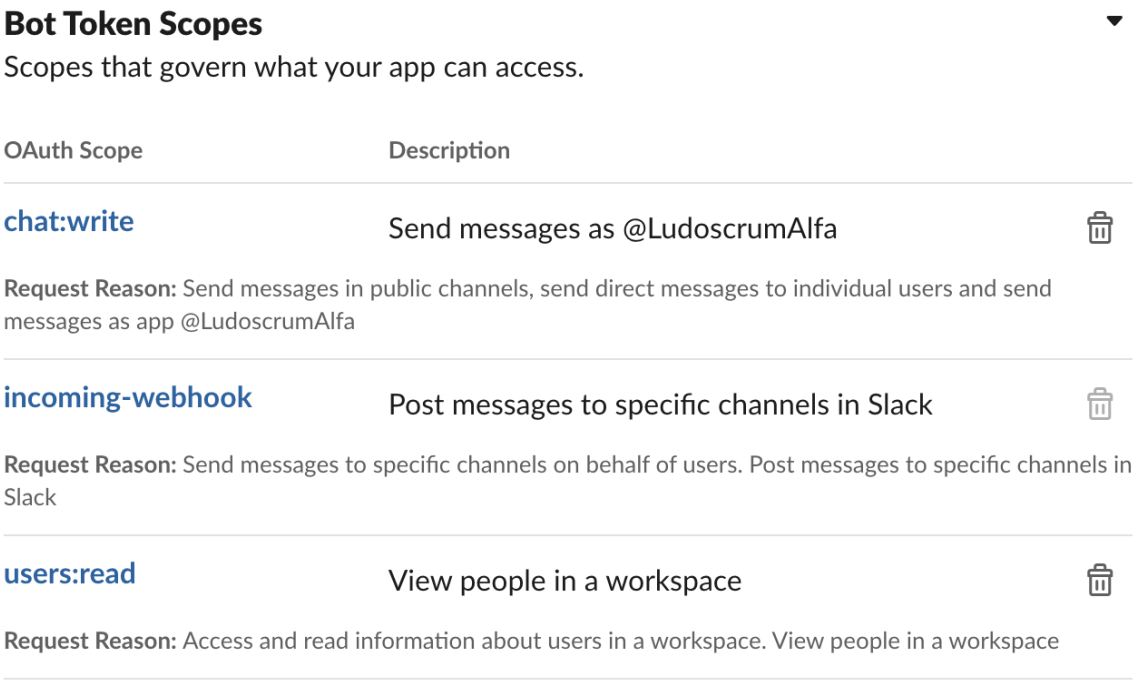


Figura 3.34. Vista l'apartat Scopes d'OAuth & Permissions, a la fitxa de l'aplicació de Slack "LudoscrumAlfa", de Slack api.

Després, quan un usuari instal·la l'aplicació al seu espai de treball de Slack, sol·licito aquests mateixos permisos. Per això s'afegeixen paràmetres a l'URL del botó de Slack a la meva app. Aquest URL inclou el meu ID de client de l'aplicació i una llista de permisos separats per comes.

```
<a
href="https://nailsoup.slack.com/oauth?client_id=532998036867.5247472208531&abast=chat%3Awrite%2Cincoming-webhook%2Cusers%3Aread&user_scope=&redirect_uri=&state=&granular_bot_scope=1&single_channel=0&install_redirect=&tracked=1&team="><img
alt="Afegeix a Slack" height="40" width="139"
src="https://platform.slack-edge.com/img/add_to_slack.png" srcSet="https://platform
.slack-edge.com/img/add_to_slack.png 1x,
https://platform.slack-edge.com/img/add_to_slack@2x.png 2x" /></a>
```

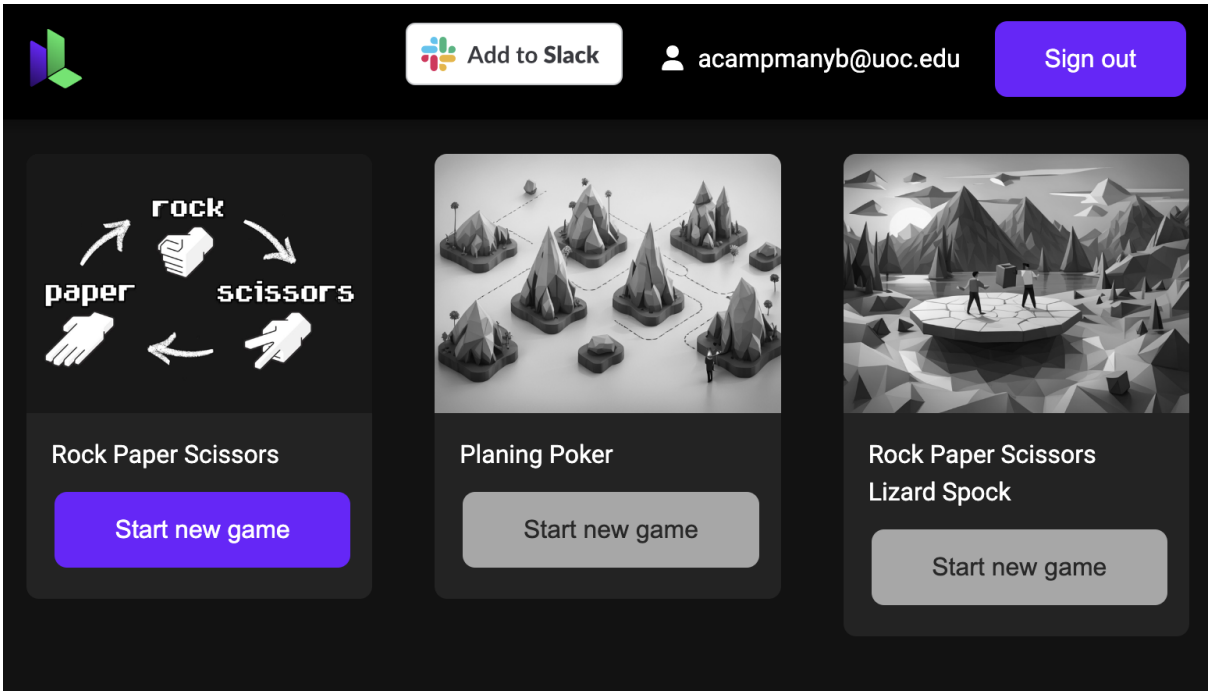


Figura 3.35. Vista de la home de Ludoscrum, on es veu el botó de "Add to Slack".

