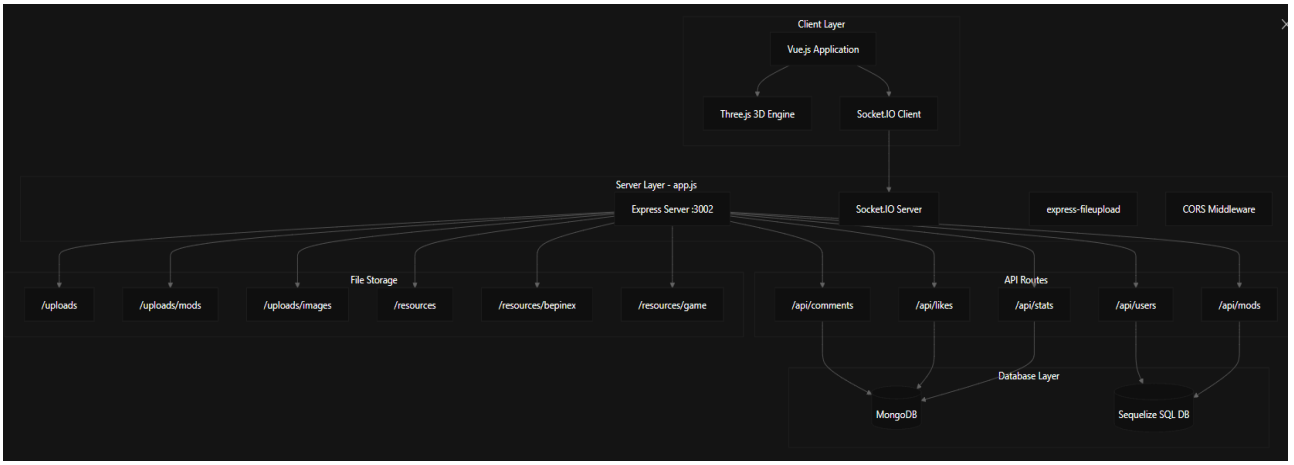
Hasham Hussain
Albert Robert
Marc Castro
David Muñoz
Matthew Castilla

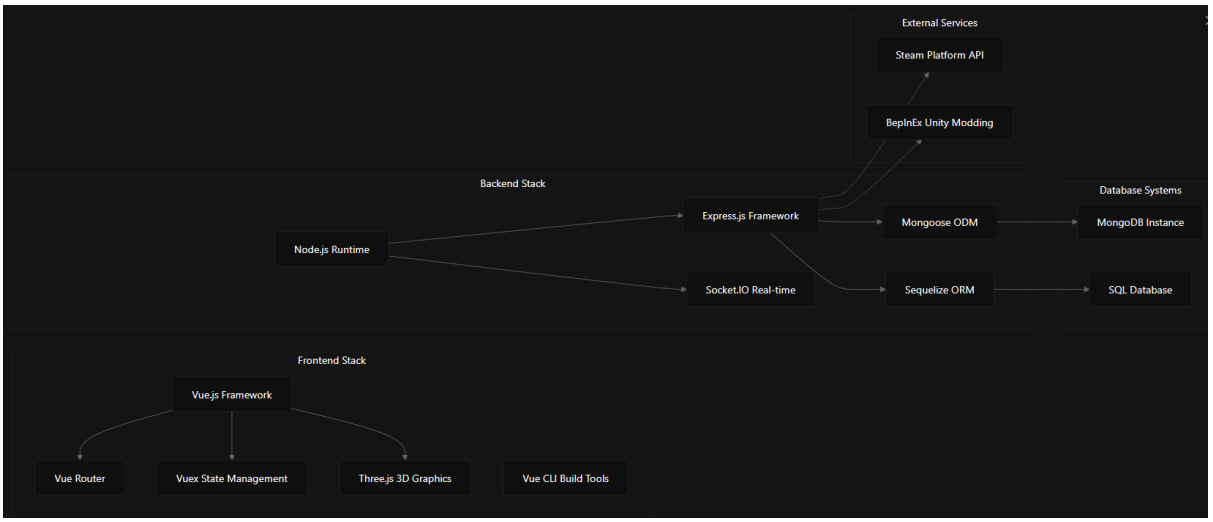
Documentació Tècnica Front/Back
Darkness Unseen

Aquest document ofereix una visió general completa de Darkness Unseen, un sistema de jocs de terror cooperatiu que combina tecnologies web amb el desenvolupament de jocs Unity. El sistema inclou una plataforma de jocs completa amb integració de Steam, capacitats multijugador en temps real, gestió de contingut generat pels usuaris i una arquitectura híbrida que admet tant interfícies web com clients de jocs Unity.

