TRABAJO DE FIN DE GRADO

REVRB

DIGITAL AUDIO WORKSTATION

Lluís Enric Santos Linares

// Institut El Puig Castellar

// Desarrollo de Aplicaciones Multiplataforma

2025 – 2026

Resum del projecte

REVRB és una estació de treball d'àudio digital (DAW) d'escriptori desenvolupada íntegrament en Java amb JavaFX. L'aplicació permet als usuaris crear composicions musicals amb múltiples pistes MIDI, editar notes, aplicar efectes de so i reproduir les seves creacions en temps real gràcies al sintetitzador MIDI integrat en la plataforma Java. A diferència de les solucions comercials existents, REVRB és completament gratuïta i incorpora funcionalitats socials com una biblioteca comunitària on els usuaris poden compartir, descarregar i valorar composicions d'altres creadors.

Com a element diferenciador, el projecte integra un assistent d'intel·ligència artificial basat en l'API de GPT-4o d'OpenAI que permet interactuar amb el DAW mitjançant el llenguatge natural, tant per resoldre dubtes de producció musical com per generar estructures musicals completes de forma automàtica.

El desenvolupament s'ha gestionat amb metodologia àgil Kanban mitjançant Trello, control de versions amb GitHub i persistència híbrida de dades amb fitxers locals en format propi .revrb (fitxers JSON) i base de dades MySQL remota allotjada al núvol (Aiven).

Paraules clau:

- DAW
- MIDI
- JavaFX
- Java Sound API
- Producció musical
- Intel·ligència artificial
- Base de dades relacional
- Biblioteca comunitària
- MVC

Abstract

REVRB is a desktop Digital Audio Workstation (DAW) developed entirely in Java using JavaFX. The application enables users to create musical compositions featuring multiple MIDI tracks, edit notes, apply sound effects, and play back their creations in real time using the built-in MIDI synthesizer from the Java platform. Unlike existing commercial solutions, REVRB is completely free and incorporates social features, such as a community library where users can share, download, and rate compositions from other creators.

As a key differentiating factor, the project integrates an artificial intelligence assistant powered by OpenAI's GPT-4o API. This allows users to interact with the DAW using natural language, both to resolve music production queries and to automatically generate complete musical structures.

The development process was managed using the Agile Kanban methodology via Trello, version control with GitHub, and a hybrid data persistence system utilizing custom local files in .revrb format (JSON files) alongside a remote MySQL database hosted in the cloud (Aiven).

Keywords:

- DAW
- MIDI
- JavaFX
- Java Sound API
- Music production
- Artificial Intelligence
- Relational database
- Community library
- MVC

Índex

BLOC 1 — PRESENTACIÓ DEL PROJECTE

1.1 Introducció

La **producció musical digital** ha experimentat una transformació radical en les últimes dècades. Gràcies al desenvolupament de programari especialitzat, avui en dia qualsevol persona amb un ordinador pot crear, editar i compartir música sense necessitat d'un estudi professional ni d'instruments físics de cost elevat. Les eines que fan possible aquest procés s'anomenen **DAW** (Digital Audio Workstation) i representen l'entorn de treball principal de productors, compositors i músics arreu del món.

Un **DAW** és, en essència, un programa que permet enregistrar, editar, barrejar i reproduir música de forma digital. Els més coneguts, com FL Studio, Ableton Live o Logic Pro, ofereixen entorns de treball molt complets però sovint complexos i de preu elevat, la qual cosa els fa poc accessibles per a usuaris sense experiència prèvia o amb recursos econòmics limitats.

En aquest context neix REVRB, un **DAW** d'escriptori desenvolupat íntegrament en **Java** amb **JavaFX 21**. El seu nom fa referència al terme anglès "reverb" (reverberació), un efecte d'àudio molt comú en la producció musical, i al mateix temps evoca la idea de ressò, de crear quelcom que perduri. REVRB permet crear projectes musicals amb múltiples pistes de tipus diferent (percussió, baix, sintetitzador, veu i altres), editar les notes de cada pista de forma visual, aplicar efectes de so i reproduir les composicions en temps real.

A banda de les funcionalitats pròpies d'un DAW, REVRB incorpora un sistema complet de gestió d'usuaris amb registre i inici de sessió, persistència dels projectes en una **base de dades relacional** remota allotjada al núvol, i una **biblioteca comunitària** on els usuaris poden compartir i descobrir composicions d'altres creadors. Com a element diferenciador, el projecte integra un **assistent d'intel·ligència artificial** capaç de generar estructures musicals completes a partir d'una petició en llenguatge natural.

El projecte s'emmarca dins el Cicle Formatiu de Grau Superior en Desenvolupament d'Aplicacions Multiplataforma i representa la integració pràctica de les competències adquirides al llarg del cicle: disseny d'interfícies gràfiques, accés a bases de dades, programació orientada a objectes, integració amb serveis externs i programació d'àudio.

Figura 1.1: Vista general de la interfície principal del DAW REVRB.

1.2 Context

El mercat del programari de **producció musical** és ampli i consolidat. Eines com FL Studio, Ableton Live, Logic Pro o GarageBand s'han convertit en referents del sector i han democratitzat la creació musical a escala global. Gràcies a elles, músics amateurs i professionals poden treballar des de casa amb eines de qualitat professional.

No obstant això, la majoria d'aquestes solucions presenten limitacions rellevants per a determinats perfils d'usuari. En primer lloc, les eines professionals com FL Studio o Ableton Live compten amb llicències de cost elevat, sovint superiors als 200-400 euros, la qual cosa les fa poc accessibles per a estudiants, músics aficionats o persones que volen aprendre producció musical sense una inversió econòmica significativa. En segon lloc, les alternatives de codi obert existents, com LMMS o Ardour, tot i ser funcionals i gratuïtes, presenten corbes d'aprenentatge elevades i interfícies menys intuïtives que poden desanimar els usuaris sense experiència prèvia.

A més, cap d'aquestes solucions no incorpora de forma nativa funcionalitats socials com una **biblioteca comunitària** integrada a l'aplicació, ni integració directa amb assistents d'**intel·ligència artificial** generativa per facilitar el procés de composició.

Des del punt de vista tecnològic, l'ecosistema **Java** ofereix eines madures i ben documentades per al desenvolupament d'aplicacions d'escriptori multiplataforma. JavaFX permet construir interfícies gràfiques riques i reactives, mentre que Java Sound API proporciona accés natiu al sintetitzador MIDI del sistema operatiu sense necessitat de

dependències externes. Això converteix Java en una plataforma interessant per al desenvolupament d'eines de producció musical de nivell mitjà.

En els últims anys, la irrupció de la **intel·ligència artificial generativa** ha obert noves possibilitats en tots els àmbits creatius, inclosa la música. Serveis com ChatGPT o Gemini demostren que és possible generar contingut creatiu a partir de simples peticions en llenguatge natural. Integrar aquesta capacitat dins d'un DAW representa una oportunitat única per acostar la producció musical a persones sense coneixements tècnics previs.

1.3 Justificació

El desenvolupament de REVRB respon a diverses necessitats i motivacions que justifiquen la realització d'aquest projecte.

En primer lloc, existeix una necessitat real d'eines de **producció musical** gratuïtes, lleugeres i fàcils d'utilitzar. Una aplicació d'escriptori desenvolupada en **Java** pot executar-se en qualsevol sistema operatiu amb JVM instal·lada (Windows, macOS i Linux), la qual cosa la converteix en una solució autènticament **multiplataforma** sense cost addicional per a l'usuari final.

En segon lloc, la dimensió social que aporta la **biblioteca comunitària** transforma REVRB d'una eina de producció individual en una plataforma col·laborativa. Aquesta funcionalitat permet als usuaris compartir les seves composicions, descarregar projectes d'altres creadors i valorar-los amb un sistema d'estrelles, afegint un valor diferencial significatiu respecte a les solucions existents al mercat.

En tercer lloc, la integració d'un assistent d'**intel·ligència artificial** representa una aposta clara per la innovació, explorant com la IA generativa pot facilitar el procés de composició musical. L'assistent no tan sols respon preguntes sobre producció musical, sinó que és capaç de generar projectes musicals complets (pistes, notes, instruments i BPM) a partir d'una petició de l'usuari, que s'injecten directament al **DAW**.

Finalment, des del punt de vista formatiu, el projecte permet aplicar de forma integrada les competències clau del Cicle Formatiu: **disseny** i implementació d'interfícies gràfiques amb **JavaFX**, arquitectura **MVC**, accés a bases de dades relacionals amb **JDBC**, persistència de dades en formats propis basats en **JSON**, integració amb serveis al núvol i programació d'àudio amb **Java Sound API**.

1.4 Objectius

Objectiu general

L'objectiu principal d'aquest projecte és desenvolupar una aplicació d'escriptori de **producció musical** completa i funcional que permeti als usuaris crear, editar i compartir composicions musicals **MIDI** de forma intuïtiva, integrant funcionalitats socials i d'**intel·ligència artificial** com a elements diferenciadors.

Objectius generals

- Implementar un entorn **DAW** funcional amb suport per a múltiples pistes **MIDI** amb reproducció en temps real.
- Oferir una **interfície d'usuari** professional i usable, inspirada en DAWs comercials com FL Studio, amb una estètica Total Black i accents de color.
- Proporcionar persistència completa dels projectes tant en fitxer local en format propi (.revrb) com en **base de dades relacional** remota.
- Crear una plataforma social que permeti compartir, descobrir i valorar composicions d'altres usuaris a través d'una **biblioteca comunitària**.
- Integrar un assistent d'**intel·ligència artificial** que faciliti la composició i la interacció amb el **DAW** mitjançant el llenguatge natural.

Objectius específics

- Implementar un sistema de pistes amb suport per als tipus AUDIO, MIDI, DRUMS, BASS, SYNTH, VOCALS i FX, cadascuna amb el seu propi canal **MIDI** independent.
- Desenvolupar un **Piano roll** interactiu en finestra independent per a l'edició de notes **MIDI** amb visualització de teclat de piano i graella de compassos.
- Integrar el sintetitzador **Java Sound API** (Gervill) amb suport per als 128 instruments General MIDI seleccionables per pista.
- Implementar la reproducció sincronitzada de múltiples pistes respectant el **BPM** configurat, amb control de mute i solo per pista.
- Crear un sistema de registre i autenticació d'usuaris connectat a una base de dades **MySQL** remota allotjada a **Aiven Cloud**.
- Dissenyar i implementar una **biblioteca comunitària** amb filtres per data de publicació, nombre de descàrregues i valoració mitjana.
- Implementar un sistema de recuperació de contrasenya per correu electrònic amb token d'un sol ús i expiració de 30 minuts.
- Integrar l'API de GPT-4o per permetre la generació automàtica d'estructures musicals completes i la interacció conversacional amb el **DAW**.

- Garantir la persistència de les notes **MIDI**, els instruments seleccionats i la configuració de cada pista en el format .revrb basat en **JSON**.
- Assegurar la compatibilitat multiplataforma de l'aplicació mitjançant **Java 21** i **JavaFX 21**.

BLOC 2 — ESTRATÈGIA I PLANIFICACIÓ

2.1 Estratègia de desenvolupament i viabilitat

Decisió estratègica

Des de l'inici del projecte es va optar per desenvolupar l'aplicació des de zero, sense l'ús de plantilles ni frameworks de tercers per a la lògica principal. Aquesta decisió proporciona un control total sobre la implementació de totes les funcionalitats, especialment pel que fa al motor d'àudio **MIDI**, el sistema de clips de la playlist i el piano roll interactiu.

Per a la interfície gràfica es va triar **JavaFX 21** per la seva maduresa com a framework d'escriptori multiplataforma per a Java, la seva integració nativa amb el JDK i la seva capacitat per construir interfícies complexes i estilitzades amb **FXML** i **CSS**. Per a la base de dades es va optar per **MySQL** remot allotjat a **Aiven Cloud**, evitant la dependència d'un servidor local i permetent que l'aplicació funcioni des de qualsevol connexió a Internet sense configuració addicional per part de l'usuari.

L'arquitectura segueix el patró **MVC** (Model-Vista-Controlador), separant clarament la lògica de negoci dels components visuals i del motor d'àudio. Aquesta separació facilita el manteniment del codi, permet incorporar noves funcionalitats de forma ordenada i fa que el projecte sigui comprensible per a tercers.

Estudi de viabilitat

Des del punt de vista tècnic, el projecte és plenament viable gràcies a la maduresa de les tecnologies escollides. Java 21 és una versió LTS (Long Term Support) amb suport garantit fins al 2029. JavaFX 21 és estable i disposa d'una documentació àmplia. MySQL és un dels sistemes de bases de dades relacionals més estesos a nivell mundial, i l'API de GPT-4o-mini d'OpenAI ofereix un rendiment excel·lent per al volum de peticions d'un projecte d'aquestes característiques.

Des del punt de vista econòmic, el projecte utilitza exclusivament tecnologies gratuïtes o de codi obert durant la fase de desenvolupament. OpenJDK i OpenJFX són gratuïts. MySQL funciona en la capa gratuïta d'Aiven Cloud per a projectes de mida reduïda. La clau d'API

d'OpenAI comporta un cost per ús, però molt reduït per al nombre de peticions d'un projecte acadèmic.

Des del punt de vista temporal, el projecte s'ha desenvolupat en un únic curs escolar per un únic programador, la qual cosa ha requerit una planificació rigorosa i una priorització acurada de les funcionalitats per garantir un producte funcional i complet dins del termini disponible.

2.2 Metodologia de treball

Per gestionar el desenvolupament del projecte s'ha optat per una metodologia àgil adaptada al treball individual, combinant eines digitals per a l'organització de tasques i el control de versions del codi.