L'usuari ha d'aprovar els permisos:

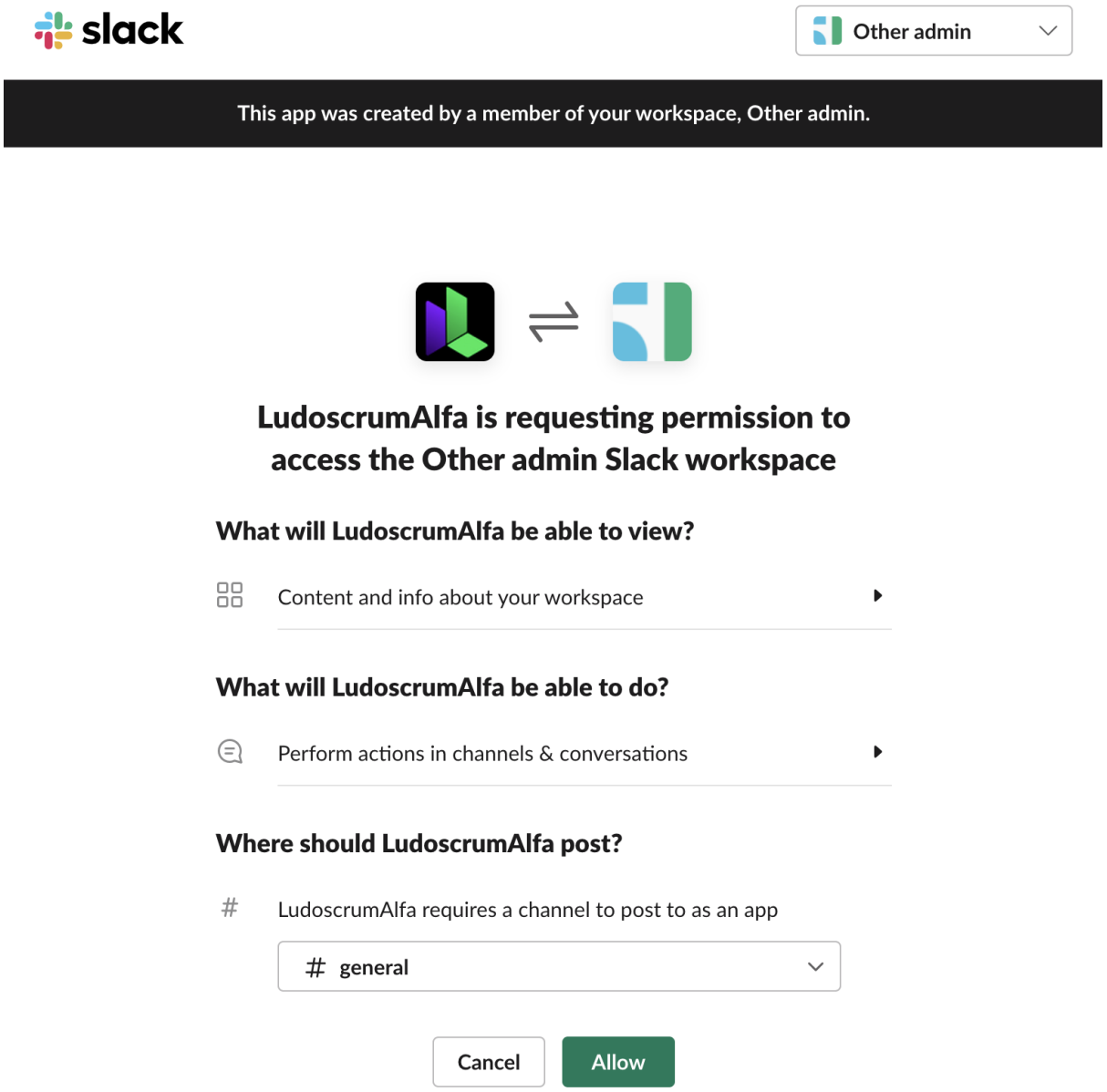


Figura 3.36. Vista per l'autorització de permisos per part de l'usuari de Slack a l'aplicació de Slack "LudoscrumAlfa"

Després Slack redirigeix l'usuari de tornada a la meva aplicació a través de l'URL de redireccionament que vaig configurar prèviament. Aquí és on va entrar en joc el segon pas.

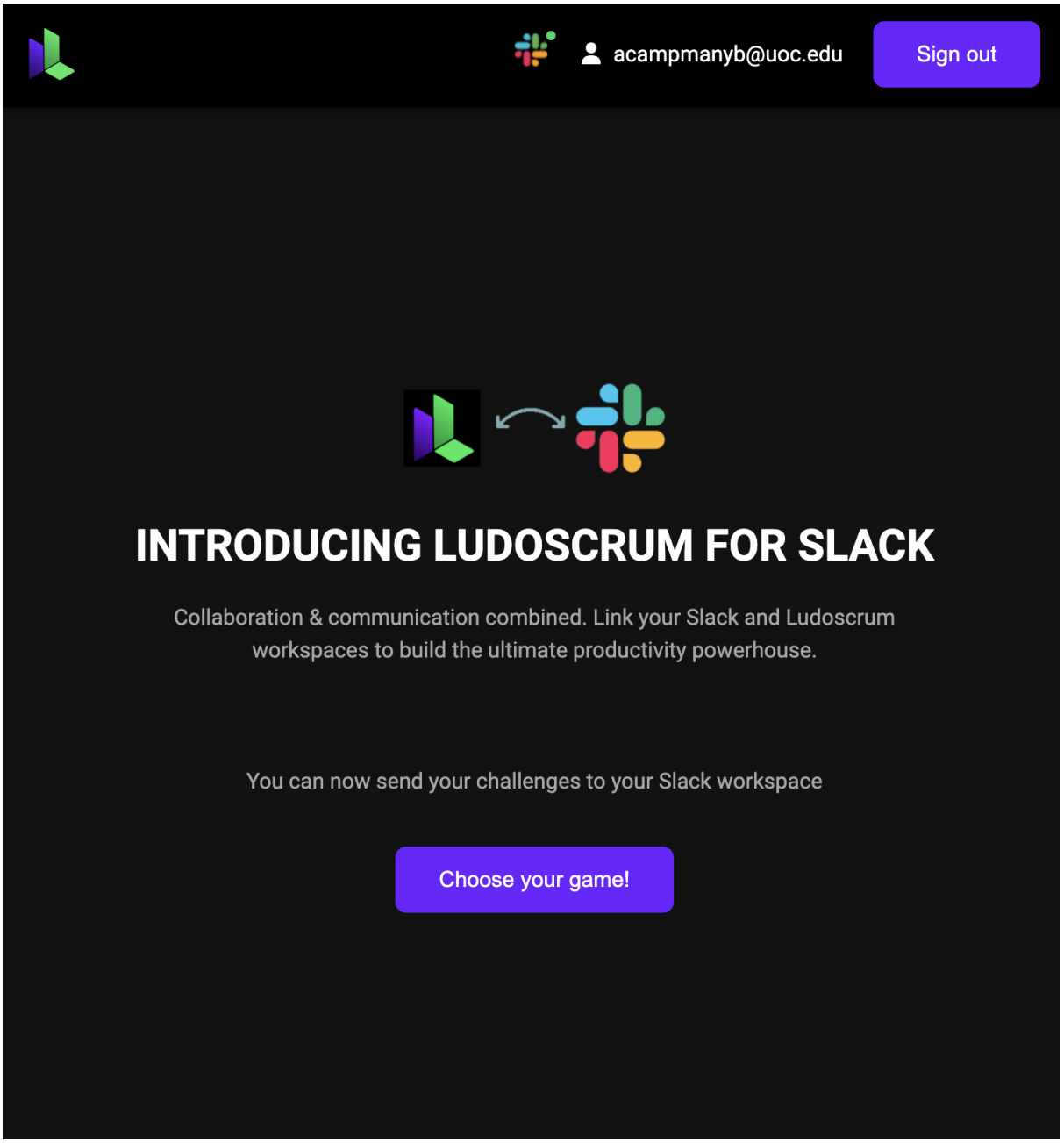


Figura 3.37. Vista de redireccionament cap a l'aplicació Ludoscrum.

Finalment, al tercer pas, s'intercanvia el codi d'autorització temporal per un token d'accés.
S'analitza la sol·licitud GET rebuda del redireccionament per obtenir el codi d'autorització temporal. Després, es realitza una trucada POST a l'API de Slack utilitzant el mètode 'oauth.v2.access', en què s'inclou el codi d'autorització temporal, l'ID de client de la meva aplicació i el "client-secret".