Visió general del sistema
Darkness Unseen implementa una moderna arquitectura de jocs basada en web que combina tecnologies web tradicionals amb renderització de gràfics en 3D i capacitats multijugador en temps real. El sistema admet la integració de Steam, el contingut generat per l'usuari mitjançant modificacions i una plataforma social integral per a la interacció amb la comunitat

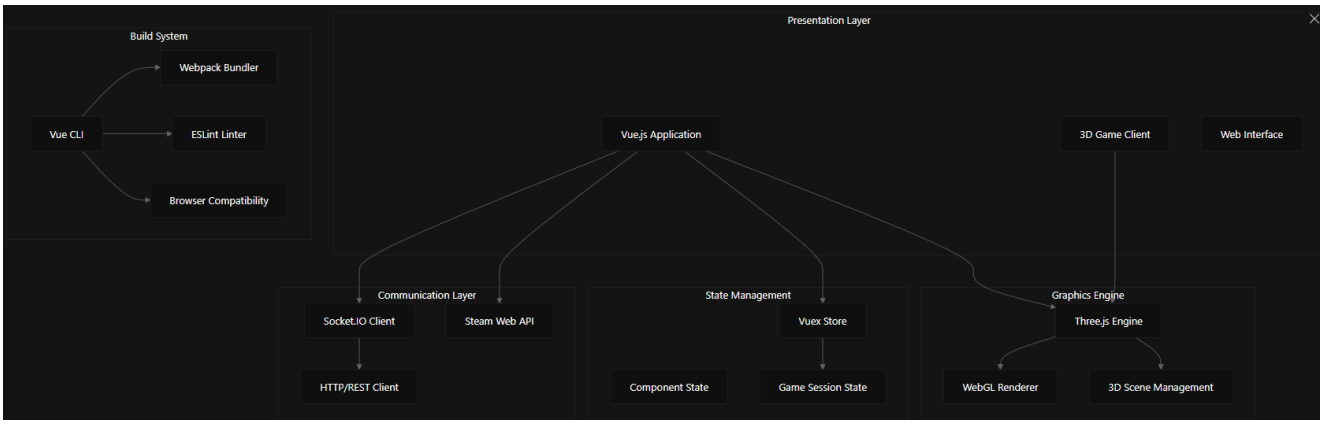


Components tecnològics primaris

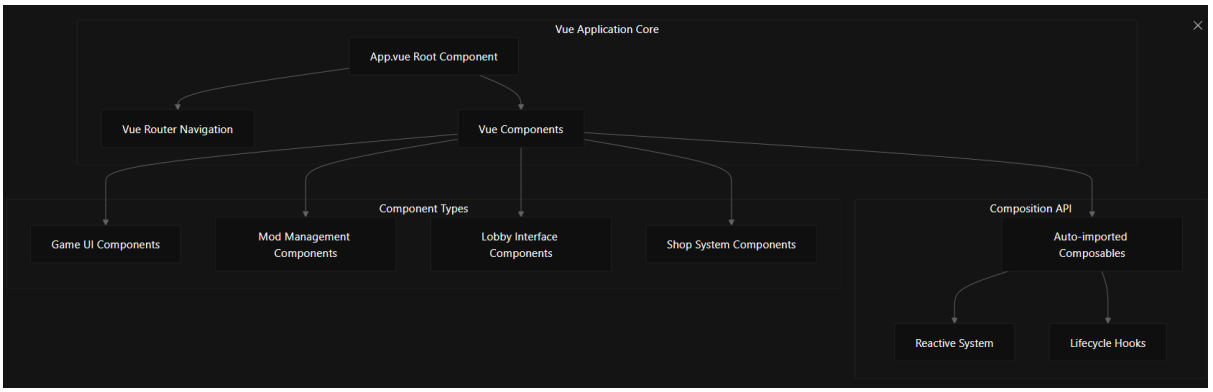


Arquitectura Frontend

La interfície segueix una arquitectura en capes que combina tecnologies web amb capacitats gràfiques en 3D:



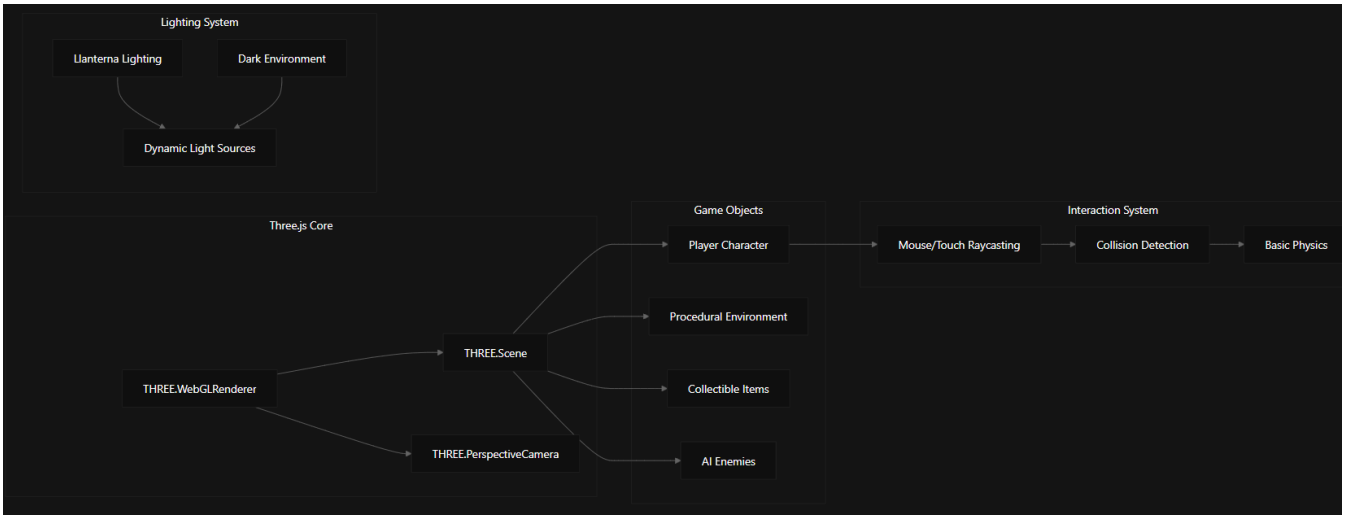
Components bàsics de Vue.js



La configuració d'ESLint permet la importació automàtica de les funcions de l'API de composició de Vue 3, com ara:

- ref, reactiu, calculat per a la reactivitat
- onMounted, onBeforeMount, onUnmounted per a la gestió del cicle de vida
- useRoute, useRouter per a la navegació
- watch, watchEfecte per a observadors reactius

Integració de gràfics 3D Three.js
El client del joc integra Three.js per a la representació de gràfics en 3D en temps real, proporcionant l'entorn fosc i la generació de nivells procedimentals:



Les responsabilitats clau de Three.js inclouen:

- Generació de nivells de procediment per a entorns de joc aleatoris
- Simulació de foscor amb visibilitat limitada de la llanterna
- Il·luminació en temps real per a l'element Llanterna
- Visualització de l'IA de l'enemic per a entitats sonores Stalker i Sensor
- Interacció d'articles per a articles de col·lecció i de botiga

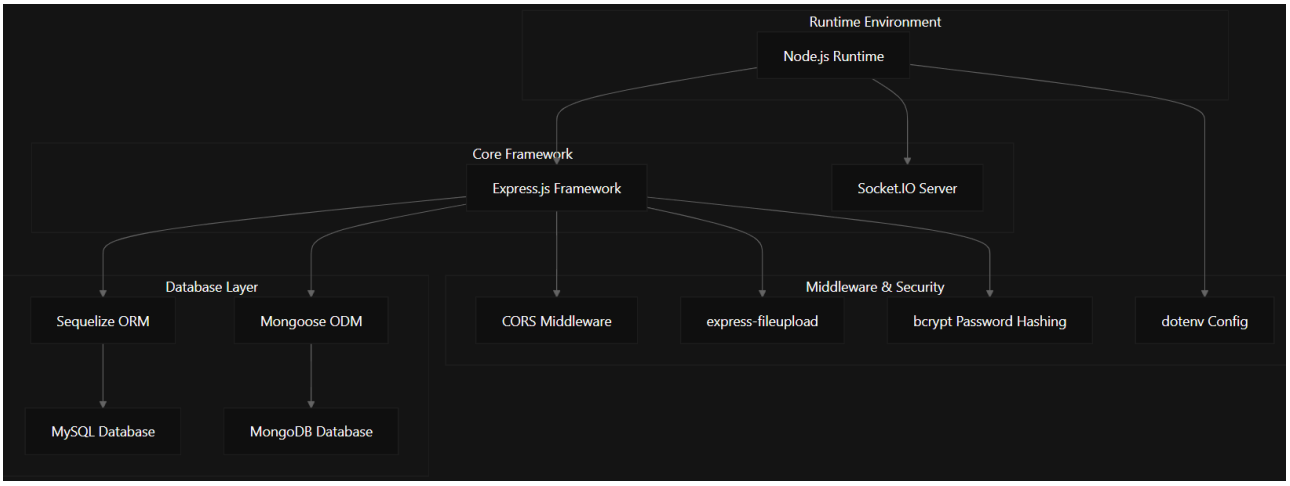
Arquitectura de back

Per a les integracions de l'API de fons, vegeu Arquitectura de fons. Per obtenir informació sobre la comunicació multijugador en temps real, consulta Comunicació en temps real. Per obtenir informació específica sobre la integració de la plataforma Steam, vegeu Integració de Steam.

Visió general de la pila de tecnologia

El backend es construeix utilitzant tecnologies modernes de Node.js amb un enfocament en la comunicació en temps real i el suport de bases de dades duals.

Pila de tecnologia de fons



Punt d'entrada de l'aplicació i configuració del servidor

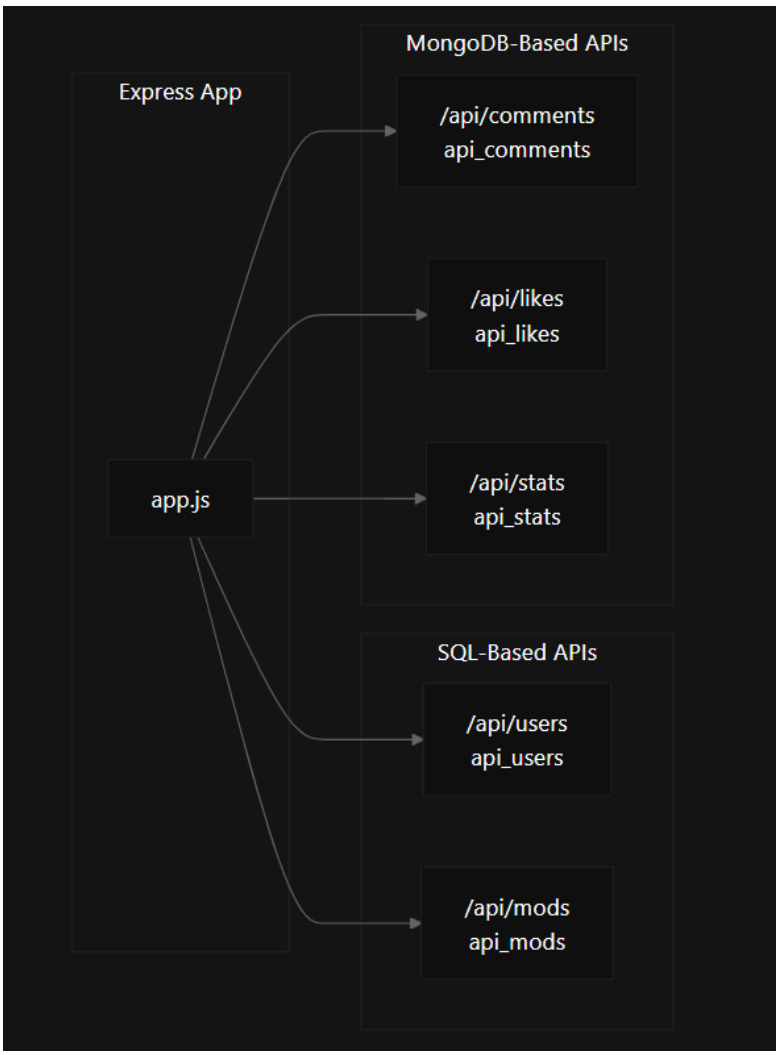
El fitxer d'aplicació principal app.js orquestra tot el sistema de fons, inicialitzant tots els components bàsics i establint connexions.

