# Leep Audio — Full MVP Backend Development Plan

Timeline: 3 Weeks | Team Size: 5 Developers

## 🎯 MVP Goal

Artist uploads → Fan or Producer remixes → Artist gains exposure → Both engage again.

## 👥 Team Structure & Responsibilities

|  |  |  |
| --- | --- | --- |
| Team Member | Primary Focus | Key Responsibilities |
| Jesus | Backend Core (Auth, Roles, Ownership) | Implement authentication, role-based access control, and ownership logic. Manage moderation and admin features. |
| Brendan | Media & Collaboration | Handle song uploads, stems, and collaboration flows. Implement Render Spaces integration. |
| Kyle | Engagement & Analytics | Build reviews, comments, tips, and basic analytics endpoints. Connect engagement data to artist dashboards. |
| Chandler | DevOps & Infrastructure | Setup database, environment configs, Docker, deployment to Render, and CI/CD workflows. |
| Yaman | Frontend Integration & QA | Clean up frontend, connect UI to backend APIs, and perform final integration and QA testing. |

## 🔐 Roles & Permissions (RBAC + ABAC)

• Fan: Can view public songs, create reviews/comments/tips, and manage private playlists.  
• Artist: Can create, update, and publish songs/projects, invite collaborators, and access analytics for their own content.  
• Producer: Can upload stems to projects they’ve been invited to, and view project files.  
• Admin: Can moderate any comment/review and take down inappropriate content.  
• Ownership Rule (ABAC): A user can update or delete a resource if they are its owner or a member of the associated project.

## 📅 Week 1 — Core Infrastructure & Upload Flow

Goal: Build foundational backend systems (auth, roles, songs, projects) and local uploads.

|  |  |  |  |
| --- | --- | --- | --- |
| Developer | Tasks | Dependencies | Deliverables |
| Jesus | Set up Fastify backend, JWT auth, and RBAC logic; build /auth and /roles endpoints. | None | Working auth + roles system. |
| Brendan | Create Song and Upload models; implement /songs, /uploads, and publish endpoints. | Dev D (DB setup) | Artist can create and publish songs locally. |
| Kyle | Define Project and Membership models; implement /projects and /invite endpoints. | Dev A (auth middleware) | Artists can create projects and invite collaborators. |
| Chandler | Configure Docker/Postgres/Prisma; prepare DigitalOcean DB & Spaces. | None | Functional local environment and ready-to-deploy DB. |
| Yaman | Clean frontend layout, set up .env for API base URL, prepare Sign In/Up UI. | Dev A (auth API) | Clean, connected frontend shell ready for integration. |

## 📅 Week 2 — Collaboration & Engagement

Goal: Enable collaboration between artists and producers; fans can engage through reviews/comments/tips.

|  |  |  |  |
| --- | --- | --- | --- |
| Developer | Tasks | Dependencies | Deliverables |
| Jesus | Expand ownership checks and validation; build consistent error handler. | Dev D (deployed DB) | Secure endpoints with ownership validation. |
| Brendan | Implement /projects/:id/stems and Spaces integration. | Dev D (storage config) | Producers can upload stems to projects. |
| Kyle | Create /comments, /reviews, and /tips endpoints; manage ownership rules. | Dev A (auth) | Fans can comment, review, and tip artists. |
| Chandler | Deploy backend to DigitalOcean; set up CORS for Vercel frontend. | Dev B (API endpoints) | Live backend reachable via public URL. |
| Yaman | Connect login, upload, and public songs pages to API endpoints. | Dev A/B (auth + songs) | UI integrated with backend for key flows. |

## 📅 Week 3 — Analytics, Admin & Final Polish

Goal: Complete analytics endpoints, moderation tools, and ensure deployment stability.

|  |  |  |  |
| --- | --- | --- | --- |
| Developer | Tasks | Dependencies | Deliverables |
| Jesus | Add admin endpoints for moderation and takedowns. | Dev D (auth roles) | Admins can moderate comments and songs. |
| Brendan | Implement /events and /analytics/artist/:id/dashboard. | Dev C (engagement data) | Basic artist dashboard with play/view stats. |
| Kyle | Add /analytics/realtime; integrate engagement events into analytics. | Dev B (events API) | Real-time trending endpoint working. |
| Chandler | Finalize deployment, add logs, caching, rate limits, and health checks. | All APIs stable | Optimized, monitored production backend. |
| Yaman | Connect analytics dashboard to API, finalize UI polish and QA. | Dev B/C (analytics APIs) | Fully functional frontend connected to backend. |

## 🤝 Collaboration & Dependencies

• Jesus ↔ Chandler: Auth system depends on database and deployment configuration.  
• Bredan ↔ Chandler: File uploads depend on Spaces configuration.  
• Kyle ↔ Jesus: Engagement endpoints rely on user authentication.  
• Brendan ↔ Kyle: Analytics connects play/review/tip events to artist data.  
• Yaman ↔ All: Frontend relies on all backend APIs for integration and testing.

## 📦 Final Deliverables by Week 3

• Fully deployed backend (Render) and frontend (Vercel).  
• Functional Auth, Projects, Uploads, Reviews, and Analytics endpoints.  
• Working collaboration loop: artist → producer → fan.  
• Admin moderation tools.  
• API documentation and environment setup guide.  
• UI connected to backend with live data flow.

## ✅ Expected Outcome

By the end of Week 3, Leep Audio will have a fully functional MVP backend and connected frontend. Artists can upload and publish songs, producers can remix through projects, fans can engage through reviews and tips, and admins can moderate the platform. Analytics will provide artists with basic engagement insights.