Aleshores, en el cas específic de l'aplicació frond de Ludoscrum, el procés implica enviar una sol·licitud GET des de l'aplicació frond cap al backend. Des del backend, es fa una sol·licitud POST per obtenir el token d'accés definitiu.

Un cop obtingut el token d'accés, també s'obté el webhookUrl entre altres paràmetres. El webhook URL de Slack és un URL única proporcionada per Slack que permet a les aplicacions enviar missatges i notificacions a canals específics a Slack. És una forma de comunicació unidireccional en què l'aplicació envia informació a Slack sense rebre respostes o interaccions directes. Aquest s'envia de tornada al frond.

```
…
onMounted(async () => {
 authorizationQuery.value = router.currentRoute.value.query;

 const accessToken = await
service.getCallback(authorizationQuery.value);

 const accessTokenString = JSON.stringify(accessToken);
 localStorage.setItem("salckAccessToken", accessTokenString);

 const storedAccessTokenString =
localStorage.getItem('salckAccessToken');
 const storedAccessToken = JSON.parse(storedAccessTokenString);
…
```

Figura 3.38. Fragment de codi, del frontend on es recull l'autorització temporal i s'envia al backend.

```
…
  const authorizationCode = req.query.code;
   // Hacer la solicitud POST a la API de Slack para intercambiar
el código de autorización por un token de acceso
   const {
       TU_CLIENT_ID,
       TU_CLIENT_SECRET
   } = process.env;

   const clientId = `${TU_CLIENT_ID}`;
   const clientSecret = `${TU_CLIENT_SECRET}`;
   const redirectUri =
'https://rps-frontend-firebase-48465.web.app/callback';
   const url = 'https://slack.com/api/oauth.v2.access';

   // Crear el objeto FormData y agregar los campos necesarios
```

```
    const formData = new FormData();
    formData.append('code', authorizationCode);
    formData.append('client_id', clientId);
    formData.append('client_secret', clientSecret);
    formData.append('redirect_uri', redirectUri);

    axios.post(url, formData, {
        headers: formData.getHeaders()
    })
        .then((response) => {
            const webhook = response.data.incoming_webhook;
            res.json(webhook);
        })
        .catch(error => {
            console.log("error", error);
        });
});
...
```

Figura 3.39. Fragment de codi, del backend s'obté finalment el token.

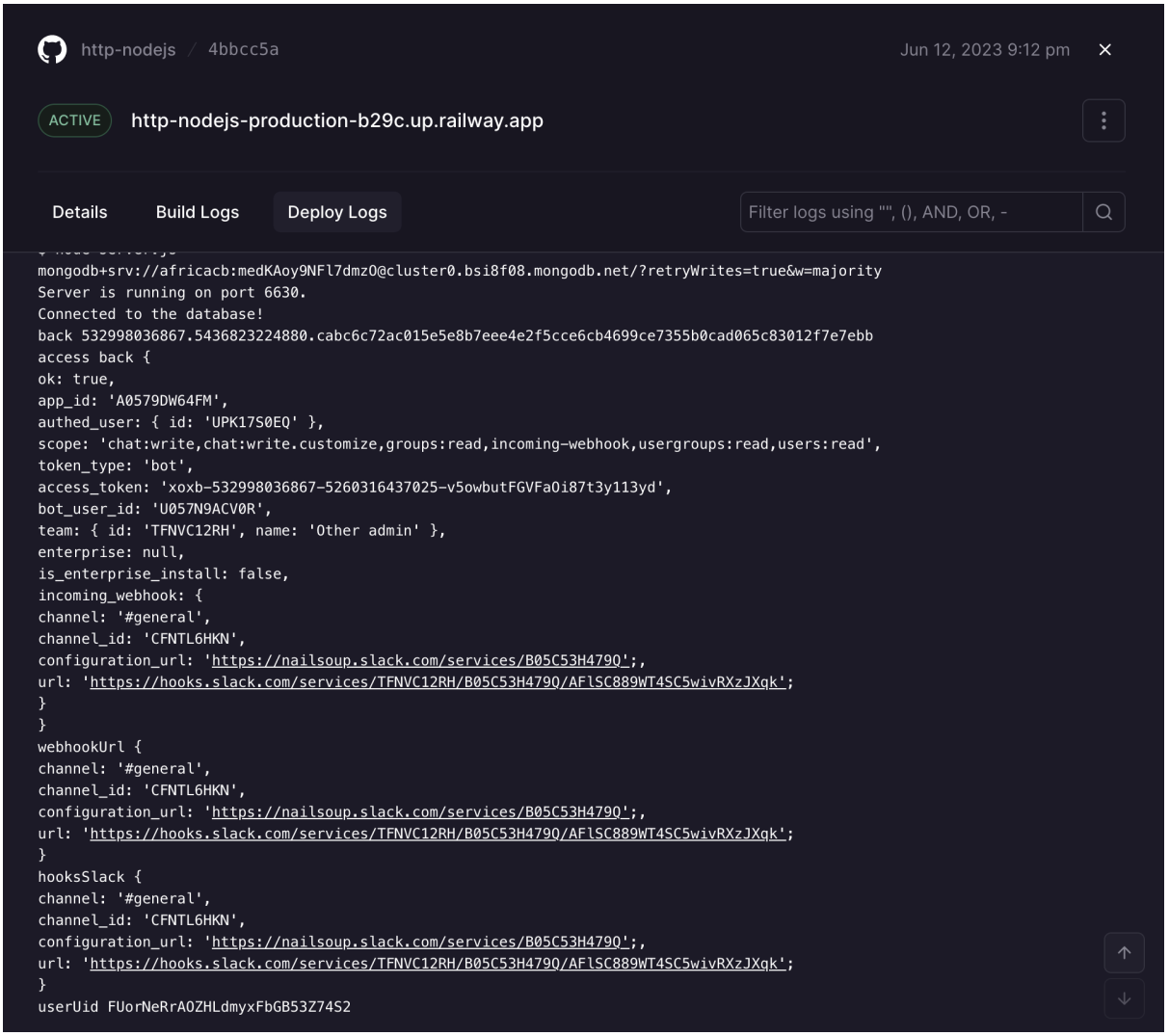


Figura 3.40. Vista de la traça obtinguda del servidor de backend.

Atès que el token d'accés rebut és únic per usuari, cal tenir un registre a l'aplicació per poder enviar els desafiaments a través d'aquest canal. Aquest registre permet associar el token d'accés amb l'usuari corresponent, cosa que garanteix que els desafiaments siguin enviats correctament i de manera segura.