Flux d'inicialització del servidor

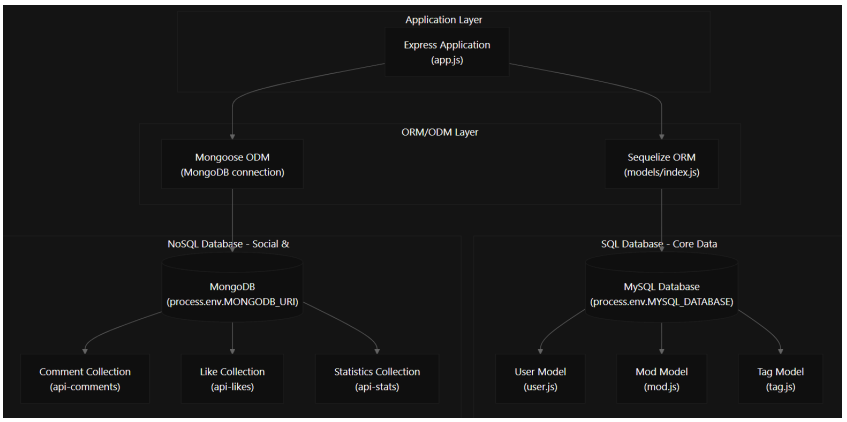
Procés d'arrencada de l'aplicació

Estructura de ruta de l'API

Punts finals de l'API i gestors de rutes



Arquitectura de bases de dades



Arquitectura MySQL amb Sequelize

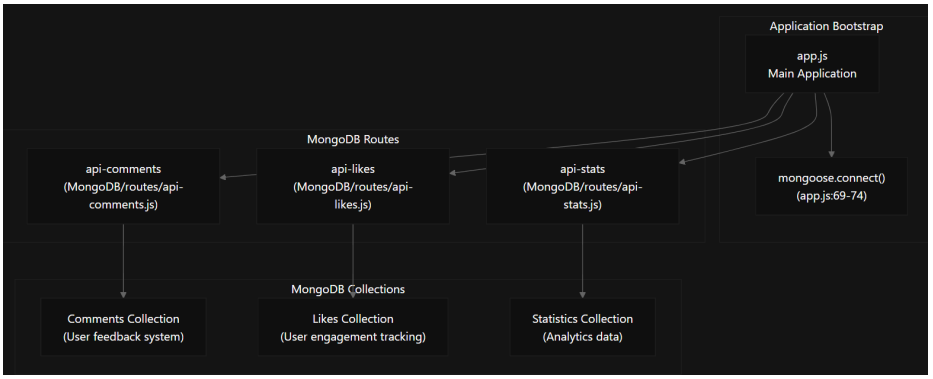
La base de dades MySQL serveix com a magatzem de dades principal per a dades d'aplicacions estructurades, gestionades mitjançant l'ORM Sequelize.