Kanban amb Trello

Per a la gestió de tasques s'ha utilitzat **Trello** seguint la metodologia **Kanba**n. El tauler s'ha organitzat en quatre columnes que representen l'estat de cada tasca: Per fer, En curs, En revisió i Completat. Cada targeta representa una funcionalitat o millora concreta del projecte, amb una descripció detallada i una estimació de temps associada. Aquesta metodologia permet tenir en tot moment una visió clara del progrés global del projecte i dels punts que requereixen atenció immediata.

El principal avantatge del **Kanban** per a un projecte individual és la seva senzillesa. No cal definir sprints fixos ni fer reunions de seguiment. Simplement es van movent les targetes d'una columna a l'altra a mesura que s'avança, la qual cosa dona una sensació de progrés constant i ajuda a identificar ràpidament quines funcionalitats estan bloquejades o requereixen revisió.

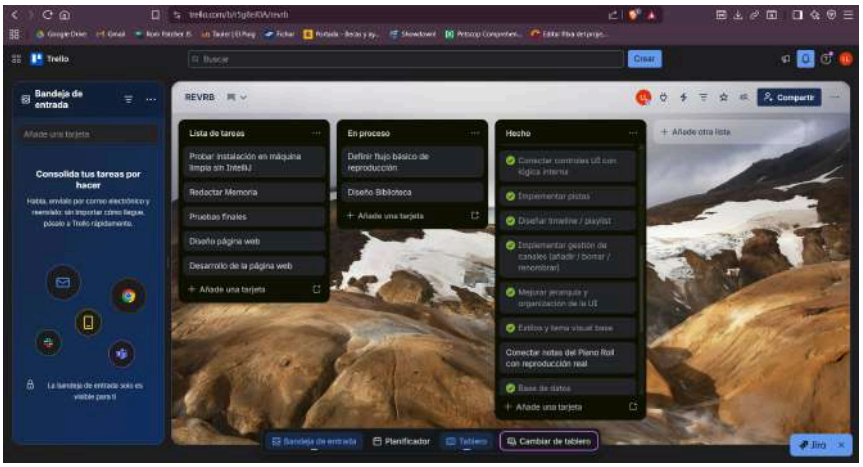


Figura 2.1: Tauler Kanban utilitzat a Trello per a la gestió de tasques del projecte.

GitHub

Per al control de versions s'ha utilitzat GitHub, mantenint un repositori on es registren tots els canvis realitzats en el codi font. Cada commit inclou un missatge descriptiu que indica les funcionalitats implementades o els errors corregits, facilitant el seguiment cronològic del procés de desenvolupament i permetent revertir canvis en cas d'errors greus o decisions de disseny incorrectes.

El control de versions ha estat especialment útil en moments on una nova funcionalitat trencava el comportament de parts ja implementades. Disposar d'un historial de commits ha permès identificar ràpidament en quin punt s'havia introduït el problema i revertir els canvis necessaris sense perdre el treball fet.

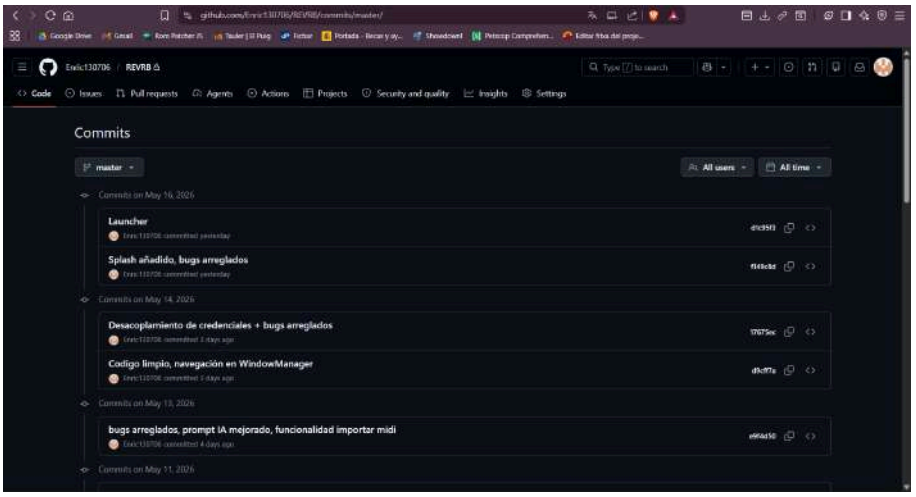


Figura 2.2: Repositori del projecte REVRB allotjat a GitHub.

IntelliJ IDEA

Com a entorn de desenvolupament integrat s'ha utilitzat IntelliJ IDEA Community Edition, que ofereix suport natiu per a **Java 21**, **Maven**, **JavaFX** i eines avançades de depuració i refactorització. L'autocompletat de codi, la navegació entre classes i la integració amb el sistema de control de versions han fet del procés de desenvolupament una experiència molt més fluïda.

JavaFX Scene Builder

Com a eina de disseny visual s'ha utilitzat JavaFX Scene Builder, un programari especialitzat que permet construir les interfícies gràfiques de l'aplicació mitjançant el mètode d'arrossegar i deixar anar (*drag-and-drop*) components visuals. Aquesta eina genera de forma automàtica el codi **FXML** que defineix l'estructura de les pantalles, el qual s'ha vinculat directament amb les fulles d'estil **CSS** per aplicar l'estètica *Total Black* del projecte i amb les classes controladores d'**IntelliJ IDEA** per gestionar la lògica dels esdeveniments.

MySQL Workbench

Per a la gestió, disseny i consulta de la base de dades remota s'ha utilitzat MySQL Workbench, que permet executar consultes SQL directament sobre la base de dades allotjada a **Aiven Cloud** i visualitzar l'estructura de taules i relacions de forma gràfica.

Maven

Com a gestor de dependències i construcció del projecte s'ha utilitzat Maven, que gestiona automàticament les dependències externes del projecte com el connector **JDBC** de **MySQL**, la llibreria **GSON** per al processament de **JSON**, i la dependència de **Jakarta Mail** per a l'enviament de correus electrònics.

2.3 Estudi econòmic i pressupostari

El desenvolupament d'una aplicació d'escriptori de la magnitud de REVRB implica una inversió considerable de temps, recursos tècnics i infraestructures que és necessari quantificar. En el context d'aquest projecte acadèmic, **s'ha establert com a prioritat absoluta evitar qualsevol despesa econòmica innecessària**. Per aquest motiu, l'estratègia de desenvolupament s'ha basat exclusivament en l'ús de **tecnologies de codi obert** (*open-source*), **eines gratuïtes** i serveis al núvol que ofereixen **capes gratuïtes** (*free tiers*) per a desenvolupadors.

A continuació, es detalla la viabilitat econòmica del projecte dividida en dues grans àrees: el cost real assumit durant el desenvolupament acadèmic i una estimació pressupostària en cas de traslladar el projecte a un entorn de mercat professional.

En l'àmbit real del projecte, l'únic cost directe associat ha estat l'adquisició del maquinari informàtic utilitzat com a estació de treball i un consum residual de l'API de serveis externs. El desenvolupament s'ha centralitzat en un ordinador portàtil **Acer Aspire 5**, adquirit expressament amb unes especificacions potents (**32 GB de memòria RAM** i **1 TB d'emmagatzematge SSD**) per garantir la fluïdesa necessària en l'execució simultània de l'IDE, l'entorn de construcció, el motor d'àudio de Java i el servidor local de proves.

La següent taula detalla la inversió real en conceptes de **maquinari**:

Component	Descripció	Cost real
Ordinador de desenvolupament	Portàtil Acer Aspire 5 (32 GB RAM, 1 TB SSD, Windows)	800,00 €
Monitor secundari	Pantalla addicional per a la depuració i disseny d'interfícies	150,00 €
Perifèrics	Teclat i ratolí estàndard	60,00 €
TOTAL MAQUINARI		**1.010,00 €**

Pel que fa al **programari** i les llicències, el cost ha estat pràcticament **nul**. S'ha fet un estudi previ de les alternatives de mercat per escollir llicències gratuïtes o lliures per a tots els àmbits (entorn de desenvolupament, gestió de base de dades, llibreries de codi i organització). L'única despesa de serveis ha estat el model de pagament per ús (*pay-as-you-go*) de la IA d'OpenAI per a les crides de text a l'assistent durant les proves de laboratori.

Eina / Servei	Tipus de llicència / Capa	Cost real
Java 21 (OpenJDK)	Codi obert (Gratuït)	0,00 €
JavaFX 21 (OpenJFX)	Codi obert (Gratuït)	0,00 €
IntelliJ IDEA Community Edition	Llicència gratuïta de desenvolupament	0,00 €
MySQL (Aiven Cloud)	Capa gratuïta (*Free Tier*) per a bases de dades	0,00 €
GitHub	Repositori privat de codi obert (Gratuït)	0,00 €
Trello	Gestió de projectes Kanban (Gratuït)	0,00 €
API GPT-4o-mini (OpenAI)	Pagament per ús (Crides de prova consumides)	< 5,00 €
TOTAL PROGRAMARI I SERVEIS		**< 5,00 €**

El veritable actiu d'aquest projecte ha estat la inversió en **capital humà**, corresponent a les hores de dedicació personal del propi programador per al disseny de l'arquitectura, la programació del motor d'àudio i la redacció de la documentació tècnica. Com a projecte d'autoria pròpia, aquest temps es tradueix en un cost financer de 0,00 €, fent que la viabilitat del projecte hagi estat absoluta.

Per tal d'analitzar el valor real de l'aplicació REVRB en el mercat de la producció de programari, és imprescindible realitzar una simulació de costos en un context professional. En aquest escenari, es consideraria la contractació d'un equip tècnic multidisciplinari i el lloguer o quota comercial de serveis al núvol escalats per a una base d'usuaris massiva.

A continuació, es pressuposta la repercussió econòmica de la força de treball calculada sobre una dedicació aproximada del projecte (estimada en unes 400 hores de desenvolupament i tasques transversals de disseny i QA):

Rol professional	Tarifa horària estimada	Dedicació (Hores)	Cost estimat
Desenvolupador Java / JavaFX sènior	20,00 €/h	400 h	8.000,00 €
Dissenyador d'interfície (UI/UX)	25,00 €/h	80 h	2.000,00 €
Administrador de bases de dades (DBA)	22,00 €/h	40 h	880,00 €
Enginyer de Proves / QA Tester	18,00 €/h	30 h	540,00 €
TOTAL DESENVOLUPAMENT PROFESSIONAL		**550 h**	**11.420,00 €**

Finalment, en consolidar tots els conceptes (recollint el maquinari inicial, el manteniment professional i el programari), el balanç econòmic total que hauria d'assumir una empresa per llançar un producte amb les característiques del DAW REVRB seria el següent:

Concepte pressupostari	Naturalesa del cost	Cost total estimat
Maquinari	Inversió inicial en actius fixos	1.010,00 €
Programari i Serveis	Despeses operatives de llicències i APIs	5,00 €
Desenvolupador i Equip Tècnic	Cost de personal i sous professionals	11.420,00 €
COST TOTAL ESTIMAT DEL PROJECTE		**12.435,00 €**

2.4 Planificació

La planificació del projecte REVRB s'ha dissenyat seguint un calendari acadèmic que s'estén des del mes de **setembre** fins al mes de **maig**. El cronograma s'ha dividit en sis blocs principals, cadascun amb tasques específiques per garantir un desenvolupament progressiu i ordenat.

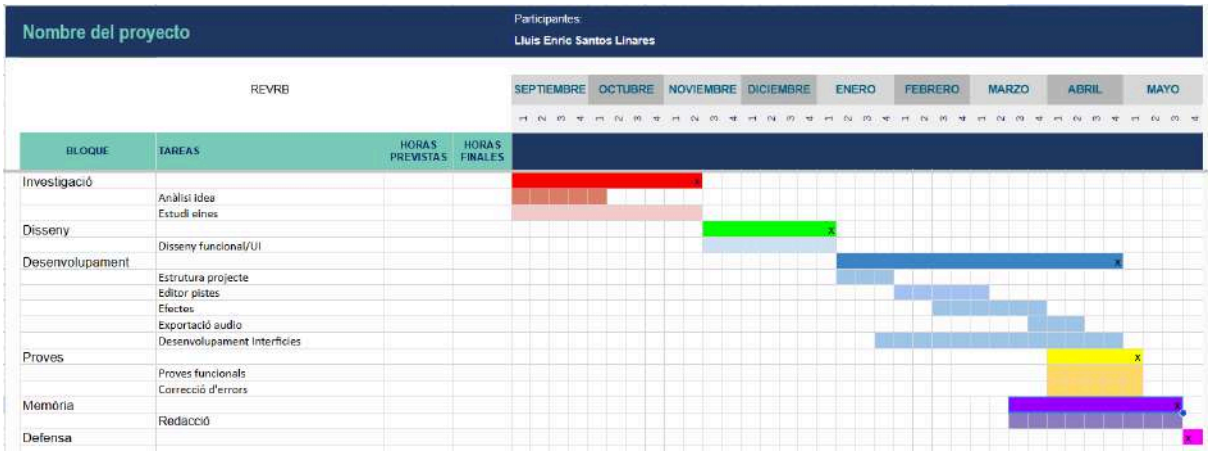


Figura 2.3: Cronograma de desenvolupament del projecte REVRB representat en un diagrama de Gantt.

Fase 1 — Investigació (Setembre - Octubre)

Aquesta fase inicial ha estat clau per definir el rumb del projecte. Durant el primer mes, s'ha realitzat l'**anàlisi de la idea**, delimitant les funcionalitats del **DAW** i la seva viabilitat. Posteriorment, s'ha dut a terme un **estudi d'eines**, on es va decidir l'ús de **Java 21**, **JavaFX** i **MySQL** com a pilars tecnològics, avaluant la seva integració i rendiment.

