DOUSIC MEDIA

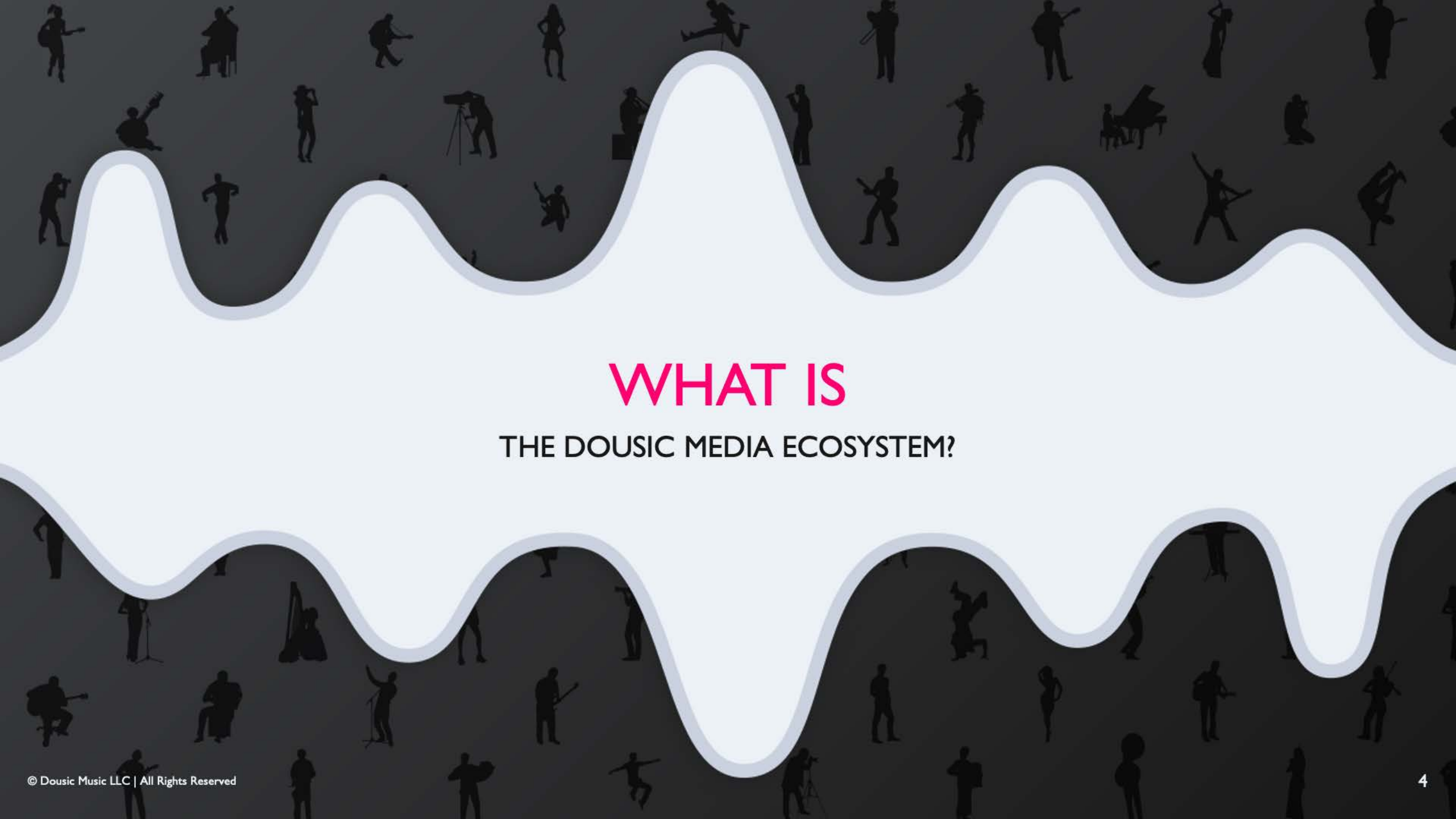
THE DOUSIC MEDIA ECOSYSTEM

A Developer's Overview

STATEMENT OF CONFIDENTIALITY & USAGE RESTRICTIONS

The content of this presentation and/or documentation contains the proprietary and confidential information of DOUSIC Music LLC and/or all of its relevant parent(s), subsidiary and/or subsidiaries and, as applicable, the proprietary and confidential information of third parties used by DOUSIC Music LLC under an appropriate license there from. In consideration of DOUSIC Music LLC making this information available to you, you agree to protect the confidentiality of the same as might reasonably be expected by DOUSIC Music LLC .Without limitation, the information contained herein is the copyright of DOUSIC Music LLC and may not be re-produced in whole or in part without the express prior permission of DOUSIC Music LLC.

TABLE OF CONTENTS



WHAT IS

THE DOUSIC MEDIA ECOSYSTEM?

The Dousic Media ecosystem is comprised of community-based and device agnostic online services and tools engineered to empower a community to create, collaborate, control, consume, and profit independently and together from original creative digital works through a collaborative and fully accessible media monetization experience.

OUR DIGITAL MEDIA CATAGORIES

- Audio Files
- Video Files
- Static Text Content
- Live Streaming Content
- Visual Art Content
- eBook Content
- Web App/Software Content
- Production Assets

END-USER DIGITAL MEDIA EXAMPLES

Audio Files

Chord Progressions
Instrument Solos
Music Songs
Podcast Episodes
Recorded Monologues
Vocal Samples
Voice Overs
Voice Solo Samples
And much more…

eBooks

Fiction Novels
Graphic Novels
Instructional Guides
eMagazines
Mixed-Media eBooks
And much more…

Live Streams

DJ Sets
Live Journaism
Live Monologue Performances
Live Troupe Performances
Live Spiritual Oration
Live Weather Reporting
And much more…

Production Assets

4K Streaming Layout Templates
DAW Default Templates
Graphic Layout Templates
HD Streaming Layout Templates
Presentation Templates
Sampler Patches
And much more…

Static Text

Long Form Fiction
Long Form Non-Fiction
Poems
Short Form Fiction
Short Form Non-Fiction
