Institut Puig Castellar
Santa Coloma de Gramenet

# Custom background sound

**Projecte de desenvolupament**
CFGM Sistemes Microinformàtics i Xarxes (SMX)

**Yuhan Zhu - Ziqi Zhu**
**Grup C**
**CFGM Sistemes Microinformàtics i Xarxes**

# Resum del projecte:

El projecte "Custom Background Sound" (CBS) consisteix en el desenvolupament d'una aplicació web interactiva que permet als usuaris reproduir, combinar i barrejar diferents sons ambientals —com ara pluja, vent, foguera, ocells o onades del mar— per crear entorns sonors completament personalitzats.

Aquestes combinacions ajuden els usuaris a millorar la concentració, reduir l'estrès, fomentar la relaxació i facilitar el descans en contextos com l'estudi, el treball o el son.

L'objectiu principal és proporcionar una eina gratuïta, intuïtiva i fàcil d'utilitzar que permeti a qualsevol persona dissenyar el seu propi espai sonor de manera senzilla, amb èmfasi en la personalització i la persistència de preferències.

Els usuaris poden ajustar el volum de cada so individualment, desar les seves configuracions mitjançant un compte d'usuari, crear modes predefinits (Estudi, Relaxació, Son) i compartir les seves mescles.

Per assolir aquests objectius, s'han aplicat metodologies àgils (Scrum) per gestionar les diferents fases del desenvolupament. La implementació tècnica s'ha realitzat amb React per al front-end, Howler.js per a la gestió eficient de l'àudio, Zustand per a l'estat global i localStorage amb xifratge SHA256 per a la persistència de dades d'usuari. Tots els arxius de so provenen de fonts amb llicències obertes CC-BY o CC0.

El resultat final és una plataforma web moderna, creativa i funcional, que combina tecnologia i benestar. El projecte demostra que és possible crear un sistema d'autenticació completament funcional i una experiència d'usuari personalitzada sense necessitat d'un backend complex, utilitzant únicament tecnologies front-end modernes.

**Paraules clau (entre 4 i 8):**

Sons ambientals · Relaxació · Concentració · Aplicació web · React · Howler.js · Persistència local · Personalització

# Abstract:

The "Custom Background Sound" (CBS) project consists of developing an interactive web application that allows users to play, combine and mix different environmental sounds —such as rain, wind, campfire, birds or sea waves— to create completely personalized sound environments.

These combinations help users improve concentration, reduce stress, promote relaxation and facilitate rest in contexts such as studying, working or sleeping.

The main objective is to provide a free, intuitive and easy-to-use tool that allows anyone to design their own sound space in a simple way, with emphasis on personalization and preference persistence.

Users can adjust the volume of each sound individually, save their settings through a user account, create preset modes (Study, Relaxation, Sleep) and share their mixes.

To achieve these objectives, agile methodologies (Scrum) have been applied to manage the different development phases. The technical implementation has been carried out with React for the front-end, Howler.js for efficient audio management, Zustand for global state management, and localStorage with SHA256 encryption for user data persistence. All sound files come from open-licensed sources (CC-BY or CC0).

The final result is a modern, creative and functional web platform that combines technology and well-being. The project demonstrates that it is possible to create a fully functional authentication system and a personalized user experience without the need for a complex backend, using only modern front-end technologies.

**Keywords (entre 4 i 8):**

Ambient sounds · Relaxation · Concentration · Web application · React · Howler.js · Local storage · Personalization

# Índex

# 1 Introducció

Aquest projecte consisteix en el desenvolupament d'una aplicació web interactiva anomenada **"Custom Background Sound" (CBS)**, destinada a la creació de paisatges sonors personalitzats. L'objectiu principal és oferir una plataforma accessible i intuïtiva que permeti als usuaris combinar diferents sons ambientals (com pluja, vent, onades, foc) i ajustar-ne individualment el volum, amb la finalitat de millorar la concentració, la relaxació o la qualitat del son.

El projecte se sustenta en tecnologies web modernes com **React** per a la interfície d'usuari i **Howler.js** per a la gestió eficient de l'àudio, i s'allotjarà a **GitHub** Pages, una plataforma gratuïta, per garantir-ne l'accessibilitat universal.

## 1.1 Context

Els sons ambientals i el soroll blanc són àmpliament reconeguts com a maneres efectives de regular l'estat d'ànim, millorar l'eficiència laboral i crear una atmosfera confortable. En l'era digital actual, cada vegada més persones busquen eines digitals que els ajudin a aïllar-se del soroll exterior i crear el seu propi espai de concentració o relaxació.

En aquest context, aquest projecte pretén construir una plataforma de mescla de so fàcilment accessible, intuïtiva i altament personalitzable. Basat en fonts d'àudio de codi obert amb llicències **CC0** i **CC-BY** i tecnologies web modernes, el projecte permetrà als usuaris crear entorns sonors únics adaptats a les seves necessitats específiques, ja sigui per estudiar, treballar, meditar o dormir.

## 1.2 Justificació

La importància d'aquest projecte rau en tres pilars fonamentals:

- **Accessibilitat universal:** Proporcionar una solució gratuïta i sense instal·lació que s'executa directament al navegador, eliminant barreres d'entrada per a qualsevol usuari.
- **Legalitat i sostenibilitat:** Utilitzar recursos de so amb llicència oberta (CC0, CC-BY) per garantir la legalitat del projecte i el respecte als drets dels autors originals.
- **Experiència personalitzada:** Oferir una experiència altament personalitzable que permeti a cada usuari dissenyar el seu propi ambient sonor segons les seves preferències i necessitats momentànies.

# 1.3 Objectius

## 1.3.1 Objectiu general

Crear una aplicació web que permeti als usuaris generar paisatges sonors personalitzats barrejant i ajustant sons ambientals per ajudar a concentrar-se, relaxar-se o dormir.

## 1.3.2 Objectius específics

- Proporcionar almenys 5 sons ambientals
- Implementar un mesclador de so amb volum ajustable independentment
- Afegir temporitzador bàsic (1-60 minuts)
- Crear modes preestablerts (Focus, Relax, Sleep, Meditate, Reading, L'Ignot)
- Permetre que la configuració es desi a través de localStorage o del compte d'usuari
- Explorar viabilitat de PWA
- Implementar gravació i compartició de vídeo de la mescla activa

# 1.4 Estratègia i planificació del projecte

L'estratègia tecnològica del projecte es basa en tres components clau:

| Component | Tecnologia | Justificació |
|---|---|---|
| Front-end | React 19 + TypeScript | Comunitat madura, ecosistema ric, altament compatible amb Howler.js i PWA |
| Motor d'àudio | Howler.js 2.2.4 | Fàcil d'utilitzar, rendiment multiplataforma estable, suport natiu per a múltiples pistes |
| Emmagatzematge | localStorage | Senzill, sense necessitat de backend, suficient per a les necessitats inicials |
| Control de versions | Git + GitHub | Seguiment de canvis, col·laboració, integració amb eines de gestió |
| Allotjament | GitHub Pages | Gratuït, integració perfecta amb el repositori |

Aquesta estratègia és altament factible perquè ofereix una alta eficiència de desenvolupament, una estructura clara, una forta escalabilitat i una fàcil implementació, alhora que simplifica el procés d'implementació per al processament d'àudio.