Fase 2 — Disseny (Octubre)

Dedicada íntegrament a la definició visual i funcional. S'ha treballat en el **disseny funcional** i de la **interfície d'usuari (UI)**, creant els primers esbossos de l'estètica. L'objectiu ha estat traslladar la complexitat d'un programari d'àudio a un entorn intuïtiu, definint el comportament de les finestres i els controls principals.

Fase 3 — Desenvolupament (Novembre - Març)

Es tracta del bloc més extens i crític del projecte, on s'ha realitzat la implementació del codi font dividida en cinc etapes:

- **Estructura del projecte:** Configuració de l'arquitectura **MVC** i el sistema de construcció **Maven**.
- **Editor de pistes:** Programació de la lògica de les pistes, el motor d'àudio i el **Piano Roll**.
- **Playlist:** Creació de la graella principal on s'ubiquen els clips musicals i la gestió del temps.
- **Integració IA i BD:** Implementació de la connexió amb l'API de **GPT-4o** d'OpenAI i la base de dades remota a **Aiven Cloud**.
- **Desenvolupament d'interfícies:** Poliment final de tots els components visuals i navegació de l'aplicació.

Fase 4 — Proves (Març - Abril)

Un cop finalitzat el desenvolupament *core*, s'ha iniciat el període de **proves funcionals** per assegurar que tots els mòduls funcionen correctament de forma integrada. Durant el mes d'abril, s'ha prioritzat la **correcció d'errors** (*debugging*), optimitzant el rendiment del motor d'àudio i l'estabilitat de les connexions amb el núvol.

Fase 5 — Memòria (Abril - Maig)

Paral·lelament a les proves finals, s'ha realitzat la **redacció de la memòria tècnica** del projecte. Aquesta fase recull tota la documentació de l'anàlisi, el disseny, el desenvolupament i l'estudi econòmic, reflectint el procés seguit i les competències adquirides durant el cicle.

Fase 6 — Defensa (Maig)

El projecte finalitza amb la **presentació i defensa** davant del tribunal, on es realitza una demostració en viu del funcionament del **DAW REVRB** i s'exposen les conclusions extretes del desenvolupament.

2.5 Eines utilitzades

Java 21

Java és el llenguatge de programació principal del projecte. S'ha optat per la versió 21 LTS (Long Term Support) per la seva estabilitat i el seu suport garantit fins al 2029. Java ofereix portabilitat nativa entre sistemes operatius gràcies a la JVM (Java Virtual Machine), la qual cosa permet executar REVRB en Windows, macOS i Linux sense canvis en el codi.

JavaFX 21

JavaFX és el framework triat per a la interfície gràfica. Permet definir les pantalles en fitxers FXML (un format basat en XML específic per a JavaFX) i aplicar estils visuales amb CSS, de manera similar a com es fa en el desenvolupament web. Gràcies a JavaFX s'han pogut crear interfícies complexes amb animacions, sliders, scrolls sincronitzats i elements personalitzats, tot mantenint el codi de la lògica separat de la presentació visual.

Maven

Maven és l'eina de gestió de dependències i construcció del projecte. Gestiona automàticament la descàrrega i actualització de les llibreries externes necessàries per al projecte (connectors de base de dades, llibreries JSON, correus electrònics) i proporciona un sistema estàndard de construcció de projectes Java.

Java Sound API

Java Sound API és la biblioteca de la plataforma Java que proporciona accés al sistema d'àudio i MIDI del sistema operatiu. REVRB l'utilitza per connectar-se al sintetitzador Gervill, inclòs en el JDK, que permet reproduir els 128 instruments del General MIDI Standard sense necessitat de cap dependència externa.

MySQL (Aiven Cloud)

La base de dades del projecte és MySQL, allotjada a Aiven Cloud. Aiven és una plataforma de gestió de bases de dades al núvol que ofereix una capa gratuïta adequada per a projectes de mida reduïda. La connexió entre l'aplicació i la base de dades es realitza mitjançant JDBC (Java Database Connectivity) amb SSL activat per garantir la seguretat de les comunicacions.

OpenAI API (GPT-4o-mini)

La integració amb la intel·ligència artificial es realitza a través de l'API d'OpenAI, concretament el model GPT-4o-mini. Les peticions es fan mitjançant el client HTTP natiu de Java (java.net.http), disponible des de Java 11, sense necessitat de cap SDK addicional.

Jakarta Mail

Per a l'enviament de correus electrònics (recuperació de contrasenya) s'utilitza la llibreria Jakarta Mail 2.0.1, que permet enviar correus a través del servidor SMTP de Gmail.

Google Gson i org.json

Aquestes dues llibreries permeten serialitzar i deserialitzar objectes Java en format JSON. S'utilitzen tant per a la persistència local dels projectes (format .revrb) com per al processament de les respostes de l'API d'OpenAI.

IntelliJ IDEA

L'entorn de desenvolupament integrat (IDE) triat ha estat IntelliJ IDEA Community Edition, que ofereix eines avançades de depuració, refactorització i navegació de codi per a projectes Java.

MySQL Workbench

Per a la gestió visual de la base de dades remota s'ha utilitzat MySQL Workbench, que permet dissenyar l'esquema de taules, executar consultes SQL i monitoritzar l'estat de la base de dades.

GitHub

El control de versions del projecte s'ha gestionat amb GitHub. El repositori privat conté tot el codi font del projecte, l'historial complet de canvis i les notes associades a cada versió.

Trello

La planificació i el seguiment de tasques s'han gestionat amb Trello, seguint la metodologia Kanban.

BLOC 3 — ANÀLISI

3.1 Casos d'ús

Els principals **casos d'ús** del sistema s'organitzen al voltant de tres actors: l'**Usuari no autenticat**, l'**Usuari autenticat** i el **Sistema** (que actua de forma autònoma en determinades operacions).

L'**Usuari no autenticat** pot registrar-se a l'aplicació introduint un nom d'usuari, una contrasenya i un correu electrònic. Un cop registrat, pot iniciar sessió amb les seves credencials. Si oblida la contrasenya, pot sol·licitar un codi de recuperació que li arribarà per correu electrònic i que li permetrà establir una nova contrasenya.

Un cop autenticat, l'**Usuari autenticat** pot gestionar els seus projectes musicals: crear-ne de nous, obrir-ne d'existents, desar-los i eliminar-los. Dins del **DAW**, pot afegir i eliminar pistes, crear clips a la *playlist*, editar les notes de cada pista al **Piano Roll**, seleccionar l'instrument de cada pista entre els 128 disponibles del **General MIDI**, ajustar el volum, silenciar (*mute*) o soloitzar (*solo*) pistes i reproduir la composició en temps real.

L'usuari autenticat també pot publicar els seus projectes a la **biblioteca comunitària**, definint un títol públic i una descripció. Des de la biblioteca, pot explorar les composicions d'altres usuaris, filtrar-les per data de publicació, nombre de descàrregues o valoració mitjana, descarregar-les i obrir-les directament al DAW, i puntuar-les amb una valoració d'1 a 5 estrelles.

Finalment, l'usuari pot interactuar amb l'assistent d'**intel·ligència artificial**, tant per fer preguntes sobre producció musical com per demanar-li que generi una estructura musical completa que s'injectarà automàticament al DAW com a nou projecte.

El **Sistema** actua de forma autònoma en operacions com el desat automàtic del projecte a la base de dades quan l'usuari el desa localment, l'expiració automàtica dels *tokens* de recuperació de contrasenya (caducitat de 30 minuts), i la detecció automàtica del tipus de resposta de la **IA** (si és una resposta en format **JSON** de projecte o un missatge de text conversacional).

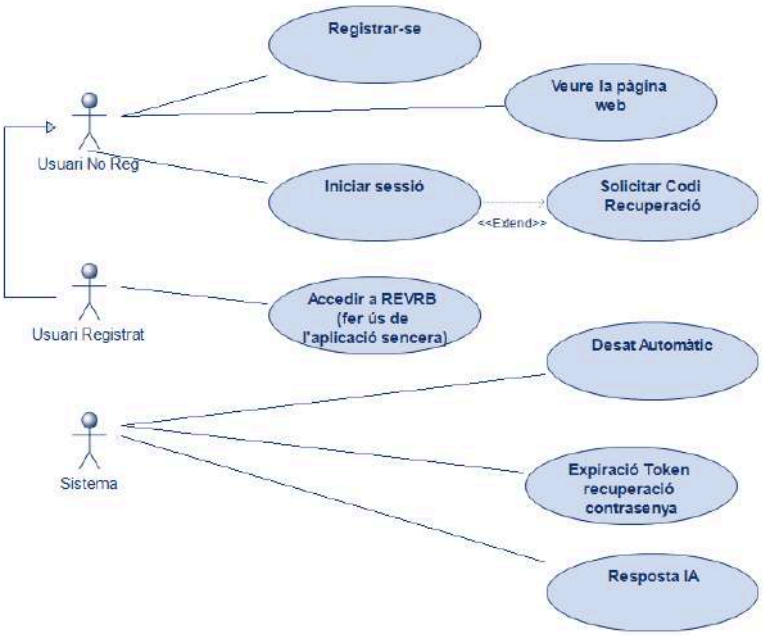


Figura 3.1: Diagrama de casos d'ús del sistema REVRB.

3.2 Requisits funcionals

3.2.1 Gestió d'usuaris

- **RF-1.1 Registre d'usuaris:** El sistema ha de permetre el registre de nous usuaris amb nom d'usuari únic, contrasenya i correu electrònic. Ha de validar les dades introduïdes i informar l'usuari en cas d'error.
- **RF-1.2 Autenticació:** L'usuari ha de poder iniciar sessió amb les seves credencials, les quals es validen contra la base de dades remota. El sistema ha de mantenir la sessió de l'usuari activa durant tota la navegació.
- **RF-1.3 Recuperació de credencials:** El sistema ha de permetre la recuperació de contrasenya mitjançant un codi de verificació d'un sol ús enviat per correu electrònic, amb una validesa màxima de 30 minuts.

3.2.2 Gestió de projectes

- **RF-2.1 Creació de projectes:** L'usuari ha de poder crear projectes musicals nous definint el nom, el **BPM** (*beats per minute*) i el tipus de *snap* (ajust) de la graella.
- **RF-2.2 Tipología de pistes:** Ha de poder afegir i eliminar pistes de qualsevol dels tipus disponibles: AUDIO, MIDI, DRUMS, BASS, SYNTH, VOCALS i FX.
- **RF-2.3 Manipulació de clips:** L'usuari ha de poder crear, moure, redimensionar i eliminar clips a la *playlist* de cada pista.

- **RF-2.4 Persistència híbrida:** El sistema ha de desar i carregar projectes tant en format local .revrb com a la base de dades remota de forma automàtica.

3.2.3 Edició musical

- **RF-3.1 Piano Roll interactiu:** L'usuari ha de poder obrir un **Piano Roll** independent per a cada pista, on es mostren les notes **MIDI** en una graella i es poden afegir, eliminar i redimensionar.
- **RF-3.2 Previsualització visual:** El sistema ha de mostrar un resum visual de les notes directament sobre els clips de la *playlist*.
- **RF-3.3 Instrumentació MIDI:** L'usuari ha de poder seleccionar l'instrument de cada pista entre els 128 instruments del **General MIDI Standard**.
- **RF-3.4 Mesclador i transport:** El sistema ha de reproduir totes les pistes de forma sincronitzada respectant el BPM, amb controls de *mute*, *solo* i volum per pista.

3.2.4 Biblioteca comunitària

- **RF-4.1 Publicación de proyectos:** L'usuari ha de poder publicar els seus projectes a la biblioteca comunitària per fer-los públics.
- **RF-4.2 Búsqueda y filtrado:** Ha de poder filtrar els projectes de la comunitat per data, descàrregues i valoració.
- **RF-4.3 Descarga de proyectos:** Ha de poder descarregar projectes d'altres usuaris i obrir-los directament al DAW.
- **RF-4.4 Sistema de valoración:** Ha de poder valorar qualsevol projecte amb una puntuació d'1 a 5 estrelles, restringit a una única vegada per usuari i projecte.

3.2.5 Assistent d'intel·ligència artificial

- **RF-5.1 Interacción conversacional:** L'usuari ha de poder conversar amb l'assistent en llenguatge natural sobre producció musical.
- **RF-5.2 Generación automatizada:** L'assistent ha de poder generar estructures musicals completes i injectar-les automàticament al DAW en format actiu.
- **RF-5.3 Modificación por comandos:** L'assistent ha de poder afegir pistes i modificar el BPM del projecte a través d'instruccions de l'usuari en el xat.

3.3 Requisits no funcionals

- **RNF-1 Rendiment (Concurrència i Asincronia):** La interfície d'usuari (**UI**) no ha de bloquejar-se en cap moment durant les peticions a la IA ni a la base de dades. Totes les operacions d'entrada/sortida (connexions de xarxa, lectura i escriptura de fitxers)

s'han d'executar en **fils secundaris** (*threads*) de forma asíncrona per mantenir l'aplicació reactiva.

- **RNF-2 Usabilitat:** La interfície ha de ser intuïtiva i consistent, seguint les convencions visuals dels DAWs comercials de referència. Els elements interactius han de donar retroalimentació visual immediata a l'usuari (canvis de color, animacions, indicadors d'estat).
- **RNF-3 Compatibilitat:** L'aplicació ha de ser **multiplataforma** i funcionar de manera nativa en Windows, macOS i Linux mitjançant la màquina virtual de **Java 21** i **JavaFX 21**, sense requerir dependències del sistema operatiu hoste.
- **RNF-4 Seguretat:** Les contrasenyes s'han d'encriptar abans de ser emmagatzemades i no han de quedar exposades en els registres de l'aplicació. La connexió a la base de dades s'ha de realitzar mitjançant el protocol **SSL** activat. Els *tokens* de recuperació han de ser d'un sol ús i amb expiració fixa de 30 minuts.
- **RNF-5 Mantenibilitat:** El codi font s'ha d'organitzar estrictament seguint el patró **MVC** (Model-Vista-Controlador), separant clarament les responsabilitats per paquets (*packages*) i encapsulant els components per facilitar futures expansions.