Song Lyrics
And much more…

Video Files

Animations
VODs
Vlog Episodes
Video Podcast Episodes
Pre-Recorded Web Series
And much more…

Visual Art Files

Profile Cover Images
Promotional Images
Song Artwork Images
And much more…

Web Software

Custom SPA Tooling
Custom Static Brochure Pages
And much more…

ECOSYSTEM

APPLICATION ARCHITECTURE

DEVELOPMENT GOALS

- Build a new Dousic Media web application to replace our not MVC/DRY enough legacy CodeIgniter 3 application with a properly separated and DRY Laravel 8.x code base(s) and server(s) that are (all together) appropriately de-coupled for horizontal scaling via load balancing.
- 100% of Dousic Media's developer's will be required to work via local instances of our new Laravel application, which should therefore be set up, maintained, and fully documented (including the pesky local installs process) for the Laravel workflow(s) using the new "Laravel Sail" package (IE Docker!)
- All code must be versioned and stored in Dousic Media GitHub account repositories. Whatever repos are needed for both DevOps and de-coupling are open to be created as needed/advised.
- Application Keys and sensitive authentication information (IE CD/CI/services/api creds etc) need to ideally be kept as obfuscated and centralized as possible - and away from local developer installs as much as possible. Our current policy is to manually provide new devs with keys in a CodeIgniter config file, so we'd really like a more robust (and secure) solution for app credentials - so we can handle bringing in a LOT of developers as we grow - but without risking the kingdom security in the process.
- Also, if we need to change a secret key or service partner API password, we'd ideally want that to be a single source of truth (DRY change) that in-turn ripples out to servers and builds naturally during deployment processes (assuming keys/passwords are obfuscated as in the bullet above).
- Generally speaking, we want to follow any/all possible Laravel community guidelines, best practices, and coding style guidelines - with a persistent goal of only pushing fully refactored, DRY, and fully documented code (for the next devs) into our code base repos.