# 1.5 Metodologia de treball

Aquest projecte adoptarà el mètode de desenvolupament **àgil (Agile)**, prenent com a referència les idees de treball de **Scrum**. L'elecció d'un mètode àgil en lloc del model tradicional en cascada (Waterfall) es basa principalment en els motius següents:

**Flexibilitat i iteració**

El lloc web del mesclador de sons ambientals inclou una gran quantitat de funcionalitats ajustables (control de volum, temporitzador, etc.). Aquestes funcionalitats poden requerir modificacions durant el procés de desenvolupament segons el retorn de l'experiència d'usuari. Per això, els curts cicles d'iteració (sprints) i la millora contínua del mètode àgil s'adapten millor a un projecte en evolució constant.

**Alta variabilitat dels requisits**

A mesura que avança el projecte, el nombre de fonts sonores, el disseny de la interfície o el mecanisme de desament poden variar en funció dels resultats de les proves. Les iteracions de Scrum permeten ajustar ràpidament el backlog sense haver de fixar els requisits en cada fase, com passa en el model en cascada.

**Fàcil aplicació de passos petits i continus**

Com que el projecte està format per diversos mòduls —front-end, motor d'àudio, UI/UX— Scrum permet implementar i provar aquests mòduls de manera progressiva:

- Sprint 1: Implementació de la interfície bàsica i la reproducció d'àudio
- Sprint 2: Afegir el mesclador de sons
- Sprint 3: Afegir temporitzador
- Sprint 4: Modes predefinits i mecanisme de desament
- Sprint 5: Exploració de PWA i millores finals

**Eines de gestió del projecte**

Per garantir una execució efectiva del mètode àgil, en aquest projecte s'utilitzarà GitHub Projects com a eina principal de gestió, basant-nos en l'anàlisi comparativa realitzada:

| Eina | Descripció | Punts forts | Punts febles |
|---|---|---|---|
| Trello | Eina Kanban visual | Interfície intuïtiva, compatible amb mòbils | Funcions avançades limitades en versió gratuïta |
| Notion | Plataforma integral | Flexibilitat, bases de dades | Corba d'aprenentatge pronunciada |
| GitHub Projects | Tauler integrat a GitHub | Sincronització automàtica amb el codi, ideal per seguiment | Menys versàtil per a tasques no relacionades amb codi |

Eina triada: GitHub Projects

**Justificació de l'elecció:**
Integració directa amb el codi: Com el nostre projecte es desenvolupa a GitHub, podem vincular tasques amb commits, pull requests i incidències.

**Seguiment complet:** Permet seguir el progrés de cada funcionalitat des de la idea fins a la implementació final.

**Accessibilitat:** Tots els membres de l'equip i el professor poden accedir fàcilment al tauler sense necessitat de comptes externs.

**Suport amb Diagrama de Gantt**
Tot i que l'àgil posa més èmfasi en la flexibilitat, s'utilitzarà un **Diagrama de Gantt** a l'inici del projecte per planificar de manera global el temps i l'ordre de les tasques, proporcionant una visió general dels terminis estimats.

## 1.6 Estudi econòmic i pressupostari

Aquest apartat presenta una anàlisi detallada dels recursos necessaris, els costos associats i la càrrega de treball estimada per al desenvolupament del projecte Custom Background Sound (CBS). Donat que es tracta d'un projecte acadèmic sense ànim de lucre, l'enfocament principal és minimitzar costos utilitzant eines gratuïtes i de codi obert.

### 1.6.1 Tasques principals

| ID | Tasca | Descripció |
|---|---|---|
| T1 | Recopilació d'àudio | Cercar sons ambientals amb llicència CC0 o CC-BY a Freesound.org |
| T2 | Verificació de llicències | Revisar manualment les llicències i crear un registre de fonts |
| T3 | Processament d'àudio | Editar, reduir soroll i comprimir àudio amb Audacity i FFmpeg |
| T4 | Desenvolupament Front-end | Implementar la interfície d'usuari amb React + TypeScript |
| T5 | Integració d'àudio | Integrar Howler.js per a la reproducció multicanal |
| T6 | Implementació de funcionalitats | Afegir temporitzador, modes predefinits i localStorage |
| T7 | Desenvolupament PWA (exploració) | Configuració de Service Worker i mode fora de línia |
| T8 | Desplegament i proves | Desplegament a GitHub Pages i proves de compatibilitat |
| T9 | Documentació | Redacció de la guia d'ús i manteniment del repositori |

### 1.6.2 Recursos humans

| Rol | Responsabilitats | Dedicació | Cost |
|---|---|---|---|
| Desenvolupador Front-end | Implementació de la interfície d'usuari, integració amb React | 100% (2 persones) | Projecte acadèmic |
| Dissenyador UI/UX | Disseny de la interfície i experiència d'usuari | 30% (compartit) | Projecte acadèmic |
| Tècnic d'àudio | Processament i optimització dels fitxers d'àudio | 20% (compartit) | Projecte acadèmic |
| Tester | Proves de compatibilitat i detecció d'errors | 15% (compartit) | Projecte acadèmic |

**Nota:** Tractant-se d'un projecte acadèmic, els costos de personal no es contemplen en termes monetaris, però sí en hores de dedicació.

### 1.6.3 Recursos materials i tecnològics

| Categoria | Recurs | Cost | Justificació |
|---|---|---|---|
| Recursos d'àudio | Freesound.org (CC0, CC-BY) | Gratuït | Més de 600.000 sons disponibles amb llicències obertes |
| Eines d'edició d'àudio | Audacity | Gratuït | Editor d'àudio de codi obert, funcionalitats professionals |
| Eines de conversió | FFmpeg | Gratuït | Eina potent per a la conversió i compressió d'àudio |
| Framework Front-end | React 19 + TypeScript | Gratuït | Ecosistema madur, documentació extensa |
| Llibreria d'àudio | Howler.js 2.2.4 | Gratuït | Gestió eficient de reproducció multicanal |
| Control de versions | Git + GitHub | Gratuït | 2000 minuts d'execució/mes (suficient) |
| Allotjament | GitHub Pages | Gratuït | 1 GB d'emmagatzematge, 100 GB d'ample de banda/mes |
| Gestió de projectes | GitHub Projects | Gratuït | Integració directa amb el repositori de codi |
| Disseny UI | CSS Modules | Gratuït | Estils locals, sense dependències externes |
| Icones | React Icons | Gratuït | Més de 20.000 icones, càrrega sota demanda |

### 1.6.4 Estimació de càrrega de treball

| Fase | Tasques | Hores estimades |
|---|---|---|
| Fase 1: Preparació d'àudio | T1, T2, T3 | 20 hores |
| Fase 2: Desenvolupament base | T4, T5 | 150 hores |
| Fase 3: Funcionalitats avançades | T6, T7 | 60 hores |
| Fase 4: Desplegament i documentació | T8, T9 | 30 hores |
| **TOTAL** | | **~260 hores** |

### 1.6.5 Taula resum de costos

| Concepte | Cost directe | Cost indirecte | Observacions |
|---|---|---|---|
| Recursos humans | 0 € | 260 hores | Projecte acadèmic |
| Recursos d'àudio | 0 € | 0 € | Llicències CC0/CC-BY |
| Eines de desenvolupament | 0 € | 0 € | Programari gratuït/codi obert |
| Allotjament | 0 € | 0 € | GitHub Pages |
| Domini personalitzat | 0 € | 0 € | No necessari inicialment |
| **TOTAL** | **0 €** | **260 hores** | |