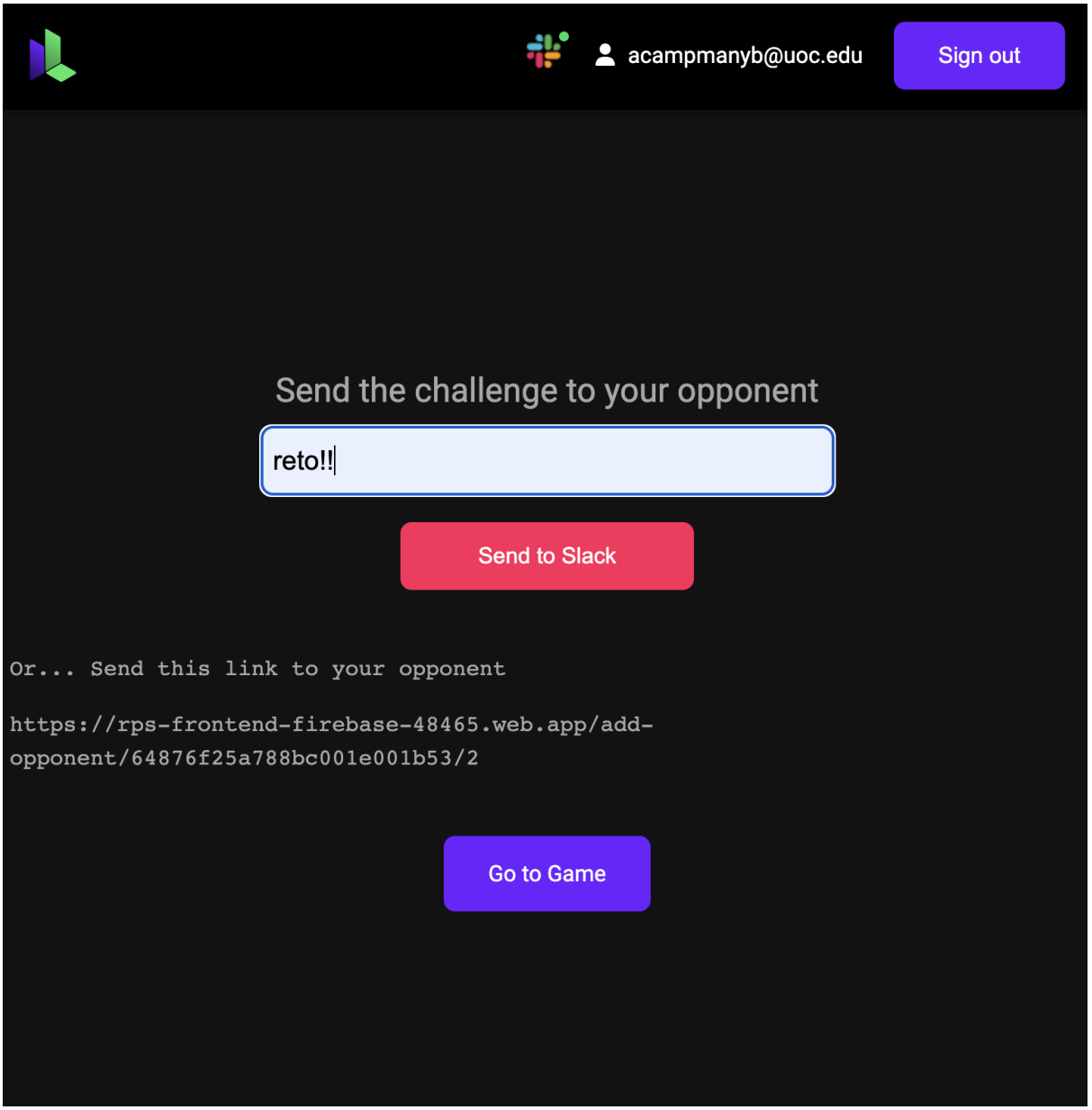


Figura 3.41. Vista de l'aplicació de Ludoscrum, on gràcies a estar logat amb un usuari amb token de Slack, s'habilita l'opció d'enviar el repte a través d'aquest canal.

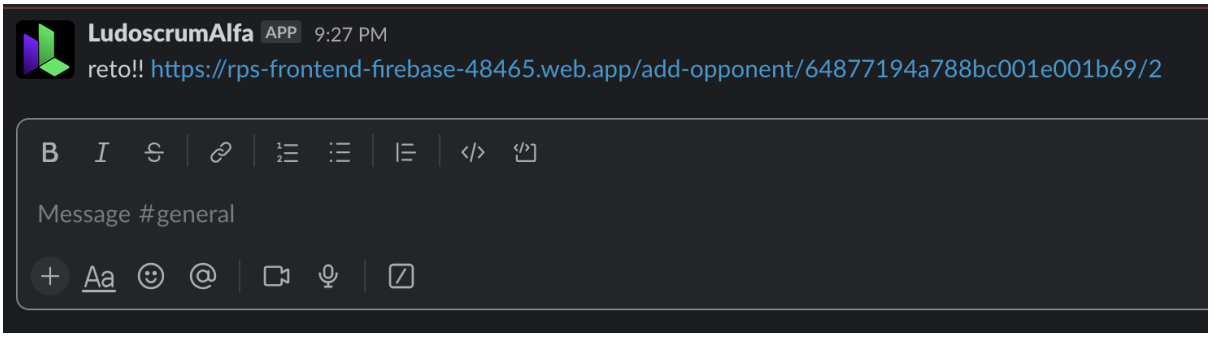


Figura 3.42. Vista de missatge del repte a l'espai de treball de Slack.

3.5.Presentació Online

Estic emocionada de compartir l'aplicació que he creat amb tant d'entusiasme i dedicació!

Es pot accedir-hi a través de la següent URL:
https://rps-frontend-firebase-48465.web.app/

A la pàgina d'inici, es troba l'opció de registrar-se, i accés al joc generat específicament per a aquesta prova de concepte (PoC).

Un cop fet el registre, es desbloquejarà la possibilitat de connectar l'aplicació a Slack. Això permetrà gaudir d'una major integració amb la plataforma de missatgeria i aprofitar al màxim l'experiència de l'aplicació.

4. Conclusions i treballs futurs

4.1. Conclusions

Les conclusions extretes del treball revelen diversos punts clau. En primer lloc, s'ha observat que és relativament senzill crear una plataforma de gamificació que s'integri amb les eines existents i utilitzades als equips de desenvolupament. Això és degut sobre manera al fet que la majoria d'aquestes plataformes compten amb API obertes per als desenvolupadors. Aquesta circumstància permet una àmplia gamma d'opcions per combinar les funcionalitats d'aquestes eines amb petites aplicacions de gamificació que s'integrin en els processos de treball.

Tanmateix, la conclusió més significativa, al meu entendre, és la importància que els elements de gamificació siguin petits, simples i perfectament integrats. És en aquest aspecte de la integració on cal concentrar tots els esforços. La qualitat de la integració determinarà molt l'èxit i l'acceptació de les estratègies de gamificació per part dels usuaris.

Conclusions sobre els objectius de la PoC

Si es re-avalua els objectius plantejats inicialment, es pot afirmar que se n'ha aconseguit complir satisfactòriament molts. A continuació, es fa una avaluació crítica dels objectius assolits i s'exploren les raons per les quals es consideren finalitzats amb èxit:

- S'ha buscat i trobat un element de ludificació que és natural i no artificiós ni complex. S'espera que això afavoreixi l'acceptació i la participació dels usuaris.
- S'ha dissenyat un joc que involucra activament els companys de feina, fomentant interaccions positives. Amb això es pretén promoure un ambient de treball dinàmic, impulsant la motivació i la col·laboració en l'equip.
- Els elements ludificats han estat dissenyats de manera inclusiva, evitant estereotips de gènere i promovent la igualtat d'oportunitats. S'ha procurat que totes les persones, sense importar-ne el gènere, puguin participar i gaudir de l'experiència de gamificació sense discriminació.
- S'ha aconseguit que l'aplicació sigui adaptable a diferents dispositius i resolucions de pantalla, cosa que garanteix que els usuaris hi puguin accedir de manera còmoda des de diferents equips i dispositius mòbils.