DOUSIC MEDIA ECOSYSTEM ARCHITECTURE SKETCH (PROPOSED)

LOAD BALANCED SERVER PROVISIONING-BASED SYSTEM

DATA LAKE/ WAREHOUSING

xxx.redbee.com
RED BEE SERVICE
(Internal and Stream/Video End-User Portal Service)

app.dousic.com
LARAVEL SERVER
(Horizontally Scalable App Site)

App Site Release Repo
GitHub Repo & Deployment Docs

db.dousic.com
HEADLESS API LARAVEL SERVER
(App Site Data API)

Db API Site Dev Repo
GitHub Repo & Docs

admin.dousic.com
LARAVEL SERVER
(App Analytics, User Management, Tooling)

Admin Site Release Repo
GitHub Repo & Deployment Docs

livestage.dousic.com
LARAVEL SERVER
(Www Staging Site)

Admin Staging Site Dev Repo
GitHub Repo , CI/CD Actions & Dev Docs

Staging Site Dev Repo
GitHub Repo & Dev Docs

QA Site Release Repo
GitHub Repo & QA Docs

localhost
LARAVEL SAIL/DOCKER & MySQL INSTANCE
(Data Seeded Instance via Docker w/ Server Env's)

staging.dousic.com
LARAVEL SERVER
(Dev Staging Site)

qa.dousic.com
LARAVEL SERVER
(QA Staging Site)