3.4 Anàlisi d'alternatives tecnològiques

3.4.1 Llenguatge de programació

L'alternativa principal a **Java** hauria estat Python (amb biblioteques gràfiques com Tkinter o PyQt) o bé una aplicació web distribuïda amb JavaScript/TypeScript. No obstant això, Java s'ha escollit per ser el llenguatge vehicular del cicle formatiu i per oferir una combinació òptima de portabilitat multiplataforma, rendiment d'escriptori i, especialment, per l'accés natiu al sistema d'àudio MIDI a través de **Java Sound API**, evitant la dependència de llibreries externes que poguessin comprometre l'estabilitat del sistema.

3.4.2 Framework d'interfície gràfica

Dins l'ecosistema Java, l'alternativa clàssica hauria estat **Swing**. Tot i ser funcional, Swing presenta una estètica completament obsoleta per als estàndards actuals i un model de renderitzat més rígid. Es va triar **JavaFX 21** perquè permet una separació neta entre la lògica i la presentació mitjançant fitxers **FXML** i fulles d'estil **CSS**, a més d'oferir un millor rendiment gràfic gràcies a l'acceleració per maquinari i l'ús de components moderns com el Canvas.

3.4.3 Sistema de base de dades

Les alternatives estudiades a **MySQL** van ser PostgreSQL, SQLite o solucions NoSQL com MongoDB. SQLite es va descartar perquè, en treballar en local, impedia el desplegament de la biblioteca comunitària. MongoDB hauria facilitat l'emmagatzematge dels projectes en format JSON, però hauria complicat la integritat i consistència de les relacions estructurals (usuaris, projectes, taules de valoracions i claus primàries). Finalment, la maduresa de **MySQL** i la facilitat de desplegament remot a la plataforma **Aiven Cloud** van determinar la seva elecció.

3.4.4 Motor d'àudio

L'ús de **Java Sound API** juntament amb el sintetitzador **Gervill** integrat en el JDK constituïa l'opció més robusta per evitar dependències externes del sistema operatiu. L'alternativa hauria estat la integració de llibreries de tercers com JavaZoom o FX-MIDI-SYNTH. Tot i que podien oferir algun avantatge de configuració, introduïen dependències alienes que xocaven amb l'objectiu de dissenyar un nucli de codi compacte, segur i portable.

3.4.5 Intel·ligència artificial

Com a alternativa al model **GPT-4o** d'OpenAI, es va valorar inicialment l'API de **Gemini** de Google o el desplegament local de models de codi obert com **LLaMA** via Ollama. Es va descartar l'opció local per l'alt requisit de maquinari (targetes gràfiques dedicades) que hauria d'exigir a l'usuari final per fer córrer el DAW. Entre Gemini i OpenAI, es va triar el model **GPT-4o** per la seva excel·lent relació qualitat-preu, la velocitat de resposta en fils asíncrons i la precisió estricta a l'hora de retornar estructures en format **JSON** net per ser injectades directament a l'aplicació.

BLOC 4 — DISSENY

4.1 Arquitectura del sistema

REVRB implementa el patró arquitectònic **MVC** (**Model-Vista-Controlador**), garantint una separació estricta del sistema en tres capes lògiques amb responsabilitats clares i independents per afavorir la escalabilitat i el manteniment del codi.

- **La capa Model:** Conté tota la **lògica de negoci** de l'aplicació. S'encarrega de gestionar la connexió a la base de dades remota, la serialització i deserialització de projectes, la representació en memòria de les estructures musicals (pistes, clips i notes), la integració de l'API d'OpenAI, el servei d'enviament de correus electrònics i el control de la sessió de l'usuari actiu.
- **La capa Vista:** Està constituïda pels fitxers **FXML** que defineixen l'estructura jeràrquica de nodes de cada pantalla, així com per un fitxer **CSS** centralitzat que unifica l'estètica de l'aplicació. Aquestes vistes són totalment declaratives i estan exemptes de lògica de control.
- **La capa Controlador):** Classes Java que actuen com a pont de comunicació entre el Model i la Vista. Cada pantalla disposa d'un controlador propi acoblat que captura els esdeveniments de la interfície gràfica (clics de ratolí, esdeveniments de teclat, interaccions amb *sliders*), invoca els mètodes operatius del model i actualitza de manera reactiva els components de la vista.

A banda del nucli MVC, el sistema incorpora un mòdul desacoblat i especialitzat: el **motor d'àudio**.. Aquest component encapsula de manera independent la interacció amb la **Java Sound API** i el sintetitzador **MIDI** (Gervill). Està dissenyat seguint un model de caixa negra, de manera que el controlador principal del **DAW** el pot instanciar i governar (reproduir, pausar, aturar) sense conèixer-ne la implementació interna de baix nivell.

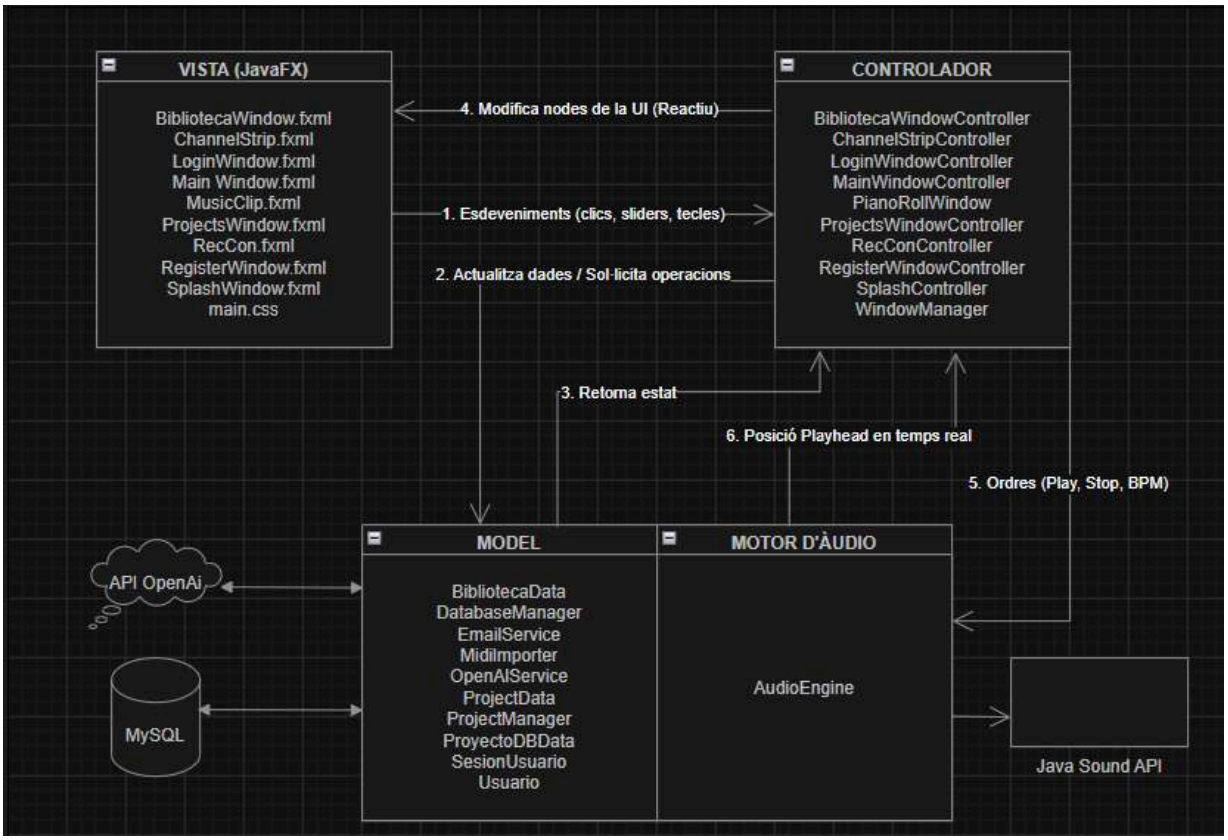


Figura 4.1: Diagrama de l'arquitectura de blocs basada en el patró Model-Vista-Controlador (MVC) i el mòdul d'àudio.

El flux d'arrencada de l'aplicació s'inicia a la classe principal, la qual inicialitza l'escenari (Stage) de **JavaFX** carregant la vista de Login. A partir d'aquest punt d'entrada, el cicle de navegació permet derivar l'usuari cap al Registre o la Recuperació de contrasenya. Un cop superada l'autenticació, es transfereix el control a la Llista de Projectes, que actua com a pas previ per obrir l'entorn de treball del **DAW** principal, des d'on finalment es pot desplegar la Biblioteca comunitària.

4.2 Model de dades

4.2.1 Base de dades relacional

L'esquema de persistència al núvol allotjat a **Aiven Cloud** es basa en un model **entitat-relació** compost per quatre taules principals altament cohesionades:

- **Taula d'usuaris (users):** Emmagatzema la informació d'identitat de cada compte. Conté la clau primària (*id*), el nom d'usuari (amb restricció d'unicitat), el correu electrònic i la contrasenya encriptada. També inclou els camps de control per al flux de recuperació: el *token* temporal d'un sol ús i la seva marca de temps d'expiració (*expires_at*).

- **Taula de projectes (projects):** Emmagatzema els projectes musicals associats a cada compte de forma indexada mitjançant una **clau estrangera** (*foreign key*) connectada a la taula d'usuaris. Inclou metadades com el títol, data de creació, recompte de pistes, estat (esborrany o publicat) i una columna de tipus LONGTEXT que emmagatzema l'estructura de dades completa en format **JSON**. Aquest disseny de persistència híbrida simplifica l'esquema de la base de dades, evitant la sobrecàrrega de taules relacionals per a cada nota o clip individual i agilitzant les operacions de lectura i escriptura.

- **Taula de biblioteca (community_library):** Registra les publicacions compartides a la plataforma col·laborativa. Vincula una clau estrangera a la taula de projectes (configurada com a ON DELETE SET NULL per no perdre l'entrada si l'usuari elimina el seu projecte personal) i una altra clau a l'usuari autor. Desa camps com el títol comunitari, descripció, data de publicació i un comptador incremental de descàrregues.

- **Taula de valoracions (ratings):** Registra les puntuacions atorgades per la comunitat. Relaciona l'entrada de la biblioteca amb l'usuari que puntua i un valor numèric discret de l'1 al 5. S'ha aplicat una **restricció d'unicitat composta** (*unique key*) sobre el parell d'atributs (id_biblioteca, id_usuari) per garantir a nivell de motor de base de dades la restricció de vot únic per usuari.

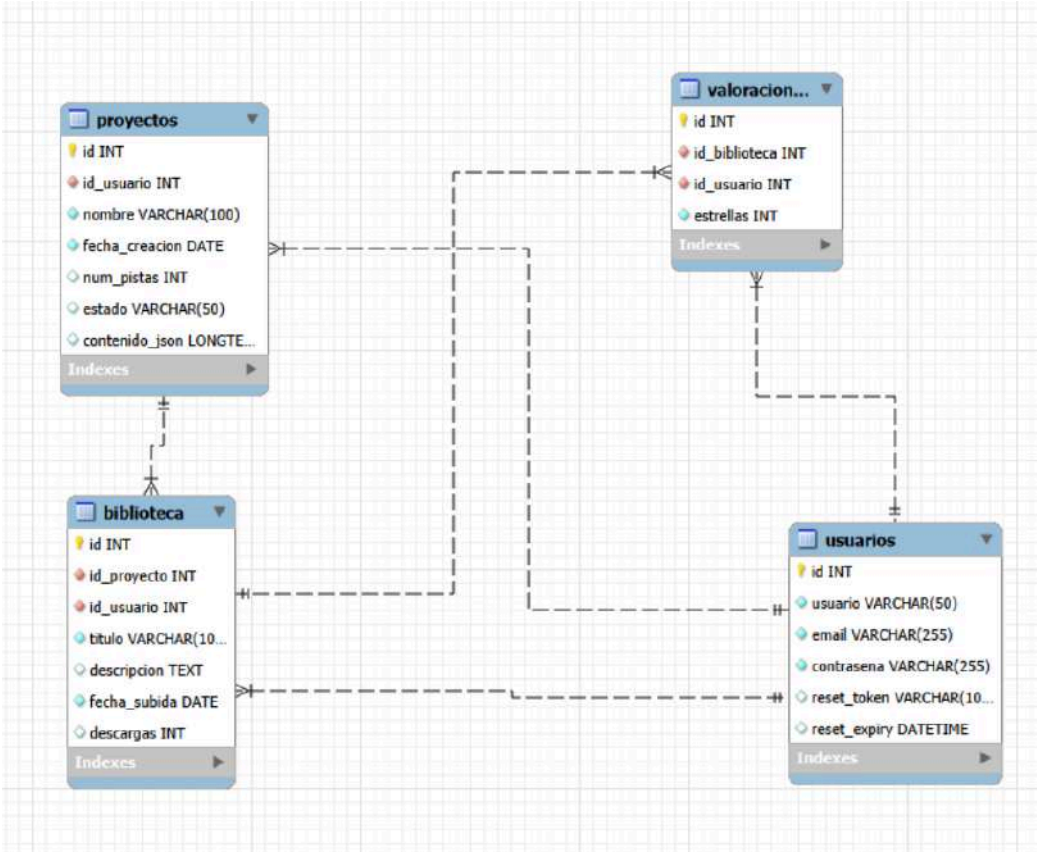


Figura 4.2: Model entitat-relació de la base de dades de REVRB allotjada a Aiven Cloud.