- S'ha avaluat la usabilitat de l'aplicació, assegurant que sigui fàcil fer servir i intuïtiva per als usuaris. Tot i que es reconeix que encara hi ha marge de millora en aquest aspecte, es considera que s'ha superat satisfactòriament a la Prova de Concepte (PoC) del projecte.
- L'aplicació és escalable i capaç de manejar un nombre creixent d'usuaris i dades sense comprometre el seu rendiment. Gràcies a l'elecció de tecnologies adequades, s'ha garantit que l'aplicació pugui créixer i adaptar-se a necessitats futures.

Tot i això, també és important assenyalar els requisits que no s'han pogut complir plenament:

- Identificar els processos dins l'entorn àgil que es poden ludificar i que fomenten la cooperació i la comunicació: Tot i que s'ha buscat la simplicitat i la senzillesa en el joc, es reconeix que la solució implementada és força genèrica i no s'ajusta específicament als processos àgils.
- El disseny ha de ser accessible per a totes les persones, independentment de les seves habilitats o discapacitats: Tot i que s'ha considerat l'accessibilitat en el disseny de l'aplicació, es reconeix que no s'han abordat totes les necessitats d'accessibilitat de manera completa.
- El 50% dels jocs estaran dissenyats per privilegiar alguna limitació concreta: Malauradament, aquest objectiu no s'ha complert, ja que només s'ha desenvolupat un joc i aquest no compleix aquest requisit. La manca de temps i els recursos van limitar la capacitat d'implementar jocs addicionals que aborden específicament limitacions concretes.

En resum, s'han aconseguit la majoria dels objectius plantejats inicialment. No obstant això, hi ha aspectes en què es poden fer millores en futurs projectes per abordar els requisits no complerts i ampliar l'aplicació Ludoscrum.

Conclusions sobre la planificació

En general, es va poder seguir el curs de la planificació establerta per a la feina. Tot i això, es reconeix que hi va haver algunes deficiències en l'avaluació inicial de certs aspectes, com el cost de la integració amb l'API de l'eina de desenvolupament, en aquest cas, Slack. A més, la inclusió del registre d'usuari no va ser considerada com a part de l'abast inicial de la Prova de Concepte (PoC).

Tot i això, durant la fase d'implementació, es va evidenciar que era més pertinent incloure aquests desenvolupaments en lloc d'ampliar l'apartat de jocs. Això va implicar una reavaluació més detallada de la planificació per ajustar-la a les noves necessitats identificades.

El procés de reavaluació va permetre tenir una visió més precisa i realista dels recursos, temps i objectius requerits per assolir els resultats desitjats. Es van prendre decisions estratègiques i es va prioritzar la integració amb l'API de Slack i la implementació del registre d'usuari, considerats fonamentals per a l'èxit de la PoC.

Aquests ajustos en la planificació van demostrar ser encertats, així es va poder enfocar els esforços en aspectes crítics i necessaris. Tot i que va implicar una adaptació en el camí, considero que va ser una decisió correcta, jaqué en última instància, van contribuir de manera significativa a demostrar el potencial d'una aplicació de ludificació en entorns SCRUM.

4.2. Línies futures

Pel que fa a les accions a futur, es planteja implementar WebSocket com a solució per millorar la comunicació entre el frontend i el backend a curt termini. Aquesta integració permetrà una interacció més fluida i en temps real entre els diferents components del sistema. També està previst establir entorns de desenvolupament (DEV) i producció (PRO) en un flux d'integració continua, facilitant així el procés de desenvolupament i desplegament del projecte.

A mesura que avancem cap al mitjà i llarg termini, es contempla la tasca emocionant d'explorar les nombroses possibilitats de desenvolupar elements de ludificació sota la filosofia de Ludoscrum. Es planteja enfocar-se a la creació de petites peces de ludificació que siguin intuïtives, altament integrades i capaces d'enriquir l'experiència dels usuaris. L'objectiu és implementar aquestes peces en entorns reals, obrint així la porta a nous reptes i oportunitats. La retroalimentació directa dels usuaris serà especialment valuosa per millorar i adaptar les solucions de ludificació a mesura que avança el projecte.

Bibliografia

[1] American Tobacco Company and the Rise of Modern Advertising. Richard W. Pollay (1997).

[2] The Agile Manifesto, http://agilemanifesto.org/authors.html. Últim accés 2023-06-17.

[3] Agile Project Management with Scrum. Últim accés 2023-06-17.

[4] Gamification Global Market Report 2022: Ukraine-Russia War Impact. https://www.researchandmarkets.com/reports/5700430/gamification-global-market-report-2022-ukraine#src-pos-3. Últim accés 2023-06-17.

[5] Spain E-Learning Market Overview, 2027. https://www.researchandmarkets.com/reports/5671742/spain-e-learning-market-overview-2027. Últim accés 2023-06-17.

[6] NACIONES UNIDAS, Lenguaje inclusivo en cuanto al género. https://www.un.org/es/gender-inclusive-language/toolbox.shtml. Últim accés 2023-06-17.

[7] WCAG (Web Content Accessibility Guidelines). https://www.w3.org/WAI/fundamentals/accessibility-intro/es. Últim accés 2023-06-17.

[8] UML 2.5. https://www.omg.org/spec/UML/2.5. Últim accés 2023-06-17.

[9] MVC (Model-Vista-Controlador). https://www.gwtproject.org/articles/mvp-architecture.html. Últim accés 2023-06-17.

[10] Introducción a MEVN. https://bluuweb.github.io/mevn. Últim accés 2023-06-17.

[11] Patrones de diseño. https://refactoring.guru/es/design-patterns. Últim accés 2023-06-17.

[12] WCAG - Contrast Checker, Check the contrast of your color design for accessibility base on Web Content Accessibility Guideline (WCAG). https://contrastchecker.com. Últim accés 2023-06-17.

[13] Roboto. https://fonts.google.com/specimen/Roboto. Últim accés 2023-06-17.

[14] Usability 101: Introduction to Usability, World Leaders in Research-Based User Experience. https://www.nngroup.com/articles/usability-101-introduction-to-usability. Últim accés 2023-06-17.

[15] NoSQL. http://www.strozzi.it/cgi-bin/CSA/tw7/I/en_US/nosql/Home%20Page. Últim accés 2023-06-17.

[16] Docke. https://www.docker.com. Últim accés 2023-06-17.

[17] RFC 6749: The OAuth 2.0 Authorization Framework, Hardt. https://datatracker.ietf.org/doc/html/rfc6749. Últim accés 2023-06-17.

[18] ISO 9241-11:1998. https://www.iso.org/standard/16883.html. Últim accés 2023-06-17.

[19] Home, Affairs https://www.usability.gov. Últim accés 2023-06-17.

Glossari

Docker

Docker és una plataforma de codi obert que permet desenvolupar, empaquetar i desplegar aplicacions de manera eficient i portàtil. Es basa en la tecnologia de contenidors, que és una forma de virtualització lleugera que permet executar aplicacions de manera aïllada en un entorn controlat.

A Docker, una aplicació i totes les seves dependències s'empaqueten en un contenidor, que és una unitat estàndard i portàtil que es pot executar en qualsevol sistema operatiu que tingui Docker instal·lat. Els contenidors són lleugers, ràpids i comparteixen el mateix nucli del sistema operatiu subjacent, cosa que els fa més eficients en termes de recursos en comparació amb les màquines virtuals tradicionals.

Docker proporciona eines i ordres per crear, gestionar i desplegar contenidors de manera senzilla. Permet la creació d'imatges de contenidor a partir de fitxers de configuració anomenats "Dockerfile", que especifiquen l'entorn i les dependències necessàries per a l'aplicació. Aquestes imatges es poden compartir i distribuir mitjançant un registre centralitzat anomenat "Docker Hub" o en repositoris privats.