### 1.6.8 Valoració final

| Concepte | Cost directe | Cost indirecte | Observacions |
|---|---|---|---|
| Recursos humans | 0 € | 260 hores | Projecte acadèmic |
| Recursos d'àudio | 0 € | 0 € | Llicències CC0/CC-BY |
| Eines de desenvolupament | 0 € | 0 € | Programari gratuït/codi obert |
| Allotjament | 0 € | 0 € | GitHub Pages |
| Domini personalitzat | 0 € | 0 € | No necessari inicialment |
| **TOTAL** | **0 €** | **260 hores** | |

# 2 Descripció del projecte

## 2.1 Anàlisi de requisits CBS

CBS és una aplicació web de reproducció de sons de fons personalitzats que permet als usuaris barrejar i ajustar diferents sons ambientals per crear l'entreniment d'àudio perfecte per a l'estudi, el treball, la relaxació o el son.

### 2.1.1 Requisits funcionals

**Interfície d'usuari**

- **RF1:** Mostra 8 targetes de so de fons predefinides
- **RF2:** Cada targeta de so inclou una icona i un nom
- **RF3:** Interfície en català

**Funcions de control d'àudio**

- **RF4:** Control independent de reproducció/pausa per a cada so
- **RF5:** Control lliscant de volum independent per a cada so (0-100%)
- **RF6:** Control global de reproducció/pausa
- **RF7:** Control global de volum

**Funció de temporitzador**

- **RF8:** Temporitzador (Atura automàticament tots els sons en finalitzar el temporitzador. Màxim 60 minuts)

**Visualització d'estat**

- **RF9:** Mostra l'estat actual de la reproducció
- **RF10:** Mostra el volum mitjà
- **RF11:** Mostra el nombre de sons actius
- **RF12:** Guarda i càrrega configuracions de combinacions de sons

**Funcions del compte d'usuari**

- **RF13:** Inici de sessió/registre d'usuari
- **RF14:** Preferits d'usuari i gestió de presets
- **RF15:** Permet als usuaris crear comptes amb noms d'usuari i contrasenyes.
- **RF16:** Permet als usuaris iniciar sessió per accedir a les seves preferències personals.
- **RF17:** Desa automàticament les preferències de cada usuari.
- **RF18:** Emmagatzema les contrasenyes mitjançant hashes SHA256.

### 2.1.2 Requisits no funcionals

**Rendimiento:**

- **RNF1:** Temps de càrrega de la pàgina inferior a 2 segons
- **RNF2:** Animacions i interaccions suaus
- **RNF3:** Sense retard ni entrebancs en la reproducció d'àudio
- **RNF4:** Admet la reproducció simultània de diverses pistes d'àudio sense problemes de rendiment

**Usabilidad:**

- **RNF5:** Disseny responsiu, adaptable a ordinadors d'escriptori i dispositius mòbils
- **RNF6:** Interfície d'usuari intuïtiva, no cal tutorial
- **RNF7:** Localització al català
- **RNF8:** Compleix amb les directrius bàsiques de les WCAG, admet la navegació amb teclat
- **RNF9:** Proporciona etiquetes de text significatives o atributs aria-label per als botons d'icona
- **RNF10:** Retroalimentació visual clara (estat dels botons, canvis de volum, etc.)

**Tecnología**

- **RNF11:** Uso de tecnologías web modernas
- **RNF12:** Mantenibilidad y escalabilidad del código
- **RNF13:** Seguridad de tipos en TypeScript
- **RNF14:** Compatibilidad con navegadores (compatible con las últimas versiones de Chrome, Firefox, Safari y Edge)

## 2.2 Previsió de tasques d'investigació CBS

Abans de desenvolupar el projecte CBS, és necessari investigar i verificar les següents qüestions tècniques:

### 2.2.1 Prova de selecció de biblioteques d'àudio

**Pregunta:** Hauríem d'utilitzar Howler.js o l'API d'àudio web nativa?

**Contingut de la recerca:**

- Provar la compatibilitat del navegador de les dues solucions
- Comparar el rendiment de la reproducció de múltiples pistes
- Avaluar la dificultat de desenvolupament i la complexitat del codi
  **Resultat esperat:** Determinar la solució de processament d'àudio més adequada

### 2.2.2 Verificació de la tecnologia de mescla multipista

**Problema:** Com assegurar que 8 sons es reprodueixin simultàniament sense entrebancs?

**Contingut de la recerca:**

- Provar el rendiment de diferents formats d'àudio (MP3, OGG)
- Verificar la precisió del control de volum independent
- Mesurar l'ús de CPU i memòria
  **Resultat esperat:** Establir les millors pràctiques per a la reproducció de múltiples pistes

### 2.2.3 Recerca sobre les limitacions de la reproducció d'àudio mòbil

**Problema:** Quines limitacions tenen iOS i Android per a la reproducció d'àudio?

**Contingut de la recerca:**

- Provar les estratègies de reproducció automàtica dels navegadors mòbils
- Verificar la viabilitat de la reproducció en segon pla
- Comprovar la latència d'àudio en diferents dispositius
  **Resultat esperat:** Determinar el pla de compatibilitat per a dispositius mòbils

### 2.2.4 Comparació de solucions de gestió d'estat

**Pregunta:** Quina és més adequada: Zustand, Context+Reducer o Redux?

**Contingut de la recerca:**

- Tres solucions per aconseguir la mateixa funcionalitat
- Comparar la complexitat i la mantenibilitat del codi
- Provar les diferències de rendiment en les actualitzacions d'estat
  **Resultat esperat:** Seleccionar la solució de gestió d'estat més adequada

### 2.2.5 Estratègia d'optimització de fitxers d'àudio

**Pregunta:** Com equilibrar la qualitat de l'àudio i la mida del fitxer?

**Contingut de la recerca:**

- Provar les diferències de qualitat d'àudio a diferents taxes de bits
- Comparar la mida del fitxer i el temps de càrrega
- Avaluar l'impacte de la compressió en la qualitat de l'àudio
  **Resultat esperat:** Determinar la solució òptima de compressió d'àudio

### 2.2.6 Estudi de viabilitat de la reproducció fora de línia

**Pregunta:** És factible la funcionalitat de reproducció fora de línia?

**Contingut de la recerca:**

- Emmagatzematge en memòria cau de fitxers d'àudio per part del Service Worker de proves
- Avaluació dels requisits d'espai d'emmagatzematge
- Estudi de la viabilitat de les PWA (aplicacions web progressives)
  **Resultat esperat:** Avaluació de la viabilitat tècnica de la funcionalitat fora de línia

### 2.2.7 Estudi de viabilitat d'autenticació sense backend

**Pregunta:** És possible implementar un sistema d'autenticació completament frontal?

**Contingut de la recerca:**

- Emmagatzematge d'usuaris a localStorage amb clau silence_users_db
- Xifrat de contrasenyes amb SHA256 (hash irreversible)
- Restauració d'estat en iniciar l'aplicació
- Avaluació de límits d'emmagatzematge i rendiment
  **Resultat esperat:** Validar la viabilitat tècnica d'un sistema d'autenticació i persistència de preferències sense servidor.