4.2.2 Model de projecte en format JSON (.revrb)

Per a la persistència en l'emmagatzematge local, s'ha dissenyat una extensió de fitxer pròpia anomenada .revrb. Aquest fitxer conté un objecte **JSON** estructurat que encapsula l'estat absolut de la sessió de treball del DAW per permetre la portabilitat total entre plataformes:

- **Metadades globals:** Nom del projecte, tempo configurat en **BPM** i factor d'ajust de la graella (*snap to grid*).
- **Estructura de llistat de pistes:** Un vector (*array*) on cada objecte representa una pista amb el seu nom d'usuari, tipologia (AUDIO, MIDI, DRUMS, etc.), color de representació en format hexadecimal, volum decimal, i els estats booleans de *mute* i *solo*, a més de la id de l'instrument assignat.
- **Mòdul de clips:** Posició de l'eix X a la línia de temps, amplada de l'objecte gràfic (durada) i l'enllaç de la pista pare on resideix.

- **Matriu de notes MIDI:** Cada nota es serialitza com un subobjecte indexat amb tres variables primitives: el valor de nota **MIDI** (estàndard 0-127), la columna d'inici dins de la graella de compassos i la longitud o durada d'execució mesurada en columnes de graella.

La llibreria **Google Gson** s'encarrega d'analitzar i mapar aquesta estructura d'objectes cap a text de manera immediata, garantint un format lleuger i portable.

4.3 Disseny de la interfície

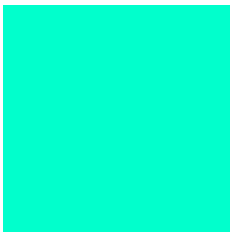
4.3.1 Estètica general

El disseny visual de REVRB s'ha concebut sota les pautes d'una estètica **Total Black**, emulant directament el patró cromàtic de les estacions de treball d'àudio digitals de referència de la indústria com FL Studio o Ableton Live. L'objectiu d'aquesta decisió és reduir la fatiga visual durant les sessions d'edició prolongades i oferir un entorn professional.

La paleta de colors es governa de manera estricta sota els següents codis cromàtics aplicats mitjançant **CSS**:

- **Fons principals i espais de treball:** Negres profunds (#0A0A0A) i grisos de baix contrast (#1E1E1E).

- **Color d'accent principal (Interactius / Selecció):** Cyan brillant (#00FFCC), utilitzat per a playheads, botons actius i línies de selecció.



- **Color d'accent secundari (Alertes / Estats):** Taronja (#FF8800), destinat a controls d'efectes de so i indicadors auxiliars.

Pel que fa a la tipografia, s'utilitza la font **Segoe UI** per a l'entorn de navegació general per la seva claredat i legibilitat en pantalles d'alta densitat. Per contra, per als indicadors mètrics de naturalesa digital (el rellotge de transport, el visualitzador de BPM i comptadors), s'ha optat per la font monoespaiada **Courier New**, reforçant l'aspecte de precisió del DAW.

4.3.2 Flux de pantalles

L'aplicació es fragmenta en un recorregut guiat d'interfícies instanciades mitjançant escenes de JavaFX:

1. **Pantalla de Login:** Finestra inicial d'accés que disposa d'un formulari d'autenticació asimètric centrat i hipervincles cap als fluxos alternatius de gestió de credencials.
2. **Pantalla de Registre:** Dissenyada amb controls de validació que canvien el color de la vora del component en temps real per oferir *feedback* neta a l'usuari abans d'enviar la sentència a la base de dades.
3. **Pantalla de Recuperació:** Interfície seqüencial dividida en dues etapes funcionals: formulari de petició (enviament del correu de verificació) i formulari de validació (verificació del *token* i reescriptura de contrasenya).
4. **Pantalla de Gestió de Projectes:** Un tauler tipus *Dashboard* que llista els projectes personals continguts a la base de dades en un component ListView personalitzat, exposant botons d'acció ràpida per a obertura, eliminació o publicació directa a la xarxa.
5. **Entorn DAW Principal:** Pantalla mestra de l'aplicació dissenyada per a una resolució maximitzada. Es divideix geomètricament en quatre blocs operatius de control:
    - **Barra de transport superior:** Conté el panell de reproducció, metrònom, rellotge digital, selector de *snap*, ajust de BPM i l'accés directe al xat asíncron de la Intel·ligència Artificial.
    - **Panell de control de canals (Esquerra):** Conté la llista vertical de pistes, on cada ítem actua com un panell independent amb barres de volum, selectors d'instruments MIDI (desplegable dels 128 elements) i botons commutadors de *Mute* i *Solo*.

- **Playlist central:** Un espai de treball visual basat en un component de tipus ScrollPane sincronitzat amb la llista de canals que permet allotjar, desplaçar i redimensionar els clips sobre una graella de composició.
- **Inspector de control (Dreta):** Panell contextual on es parametritzen els efectes digitals (Reverb, Delay, EQ) del clip actiu seleccionat.

6. **Biblioteca Comunitària:** Finestra emergent en modalitat *Dialog* o vista independent que renderitza el banc de projectes mundials en una graella interactiva, habilitant botons de descàrrega i barres interactives de valoració d'estrelles.

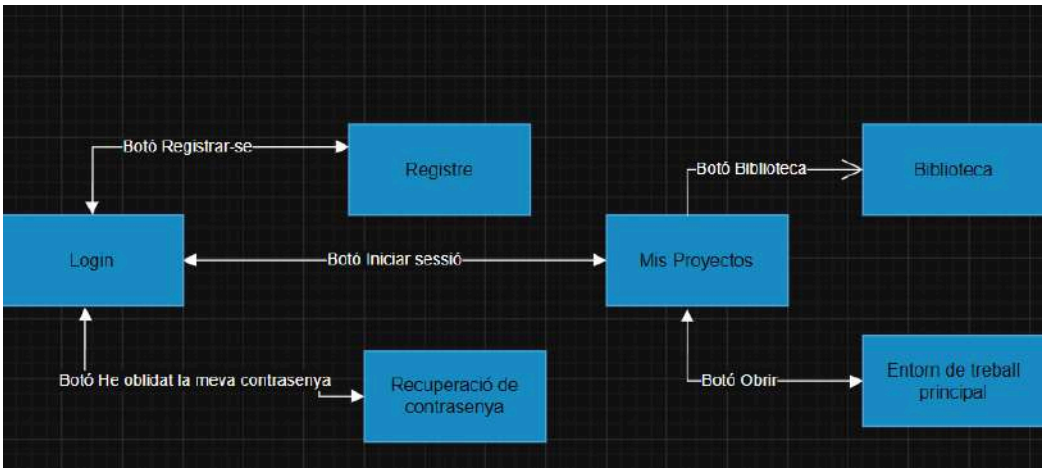


Figura 4.3: Diagrama de flux de navegació i arquitectura d'interfícies de l'aplicació REVRB.

BLOC 5 — DESENVOLUPAMENT / IMPLEMENTACIÓ

5.1 Estructura del projecte

L'aplicació s'ha desenvolupat seguint l'estàndard d'estructura de directoris de **Maven**, garantint la portabilitat i la gestió automatitzada del cicle de vida de l'aplicació. El codi font en Java es modularitza en paquets (*packages*) seguint criteris d'alta cohesió i baix acoblament:

- **com.revrb.app:** Conté la classe principal d'arrencada (Main) que hereta d'application.Application de **JavaFX**, inicialitzant el contenidor d'escenes primari (*Stage*).
- **com.revrb.audio:** Allotja el motor d'àudio polifònic i la lògica d'interacció amb el sintetitzador per programari.
- **com.revrb.model:** Conté el model de domini (entitats reflectides de la base de dades), els serveis de connexió HTTP i la lògica de persistència.
- **com.revrb.ui:** Agrupa els controladors de la interfície gràfica encarregats de capturar els esdeveniments de l'usuari (*Event Handlers*).

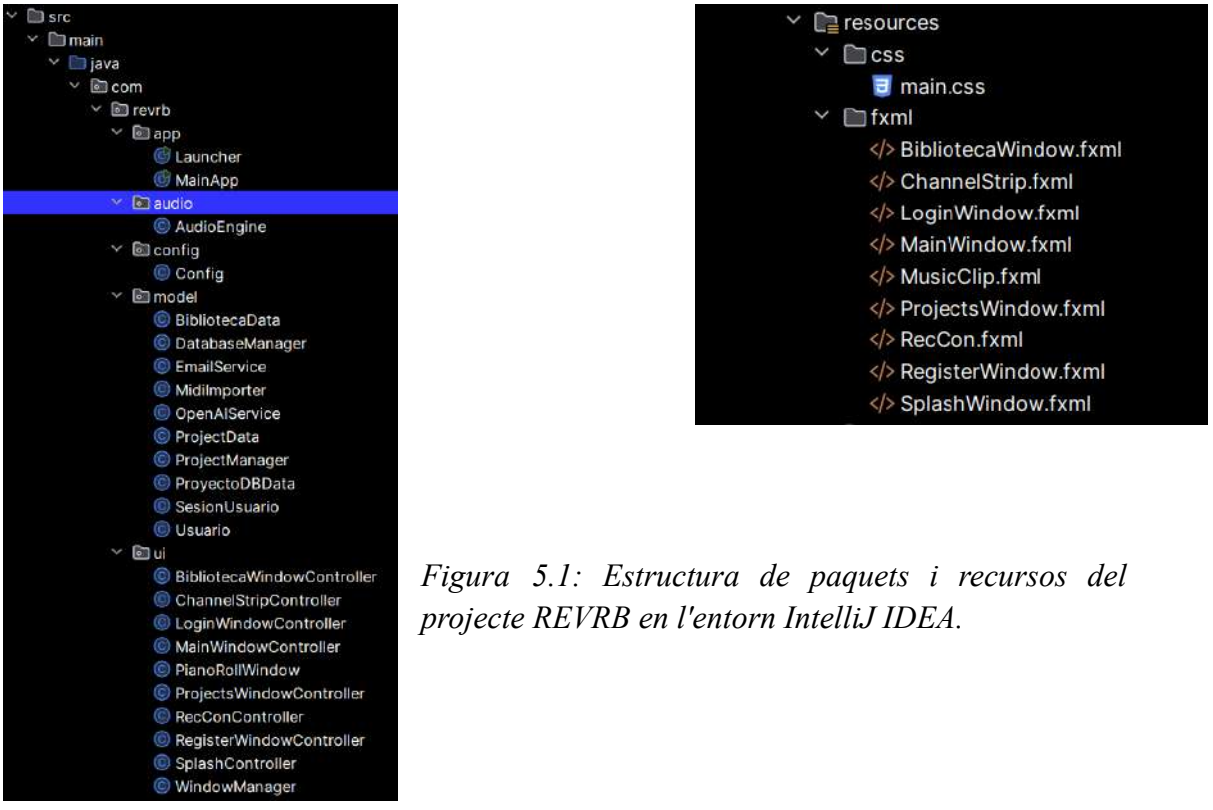


Figura 5.1: Estructura de paquets i recursos del projecte REVRB en l'entorn IntelliJ IDEA.

La gestió de llibreries es centralitza en el fitxer de configuració **pom.xml**. Les dependències externes de tercers incloses en el projecte són:

- mysql-connector-j: Driver JDBC per a la connexió nativa i segura amb el servidor MySQL al núvol.
- gson (Google): Llibreria de serialització i mapeig d'objectes Java a cadenes de text en format JSON.
- jakarta.mail: API per a la creació de sessions de correu, autenticació SMTP i enviament de missatges electrònics.
- org.json: Utilitats de baix nivell per al processament dinàmic de dades estructurades.

5.2 Sistema d'usuaris i autenticació

5.2.1 Registre

El mètode de registre implementa validacions en la capa de client abans d'interactuar amb la base de dades per estalviar peticions innecessàries a la xarxa. El controlador comprova mitjançant expressions regulars que el correu elèctronic sigui formalment vàlid, que l'identificador d'usuari superi una longitud mínima de caràcters i que la contrasenya disposi de mecanismes de robustesa bàsics. Si les precondicions es compleixen, s'executa una sentència INSERT contra la base de dades. Si es viola la restricció d'unicitat (UNIQUE KEY) dels camps usuario o email, el driver de MySQL llança una excepció SQLException que el controlador captura per mostrar una alerta visual a l'usuari sense que l'aplicació falli.

5.2.2 Inici de sessió

L'autenticació valida les credencials de l'usuari realitzant una consulta parametritzada (PreparedStatement) de tipus SELECT per evitar atacs d'injecció SQL (SQLi). Si la contrasenya emmagatzemada coincideix amb la introduïda, el sistema instancia un **objecte de sessió global** utilitzant el patró de disseny **Singleton**. Aquesta classe manté en memòria els identificadors de l'usuari autenticat durant tot el cicle de vida de l'execució, actuant com a proveïdor de context centralitzat per a les accions de desat de projectes, publicació o votació a la comunitat.