Els avantatges principals d'utilitzar Docker inclouen la portabilitat de les aplicacions, la facilitat de desplegament en diferents entorns, l'eficiència en l'ús de recursos i la capacitat d'escalar ràpidament aplicacions de forma horitzontal. Docker ha guanyat popularitat en el desenvolupament d'aplicacions modernes, facilitant la creació d'entorns de desenvolupament consistents, el desplegament de microserveis i la implementació d'arquitectures basades en contenidors.

Docker Compose

Docker Compose és una eina que s'utilitza en conjunt amb Docker per facilitar la definició i gestió d'aplicacions multicontenidor. Permet descriure la configuració de diversos contenidors en un sol fitxer de configuració anomenat "docker-compose.yml".

Amb Docker Compose, podeu definir els serveis, xarxes i volums necessaris per a la vostra aplicació, així com les configuracions específiques de cada contenidor. Això inclou detalls com les imatges de Docker a utilitzar, els ports a exposar, les variables dentorn, els volums compartits, entre altres.

Un cop teniu definit el fitxer docker-compose.yml, podeu utilitzar l'ordre "docker-compose" per crear i gestionar tots els contenidors definits a la configuració de manera conjunta. Això simplifica el procés d'execució i coordinació de múltiples contenidors, ja que Docker Compose s'encarrega de crear les xarxes i els volums necessaris, i assegura que els contenidors s'iniciïn i es comuniquin correctament.

Docker Compose és especialment útil en entorns de desenvolupament i proves, on és comú tenir una combinació de serveis i dependències que cal executar conjuntament. Amb una sola instrucció, podeu crear i configurar tot l'entorn necessari per a la vostra aplicació.

En resum, Docker Compose és una eina que simplifica la gestió d'aplicacions multicontenidor en permetre definir i orquestrar fàcilment tots els components necessaris per a la teva aplicació en un únic fitxer de configuració. Això facilita el desenvolupament, el desplegament i el manteniment d'aplicacions basades en contenidors.

GitHub

GitHub és una plataforma web que proporciona un sistema de control de versions distribuït i una funcionalitat d'allotjament de repositoris de codi font. GitHub permet als desenvolupadors col·laborar en projectes de programari, compartir codi i realitzar un seguiment de les revisions i canvis realitzats en el codi.
Els repositoris de codi en GitHub poden ser públics, la qual cosa permet que qualsevol persona accedeixi al codi i contribueixi al projecte, o privats, la qual cosa limita l'accés al codi als membres específics de l'equip. Els usuaris poden fer canvis en el codi i enviar sol·licituds d'extracció (pull requests), que permeten a uns altres revisar i aprovar o rebutjar els canvis abans que es fusionin amb el codi principal.
A més del control de versions i la col·laboració en el codi, GitHub també ofereix característiques addicionals, com un sistema de seguiment de problemes, wikis i un sistema de gestió de projectes. És àmpliament utilitzat per la comunitat de desenvolupadors i és particularment popular entre els desenvolupadors de programari de codi obert, a causa de la seva facilitat d'ús i àmplia gamma de característiques de col·laboració.

Jira

Jira és un programari de gestió de projectes i seguiment d'incidències (també conegut com a eina de seguiment de problemes) desenvolupat per l'empresa Atlassian. És àmpliament utilitzat en la gestió de projectes de programari i en el seguiment d'incidències.
Permet als equips de treball crear i gestionar tasques, incidències i projectes a través d'un sistema de seguiment en línia. Els usuaris poden crear projectes i tasques, assignar-les, establir dates límit i fer seguiment del progrés al llarg del temps. També poden afegir comentaris, adjuntar arxius, etiquetes, realitzar cerques i generar informes.

MongoDB Atles

MongoDB Atlas és un servei de base de dades al núvol ofert per MongoDB, l'empresa darrere de la base de dades NoSQL MongoDB. És una plataforma de base de dades com a servei (DBaaS) que permet als desenvolupadors crear, configurar i administrar clústers de bases de dades MongoDB de manera senzilla i escalable al núvol.

MongoDB Atlas proporciona una infraestructura de base de dades completament administrada, cosa que significa que MongoDB s'encarrega de les tasques de manteniment, la configuració del clúster, les còpies de seguretat automàtiques i el monitoratge de la base de dades. Això permet als desenvolupadors centrar-se en el desenvolupament d'aplicacions i no haver de preocupar-se de la gestió de la infraestructura de la base de dades.

En utilitzar MongoDB Atlas, els desenvolupadors poden aprofitar les característiques i avantatges de MongoDB, com la flexibilitat d'esquema, la capacitat d'escalar horitzontalment i el potent llenguatge de consulta. També ofereix opcions avançades de seguretat i compliment de normatives per protegir les dades emmagatzemades.

A més, MongoDB Atlas és compatible amb múltiples proveïdors de núvol, cosa que permet als desenvolupadors triar la plataforma on vulguin implementar la seva base de dades, com Amazon Web Services (AWS), Microsoft Azure o Google Cloud Platform (GCP).

Slack

Slack és una eina de comunicació i col·laboració que permet als usuaris comunicar-se i col·laborar en temps real. És una aplicació de missatgeria que s'utilitza principalment en entorns empresarials.
Ofereix diferents canals per a la comunicació, on els usuaris poden enviar missatges de text, veu o vídeo, compartir arxius i documents, així com integrar altres eines de programari que fan servir en el seu treball diari. A més, Slack permet la creació de grups de treball i la segmentació de canals segons els projectes o àrees de treball.

Zeplin

Zeplin és una eina de disseny i col·laboració que ajuda els dissenyadors i desenvolupadors a treballar junts en projectes de disseny d'interfície d'usuari (UI) i disseny d'experiència d'usuari (UX).
Zeplin permet als dissenyadors carregar els seus dissenys de UI i UX en la plataforma i compartir-los amb el seu equip de desenvolupament, proporcionant un espai centralitzat per a la col·laboració en els projectes de disseny. Zeplin també genera automàticament especificacions precises dels dissenys, la qual cosa ajuda als desenvolupadors a comprendre els detalls i les dimensions de cada element en el disseny.

A més, Zeplin permet als desenvolupadors descarregar assets, com a imatges, icones i fonts, directament des de la plataforma, la qual cosa accelera el procés de desenvolupament i redueix la possibilitat d'errors.