## 2.3 Tecnologies

### 2.3.1 Comparativa de les tecnologies valorades

| Categoria Tecnològica | Opció A | Opció B | Raons de l'elecció i notes |
|---|---|---|---|
| Framework Frontend | React 19 + TypeScript | Vue 3 / Svelte | React té un ecosistema extens, bon suport comunitari, TypeScript ofereix seguretat de tipus, adequat per a aplicacions amb interaccions complexes. L'equip està familiaritzat amb React. |
| Llibreria de Processament d'Àudio | Howler.js | API Web Audio nativa | Howler.js proporciona una API senzilla, suport perfecte per a reproducció de múltiples pistes, control de volum, bona compatibilitat entre navegadors, alta eficiència de desenvolupament. |

| | | | |
|---|---|---|---|
| Gestió d'Estat | Zustand | React Context + useReducer / Redux | Zustand és lleuger i senzill, amb una API intuïtiva, bon rendiment, adequat per a les necessitats de gestió d'estat de l'aplicació de reproducció d'àudio. |
| Eina de Construcció | Create React App (CRA) | Vite | CRA és l'entorn oficial de React, configuració senzilla i estable, adequat per a iniciar projectes ràpidament. Vite és més ràpid però requereix més configuració. |
| Solució d'Estils UI | CSS Modules + Estils personalitzats | Tailwind CSS / Styled Components | CSS Modules ofereix CSS amb àmbit local, evita conflictes d'estils, mentre manté llibertat creativa total en el disseny. |
| Llibreria d'Icones | React Icons | SVG personalitzat / Icones de font | React Icons ofereix una àmplia selecció d'icones, suport per a Tree-shaking, càrrega sota demanda, redueix la mida del paquet. |
| Plataforma de Desplegament | GitHub Pages | Vercel / Netlify | GitHub Pages és gratuït i estable, integració perfecta amb GitHub, adequat per al desplegament inicial de projectes de codi obert. |
| Dades de l'usuari | localStorage | IndexedDB / Firebase | localStorage ofereix API simple i suficient capacitat (5-10 MB) per a desenes d'usuaris i preferències |
| Gestió de sessió | Zustand + localStorage | Redux Persist | Zustand permet sincronització senzilla amb localStorage mitjançant middleware personalitzat |
| Xifrat de contrasenyes | Crypto-JS (SHA256) | bcrypt / Argon2 | SHA256 proporciona hash irreversible, simulant pràctiques de seguretat reals en un entorn acadèmic |

### 2.3.2 Tecnologies escollides

**1. React 19 + TypeScript 4.9.5**

React ofereix un model de components ideal per a interfícies interactives i un ecosistema madur. TypeScript afegeix verificació estàtica de tipus, redueix errors en temps d'execució i millora la mantenibilitat del codi.

**2. Framer Motion 11.18.2**

Llibreria d'animació declarativa per a React, utilitzada en transicions de pàgina, modals, efectes de desplaçament, animacions del logo i el cursor personalitzat.

**3. Zustand 5.0.11**

Gestió d'estat lleugera amb API intuïtiva i bon rendiment. Respecte a Redux té menys codi boilerplate i és més adequat que Context per a estats complexos d'àudio.

**4. CSS Modules + Estils personalitzats**

Proporcionen àmbit local als estils, eviten conflictes globals i mantenen control creatiu total sobre el disseny fosc i immersiu, sense dependències externes.

**5. React Icons 4.12.0**

Àmplia selecció d'icones de múltiples conjunts, amb càrrega sota demanda (tree-shaking) per reduir la mida del paquet.

**6. Create React App + GitHub Pages**

CRA ofereix un entorn de desenvolupament React sense configuració (Babel, Webpack integrats). GitHub Pages proporciona allotjament gratuït per a llocs estàtics, amb integració directa amb el repositori.

**7. localStorage API**

API simple i suficient (5-10 MB) per persistir dades d'usuaris i preferències sense necessitat de backend.

**8. Crypto-JS (SHA256)**

Genera hashes irreversibles de les contrasenyes, evitant emmagatzemar-les en text pla i simulant pràctiques de seguretat reals en un entorn acadèmic.

**9. Howler.js 2.2.4**

Gestiona la reproducció simultània de múltiples pistes, control de volum independent, fades i compatibilitat entre navegadors, encapsulant la complexitat de la Web Audio API.

**10. @ffmpeg/ffmpeg 0.12.15**

Executa FFmpeg al navegador mitjançant WebAssembly per transcodificar la gravació WebM a MP4, utilitzada a la funció de compartir vídeo.

## 2.4 Estructura del projecte

```
CBS/  # Arrel del projecte
├── build/  # Directori de sortida de producció
│   └── static/  # (creat per 'npm run build')
│       ├── css/  # Fulls d'estil optimitzats i minimitzats
│       ├── js/  # Paquets JavaScript i code-splitting
│       └── media/  # Imatges i fitxers d'àudio processats
├── node_modules/  # Dependències (no es puja al repositori)
├── public/  # Recursos estàtics accessibles directament
│   ├── favicon.ico  # Icona del lloc
│   ├── index.html  # Plantilla HTML principal (arrel de React)
│   ├── logo192.png  # Icona petita per a PWA
│   ├── logo512.png  # Icona gran per a PWA i pantalla de càrrega
│   ├── manifest.json  # Configuració de l'Aplicació Web Progressiva
│   ├── robots.txt  # Directives per a motors de cerca
│   └── sw.js  # Service Worker (mode offline, només producció)
├── src/  # Codi font de l'aplicació
│   ├── assets/  # Recursos estàtics
│   │   └── images/  # Imatges de fons (pluja, mar, foc…) i logo
│   │       ├── rain-on-grasst.png
│   │       ├── waves.png
│   │       ├── bonfire.png
│   │       ├── wind.png
│   │       ├── bird.png
│   │       ├── cricket.png
│   │       ├── thunder.png
│   │       ├── woodcrack.png
│   │       └── logo-512x512.png
│   ├── sounds/  # Fitxers d'àudio amb llicència oberta (CC0/CC-BY)
│   │   ├── rain-on-grasst.mp3
│   │   ├── waves.mp3
│   │   ├── bonfire.mp3
│   │   ├── wind.mp3
│   │   ├── bird.mp3
│   │   ├── cricket.mp3
│   │   ├── thunder.mp3
│   │   └── woodcrack.mp3
│   ├── stores/  # Gestió de l'estat global amb Zustand
│   │   └── useSoundStore.ts  # Store únic: àudio, temporitzador,
│   │                         # autenticació, presets, preferències i
│   │                         # persistència a localStorage amb SHA256
│   ├── App.tsx  # Component principal: tota la UI integrada
│   │            # (targetes de so, hero, navbar, modals de
│   │            # login/share, illa dinàmica, cursor, etc.)
│   ├── App.css  # Full d'estils global (CSS vanilla amb
│   │            # variables, keyframes i responsive)
│   ├── index.tsx  # Punt d'entrada: renderitza <App /> i
│   │              # registra el Service Worker en producció
│   ├── react-app-env.d.ts  # Tipus per a react-scripts
│   └── declarations.d.ts  # Declaració de mòduls per a .mp3, .png, .css
└── Fitxers de configuració
    ├── package.json  # Nom, versió, dependències i scripts
    │                 # (start, build, test, predeploy, deploy)
    ├── package-lock.json  # Arbre exacte de versions de dependències
    ├── tsconfig.json  # Opcions del compilador TypeScript
    ├── .gitignore  # Fitxers i carpetes ignorats per Git
    └── README.md  # Documentació bàsica del projecte
```

## 2.5 Descripció dels components CBS

### 2.5.1 Component 1: Component arrel de l'aplicació (App.tsx)

**Descripció:** Component principal que inicialitza l'estat global amb Zustand i orquestra tota la interfície: barra de navegació, hero, galeria de sons, panell de control flotant i peu de pàgina.