Seqüelitzar la configuració i la connexió

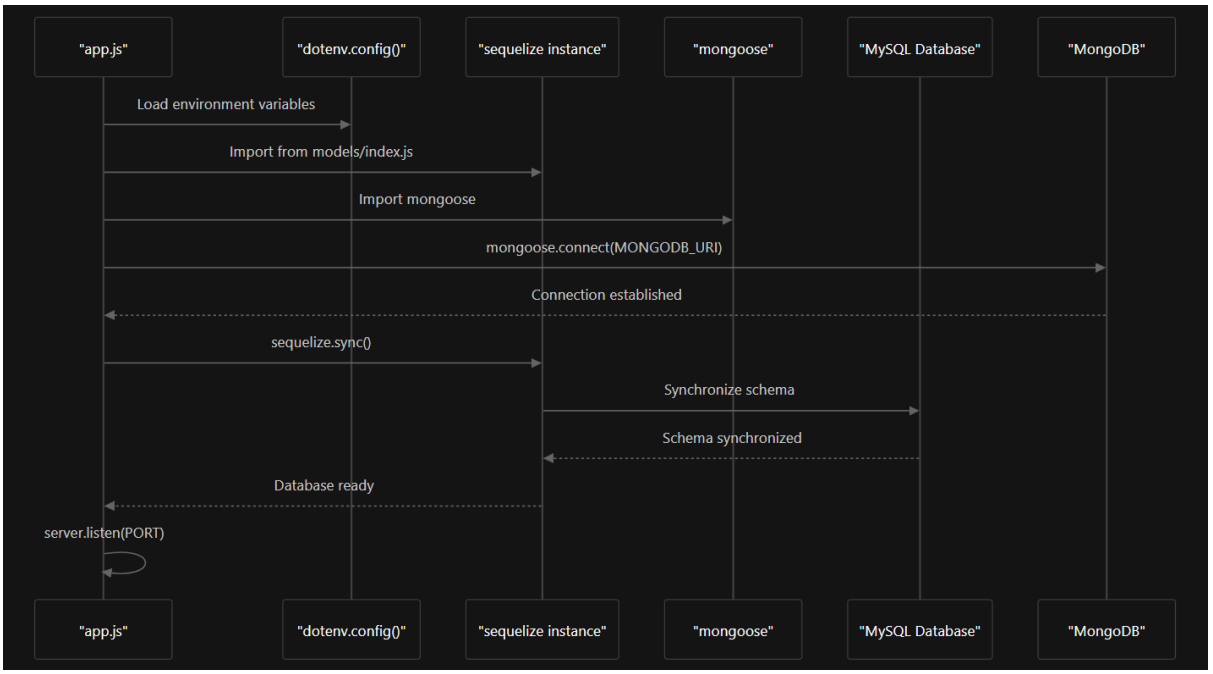
Arquitectura MongoDB amb Mongoose

MongoDB gestiona dades dinàmiques basades en documents per a funcions i anàlisis socials mitjançant Mongoose ODM.

Connexió de MongoDB i integració de rutes



Flux d'inicialització de la connexió



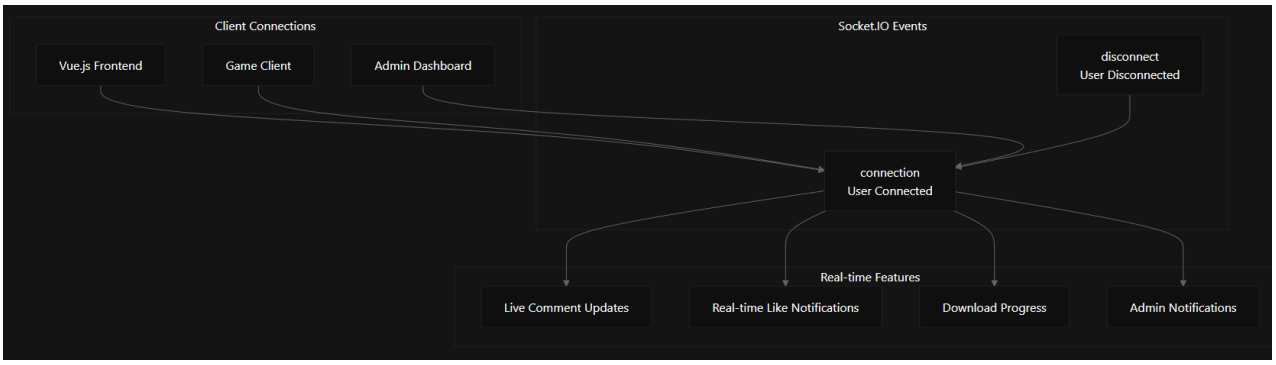
Documentació de l'API
Aquesta part proporciona documentació completa per a l'API REST Darkness Unseen, que cobreix tots els punts finals disponibles, mecanismes d'autenticació, formats de sol·licitud/resposta i funcions de comunicació en temps real. L'API serveix com a interfície principal entre l'interfície Vue.js i el backend Node.js, permetent la gestió d'usuaris, la distribució de modificacions, les funcions socials i les estadístiques del joc.

Estructura d'encaminament de l'API

Comunicació en temps real

Configuració de Socket.IO

L'API inclou suport WebSocket per a funcions en temps real mitjançant Socket.IO.



Esdeveniments Socket.IO

El servidor Socket.IO gestiona els esdeveniments bàsics següents:

- connexió: s'activa quan un client es connecta
- desconnectar: s'activa quan un client es desconnecta
- S'implementen esdeveniments addicionals en mòduls d'API individuals per a actualitzacions en temps real.

Càrrega de fitxers i recursos estàtics

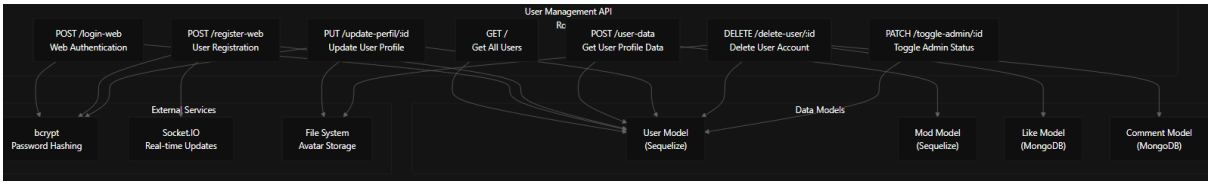
Configuració de càrrega de fitxers

L'API admet càrregues de fitxers mitjançant programari intermediari de pujada de fitxers express per a fitxers de modificacions, avatars d'usuari i recursos de joc.