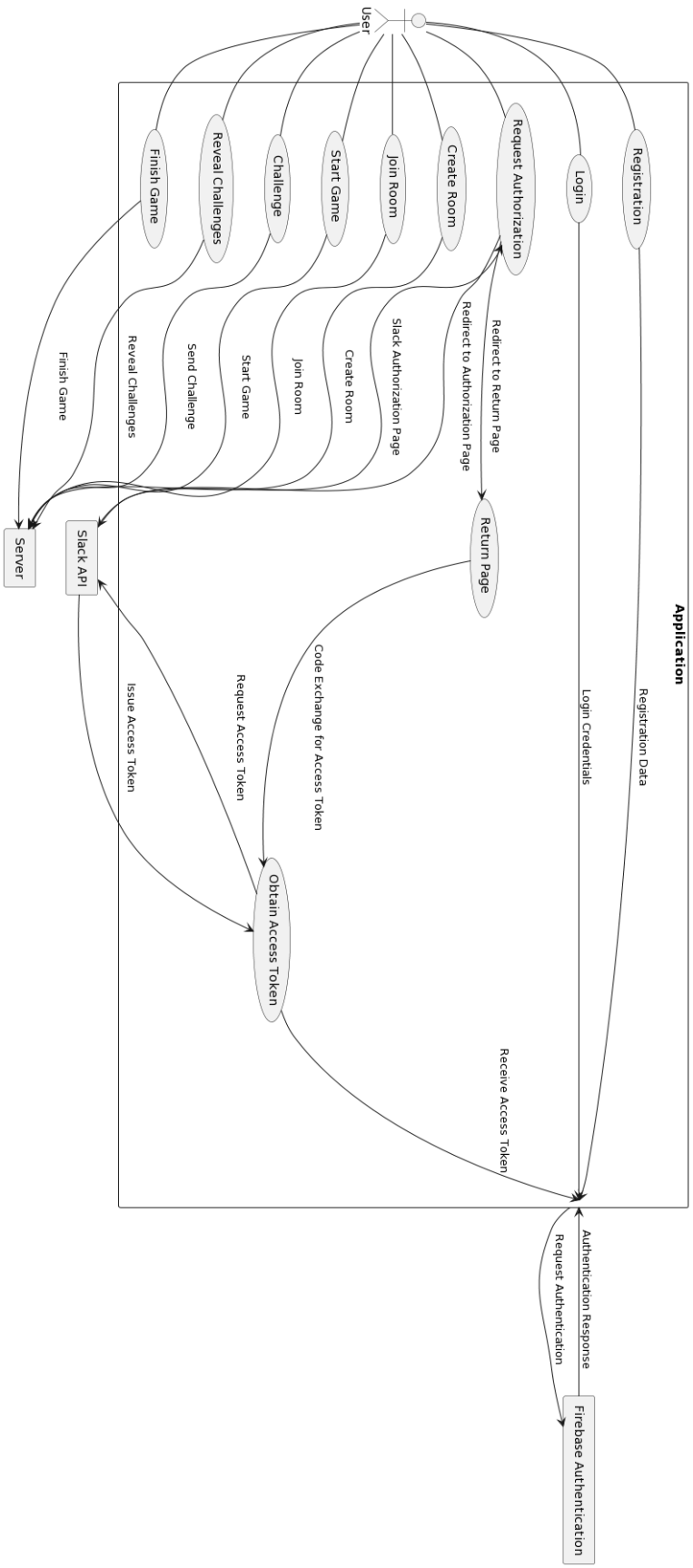
Apèndix

Diagramés UML i codis Plantuml

Casos d'ús crítics

```
@startuml

left to right direction

actor User as user
rectangle "Application" as app {
  usecase "Registration" as registration
  usecase "Login" as login
  usecase "Request Authorization" as request_authorization
  usecase "Obtain Access Token" as obtain_token
  usecase "Return Page" as return_page
  usecase "Create Room" as create_room
  usecase "Join Room" as join_room
  usecase "Start Game" as start_game
  usecase "Challenge" as challenge
  usecase "Reveal Challenges" as reveal_challenges
  usecase "Finish Game" as finish_game
}
rectangle "Slack API" as slack_api
rectangle "Firebase Authentication" as firebase_auth
rectangle "Server" as server

user -- login
user -- registration
user -- request_authorization
user -- create_room
user -- join_room
user -- start_game
user -- challenge
user -- reveal_challenges
user -- finish_game
login --> app : Login Credentials
registration --> app : Registration Data
request_authorization --> slack_api : Redirect to Authorization Page
slack_api --> request_authorization : Slack Authorization Page
request_authorization --> return_page : Redirect to Return Page
return_page --> obtain_token : Code Exchange for Access Token
```

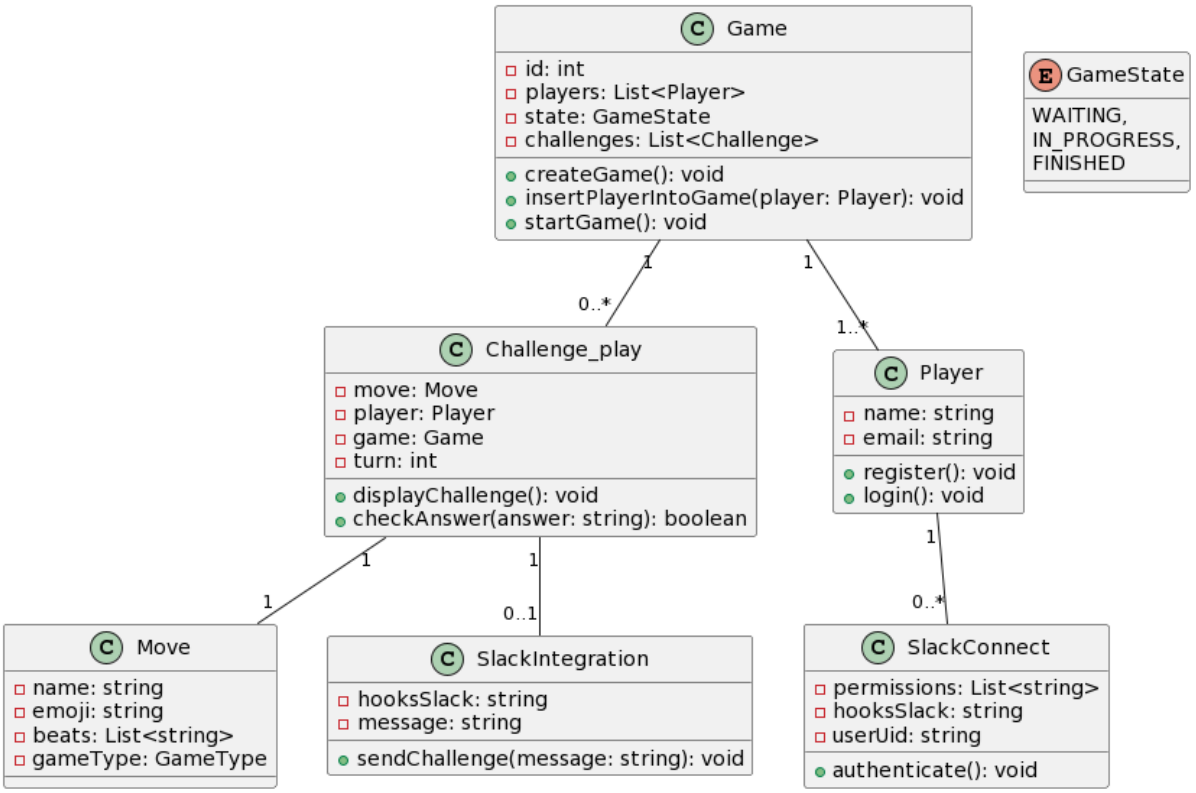
```
obtain_token --> slack_api : Request Access Token
slack_api --> obtain_token : Issue Access Token
obtain_token --> app : Receive Access Token
app --> firebase_auth : Request Authentication
firebase_auth --> app : Authentication Response
create_room --> server : Create Room
join_room --> server : Join Room
start_game --> server : Start Game
challenge --> server : Send Challenge
reveal_challenges --> server : Reveal Challenges
finish_game --> server : Finish Game

@enduml
  - userUid: string
  + authenticate(): void
}

enum GameState {
  WAITING,
  IN_PROGRESS,
  FINISHED
}

Game "1" -- "1..*" Player
Game "1" -- "0..*" Challenge_play
Challenge_play "1" -- "1" Move
Challenge_play "1" -- "0..1" SlackIntegration
Player "1" -- "0..*" SlackConnect

@enduml
```