**Funcions:**

- Gestiona l'estat global (àudio, autenticació, preferències) via Zustand
- Defineix l'estructura i el disseny general de l'aplicació
- Aplica el tema fosc i els estils visuals mitjançant CSS personalitzat i animacions amb Framer Motion
- Gestiona la interacció amb el logotip (efecte d'ona en fer clic i animació de respiració)

**Característiques tècniques:**

- Components funcionals amb React Hooks
- Animacions CSS avançades (keyframes, pseudo-elements) per als efectes del logotip
- Integració amb Framer Motion per a transicions de pàgina, modals i el cursor personalitzat

### 2.5.2 Component 2: Component de targeta de so (SoundCard.tsx)

**Descripció:** Representa un so ambiental independent, és la unitat central d'interacció de l'aplicació.

**Funcions:**

- Mostra la icona, nom i categoria del so
- Proporciona control de botó de reproducció/pausa
- Proporciona control lliscant de volum (0-100%)
- Mostra el volum actual i l'estat de reproducció
- Retorn visual (ressaltat en estat actiu)

### 2.5.3 Component 3: Illa dinàmica (Dynamic Island)

**Descripció:** Barra de control flotant integrada directament a App.tsx, situada a la part inferior de la pantalla. Agrupa els controls globals de manera compacta i sempre accessible.

**Funcions:**

- Botó global de reproducció/pausa
- Indicador d'estat (actiu / en espera)
- Visualització del compte enrere del temporitzador
- Selector de temps del temporitzador
- Control lliscant de volum global

**Subcomponents:**
- Botó mestre de reproducció/pausa
- Indicador d'estat i temporitzador
- Selector de durada del temporitzador
- Control lliscant de volum global

### 2.5.4 Component 4: Sistema d'autenticació d'usuaris

**Descripció:** Conjunt de components i lògica que permeten el registre, inici de sessió i gestió de perfils d'usuari, amb persistència a localStorage.

**Funcions:**
- Gestiona el registre de nous usuaris amb hash SHA256 de contrasenya
- Controla l'inici de sessió i verificació d'identitat
- Manté la sessió de l'usuari actiu a localStorage
- Restaura les preferències de l'usuari en iniciar l'aplicació
- Proporciona tancament de sessió

**Subcomponents:**
- `LoginForm`: Formulari d'inici de sessió
- `RegisterForm`: Formulari de registre
- `UserMenu`: Menú amb informació de l'usuari i opció de tancar sessió

**Característiques tècniques:**
- Zustand com a única font de veritat per a l'estat d'autenticació
- localStorage per a persistència d'usuaris i sessió activa
- SHA256 per a hash de contrasenyes (emmagatzematge segur i irreversible)
- Rehydration automàtica en iniciar l'aplicació

### 2.5.5 Component 5: Modal de compartir vídeo (ShareModal)

**Descripció:** Modal que permet gravar 15 segons de la mescla activa amb visualització d'espectre d'àudio sobre la imatge de fons del so principal, i transcodificar el resultat a MP4 per descarregar o compartir.

**Funcions:**
- Gravació de pantalla amb àudio mitjançant MediaRecorder i Canvas
- Transcodificació de WebM a MP4 amb FFmpeg (WebAssembly)
- Descàrrega del vídeo generat
- Compartició nativa (navigator.share)

**Característiques tècniques:**

- Utilitza Howler.ctx per connectar el flux d'àudio al MediaStream
- AnalyserNode per dibuixar l'espectre d'àudio en temps real
- FFmpeg carregat dinàmicament des de CDN

### 2.5.6 Component 6: Cursor personalitzat (CustomCursor)

Descripció: Substitueix el cursor del sistema per un cercle blanc amb efecte de difuminat (goo) i un segon cercle de seguiment amb molla. S'amaga automàticament en dispositius tàctils.

Funcions:
- Seguiment del ratolí amb Framer Motion (useMotionValue + useSpring)
- Detecció d'elements interactius per canviar l'escala
- Filtre SVG per a l'efecte visual
- Compatible amb mode Zen (desapareix durant inactivitat)

### 2.5.7 Component 7: Superposició Zen (Zen Overlay)

Descripció: Pantalla negra amb frase meditativa que apareix després de 30 segons d'inactivitat de l'usuari.

Funcions:
- Detecció d'inactivitat (manca de moviments de ratolí, clics o tecles)
- Selecció aleatòria de frases meditatives
- Transició suau d'opacitat (2 segons)

### 2.5.8 Component 8: Selector d'idioma

Descripció: Botó integrat a la barra de navegació que alterna entre català i espanyol.

Funcions:
- Canvi instantani d'idioma
- Persistència de la preferència a localStorage
- Actualització de tots els textos de la interfície

## 2.6 Definició de les tasques CBS

### 2.6.1 Prova 1: Prova de rendiment de reproducció de múltiples pistes

**Descripció:** Provar el rendiment de Howler.js en la reproducció simultània de múltiples pistes.

**Objectius de la prova:**

- Verificar l'estabilitat de la reproducció simultània de 8 pistes
- Provar la qualitat d'àudio amb control independent del volum
- Mesurar l'ús de CPU i memòria

**Mètode de prova:**

1. Crear una pàgina de prova, carregar 8 fitxers d'àudio diferents
2. Reproduir totes les pistes simultàniament
3. Ajustar el volum de cada pista independentment
4. Monitoritzar el panell de rendiment del navegador
5. Provar en diferents dispositius (PC, telèfon mòbil)

**Resultats esperats:**

- Confirmar que Howler.js pot gestionar establement la reproducció de múltiples pistes
- Determinar colls d'ampolla de rendiment i punts d'optimització
- Establir les millors pràctiques per a fitxers d'àudio

### 2.6.2 Prova 2: Prova de compatibilitat amb dispositius mòbils

**Descripció:** Provar la compatibilitat de reproducció d'àudio en dispositius mòbils.

**Objectius de la prova:**

- Verificar la reproducció d'àudio en dispositius iOS i Android
- Provar la responsivitat de les interaccions tàctils
- Verificar l'estratègia de reproducció automàtica

**Mètode de prova:**

1. Provar l'aplicació en dispositius mòbils reals
2. Provar diferents navegadors mòbils (Safari, Chrome mòbil)
3. Verificar el maneig d'esdeveniments tàctils
4. Provar el comportament de reproducció d'àudio en segon pla

**Resultats esperats:**

- Establir les millors pràctiques per a dispositius mòbils
- Determinar els requisits necessaris d'interacció de l'usuari
- Optimitzar l'experiència d'interacció tàctil

### 2.6.3 Prova 3: Comparativa de solucions de gestió d'estat

**Descripció:** Comparar el rendiment i l'experiència de desenvolupament de diferents solucions de gestió d'estat.

**Objectius de la prova:**
- Comparar el rendiment de Zustand, Context+Reducer i Redux
- Avaluar la complexitat de desenvolupament i la mantenibilitat del codi
- Provar el suport de TypeScript

**Mètode de prova:**
1. Implementar la mateixa gestió d'estat de reproducció d'àudio amb cada solució
2. Mesurar el rendiment de l'actualització d'estat
3. Avaluar la complexitat i llegibilitat del codi

**Resultats esperats:**
- Seleccionar la solució de gestió d'estat més adequada (Zustand)
- Establir les millors pràctiques de gestió d'estat

### 2.6.4 Prova 4: Sistema d'autenticació i persistència d'usuaris

**Descripció:** Provar el sistema d'usuaris amb localStorage: registre, inici de sessió i persistència de preferències individuals.