API de gestió d'usuaris

Visió general del punt final de l'API

L'API de gestió d'usuaris proporciona punts finals RESTful per a totes les operacions relacionades amb l'usuari, des del registre inicial fins a la gestió de perfils i controls administratius. L'API s'integra tant amb sistemes SQL (dades principals de l'usuari) com MongoDB (dades d'activitat de l'usuari).



Endpoint d'autenticació

Registre d'usuari

El punt final de registre crea nous comptes d'usuari amb hash de contrasenya i validació duplicada.

/registre-web

Body de la sol·licitud:

- nom d'usuari: cadena, identificador únic
- correu electrònic: cadena, adreça electrònica única
- contrasenya: cadena, contrasenya de text sense format

/login-web

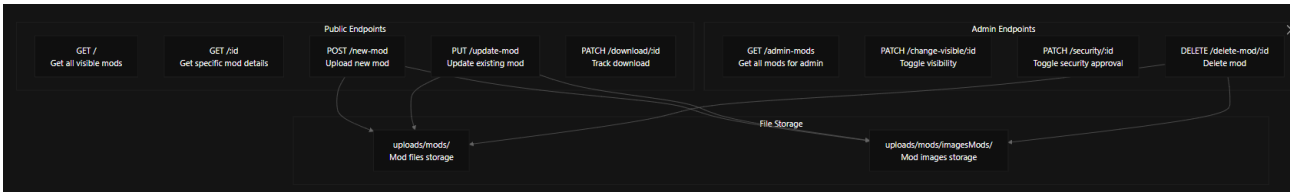
Òrgan de la sol·licitud:

- correu electrònic: cadena, adreça de correu electrònic registrada
- contrasenya: cadena, contrasenya de text sense format

API de gestió de modificacions

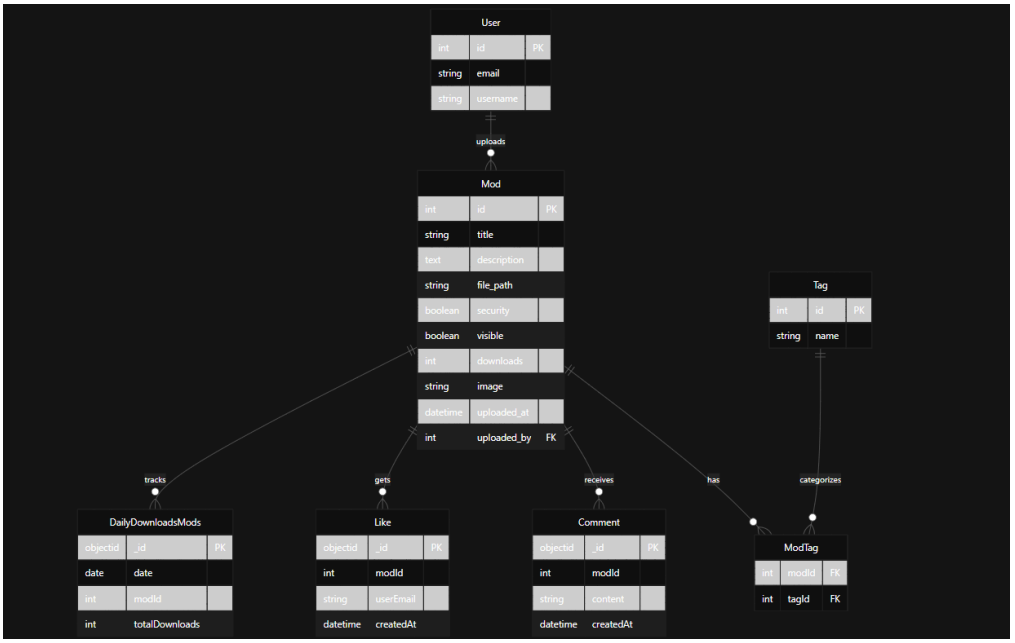
Visió general dels punts finals de l'API

El sistema de gestió de modificacions exposa els punts finals REST mitjançant el prefix de ruta /api/mods, gestionant tant l'accés públic a les modificacions com les funcions administratives.



Models de dades i relacions

El sistema de modificacions utilitza un enfocament híbrid SQL-MongoDB amb models Sequelize per a dades bàsiques i models Mongoose per a l'anàlisi.



Pujada i creació de mods

El procés de creació de modificacions gestiona les càrregues de fitxers de diverses parts tant per al fitxer de modificacions com per a la imatge de vista prèvia:

La funció handleFileUpload a
back/routes/api-mods.js-117-129
genera noms de fitxer únics mitjançant prefixos UUID i gestiona l'emmagatzematge de fitxers als directoris designats.

API d'estadístiques

Visió general

L'API d'estadístiques implementa un sistema d'anàlisi basat en MongoDB que fa un seguiment de les mètriques agregades diàries per a la plataforma de modificació. El sistema captura dues categories principals d'estadístiques:

Estadístiques de descàrrega: recompte de descàrregues diàries per mod individual
Estadístiques del cicle de vida del mod: recomptes diaris de noves creacions i supressions de modificacions
L'API utilitza MongoDB per a l'emmagatzematge flexible de dades d'anàlisi d'esquemes, independent de la base de dades SQL principal que s'utilitza per a les dades de l'aplicació bàsica.