User
Application
Registration
Login
Request Authorization
Create Room
Join Room
Start Game
Challenge
Reveal Challenges
Finish Game
Registration Data
Login Credentials
Redirect to Authorization Page
Slack Authorization Page
Create Room
Join Room
Start Game
Send Challenge
Reveal Challenges
Finish Game
Return Page
Code Exchange for Access Token
Request Access Token
Issue Access Token
Receive Access Token
Obtain Access Token
Server
Slack API
Firebase Authentication
Request Authentication
Authentication Response

Entitats i les seves relacions

```
@startuml

class Game {
  - id: int
  - players: List<Player>
  - state: GameState
  - challenges: List<Challenge>
  + createGame(): void
  + insertPlayerIntoGame(player: Player): void
  + startGame(): void
}

class Challenge_play {
  - move: Move
  - player: Player
  - game: Game
  - turn: int

+ displayChallenge(): void
  + checkAnswer(answer: string): boolean
}

class Move {
  - name: string
  - emoji: string
  - beats: List<string>
  - gameType: GameType
}

class Player {
  - name: string
  - email: string
  + register(): void
  + login(): void
}


class SlackIntegration {
  - hooksSlack: string
  - message: string
  + sendChallenge(message: string): void
}

class SlackConnect {
  - permissions: List<string>
```

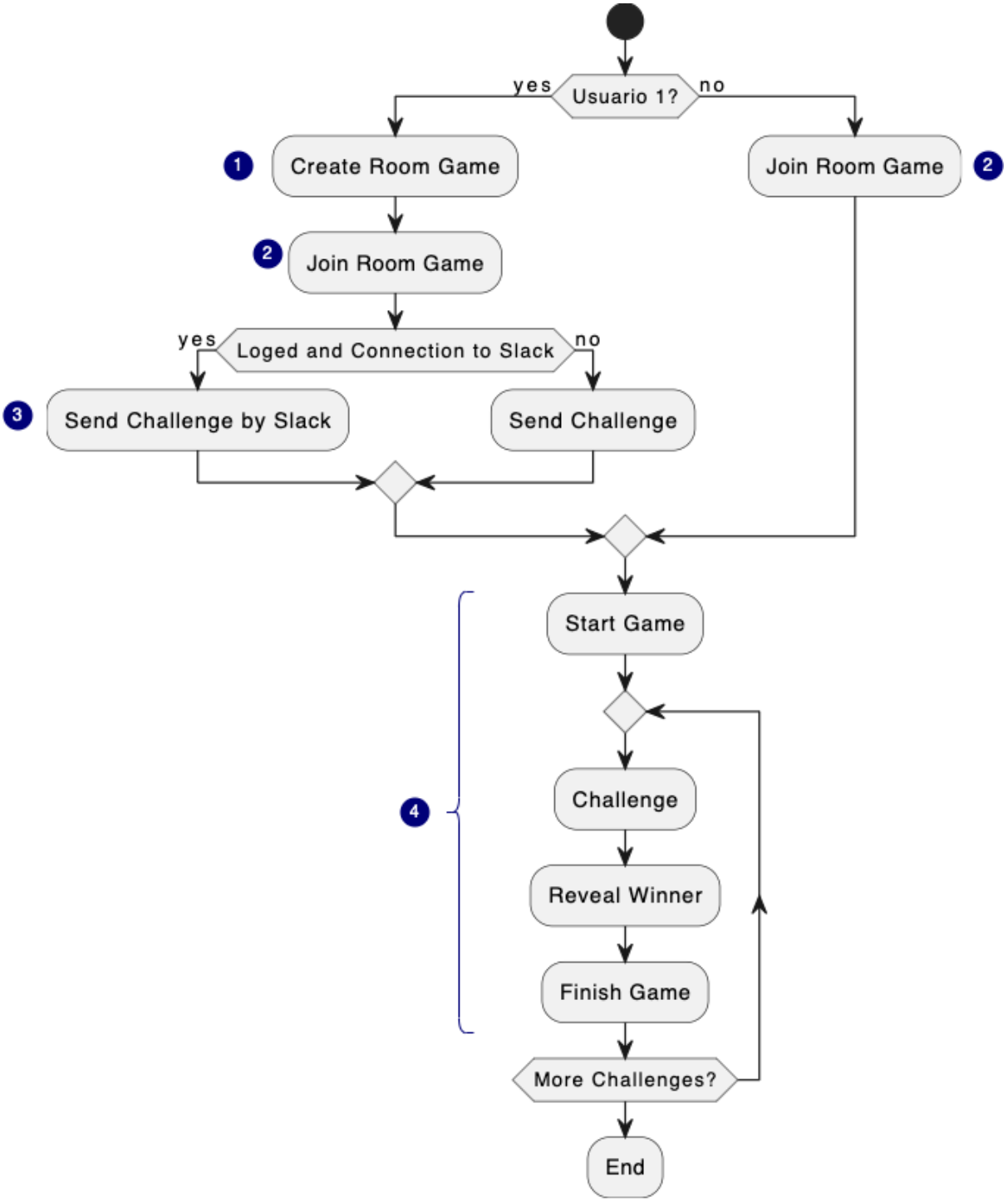
```
  - hooksSlack: string
  - userUid: string
  + authenticate(): void
}

enum GameState {
  WAITING,
  IN_PROGRESS,
  FINISHED
}

Game "1" -- "1..*" Player
Game "1" -- "0..*" Challenge_play
Challenge_play "1" -- "1" Move
Challenge_play "1" -- "0..1" SlackIntegration
Player "1" -- "0..*" SlackConnect

@enduml
```



Navegació i usabilitat

```
@startuml

class Game {
  - id: int
  - players: List<Player>
  - state: GameState
  - challenges: List<Challenge>
  + createGame(): void
  + insertPlayerIntoGame(player: Player): void
  + startGame(): void
}

class Challenge_play {
  - move: Move
  - player: Player
  - game: Game
  - turn: int

+ displayChallenge(): void
  + checkAnswer(answer: string): boolean
}

class Move {
  - name: string
  - emoji: string
  - beats: List<string>
  - gameType: GameType
}

class Player {
  - name: string
  - email: string
  + register(): void
  + login(): void
}


class SlackIntegration {
  - hooksSlack: string
  - message: string
  + sendChallenge(message: string): void
}

class SlackConnect {
  - permissions: List<string>
```

```
  - hooksSlack: string
  - userUid: string
  + authenticate(): void
}

enum GameState {
  WAITING,
  IN_PROGRESS,
  FINISHED
}

Game "1" -- "1..*" Player
Game "1" -- "0..*" Challenge_play
Challenge_play "1" -- "1" Move
Challenge_play "1" -- "0..1" SlackIntegration
Player "1" -- "0..*" SlackConnect

@enduml
```


Usuario 1?
yes
no
1
Create Room Game
Join Room Game
2
2
Join Room Game
Loged and Connection to Slack
yes
no
3
Send Challenge by Slack
Send Challenge
Start Game
4
Challenge
Reveal Winner
Finish Game
More Challenges?
End

Fitxes adjunts de l'apèndix

Al directori adjunt "Fixes-d-Apendix" es trobaran els següents documents::

- Temporitzacio-diagrama-Gantt.pdf
- Temporitzacio-diagrama-Gantt.xlsx

Els dos documents pertanyen a l'apartat planificació i contenen el diagrama de Gantt de la planificació del projecte en els formats .pdf i .xlsx, respectivament.