**Objectius de la prova:**
- Registre correcte sense duplicats
- Validació de credencials amb SHA256
- Preferències independents per usuari

**Mètode de prova:**
- Registre usuari nou → desat amb hash
- Login incorrecte → accés denegat
- Recarregar pàgina → estats es mantene

**Resultats:**
- Distinció correcta entre usuaris
- Contrasenyes amb hash (no text pla)
- Preferències individuals persistents i sessió mantinguda després de reiniciar navegador

### 2.6.5 Prova 5: Prova de gravació i transcodificació de vídeo

**Descripció:** Verificar el funcionament de la funcionalitat de compartir vídeo en diferents navegadors i dispositius.

**Objectius de la prova:**

- Verificar que MediaRecorder captura correctament el canvas i l'àudio
- Verificar la compatibilitat del còdec WebM (VP9 / VP8)
- Provar la transcodificació a MP4 amb FFmpeg
- Verificar la descàrrega i la compartició nativa

**Mètode de prova:**

- Activar diversos sons i obrir el modal de compartir
- Iniciar la gravació i verificar que l'espectre es dibuixa
- Esperar que finalitzi la transcodificació
- Reproduir el vídeo generat
- Provar la descàrrega i, si escau, la compartició nativa

**Resultats:**

- Vídeo MP4 generat correctament amb àudio i imatge
- Funcionament estable a Chrome, Firefox i Edge

## 2.7 Definició de les funcionalitats CBS

### 2.7.1 Funcionalitat 1: Reproducció de barreja de múltiples pistes

**Descripció:** Permet als usuaris reproduir múltiples sons ambientals simultàniament i controlar independentment el volum de cada pista.

**Procés d'implementació:**

1. Utilitzar Howler.js per carregar múltiples fitxers d'àudio
2. Crear una instància de Howl independent per a cada pista
3. Implementar control independent de reproducció/pausa
4. Implementar control independent de lliscadors de volum
5. Gestionar la sincronització d'estat entre pistes

**Estat:** Totalment implementat

### 2.7.2 Funcionalitat 2: Funció de temporitzador

**Descripció:** Establir un temporitzador que atura automàticament tota la reproducció de sons quan finalitza el temps.

**Procés d'implementació:**

1. Crear component de UI del temporitzador
2. Implementar lògica de compte enrere
3. Cridar al mètode d'aturada de reproducció quan finalitza el temporitzador
4. Proporcionar retorn de l'estat del temporitzador

**Estat:** Totalment implementat

### 2.7.3 Funcionalitat 3: Control global

**Descripció:** Proporciona control global de reproducció i control de volum, a més d'elements visuals que reforcen la identitat de l'aplicació. Aquesta funcionalitat inclou el botó global de play/pausa, el lliscador de volum màster.

**Procés d'implementació:**

1. Implementar botó global de reproducció/pausa
2. Implementar lliscador global de volum
3. Afegir efectes visuals al logotip amb animacions CSS avançades

**Estat:** Totalment implementat

### 2.7.4 Funcionalitat 4: Modes predefinits i personalització

**Descripció:** Inclou 5 modes predefinits (Focus, Relax, Sleep, Meditate, Reading), un botó de mescla aleatòria (L'IGNOT) i 3 slots per desar mescles personalitzades.

**Procés d'implementació:**

1. Aplicar configuracions predefinides de sons i volums per a cada mode
2. Generar combinacions aleatòries de 2-4 sons amb volums entre 20% i 80%
3. Desar, carregar i esborrar fins a 3 mescles personalitzades per usuari

**Estat:** Totalment implementat

### 2.7.5 Característica 5: Canvi d'idioma (català / espanyol)

**Descripción:** Proporciona una interfície completa en català i espanyol, amb alternança instantània mitjançant un botó a la barra de navegació.

**Proceso de implementación:**

1. Tots els textos de la interfície es defineixen en un diccionari amb claus comunes.
2. L'estat de l'idioma es guarda a l'estat global (Zustand) i a localStorage.
3. El canvi d'idioma actualitza immediatament tots els textos visibles.

**Estado:** Totalment implementat

### 2.7.6 Funcionalitat 6: Persistència d'estat

**Descripció:** Guardar les configuracions de combinacions de sons dels usuaris, associades a cada compte individual, utilitzant localStorage.

**Procés d'implementació:**

1. Utilitzar localStorage amb les claus `silence_users_db` (base de dades d'usuaris) i silence_curr_user (sessió activa).
2. En registrar-se, es crea un objecte User amb preferències per defecte.
3. Cada vegada que l'usuari ajusta un volum o canvia l'estat d'un so, es crida una funció que actualitza l'usuari a localStorage.
4. En iniciar sessió, es carreguen les preferències de l'usuari i s'apliquen a l'estat global.

**Estat:** Totalment implementat

### 2.7.7 Funcionalitat 7: Sistema de compte d'usuari

**Descripció:** Permetre als usuaris registrar-se, iniciar sessió i gestionar les seves preferències personals.

**Procés d'implementació:**

1. Registre: Comprova duplicats, genera hash SHA256 de la contrasenya, crea usuari amb preferències per defecte i desa a localStorage
2. Inici de sessió: Cerca l'usuari, compara hashes i inicia sessió si coincideixen
3. Rehydration: En iniciar l'aplicació, restaura volums i sons actius de l'usuari guardat

**Estat:** Totalment implementat

### 2.7.8 Funcionalitat 8: Gravació i compartició

**Descripció:** Permet gravar 15 segons de la mescla activa amb visualització d'espectre d'àudio sobre la imatge de fons. El vídeo es transcodifica a MP4 i es pot descarregar o compartir.

**Procés d'implementació:**

1. Capturar el flux del canvas amb MediaRecorder i l'àudio amb Howler.ctx
2. Transcodificar el WebM generat a MP4 amb @ffmpeg/ffmpeg (WebAssembly)
3. Oferir descàrrega del fitxer i compartició nativa (navigator.share)

**Estat:** Totalment implementat

### 2.7.9 Funcionalitat 9: Experiència visual immersiva

**Descripció:** Cursor personalitzat amb efecte de seguiment i mode Zen que mostra frases meditatives després de 30 segons d'inactivitat.