Laravel Forge
APP KEYS & LOAD BALANCING

Www Site Release Repo
GitHub Repo & Deployment Docs

Www Staging Site Dev Repo
GitHub Repo , CI/CD Actions & Dev Docs

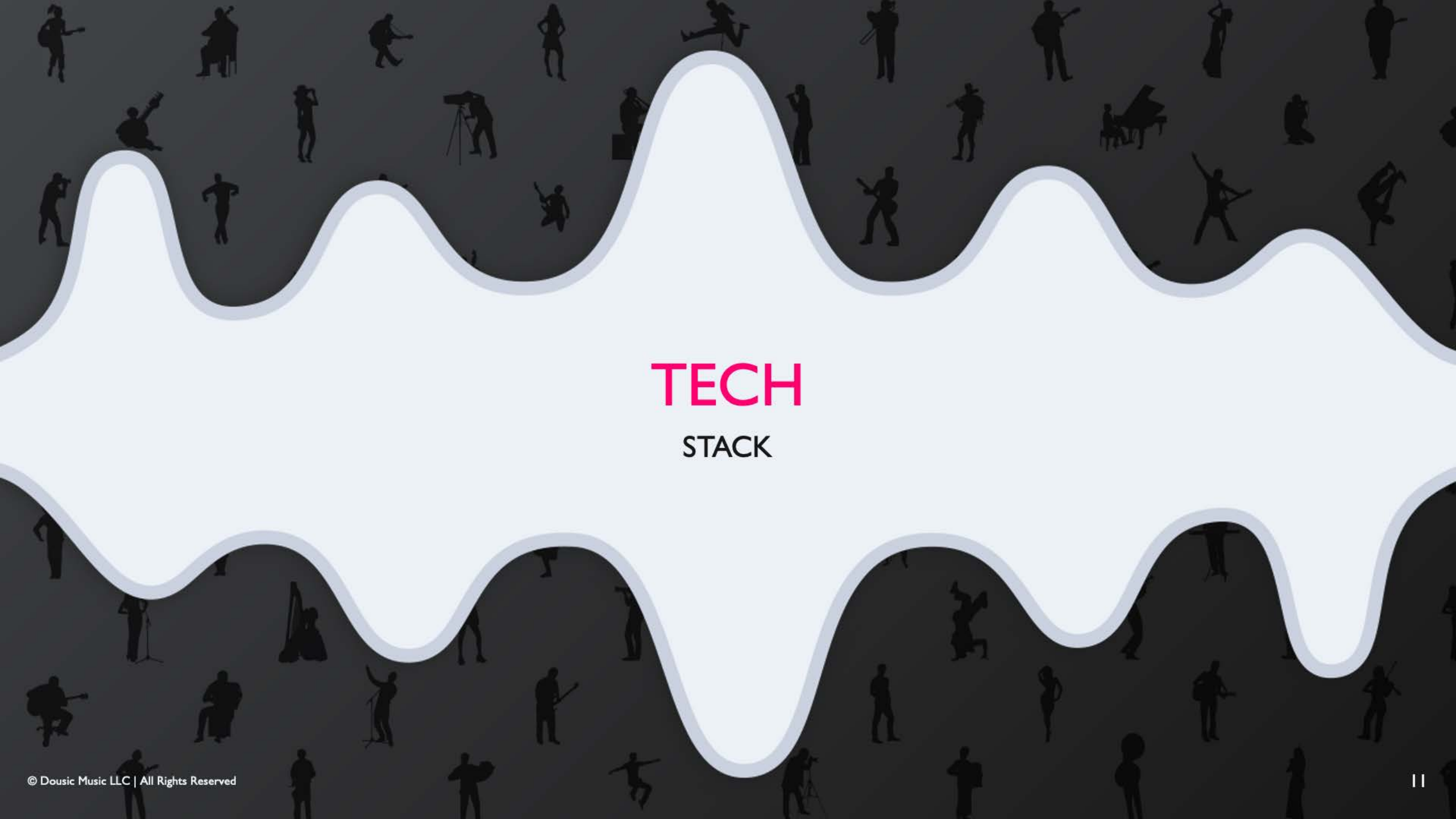
Laravel Envoyer
DEPLOYMENT MANAGEMENT

dousic.com
LARAVEL SERVER
(Public Site + Blog)

ISP
CLOUD SERVER PROVIDER

livestage.dousic.com
LARAVEL SERVER
(Www Staging Site)

Developer Workflow
Repo Release Triggered Automation
Forge/Envoyer/AWS Workflow
App Data API Serviced
Client-Side Link Pathways



TECH

STACK

WEB ECOSYSTEM TECH STACK

10,000-FOOT VIEW OF OUR TECH STACK

- We want a predominantly and/or pure Composer + LAMP (PHP) stack back-ends, but need the ability to add in a Python server into the mix for future data mining and machine learning needs (thus the cloud-based server architecture)
- The front end of the application is over-tooled for now. However, this is ok, as it's likely we'll be using mostly Livewire/Blade components for all of our server-side rendered interactive components, but need the ability to use JavaScript ES Module components and/or Vue 3 components from the client side too (IE a hybrid server-side/SPA client-side architecture)
- Additionally, concerning front end engineering at Dousic Media, we're not looking for any FED's that use modern JS libs as crutches for employees. As contractors, sure – but not employees. Instead, we will be looking for FED's that decided to focus on vanilla JS so they can jump between any FE framework without much delay! Thus, our options for FE development tasks!
  - PS – we're definitely going to have some VERY complex interfaces to build over in the next 12 months… so having the preferred Laravel shadow DOM JS lib of Vue is a necessary evil… but that means we're going to be at least in some part a Vue.js firm vs a React.js or Angular 3 firm for now (especially thanks to Vue 3's new inclusion of React-like hooks!)
- Otherwise back to the back-end, we want to use as many core Laravel Packages as needed (including paid ones!) as we want to be able to quickly and reliably hire adjunct Back-End Laravel experts to aid us at any time in the future with code base updates/changes/patches/etc.

CROSS-APP COMPONENT LIBS AS DEPENDENCIES

- We will very likely find a need to centralize, standardize, and synchronize front-end components for brand consistency across our different urls/sub-domains
- Possible front-end component libs (that should possibly mirror each other's output so the same interfaces can be rendered either server or client side and still match):
  - Laravel/Livewire components (Blade, Tailwind & alpine.js)
  - OBE:WDS Components (Bootstrap 4, jQuery, & Molecular Components ES Modules)
  - Vue 3 Components (not needed as of now but could come into play suddenly any minute)