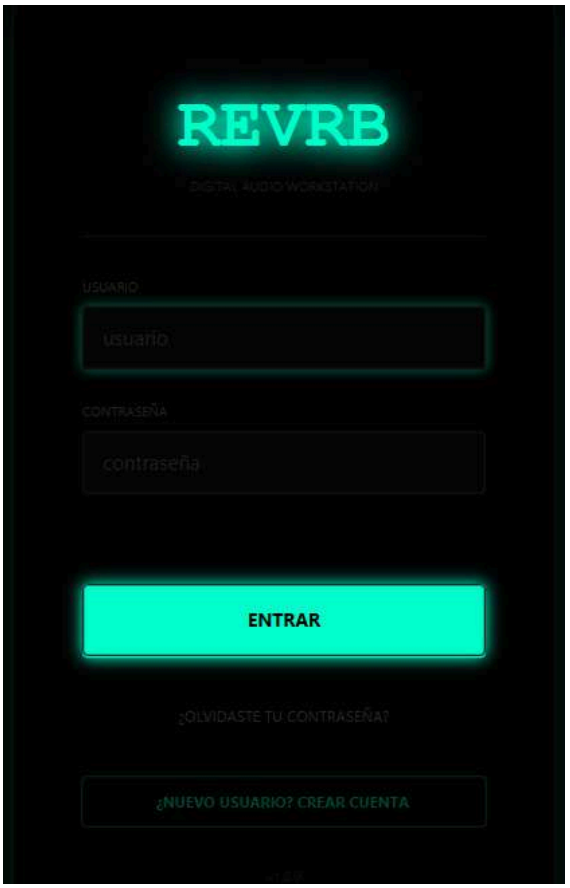


Figura 5.2: Pantalla Login

5.2.3 Recuperació de contrasenya

El mòdul de recuperació s'ha dissenyat com un procés segur en dues fases asíncrones:

1. **Fase de sol·licitud:** L'usuari introdueix el seu correu electrònic. El sistema genera un codi alfa-numèric pseudoaleatori actuant com a *token* de seguretat. Aquesta dada s'injecta a la taula usuarios en un registre temporal acompanyat d'una marca de temps (DATETIME) de caducitat fixada en 30 minuts.
2. **Fase d'enviament i validació:** S'instancia un fil secundari que utilitza l'API de **Jakarta Mail** per connectar-se de forma segura mitjançant el protocol **TLS** al servidor **SMTP** de Gmail. El sistema envia el correu de manera asíncrona perquè la

interfície d'usuari no es congeli. Quan l'usuari rep el codi i l'introdueix juntament amb la nova credencial, el sistema llegeix el registre i comprova la igualtat del codi i que la funció NOW() del servidor no hagi superat el camp reset_expiry.

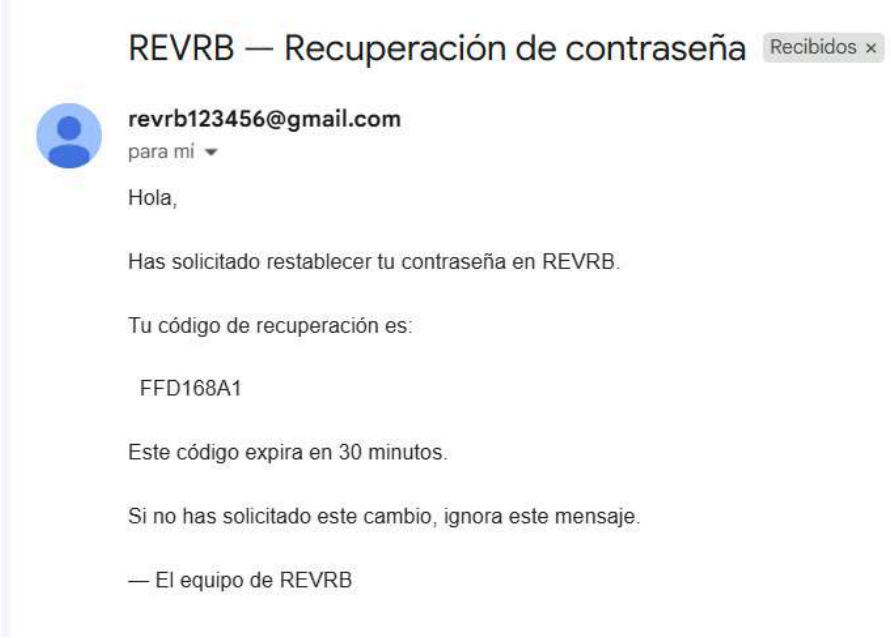


Figura 5.2.1: Email de recuperació de contrasenya

5.3 El DAW: pistes, clips i playlist

5.3.1 Sistema de pistes

Cada canal del seqüenciador es gestiona com un objecte de tipus Track. L'aplicació suporta una enumeració lògica de set variants (AUDIO, MIDI, DRUMS, BASS, SYNTH, VOCALS i FX). Cada tipologia defineix de manera nativa tres paràmetres: un prefix d'identificació visual en format emoji (com 🥁 per a percussió o 🎸 per a baix), el canal d'encaminament **MIDI** dedicat i l'instrument predeterminat del banc de sons.

La interfície gràfica de control de cada pista (*Channel Strip*) es construeix de manera dinàmica utilitzant nodes de JavaFX (HBox i VBox). Inclou elements com un component ComboBox per canviar l'instrument en temps real entre els 128 elements estàndard, selectors tipus botó (*ToggleButtons*) per als estats booleans de Mute i Solo, i un component Slider que modifica de forma contínua el valor de ganància (volum) de la pista.

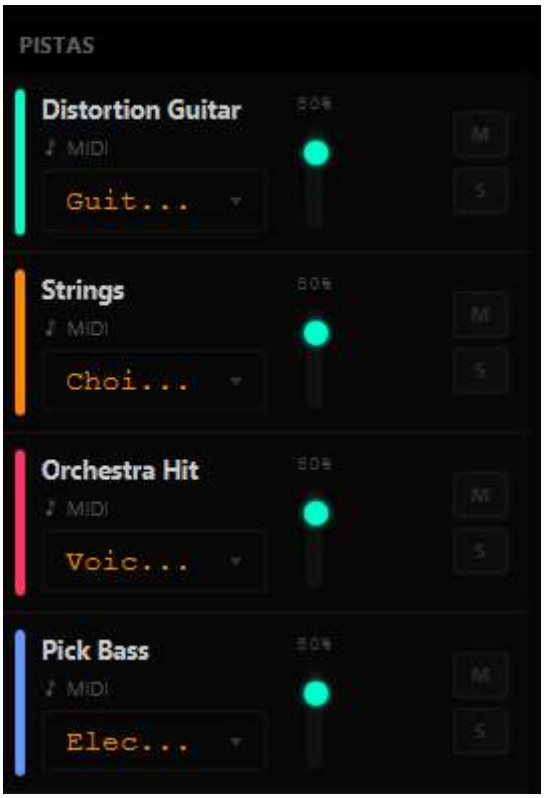


Figura 5.3: Pistas

5.3.2 Playlist i clips

La *playlist* és l'espai de treball on es renderitza el programari de línia de temps. Utilitza un disseny matricial on les files s'associen amb les instàncies de Track i l'eix X (columnes) representa la discretització del temps. Els fragments musicals es representen mitjançant instàncies de la classe Clip, dibuixades com a rectangles interactius sobre el llenç.

L'usuari interactua amb el panell gràfic gràcies als controladors d'esdeveniments del ratolí (setOnMousePressed, setOnMouseDragged). El sistema restringeix el moviment i el canvi d'amplada de l'objecte gràfic forçant una operació de **arrodoniment matemàtic** dependent de la configuració del *Snap to Grid* (ajust a la graella). Això garanteix que els clips se situïn exactament en els inicis dels compassos, evitant desviaments temporals (*jitters*) en la posterior reproducció.

A més, cada clip implementa un mètode de dibuix customitzat que renderitza de forma interna un resum gràfic de les seves notes MIDI subjacents mitjançant petites línies blanques

vectorials, oferint una previsualització òptica sense haver d'obrir editors ni finestres alternatives.

Figura 5.3.1: Playlist

5.3.3 Controls de reproducció i transport

La barra superior allotja la lògica del control de flux (*Transport Bar*). En prémer el botó de reproducció, s'activa un temporitzador gràfic (AnimationTimer o Timeline de JavaFX) encarregat de desplaçar de forma lineal la barra de posició o indicador de temps (*Playhead*) d'esquerra a dreta.

La velocitat de desplaçament es calcula analíticament a partir de la variable de tempo del projecte (**BPM**), correlacionant el pas dels píxels de la pantalla amb el transcurs del temps real en segons. El sistema manté un rellotge digital actualitzat amb el format estàndard de temps (hh:mm:ss). Si el *Playhead* supera el marge d'ajust del panell de visualització, es llança un mètode que recalcula el valor de desplaçament (*scroll*) de la finestra per mantenir l'indicador sempre centrat a la pantalla.

Figura 5.3.2: Botons de reproducció

5.4 Motor d'àudio MIDI

5.4.1 Funcionament del protocol MIDI

El nucli d'àudio de REVRB no processa fitxers d'àudio digital d'ona densa (com .wav o .mp3), sinó que opera sota l'estàndard industrial **MIDI** (*Musical Instrument Digital Interface*). Aquest enfocament implica que l'aplicació actua com un **seqüenciador d'esdeveniments**, transmetent paquets de control de dades pures (missatges que indiquen accions com *Note On* per activar una tecla, *Note Off* per apagar-la, el valor d'intensitat de velocitat i el número de canal afectat).

Dins del sistema, el mapa de temps es divideix en línies verticals on la unitat mínima és la **semicorxera** (un quart de *beat* musical). El tempo en mil·lisegons d'una semicorxera es dedueix dinàmicament mitjançant la fórmula matemática:

$$t_{\text{semicorxera}} = \frac{60.000}{\text{BPM} \times 4}$$

Cada nota musical s'emmagatzema en un objecte binari format per tres valors primitius: el codi de nota MIDI (0-127), la columna d'inici (posició absoluta a la graella) i la durada total del pols mesurada en columnes de semicorxera.

5.4.2 Sintetitzador Gervill i reproducció multitràck

L'arquitectura utilitza el sintetitzador per programari **Gervill**, integrat de forma nativa a la biblioteca **Java Sound API**. Això permet carregar bancs de sons i assignar programes d'instruments General MIDI en canals de comunicació independents de l'1 al 16.

Per garantir un rendiment òptim i evitar la degradació de la interfície d'usuari (evitant que el DAW es quedi congelat en sonar moltes notes alhora), el motor d'àudio gestiona de manera concurrent la reproducció mitjançant **fils d'execució independents** (*Multithreading*).

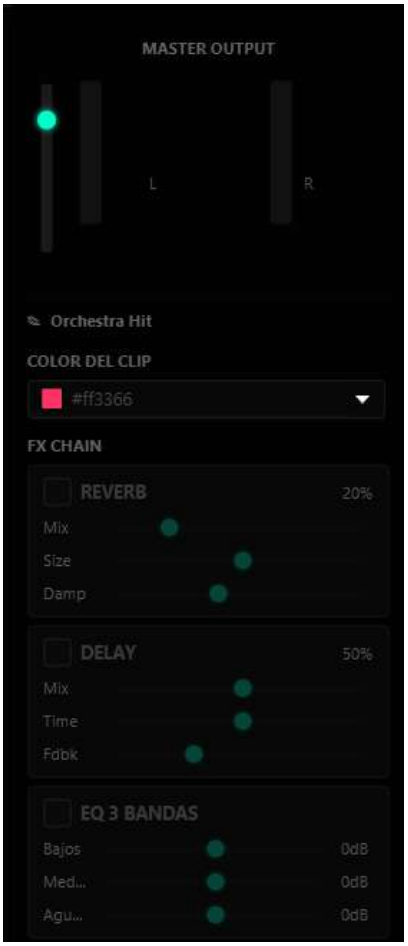
Quan s'inicia la reproducció, s'executa un fil mestre de rellotge col·laborant amb fils de treball secundaris per a cada pista activa. Aquests fils llegeixen la matriu de notes en memòria, calculen els retards d'execució en temps real mitjançant temporitzadors de precisió i envien els missatges ShortMessage al receptor del sintetitzador en el mil·lisegon exacte.

En prémer el botó d'aturada (*Stop*), els fils de concurrència s'interrompen de manera immediata i s'executa una rutina de seguretat coneguda com a **"All Notes Off"**, enviant de forma massiva el senyal de control CC 123 a tots els 16 canals del sintetitzador. Aquest

disseny garanteix la neteja absoluta de la memòria d'àudio i impedeix el bloqueig de notes o l'aparició de sons residuals suspesos de forma indefinida.

5.4.3 Control de volum i efectes auditius

La modificació dels paràmetres sonors s'efectua mitjançant missatges de canvi de control de tipus **MIDI CC** (*Control Change*). El control de volum individual s'acobla al canal enviant un missatge ShortMessage.CONTROL_CHANGE amb l'identificador de controlador CC 7 i un valor escalat d'intensitat entre 0 i 127.



Els mòduls d'efectes digitals de reverb i retard se deleguen directament a la capacitat de renderitzat d'àudio interna del motor Gervill. Per configurar el nivell d'efecte de cada canal es transmeten els controladors estàndard industrials de reverberació (CC 91) i efectes auxiliars/chorus (CC 93). Això estalvia potència de càlcul de la CPU de l'ordinador ja que el senyal es processa de forma nativa en el buffer d'àudio del sintetitzador.

La selecció de canals s'automatitza segons les directrius del protocol General MIDI, el qual dicta de forma obligatòria que el **canal 10** (índex 9 a nivell de codi) es destini exclusivament a la secció d'instruments de percussió. L'aplicació compta amb un mètode de control que intercepta la inicialització de les pistes de tipus DRUMS i les força a utilitzar aquest canal, garantint que les notes MIDI enviades es tradueixin sempre en cops de bateria i no en notes de piano tradicionals.

Figura 5.4: Control de volum i efectes

5.5 Piano Roll

El **Piano Roll** és el mòdul d'edició micro-musical a baix nivell. Està dissenyat com una escena independent implementada mitjançant un patró de finestra modal secundària que s'instancia quan l'usuari fa doble clic sobre el requadre de qualsevol pista del DAW.