**Procés d'implementació:**

1. Substituir el cursor per cercles animats amb Framer Motion i filtre SVG
2. Detectar inactivitat i mostrar frases aleatòries amb transició suau
3. Amagar automàticament el cursor personalitzat en dispositius tàctils

**Estat:** Totalment implementat

# 3. Conclusions

## 3.1 Conclusions generals del projecte

El desenvolupament de "Custom Background Sound" ha estat una experiència formativa que ens ha permès aplicar els coneixements del cicle formatiu en un projecte real i funcional. Hem consolidat competències en arquitectura front-end amb React i TypeScript, gestió d'estat amb Zustand, processament d'àudio amb Howler.js i animacions amb Framer Motion. La integració de FFmpeg mitjançant WebAssembly ha representat un repte tècnic especialment enriquidor, ja que hem obert una finestra al processament multimèdia avançat al navegador.

A nivell professional, hem simulat un cicle de desenvolupament complet: anàlisi de requisits, comparativa tecnològica, implementació iterativa amb Scrum, proves de compatibilitat i documentació. L'ús de GitHub Projects ens ha facilitat la gestió de tasques i la traçabilitat del codi. El resultat demostra que és possible crear una aplicació web completa amb autenticació, persistència i funcionalitats multimèdia avançades sense dependre d'un backend, utilitzant exclusivament tecnologies de front-end i emmagatzematge local.

## 3.2 Consecució dels objectius

A continuació valorem el grau d'assoliment de cadascun dels objectius específics:

- Proporcionar almenys 5 sons ambientals: Assolit i superat. Hem inclòs 8 sons (pluja, grill, onades, tro, foc, vent, ocells i fusta cruixint).
- Implementar un mesclador amb volum independent i temporitzador: Assolit. Cada so té control lliscant (0-100%) i el temporitzador ofereix intervals d'1 a 60 minuts amb aturada automàtica.
- Crear modes preestablerts i personalització: Assolit i superat. Hem desenvolupat 5 modes predefinits, un botó de mescla aleatòria (L'IGNOT) i 3 slots per a presets personalitzats.
- Persistència amb localStorage i autenticació: Assolit. Les preferències es guarden amb claus específiques per usuari i les contrasenyes es xifren amb SHA256.
- Explorar PWA i gravació de vídeo: Assolit. Hem configurat el Service Worker per a mode offline i implementat la gravació de 15 segons amb espectre d'àudio, transcodificació a MP4 i compartició.

En conjunt, tots els objectius s'han assolit satisfactòriament i en la majoria de casos hem superat les expectatives inicials.

## 3.3 Valoració de la metodologia i planificació

La metodologia àgil amb Scrum ha estat encertada per a aquest projecte. Els cicles curts d'iteració ens han permès desenvolupar de manera progressiva i adaptar-nos als resultats de les proves intermèdies. GitHub Projects ens ha facilitat el seguiment diari i la vinculació de tasques amb commits i pull requests.

La planificació inicial amb Diagrama de Gantt ens va proporcionar una visió global dels terminis, però cal assenyalar algunes desviacions. La Fase 2 va requerir més hores de les estimades perquè vam haver d'integrar Framer Motion i FFmpeg, tecnologies no previstes inicialment. La Fase 3 es va ampliar amb la funcionalitat de gravació de vídeo, que va enriquir el producte final. Durant les proves en dispositius mòbils vam detectar limitacions en la reproducció automàtica d'àudio, la qual cosa ens va obligar a ajustar la interfície per complir amb les polítiques dels navegadors.

En general, la flexibilitat de la metodologia àgil ens ha permès incorporar millores sense comprometre el calendari global, demostrant ser l'elecció adequada per a un projecte amb requisits canviants i un equip reduït.

## 3.4 Visió de futur

El projecte estableix una base sòlida sobre la qual es poden desenvolupar diverses millores futures:

- **PWA completa:** Implementar l'emmagatzematge en memòria cau de tots els fitxers d'àudio per a reproducció totalment offline.
- **Backend al núvol:** Substituir localStorage per Firebase o Supabase per permetre la sincronització entre dispositius i la compartició real de mescles.
- **Ampliació de la biblioteca:** Afegir noves categories de sons i permetre la càrrega de fitxers propis.
- **Mescla avançada:** Incorporar equalitzador, filtres de freqüència i control de panoràmica estèreo.
- **Accessibilitat:** Assolir el nivell WCAG 2.1 AA amb suport complet per a lectors de pantalla.
- **Internacionalització:** Ampliar els idiomes disponibles (anglès, francès, alemany).
- **Aplicació nativa:** Explorar Tauri o React Native per a una experiència mòbil optimitzada.

# 4. Glossari

Definició dels termes i acrònims més rellevants utilitzats dins la Memòria.

| Terme | Definició |
|---|---|
| AnalyserNode | Node de la Web Audio API que permet analitzar el senyal d'àudio en freqüència i temps real. S'utilitza al ShareModal per dibuixar l'espectre d'àudio sobre el canvas. |
| Backlog | Llista prioritzada de totes les funcionalitats, millores i correccions pendents d'un projecte Scrum. L'equip selecciona tasques del backlog per a cada sprint. |
| Canvas API | API del navegador que permet renderitzar gràfics 2D mitjançant JavaScript. Utilitzada per dibuixar l'espectre d'àudio en temps real al component ShareModal. |
| CSS Modules | Sistema d'estils que ofereix àmbit local per defecte a les classes CSS, evitant conflictes de noms i permetent un control total del disseny sense dependències externes. |
| FFmpeg | Eina de codi obert per a processament de vídeo i àudio. En aquest projecte s'executa al navegador mitjançant WebAssembly per transcodificar gravacions WebM a MP4. |
| Framer Motion | Llibreria d'animació declarativa per a React. Permet crear transicions, gestos i efectes visuals de manera senzilla. En aquest projecte s'utilitza per a modals, el cursor personalitzat i les animacions de desplaçament. |
| Hash | Funció matemàtica unidireccional que transforma un text d'entrada en una cadena de longitud fixa. És irreversible, és a dir, no es pot obtenir el text original a partir del hash. S'utilitza per verificar contrasenyes sense emmagatzemar-les en text pla. |
| Howler.js | Llibreria JavaScript per a la gestió d'àudio al navegador. Encapsula la complexitat de la Web Audio API i ofereix control de volum, reproducció multicanal i fades. |
| localStorage | API del navegador que permet emmagatzemar dades en format clau-valor de forma persistent. Les dades romanen disponibles fins i tot després de tancar el navegador i no tenen data de caducitat. |

