MiniTune Product Requirements Document

Document control

- Product: MiniTune
- Document type: Product Requirements Document
- Version: 1.0
- Audience: portfolio, hiring managers, stakeholders
- Date: March 2026

1. Executive summary

MiniTune is a macOS menu-bar mini player designed to reduce friction when controlling music across browser and app contexts. It allows users to pause, play, skip, resume, and browse content without hunting for a Safari tab, bringing Apple Music to the foreground, or re-opening a full player from the Dock.

The product solves a simple but frequent problem: users often know they want to pause or skip audio immediately, but the current playback source is buried inside a browser tab or separate music application. MiniTune brings those core controls into a lightweight menu-bar surface with a clean mini-player UI, artwork, progress, favourites, playlists, and fast source switching.

Its core value proposition is not just "a mini player". The unique selling point is a unified menu-bar control layer across YouTube in Safari and Apple Music, with a familiar mini-player interaction model and a lower-friction user experience than switching back to full applications. In the current codebase, the platform also extends to Navidrome for self-hosted playback, but the primary end-user narrative remains browser plus Apple Music convenience.

2. Problem statement

Users listening to music while working on macOS regularly face avoidable context switching:

- They need to find the correct Safari tab to pause or skip a YouTube track.
- They need to bring Music.app forward, or reopen it from the Dock, to change a song.
- They lose focus when moving away from the task they were doing just to perform a simple playback action.
- They often want to switch to a different song quickly, but the current player UI is hidden behind windows, tabs, or a full app.

This creates friction for a task that should take one click.

Core pain point

The user is unable to reliably play, pause, skip, or move to the next song without searching for tabs or re-opening their music player. This adds unnecessary effort to a simple action and breaks flow.

Opportunity

A persistent menu-bar control surface can remove that friction by making playback controls, now-playing context, and quick content browsing available from anywhere in macOS.

3. Product vision

MiniTune should feel like the fastest way to control personal audio on macOS without leaving the current task.

The product should:

- stay lightweight and always available
- minimise on-screen noise
- support the most common playback actions immediately
- provide a familiar mini-player mental model
- reduce the need to search for windows, tabs, or full players

4. Product positioning and USP

Positioning

MiniTune is a compact macOS menu-bar player for users who consume music while working and want immediate access to playback controls without full-app interaction.

Unique selling proposition

Unlike a traditional mini player tied to a single music service, MiniTune provides a unified, lightweight control experience across:

- YouTube playback inside Safari
- Apple Music playback
- optional self-hosted music playback via Navidrome in the current codebase

This makes MiniTune closer to a universal control surface than a single-service player.

Competitive angle

The product is similar to a native music mini player in familiarity, but differentiated by:

- cross-source control
- menu-bar availability
- reduced context switching
- quick playback control without reopening the source player

5. Target users

Primary persona: focused macOS worker

- Works with multiple windows and browser tabs open
- Listens to YouTube or Apple Music during work
- Wants quick playback control without breaking concentration

Secondary persona: casual productivity listener

- Uses a laptop for study, admin, coding, or creative work
- Frequently needs to pause or skip tracks during calls, notifications, or focus changes

Tertiary persona: self-hosted audio user

- Uses a personal Navidrome server
- Wants the same lightweight control experience without a full streaming client

6. Jobs to be done

- When I am listening to YouTube while working, I want to pause or skip without finding the correct tab.
- When I am using Apple Music, I want to control playback from the menu bar instead of opening the app.
- When I want a different song, I want to access library, favourites, or playlists quickly from the same compact UI.
- When I glance at the player, I want immediate clarity on what is playing and what action I can take.

7. Product goals

Goal 1: Reduce context switching

Users should be able to control playback from the menu bar in one interaction.

Goal 2: Deliver a cleaner mini-player experience

The UI should remain minimal, readable, and fast, without unnecessary status noise.

Goal 3: Support common playback scenarios end to end

Users should be able to play, pause, skip, resume, and choose different tracks from a compact surface.

Goal 4: Unify cross-source playback behaviour

The product should make YouTube and Apple Music feel coherent from the user’s point of view.