L'entorn gràfic es divideix asimètricament en dos nodes de disseny connectats gràcies a un component ScrollPane:

1. **Zona de teclat (Esquerra):** Un llistat vertical de components que emulen de forma realista el disseny i disposició de les tecles d'un piano. Cada botó de tecla s'associa a un mètode que, en ser premut, transmet una ordre directa de *Note On* al sintetitzador per generar feedback auditiu instantani, servint com a referència tonal per a l'usuari.
2. **Graella de notes (Dreta):** Un panell interactiu basat en un component Canvas o un entorn matricial de nodes. L'eix horitzontal representa la seqüenciació en columnes de semicorxera, mentre que l'eix vertical s'alinea simètricament amb la línia de tecles del piano del costat esquerre.

Les notes s'introdueixen en un entorn gràfic de tipus vectorial. En fer un clic esquerre sobre una cel·la, el controlador llegeix les coordenades (X, Y), calcula a quina nota i a quin temps corresponen i dibuixa un rectangle de color blau que s'afegeix instantàniament a la col·lecció de notes del projecte. L'esborrat s'efectua mitjançant el clic dret del ratolí, eliminant l'objecte gràfic de la pantalla i la instància binaria de la memòria RAM del sistema.

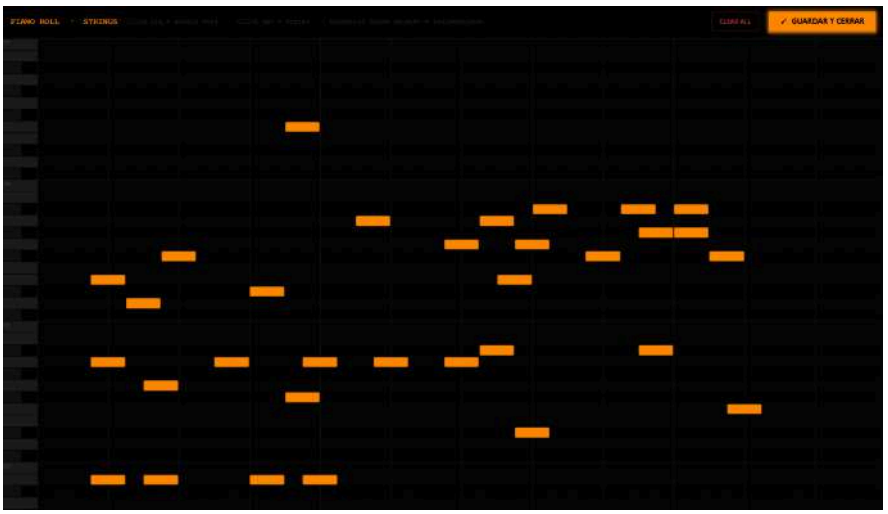


Figura 5.5: Piano Roll

5.6 Persistència de projectes

L'aplicació implementa una estratègia de **persistència híbrida**, unificant l'agilitat de l'emmagatzematge local en text pla amb la seguretat de l'allotjament centralitzat al núvol.

5.6.1 Emmagatzematge local .revrb (JSON)

L'extensió de fitxer de l'aplicació (.revrb) és una abstracció d'un fitxer de text estructurat en format **JSON**. El procés de desat es realitza gràcies a la llibreria **Google Gson**, la qual realitza un procés de **serialització** bolcant tot l'arbre de dependències de la sessió activa (metadades, arrays de pistes, llistat de clips i col·leccions de notes de cada piano roll) en una única cadena de text legible.

Durant l'operació inversa de lectura (**deserialització**), s'ha codificat un adaptador de tipus personalitzat (*JsonDeserializer*). Aquest mòdul actua com un filtre de seguretat per corregir les desviacions mètriques que generen els models de llenguatge d'IA (els quals, a causa de la naturalesa probabilística dels tokens, de vegades retornen variables numèriques senceres com a decimals, escrivint valors com 1.0 o 0.5 en lloc de dades de tipus int). El parsejador neteja i arrodoneix automàticament aquests valors a tipus primitius enters, evitant llançar excepcions d'error de format gràfic o d'àudio en carregar el projecte.

5.6.2 Persistència remota (Aiven Cloud)

Simultàniament al desat local, l'aplicació utilitza el controlador de connexió JDBC per enviar la mateixa cadena de text JSON a la base de dades remota. Aquesta dada s'emmagatzema directament dins d'una columna definida amb la tipologia de dades **LONGTEXT** a la taula projects.

Aquesta arquitectura de base de dades documental integrada dins d'una estructura relacional és altament eficient per a aquesta finalitat: permet desar configuracions i milers de notes musicals sense necessitat de dissenyar un esquema complex de múltiples taules relacionades per a les notes o realitzar centenars de consultes INSERT simultànies, reduint la latència de connexió a la xarxa a una única operació.

5.7 Biblioteca comunitària

5.7.1 Publicació de projectes

La publicació a la biblioteca es gestiona des de la interfície de gestió de projectes. En activar l'opció de compartir, l'aplicació obre un formulari on es recullen les metadades públiques (títol comunitari i descripció informativa).

El sistema realitza una transacció SQL que afegeix un registre a la taula community_library. Aquest registre guarda l'identificador del projecte base i de l'usuari autor. Com a mesura de seguretat a nivell d'esquema de dades, la clau estrangera d'enllaç compta amb la propietat **ON**

DELETE SET NULL. Això garanteix que si un usuari decideix eliminar un projecte del seu panell privat, l'entrada pública de la biblioteca no s'esborra de la xarxa, mantenint la cançó accessible per a la comunitat.

5.7.2 Exploració, descàrrega i valoració

La interfície de la biblioteca llegeix de forma dinàmica el llistat d'entrades mitjançant consultes que afegeixen clàusules d'ordenació segons els filtres seleccionats per l'usuari:

- **Data de publicació:** Ordre cronològic mitjançant ORDER BY fecha_subida DESC.
- **Descàrregues populars:** Ordre d'interès mitjançant ORDER BY descargas DESC.
- **Valoració reputacional:** Ordre de qualitat mitjançant una consulta amb agrupació que utilitza la funció d'agregació AVG(estrellas).

Quan un usuari executa l'acció de descàrrega, l'aplicació realitza una petició SELECT per llegir el camp contenido_json del projecte relacionat, el desa de forma local com a fitxer .revrb a la memòria de l'ordinador hoste i invoca el mètode d'obertura del DAW per carregar-lo. Al mateix temps, es llança una sentència asíncrona d'actualització (UPDATE) per incrementar en una unitat el comptador de la columna descargas d'aquell registre.

El mòdul de valoració permet assignar de l'1 al 5 un valor enter d'estrelles. La integritat d'aquest procés es delega directament al motor de la base de dades a través de la restricció d'unicitat composta **UNIQUE KEY (id_biblioteca, id_usuario)**. Si un usuari intenta enviar una segona puntuació sobre un mateix element, la base de dades denega la inserció per duplicitat, assegurant un sistema de vots net i democràtic.

Figura 5.7: Biblioteca

5.8 Assistent d'intel·ligència artificial

5.8.1 Integració de l'API asíncrona

La comunicació amb la intel·ligència artificial s'efectua realitzant connexions HTTPS contra l'API d'OpenAI utilitzant el model **GPT-4o-mini**. La integració s'ha programat de forma nativa fent servir el connector d'estil asíncron **java.net.http.HttpClient** introduït en les versions modernes de Java, evitant sobrecarregar el projecte amb frameworks externs.

Les peticions s'encapsulen en format JSON incloent variables de configuració de temperatura i un vector d'historial on es guarden estructuradament els darrers 30 missatges intercanviats de la sessió. Això dota l'assistent de memòria contextual durant el fil de la conversa.

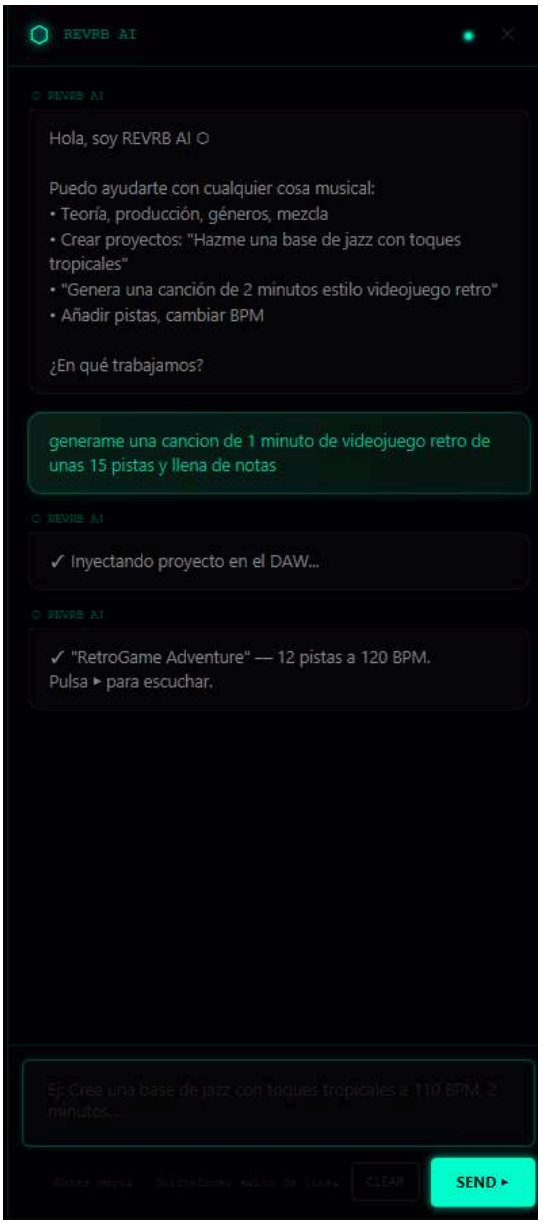


Figura 5.8: Asistent Revrb AI

5.8.2 Control d'enrutament de modes de resposta

El motor de REVRB compta amb un algorisme de detecció dinàmica que analitza la resposta del model de llenguatge per bifurcar el flux d'execució en dos entorns:

1. **Mode Conversacional (Text / Ordres):** Si l'usuari realitza consultes informatives, l'aplicació renderitza el text a la llista de missatges utilitzant etiquetes clàssiques. No obstant això, el *System Prompt* de la IA està configurat perquè, si l'usuari demana una modificació de l'entorn (com canviar el BPM o afegir canals), el model respongui en text però afegeixi al final un objecte d'ordre ocult. El controlador de REVRB analitza el text, extreu l'ordre i crida directament els mètodes operatius de la interfície d'usuari (per exemple, executant un canvi a la variable BPM o instanciant una nova fila de Track).
2. **Mode Generació de Projectes (JSON de persistència):** Si l'usuari demana explícitament la creació d'una composició, el prompt força el model a respondre **únicament i exclusiva** amb un objecte JSON estructurat que segueixi el patró complet d'un fitxer .revrb. L'aplicació intercepta la cadena, valida mitjançant un bloc try-catch que el format es pugui parsejar correctament amb *Gson* i obre un quadre de diàleg oferint a l'usuari l'opció d'injectar les pistes generades per la IA directament a la línia de temps de la seva sessió de treball de forma automatitzada.

5.9 Importació de fitxers MIDI

L'aplicació incorpora un servei lector de fitxers d'àudio externs amb format estàndard **MIDI** (fitxers .mid). El mòdul llegeix el flux de bytes del fitxer mitjançant el mètode MidiSystem.getSequence().

L'algorisme de conversió realitza un recorregut per l'arbre de pistes externes (*Tracks*), extreu els esdeveniments de canvi de tempo per fixar el valor de BPM global de manera congruent i transforma la mètrica de pols del fitxer (*ticks*) a la mètrica de coordenades horitzontals de la graella de REVRB basada en semicorxeres. Durant aquest procés, s'avalua el canal de transmissió original: si es detecta activitat al canal indexat 9, el convertidor crea de forma automatitzada una pista del nostre tipus de domini DRUMS, assegurant la compatibilitat de les bases de percussió dissenyades en altres programes professionals del mercat.

5.10 Estètica i disseny visual

L'estètica visual es dissenya seguint els estàndards formals d'interfície fosca d'alta fidelitat (**Total Black**). Totes les propietats gràfiques es governen de forma declarativa centralitzant l'estil en un fitxer CSS únic que sobrescriu el disseny per defecte dels components gràfics de JavaFX (*Modena*).

S'utilitzen variables CSS per assegurar la consistència global del color de l'aplicació:

- -fx-background-color principal fixat en negre pur (#000000) i grisos de seguretat foscos (#030303 a #111111) per als contenidors de dades, minimitzant el llamp de llum i la fatiga ocular del desenvolupador.
- Colors d'accent vibrants per destacar elements operatius: **Cyan** (#00FFCC) per a les línies temporals de playhead i selectors actius, i **Taronja** (#FF8800) per ressaltar indicadors de control d'efectes auditius.

La jerarquia de text s'estableix diferenciant els tipus de font. S'aplica la tipografia monoespaiada **Courier New** en tots els nodes de naturalesa mètrica o canviant (el temporitzador, indicadors de velocitat de compàs i valors numèrics), garantint que tots els caràcters ocupin exactament el mateix espai en píxels i evitant oscil·lacions visuals incòmodes durant la reproducció de la música. Per a la resta de l'entorn de controls, s'aplica la font de línies netes **Segoe UI**.