SAAS/PAAS

ARCHITECTURE

SAAS SERVICE PARTNERS

LARAVEL

RED BEE

PUBNUB

PERFORCE

VONAGE

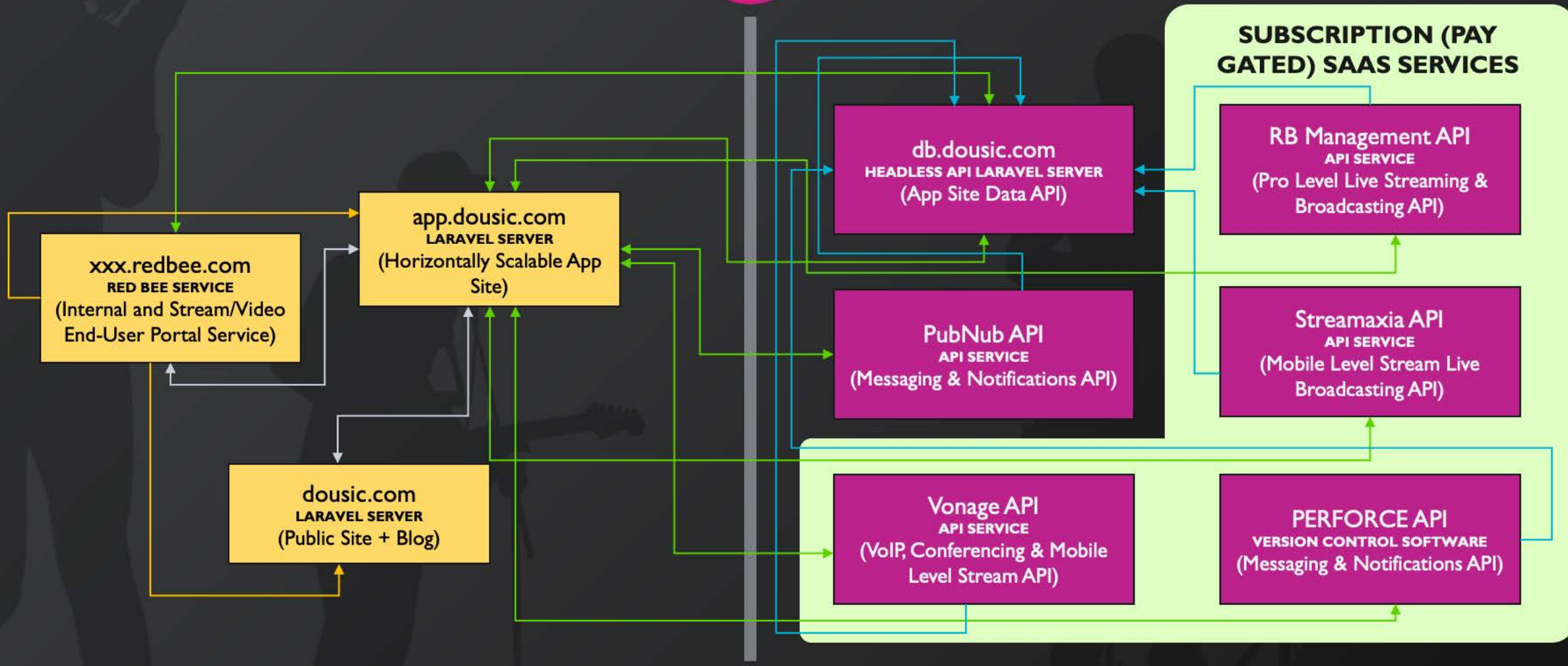
STREAMAXIA

SAAS/PAAS GOALS

- A large part of the need for a decoupled database architecture is to ensure we have read and write capabilities from/to our own application data and potentially from partner PaaS servers as well – so we can allow our users to pull in Dousic Media core application data and save data back to our app database while on an external partner PaaS box.
- For SaaS/PaaS partner services, we also want to ensure that in fringe cases where it makes sense or is needed – we can temporarily store and/or duplicate API information in the core app database for easier persistence and to reduce redundant API calls that could run up unforeseen/unbudgeted fees for Dousic Media.