8. Success metrics

Suggested product metrics for this PRD:

- Reduction in time to control playback from "search and switch" to under 2 seconds
- High success rate for pause/play/skip actions on first attempt
- Low frequency of "tab not found" or source mismatch errors
- Improved perceived usability of playback control during focused work
- Repeated usage of menu-bar controls rather than reopening source apps

9. Scope

In scope

- macOS menu-bar application
- Floating mini-player panel
- Play, pause, previous, next, resume controls
- Now-playing state with title, artist/channel, artwork, and progress
- Mode switching between supported playback sources
- Library, favourites, and playlists views
- Artwork and metadata retrieval with local caching
- Settings and setup experience
- YouTube playlists beta path through backend-assisted OAuth flow

Current source scope in code

- YouTube in Safari
- Apple Music
- Navidrome

Out of scope

- Spotify delivery in the current PRD release scope
- Mobile apps
- Social, collaborative, or recommendation features
- Full replacement of native streaming clients
- Enterprise admin or team management features

10. Current product capabilities

10.1 Menu-bar and mini-player UX

- MiniTune runs as a menu-bar app
- Offers a floating mini-player panel
- Displays artwork, now-playing text, playback controls, and progress

10.2 YouTube mode

- Detects and controls YouTube playback in Safari
- Supports menu-bar transport actions
- Uses source-specific state and artwork handling

10.3 Music mode

- Controls Apple Music playback
- Supports library and playlist interactions

- Uses MusicKit and fallback artwork paths where available

10.4 Library, favourites, and playlists

- Tracks can be browsed from compact UI surfaces
- Favourites and recent history are stored locally
- Playlist browsing exists in the current product surface

10.5 Settings and onboarding

- Playback mode and artwork settings
- Setup and permissions guidance
- Cache clearing and advanced settings in implemented areas

11. Functional requirements

FR1. Immediate playback control

The product must let users pause, play, skip to next, and go to previous track without opening the source player UI.

FR2. Unified now-playing visibility

The product must present the current title, artist/channel, artwork, and playback state in the menu-bar interface.

FR3. Quick source-aware access

The product must support source switching and route actions to the correct playback source.

FR4. Compact browsing experience

The product must let users start a different track from library, favourites, or playlists without opening a full player app where supported.

FR5. Artwork and metadata enhancement

The product must load artwork asynchronously and present a smooth placeholder while metadata resolves.

FR6. Minimal UI

The product must avoid unnecessary text and preserve a clean mini-player feel.

12. Non-functional requirements

Performance

- Artwork and network-backed operations must not block the main thread.
- The UI should remain responsive during polling and metadata updates.

Reliability

- Playback control should reflect actual state consistently.
- Source switching should not cause duplicate or conflicting actions.

Privacy

- Playback state, favourites, recents, and caches remain local-first.
- Sensitive values should be stored securely.

Security

- No secrets embedded in the app bundle
- Backend-assisted OAuth only where required

Usability

- The product should work as a quick-glance and quick-action tool
- The visual model should stay familiar to users who know mini players already

13. UX principles

- One-click access to the most common playback actions
- Minimal copy, maximal clarity
- Compact but useful information density
- Familiar controls, lower friction
- Visual calm over feature clutter

14. Key user journeys

Journey 1: Pause YouTube immediately

- User is working in another app while YouTube is playing in Safari.
- User opens MiniTune from the menu bar.
- User clicks pause.
- Playback stops without the user searching for the Safari tab.

Journey 2: Skip Apple Music track without opening Music.app

- User is listening to Apple Music in the background.
- User uses MiniTune controls from the menu bar or floating panel.
- The next track starts and the UI updates.

Journey 3: Choose a different track quickly

- User opens MiniTune.
- User navigates to library, favourites, or playlists.

- User selects a track.
- Playback changes from the compact UI.

15. Constraints and assumptions

Constraints

- Product behaviour depends on macOS platform permissions and source-specific integration paths.
- Some playback sources have different technical limitations and failure states.

Assumptions

- Primary value comes from reduced context switching, not deep library management.
- Users prefer a compact control surface to opening a full player for simple actions.

16. Risks

- Browser/source detection may occasionally fail and reduce trust.
- Permission or authorisation failures can interrupt the experience.
- Cross-source consistency is harder than single-service mini-player behaviour.

17. Roadmap framing

Now

- Stabilise current core playback experience
- Improve reliability of play, pause, next, previous, and track selection
- Preserve a minimal and polished mini-player UI

Next

- Extend reliability and setup simplicity further
- Mature beta capability where already scoped
- Keep Spotify in backlog until core stability and product direction are stronger

18. Why this product matters

MiniTune addresses a small but high-frequency frustration. The product value is not in replacing full media players. It is in making playback control immediate, clean, and available from anywhere on macOS.

That is the core user outcome:

less searching, less switching, faster control, cleaner experience.