Punts finals de l'API

GET /api/stats/stats-mods
Recupera tots els registres diaris d'estadístiques de modificacions de la plataforma.

Punt final: GET /api/stats/stats-mods

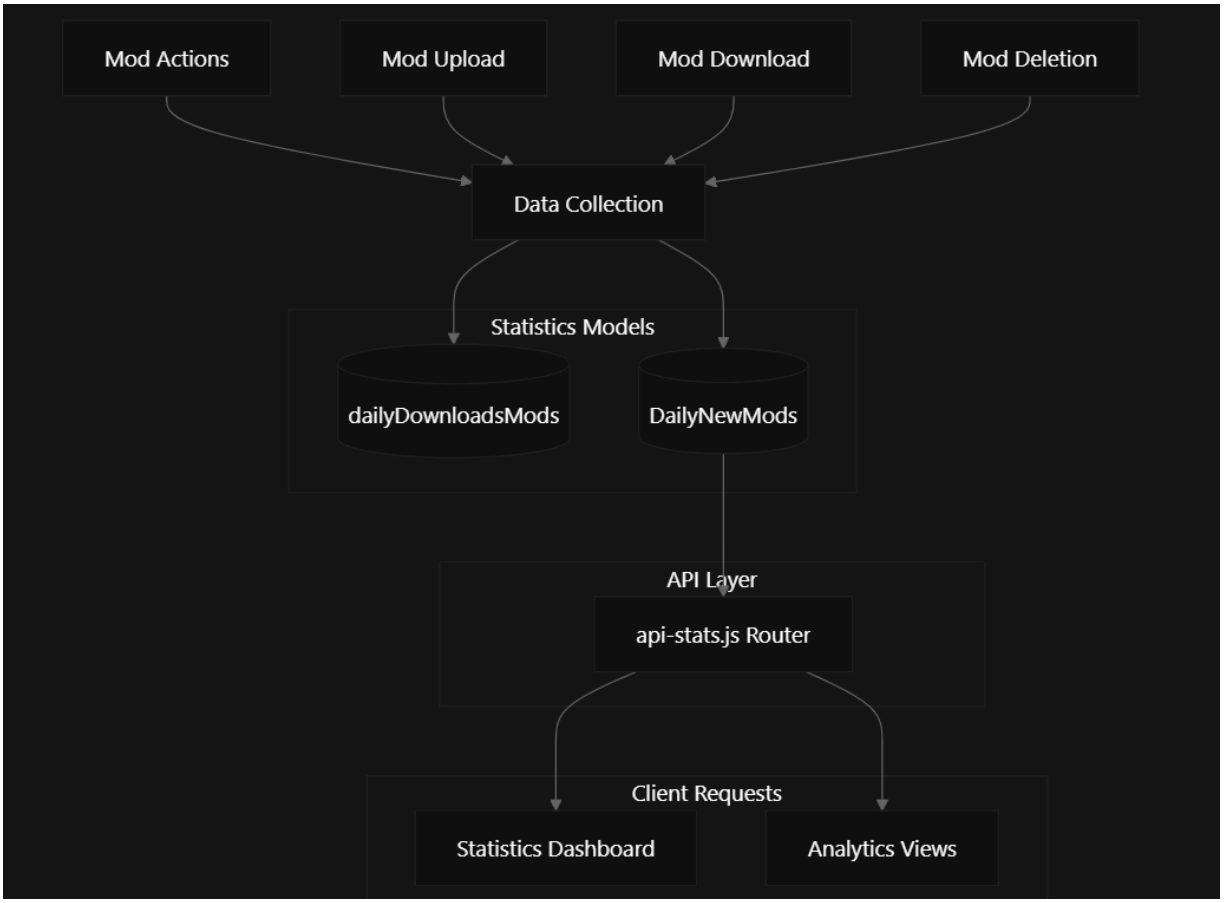
Format de resposta:

```
[
  {
    "_id": "64f1a2b3c4d5e6f7g8h9i0j1",
    "data": "2024-01-15T00:00:00.000Z",
    "newMods": 5,
    "deletedMods": 1
  }
]
```

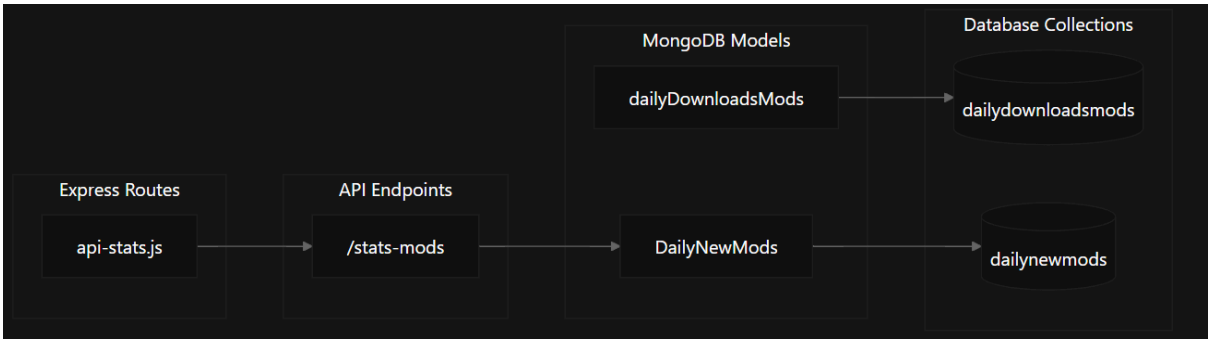
Codis de resposta:

- 200: Éxit: retorna una sèrie d'estadístiques diàries
- 500: Error del servidor: la consulta de la base de dades ha fallat

Flux de dades estadístiques



Arquitectura de l'API d'estadístiques



Detalls d'implementació

Configuració del model

Tots dos models estadístics utilitzen les estratègies de gestió i indexació de dates automàtica de MongoDB:

Gestió de dates: els valors de data predeterminats utilitzen data new().toISOString().split('T')[0] per garantir un format de data coherent diari

Estratègia d'indexació: els índexs únics impedeixen les entrades d'estadístiques duplicades

Disseny d'esquemes: estructura de document senzilla optimitzada per a consultes d'anàlisi de sèries temporals

Implementació del router

L'encaminador d'estadístiques (api-stats.js) implementa una interfície REST mínima:

Estructura d'importació: utilitza importacions de mòduls ES6 per als models MongoDB

Gestió d'errors: captura els errors de la base de dades i retorna els codis d'estat HTTP adequats

Format de resposta: retorna documents MongoDB en brut com a matrius JSON

Limitacions actuals

A partir de la implementació actual, l'API d'Estadístiques té diverses àrees per millorar-les:

Punts finals limitats: només s'ha implementat un punt final (/stats-mods)

Sense accés a les estadístiques de descàrrega: no hi ha cap punt final de l'API per recuperar dades diàries de DescàrreguesMods

Sense filtre: no hi ha capacitats de filtratge ni d'agregació d'intervals de dates

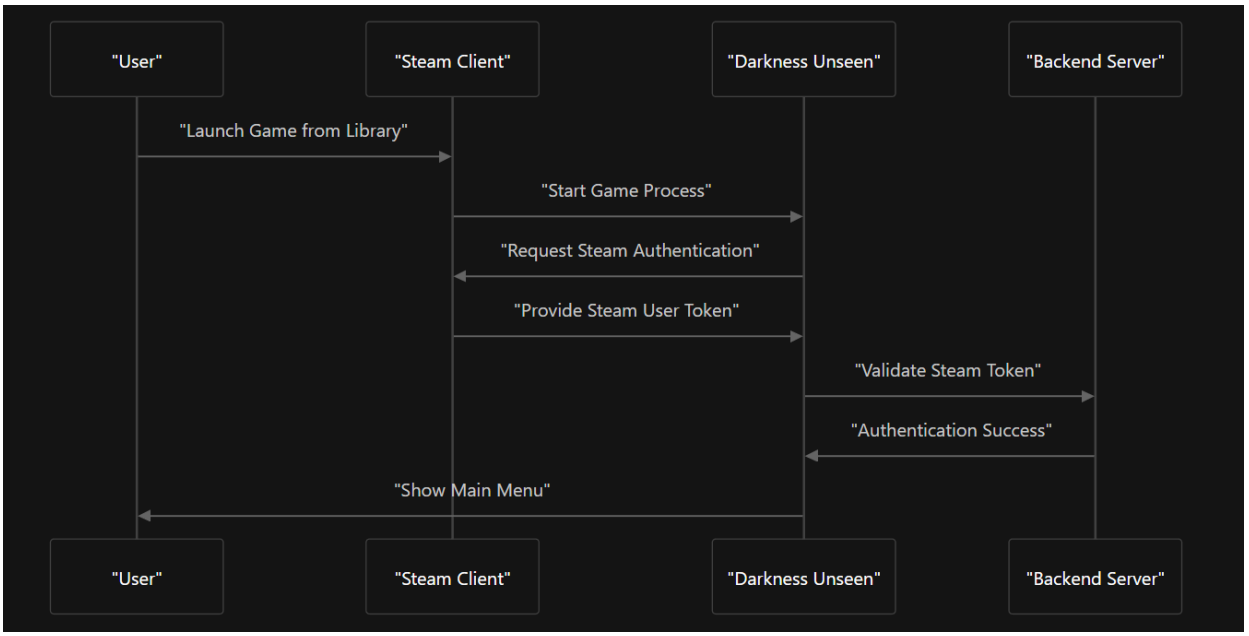
Sense autenticació: no hi ha requisits d'autenticació aparents per a l'accés a les estadístiques

Integració de Steam

La integració de Steam proporciona els serveis bàsics de la plataforma per a Darkness Unseen, que permet:

Autenticació d'usuari: inici de sessió automàtic mitjançant credencials de Steam
Funcions socials: integració de la llista d'amics de Steam per a les invitacions de jocs
Sessions multijugador: creació i unió de lobby basats en Steam
Distribució del joc: Llançament i gestió a través del client Steam
Requisits de la plataforma: integració del client Steam basat en Windows
Sistema d'autenticació
Darkness Unseen implementa l'autenticació automàtica de Steam per oferir una experiència d'inici de sessió perfecta als jugadors.

Flux d'autenticació



El procés d'autenticació elimina la necessitat de credencials d'usuari separades aprofitant la sessió de Steam existent. Els jugadors han de tenir Steam en funcionament i haver iniciat sessió per accedir al joc.