SAAS SERVICES PRODUCT DIAGRAM

HYBRID SERVER-SIDE / MICRO SERVICES ARCHITECTURE



SERVER-SIDE APPLICATIONS

Client-Side Link Pathways
API Service Functionality
Potential Data Syncing Functionality
Embedded iFrame Media Content

PARTNER/INTERNAL SAAS API & PAAS SERVICES

MVP

MILESTONES & QUESTIONS

MVP MILESTONES (WIP)

- MVP Architecture Requirements:
  - Sub-Domains Plan (incl. transitional)
  - Repositories Plan
  - Horizontal & Vertical Scaling Configurations/Plans
  - Server Management Tooling Setup (Forge)
  - Deployment Management Tooling Setup (Envoyer)
  - Sail Dev Workflows (local to production)
- Required Laravel MVP Functionality
  - Local Dev Database Migration and Faker Data Seeding Functionality
  - 2 Factor Authentication System (Livewire Starter Kit + Fortify)
  - Payment Processing (Cashier/Spark)
  - Subscription Payment Processing (Spark)
  - Cart Page (Cashier/Spark)
  - Add brand styling to out-of-box user-facing Blade files in use
- Required App MVP Functionality:
  - User Settings Sub-Portal
    - App Settings
    - Data Privacy/Visibility Settings
  - Public user profile page (custom URL)
  - User Data Privacy/Visibility Settings
  - Dousic currency shop page & controls
  - Dousic account subscription page
  - Sermons Marketplace page (filterable)
  - Upload VoD file page
  - Pay-Gated Live Streaming Functionality
    - Red Bee system ingest page (RB Management API)
  - Mobile live stream ingest page (Streamaxia &/or Vonage APIs)
  - My Content page
  - Content Preview/Buy page
    - Buy VoD access btn & pricing functionality
    - Subscribe btn & pricing functionality
      - Subscribe Options nees to offer users subs per min, hours, days, weeks, months, and 1, 2, and 3 years options
- Proprietary MVP Functionality: (Cont.)
  - Ratings System Implementation and data storage
  - Setup Special lib repo and maintain as cross-app blade FED components (as a composer dependency)
    - Off-Canvas modal main nav,
    - Sticky header & footers
    - Contact us form
    - Layout components
    - Content components
- Brochure Site Launch Site Requirements:
  - Home page
  - About page
  - Simple blog articles functionality (for launch articles and ad campaign conversion pages - could be hard-coded)
  - Dousic App DB API or RB API (to bring in featured artists and content into brochure site home/landing pages) availability
  - Ministries landing page

MVP QUESTIONS

- Do all server code bases allow for zero downtime updating (and rollbacks) along with load balancing, secured core app creds, provisioning, and basic load teasting results for an appropriate threshold of the application's concurrency maximums – all versioned properly and documented fairly for later devs to work efficiently?
- Localization?
- Data normalization?
- Are all FE components fully accessible and fully functional on all device views (even though we are mobile first)?

MVP WISH LIST

- Hopefully we can hack the Laravel Teams many-to-many structure in the Livewire kit for:
  - Solo project functionality and management
  - Collaboration Project functionality and management
- Collaboration splits functionality
- Collaboration Services Marketplace and functionality
- VCS functionality for collab projects
- Collaboration contest functionality
- Collab hours logging functionality
- Analytic data processing & charts of VCS data per project (solo and collab views)