5.11 Pla de Proves i Verificació

Per avaluar el correcte funcionament de l'aplicació i la resistència del codi davant de fallades, s'ha definit un **Pla de Proves d'Integritat de Caixa Negra**. A continuació, es detallen els casos de prova executats, el comportament esperat i el resultat obtingut durant el procés de validació:

ID	Mòdul	Descripció de la Prova	Comportament Esperat	Estat
TP-1.1	Usuaris	Intent de registre introduint un correu electrònic sense el caràcter @ ni domini formal.	El sistema intercepta el format a la UI i denega l'enviament, mostrant una alerta vermella.	**Passat**
TP-1.2	Usuaris	Sol·licitud de canvi de contrasenya utilitzant un *token* temporal que ha superat els 30 minuts de vida.	La BD detecta la caducitat temporal i l'aplicació mostra el missatge: "El codi ha expirat".	**Passat**
TP-2.1	DAW	Moure un clip gràfic a la línia de temps fins a una posició intermèdia fora de compàs.	El mètode *Snap to Grid* arrodoneix la coordenada X forçant l'alineació automàtica amb la semicorxera més propera.	**Passat**
TP-2.2	DAW	Modificar de forma dinàmica el *slider* de volum d'una pista mentre sona una melodia.	El controlador llança un missatge MIDI CC 7 en temps real, modificant la ganància de so de forma fluida i sense talls.	**Passat**

TP-3.1	Àudio	Executar l'ordre de Stop de forma immediata mentre està sonant una composició de 16 pistes simultànies.	Es deturen els fils concurrents i s'envia el senyal CC 123 (All Notes Off), silenciant el sistema a l'instant sense deixar notes suspeses.	**Passat**
TP-3.2	Àudio	Assignar una pista a la tipologia DRUMS i introduir notes musicals aleatòries.	El sistema enruta de forma automàtica la informació cap al canal MIDI 9, generant correctament sons de percussió del banc Gervill.	**Passat**
TP-4.1	Persistència	Obre un fitxer de projecte local .revrb modificat manualment on les durades de les notes s'han escrit amb decimals (0.5).	L'adaptador de tipus personalitzat de *Gson* intercepta el format, realitza un arrodoniment automàtic i carrega el DAW sense llançar cap excepció.	**Passat**
TP-4.2	Biblioteca	Intentar puntuar el mateix projecte comunitari dues vegades seguides amb valoracions diferents.	La restricció d'unicitat composta de la BD llança un error per clau duplicada, bloquejant la segona inserció de vot.	**Passat**
TP-5.1	IA	Demanar a l'assistent de veu una acció combinada: "Explica'm què és el Techno i afegeix una pista de sintetitzador".	La IA respon amb el text explicatiu i inclou un objecte d'ordre que el controlador llegeix per crear la pista automàticament al DAW.	**Passat**

TP-5.2	IA	Sol·licitar la generació d'una base de Trap de forma explícita a l'assistent.	La IA retorna exclusivament una estructura de codi JSON net, el sistema el parseja i desplega la interfície d'injecció de projecte nou, però no molt professional, falla a vegades.	Passat 50%

BLOC 6 — CONCLUSIONS

6.1 Conclusions generals

El desenvolupament de REVRB ha estat un projecte ambiciós i alhora molt gratificant. Construir un DAW funcional des de zero, integrar un sistema de base de dades remota, implementar una biblioteca comunitària i afegir un assistent d'intel·ligència artificial en un únic curs escolar i com a projecte individual ha requerit una planificació acurada, molta disciplina i la capacitat de prendre decisions tècniques importants en poc temps.

El resultat és una aplicació funcional que permet crear, editar i reproduir composicions musicals MIDI de forma visual i intuïtiva, compartir-les amb una comunitat d'usuaris i generar-les de forma automàtica a través d'un assistent d'intel·ligència artificial. Totes les funcionalitats principals han quedat implementades i funcionen de forma coherent com un producte integrat.

Des del punt de vista personal, el projecte ha representat una immersió profunda en tecnologies que no s'aborden en profunditat al llarg del cicle formatiu, com la programació d'àudio, la gestió de fils d'execució concurrents per a la reproducció multitrack, o la integració d'APIs d'intel·ligència artificial en una aplicació d'escriptori. Aquestes àrees han suposat un repte important però han enriquit molt la formació tècnica.

Un dels aprenentatges més importants del projecte ha estat la importància de la planificació i la priorització. Davant d'un projecte tan ampli, és fàcil caure en la temptació d'intentar implementar-ho tot i no acabar res. La metodologia Kanban ha ajudat a mantenir el focus en les funcionalitats prioritàries i a identificar ràpidament quines tasques eren crítiques per a l'entrega.

6.2 Consecució d'objectius

Tots els objectius generals i específics definits a l'inici del projecte s'han assolit de forma satisfactòria.

El DAW funcional amb reproducció en temps real, les múltiples pistes amb control de volum, mute i solo, el piano roll interactiu, la selecció d'instruments del General MIDI, la persistència híbrida en fitxer local i base de dades remota, el sistema complet de gestió d'usuaris amb recuperació de contrasenya, la biblioteca comunitària amb filtres i valoracions, i l'assistent d'intel·ligència artificial capaç de generar projectes musicals complets: tots aquests elements han quedat implementats i funcionen de forma integrada.

Alguns aspectes del projecte han resultat més complexos del previst, especialment la sincronització del playhead amb l'àudio en la importació de fitxers MIDI externs, que presenta inconsistències en composicions molt llargues. Aquestes limitacions s'han documentat i es consideren oportunitats de millora per a versions futures.

6.3 Valoració de la metodologia i planificació

La metodologia Kanban amb Trello ha funcionat molt bé per a un projecte individual de les dimensions de REVRB. La visibilitat constant del tauler ha permès mantenir el focus en les tasques prioritàries i identificar ràpidament quan alguna funcionalitat s'estava complicant més del previst.

El control de versions amb GitHub ha estat imprescindible. En diverses ocasions, una nova implementació ha trencat comportaments preexistents, i disposar d'un historial complet de canvis ha permès revertir de forma segura sense perdre el treball realitzat.

L'organització en fases seqüencials ha estat una bona decisió, ja que ha garantit que cada nova funcionalitat es construís sobre una base sòlida i provada. En projectes d'aquesta complexitat, intentar implementar tot simultàniament porta inevitablement a errors difícils de depurar.

Com a punt de millora, en futurs projectes hauria estat útil establir criteris de qualitat més formals per a la finalització de cada fase (definir exactament quines proves ha de superar una funcionalitat per considerar-la "completada") i dedicar temps de forma explícita a la documentació tècnica del codi a mesura que s'avancés, en lloc de deixar-ho per al final.

6.4 Visió de futur

REVRB és un projecte amb molt recorregut per davant. Les funcionalitats implementades en aquest curs representen una base sòlida sobre la qual es poden construir moltes millores i extensions.

En primer lloc, millorar la precisió de la sincronització del playhead amb l'àudio és una prioritat tècnica per a versions futures, especialment per a composicions importades de fitxers MIDI externs, però per temes de latència i limitacions no s'ha pogut fer.

En segon lloc, estendre el piano roll amb funcionalitats avançades com la selecció múltiple de notes, la transposició, la quantització automàtica i la visualització de la velocitat (intensitat) de cada nota aportaria molta potència a l'edició musical.

En tercer lloc, la possibilitat d'exportar el projecte com un fitxer d'àudio (WAV o MP3) permetria als usuaris compartir la seva música més fàcilment fora de l'aplicació. Això implicaria implementar un sistema de renderització d'àudio, que és tècnicament complex però plenament realitzable amb Java Sound API.

En quart lloc, ampliar les capacitats de l'assistent d'intel·ligència artificial per permetre la modificació de pistes existents (afegir notes, canviar instruments, modificar el volum d'una pista específica) en lloc de treballar únicament amb projectes nous obriria possibilitats molt interessants, i que pugui fer composicions més professionals.

Finalment, millorar el sistema d'efectes de so de REVRB, afegint molts més i millorant els que hi ha per a que els projectes siguin més professionals i personalitzats.

7. Glossari

DAW (Digital Audio Workstation): Estació de treball d'àudio digital. Programari que permet crear, editar, barrejar i reproduir música de forma digital.

MIDI (Musical Instrument Digital Interface): Protocol estàndard de comunicació per a instruments electrònics i programari musical. En lloc d'emmagatzemar so, emmagatzema instruccions musicals (quines notes sonar, quan i com).

General MIDI Standard: Especificació que defineix un conjunt de 128 instruments agrupats en categories, present en qualsevol dispositiu o programari compatible amb MIDI. Garanteix que el programa 0 sigui sempre un piano de cua, el programa 40 sempre un violí, etc.

Sintetitzador: Dispositiu o programa que genera so electrònicament a partir d'instruccions MIDI.

Gervill: Sintetitzador software inclòs en el JDK de Java des de la versió 7. És el motor d'àudio que REVRB utilitza per generar els sons.

JavaFX: Framework per al desenvolupament d'interfícies gràfiques d'escriptori en Java. Permet definir la interfície en fitxers FXML i aplicar estils visuals amb CSS.

FXML: Format basat en XML específic de JavaFX per definir l'estructura visual d'una pantalla de forma declarativa.

MVC (Model-Vista-Controlador): Patró d'arquitectura de software que separa el sistema en tres capes: el model (lògica de negoci i dades), la vista (interfície d'usuari) i el controlador (pont entre les dues).

JDBC (Java Database Connectivity): API de Java per connectar-se i interactuar amb bases de dades relacionals des de codi Java.

JSON (JavaScript Object Notation): Format de text lleuger per emmagatzemar i intercanviar dades estructurades. Llegible per humans i fàcilment processable per màquines.

BPM (Beats Per Minute): Unitat de mesura del tempo musical. Indica quantes batudes (beats) ocorren en un minut. Un BPM alt implica una música ràpida; un BPM baix, una música lenta.

Piano roll: Eina visual d'edició de notes MIDI que mostra les notes com a rectangles sobre una graella, on l'eix horitzontal és el temps i el vertical és el to.

Playlist: En un DAW, l'àrea visual on es col·loquen i organitzen els clips de cada pista al llarg del temps.

Channel strip: Panell de control d'una pista de so, que inclou el volum, el mute, el solo i altres paràmetres.

Snap to grid: Funcionalitat que força els clips i les notes a alinear-se automàticament a les divisions de la graella temporal.

Token: En el context de la recuperació de contrasenya, un codi aleatori d'un sol ús que s'envia per correu electrònic i permet verificar la identitat de l'usuari.

API (Application Programming Interface): Conjunt de protocols i definicions que permeten que dos programes es comuniquin entre ells.

Kanban: Metodologia de gestió de projectes basada en taulers visuals que mostren l'estat de cada tasca.

Maven: Eina de gestió de projectes Java que automatitza la compilació, les dependències i la construcció del projecte.

LTS (Long Term Support): Versió d'un programari que rep suport i actualitzacions de seguretat durant un període llarg de temps (generalment 5-8 anys).

SMTP: Protocol de comunicació per a l'enviament de correus electrònics entre servidors.

SSL (Secure Sockets Layer): Protocol de seguretat que xifra les comunicacions entre dos punts d'una xarxa, evitant que tercers puguin llegir les dades transmeses.

8. Bibliografia

OpenAI. (2024). *GPT-4o mini — OpenAI API Documentation*. Recuperat de https://platform.openai.com/docs

Oracle. (2024). *Java SE 21 Documentation*. Recuperat de https://docs.oracle.com/en/java/javase/21/

OpenJFX. (2024). *JavaFX 21 Documentation*. Recuperat de https://openjfx.io/javadoc/21/

Oracle. (2024). *Java Sound API Guide*. Recuperat de https://docs.oracle.com/en/java/javase/21/sound/

MySQL. (2024). *MySQL 8.0 Reference Manual*. Recuperat de https://dev.mysql.com/doc/

Aiven. (2024). *Aiven for MySQL Documentation*. Recuperat de https://aiven.io/docs/products/mysql

Google. (2024). *Gson User Guide*. Recuperat de https://github.com/google/gson/blob/main/UserGuide.md

Jakarta EE. (2023). *Jakarta Mail 2.0 Specification*. Recuperat de https://jakarta.ee/specifications/mail/

MIDI Manufacturers Association. (2020). *General MIDI Level 1 Specification*. Recuperat de https://www.midi.org/specifications-old/item/general-midi

JetBrains. (2024). *IntelliJ IDEA Documentation*. Recuperat de https://www.jetbrains.com/help/idea/

Atlassian. (2024). *Trello Guide: Getting Started with Kanban*. Recuperat de https://trello.com/guide

Apache Maven. (2024). *Maven Getting Started Guide*. Recuperat de https://maven.apache.org/guides/getting-started /

Pokémon Black/White - BATTLE! ELITE FOUR - FL Studio Deconstruction. Recuperat de https://www.youtube.com/watch?v=KazOZAuju40&list=PLhxjySv20GqIdKQU4Tl_TQgCXWoNI67ZT

FL Studio Piano Roll | FULL TUTORIAL. Recuperat de https://www.youtube.com/watch?v=_VDGidEjQSM

🍑 FL Studio 20 - #4: Piano Roll Basico [CURSO COMPLETO] - Tutorial. Recuperat de https://www.youtube.com/watch?v=ulAFunaTmzU

9. Annexos

Descarga del programa: https://nimble-sfogliatella-8d6424.netlify.app/

Nota sobre l'ús d'intel·ligència artificial

Durant el desenvolupament d'aquest projecte s'han utilitzat eines d'intel·ligència artificial de diverses formes. En primer lloc, l'assistent d'IA (GPT-4o) s'ha integrat directament a l'aplicació com una funcionalitat pròpia del DAW, i s'ha provat extensament per verificar el seu correcte funcionament. En segon lloc, s'han utilitzat assistents d'IA de forma puntual com a eina de suport durant el desenvolupament, principalment per resoldre dubtes tècnics concrets sobre Java Sound API, JavaFX i el format MIDI, i per revisar fragments de codi en cerca d'errors. On més he utilitzat la IA ha sigut per l'estil CSS de l'aplicació, per qüestions de velocitat i desconeixement de CSS.

En tots els casos, el codi final ha estat escrit, comprès i verificat per l'autor del projecte. Cap fragment de codi significatiu ha estat copiat directament d'una IA sense comprendre'l prèviament. L'ús de les eines d'IA ha estat un complement a l'aprenentatge, no un substitut.

Llicència