| | |
|---|---|
| MediaRecorder | API del navegador que permet gravar fluxos multimèdia (àudio i vídeo). Utilitzada per capturar el canvas i l'àudio de la mescla per a la funció de compartir. |
| MP4 / WebM | Formats contenidor de vídeo. WebM és el format generat per MediaRecorder (còdec VP8/VP9), mentre que MP4 és el format final obtingut després de la transcodificació amb FFmpeg, àmpliament compatible amb tots els dispositius. |
| PWA (Progressive Web App) | Aplicació web que utilitza Service Workers i un manifest per oferir funcionalitats pròpies d'aplicacions natives, com el mode fora de línia i la instal·lació al dispositiu. |
| Rehydration | Procés de restaurar l'estat d'una aplicació a partir de dades persistents (com les de localStorage) quan l'aplicació s'inicia. En aquest projecte, significa carregar les preferències de l'usuari desades. |
| Scrum | Metodologia àgil de desenvolupament de software basada en cicles curts d'iteració (sprints), reunions diàries i millora contínua. Utilitzada per gestionar el desenvolupament del projecte. |
| Service Worker | Script que s'executa en segon pla al navegador, independent de la pàgina web. Permet interceptar peticions de xarxa, gestionar la memòria cau i habilitar el funcionament offline de la PWA. |
| SHA256 | Algorisme de hash segur que produeix una sortida de 256 bits (32 bytes). Pertany a la família SHA-2 i és àmpliament utilitzat en aplicacions de seguretat per la seva robustesa. |
| Single Source of Truth | Principi de disseny de software on tota la informació de l'aplicació es gestiona des d'un únic lloc central (en aquest cas, el store de Zustand), evitant duplicitats i inconsistències. |
| Tree-shaking | Tècnica d'optimització del bundler que elimina automàticament el codi no utilitzat del paquet final. Permet importar només les icones de React Icons que realment s'utilitzen, reduint la mida del bundle. |
| TypeScript | Superset de JavaScript que afegeix tipat estàtic opcional. Permet detectar errors en temps de compilació, millora l'autocompletat en l'editor i facilita el manteniment del codi en projectes grans. |

| | |
|---|---|
| Web Audio API | API nativa del navegador per al processament, síntesi i anàlisi d'àudio. Howler.js l'utilitza internament per gestionar la reproducció multicanal, i s'accedeix directament a través de Howler.ctx per connectar el flux d'àudio al MediaRecorder. |
| WebAssembly | Format binari que permet executar codi d'alt rendiment al navegador. Utilitzat per carregar FFmpeg i realitzar la transcodificació de vídeo sense necessitat de servidor. |
| Zustand | Llibreria lleugera de gestió d'estat per a React. Proporciona una API senzilla i rendiment elevat sense el codi boilerplate característic de Redux. |

# 5. Bibliografia

[1] Freesound.org. *Freesound - Collaborative database of creative-commons licensed sound for musicians and sound lovers*. [En línia]. Disponible a: https://freesound.org [Consulta: 15 de gener de 2026].

[2] Audacity Team. *Audacity: Free, open source, cross-platform audio software*. [En línia]. Disponible a: https://www.audacityteam.org [Consulta: 7 de novembre de 2025].

[3] FFmpeg Project. *FFmpeg: A complete, cross-platform solution to record, convert and stream audio and video*. [En línia]. Disponible a: https://ffmpeg.org [Consulta: 19 de novembre de 2025].

[4] Meta Open Source. *React: The library for web and native user interfaces*. [En línia]. Disponible a: https://reactjs.org [Consulta: 7 de novembre de 2025].

[5] Golding, J. *Howler.js: Audio library for the modern web*. [En línia]. Disponible a: https://howlerjs.com [Consulta: 19 de novembre de 2025].

[6] GitHub, Inc. *GitHub Pages: Websites for you and your projects*. [En línia]. Disponible a: https://pages.github.com [Consulta: 12 de desembre de 2025].

[7] React Icons. *React Icons: Include popular icons in your React projects easily*. [En línia]. Disponible a: https://react-icons.github.io/react-icons [Consulta: 12 de gener de 2026].

[8] Zustand. *Zustand: Bear necessities for state management in React*. [En línia]. Disponible a: https://zustand-demo.pmnd.rs [Consulta: 3 de desembre de 2025].

[9] Microsoft. *TypeScript: JavaScript with syntax for types*. [En línia]. Disponible a: https://www.typescriptlang.org [Consulta: 7 de novembre de 2025].

[10] Mozilla Developer Network. *Web Audio API*. [En línia]. Disponible a: https://developer.mozilla.org/es/docs/Web/API/Web_Audio_API [Consulta: 7 de novembre de 2025].

[11] Scrum.org. *What is Scrum?*. [En línia]. Disponible a: https://www.scrum.org/resources/what-is-scrum [Consulta: 24 d'octubre de 2025].

[12] GitHub, Inc. *GitHub Projects: Plan and track your work*. [En línia]. Disponible a: https://github.com/features/issues [Consulta: 12 de novembre de 2025].

[13] Mozilla Developer Network. *Window: localStorage property*. [En línia]. Disponible a: https://developer.mozilla.org/es/docs/Web/API/Window/localStorage [Consulta: 19 de febrer de 2026].

[14] amether. *Bonfire (raw recording)*. [En línia]. Freesound.org, 24 de maig de 2013. Disponible a: https://freesound.org/people/amether/sounds/189237/ [Consulta: 6 de febrer de 2026].

[15] OWASP Foundation. *Password Storage Cheat Sheet*. [En línia]. Disponible a: https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html [Consulta: 19 de febrer de 2026].

[16] daveincamas. *20061111Bucket1-FX1.1.wav*. [En línia]. Freesound.org, 18 de novembre de 2006. Disponible a: https://freesound.org/people/daveincamas/sounds/25932/ [Consulta: 6 de febrer de 2026].

[17] mmiron. *waves_hit_shore_barcelona_.wav*. [En línia]. Freesound.org, 4 d'octubre de 2011. Disponible a: https://freesound.org/people/mmiron/sounds/130432/ [Consulta: 6 de febrer de 2026].

[18] klankbeeld. *Rain in forest with light wind, summer, recording 2*. [En línia]. Freesound.org, 9 d'octubre de 2023. Disponible a: https://freesound.org/people/klankbeeld/sounds/810338/ [Consulta: 6 de febrer de 2026].

[19] Framer. *Framer Motion: Production-ready animations for React*. [En línia]. Disponible a: https://www.framer.com/motion/ [Consulta: 3 de març de 2026].

[20] brix-io. *crypto-js: JavaScript library of crypto standards*. [En línia]. Disponible a: https://github.com/brix/crypto-js [Consulta: 17 de febrer de 2026].

[21] FFmpegWebAssembly. *@ffmpeg/ffmpeg: FFmpeg for browser, powered by WebAssembly*. [En línia]. Disponible a: https://github.com/ffmpegwasm/ffmpeg.wasm [Consulta: 25 de març de 2026].

[22] Meta Open Source. *Create React App: Set up a modern web app by running one command*. [En línia]. Disponible a: https://create-react-app.dev [Consulta: 11 de febrer de 2026].

[23] Mozilla Developer Network. *MediaRecorder API*. [En línia]. Disponible a: https://developer.mozilla.org/es/docs/Web/API/MediaRecorder [Consulta: 8 d'abril de 2026].

[24] Mozilla Developer Network. *Canvas API*. [En línia]. Disponible a: https://developer.mozilla.org/es/docs/Web/API/Canvas_API [Consulta: 22 de març de 2026].

[25] Mozilla Developer Network. *Service Worker API*. [En línia]. Disponible a: https://developer.mozilla.org/es/docs/Web/API/Service_Worker_API [Consulta: 14 d'abril de 2026].

[26] web.dev. *Progressive Web Apps*. [En línia]. Google Developers. Disponible a: https://web.dev/progressive-web-apps/ [Consulta: 29 de març de 2026].