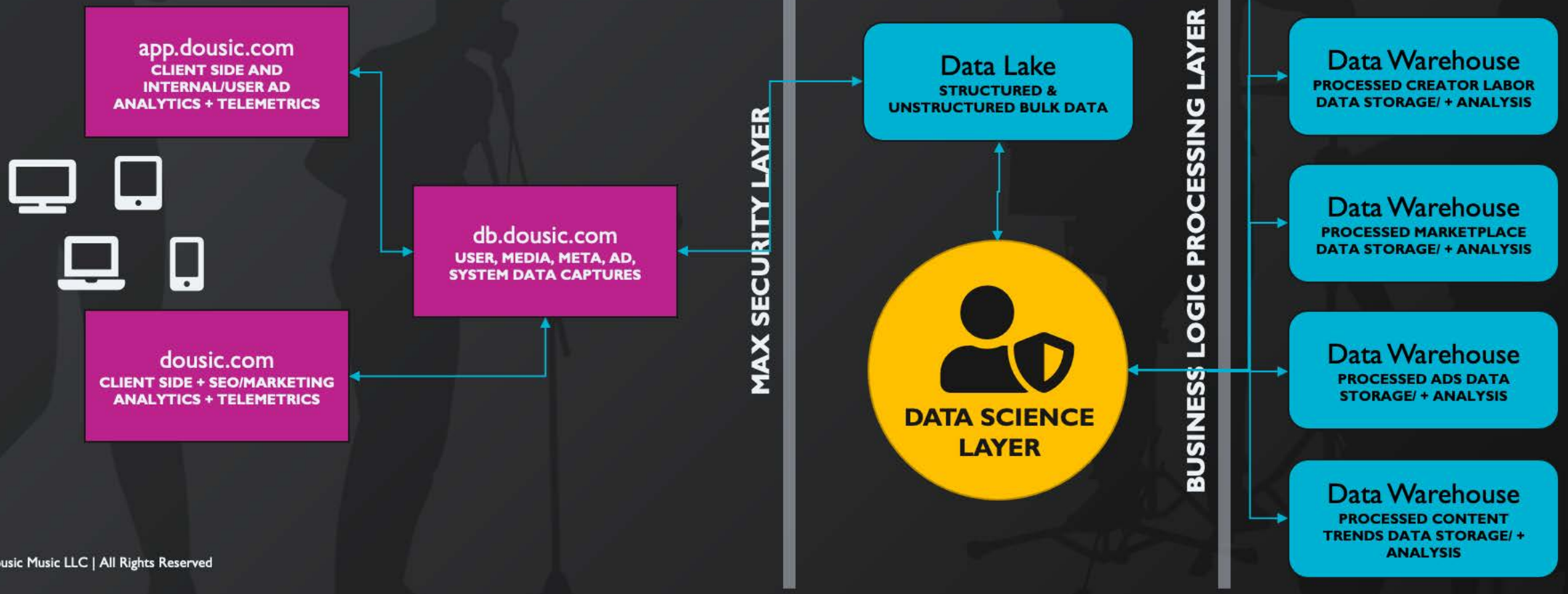
THE BIGGER

DATA PICTURE

BIG DATA GOALS

- Our combination of opportunities for market sales, collaborative splits, and runtime ad royalties, for both individual and groups of digital content creators across nearly all of the digital media content mediums, is a big data treasure trove
- Additionally, our data possibilities are vast when we consider automated creative asset valuations based on overall file data diffing (VCS) and logged labor time analysis.
- In a bigger picture view, our humble Laravel PHP application should ALWAYS be thought of as an early-stage data net – where a definite future plan of Dousic Media is to acquire:
    1. A Data Lake Solution (IoT solution) (UI interaction event tracking, geo-tracking for broadcasters, iterative creative files, labor hour patters, and much much more)
    2. A Big Data Warehousing Solution (IoT solution style to be determined out of Data Lake data storage and models)
    3. Data Mining and Machine Learning Tooling/Staff/Processes
- Big Data Goals are well beyond the scope of the MVP and initial verticals. However, we want to ensure the application is thought of as a Data Lake / Data Warehouse friendly application from its beginnings, as the analytics of using mining and machine learning will become added value profit revenue verticals.

BIG DATA FLOW (WIP)



IN THE END WHAT WE'RE DOING IS SIMPLE. WE'RE LETTING CREATORS SELL WHATEVER THEY CAN CREATE EITHER ALONE, TOGETHER, OR THROUGH SELLING SIPPETS AND SERVICES TO HELP OTHERS SUCCEED WHILE STILL GETTING A FAIR CUT.

- Together We Are Strong -

THANK YOU!