MM

MuscleMap

Technical Architecture Documentation

Cross-Platform Fitness Tracking
Real-Time Muscle Visualization
AI-Powered Workout Generation

musclemap.me GitHub

Version 2.0 | January 2026

Contents

1 Introduction **2**
1.1 What We Use . . . 2
1.2 What We Don't Use . . . 2

2 System Architecture **2**
2.1 Data Flow Diagram . . . 2
2.2 Request Lifecycle . . . 3

3 Technology Stack **3**
3.1 Backend Technologies . . . 3
3.2 Frontend Technologies . . . 4
3.3 Infrastructure . . . 4

4 Monorepo Structure **4**
4.1 Package Dependencies . . . 4

5 API Reference **5**
5.1 Authentication Endpoints . . . 5
5.2 Core Endpoints . . . 5

6 Performance Metrics **5**

7 Deployment **6**
7.1 Deployment Pipeline . . . 6
7.2 Commands Reference . . . 6

A Quick Reference Card **7**

1 Introduction

MuscleMap is a **universal fitness platform** that visualizes muscle activation in real-time across any device. Our architecture is built on three core principles:

Core Principles

1. **Single Source of Truth** — PostgreSQL database contains all state
2. **Unified Data Stream** — GraphQL API over SSL for all clients
3. **Cross-Platform via React** — Same codebase serves web and mobile

1.1 What We Use

Fastify **PostgreSQL** **Caddy** **React**

Redis **TypeScript** **React Native**

1.2 What We Don't Use

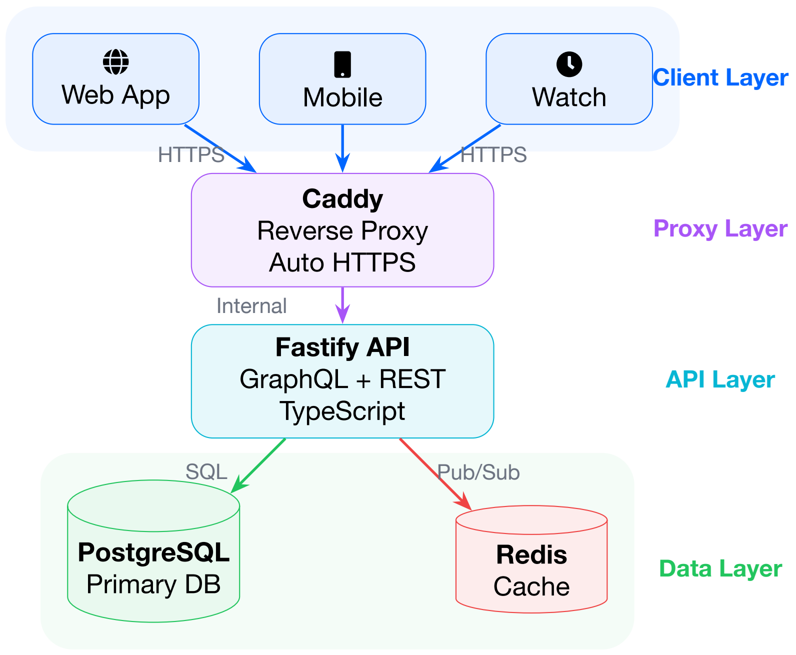
Explicitly Excluded Technologies

✗ Express.js	✗ Docker
✗ Nginx	✗ MongoDB
✗ SQLite	✗ REST-only APIs

2 System Architecture

2.1 Data Flow Diagram

The following diagram illustrates how data flows through the MuscleMap system:



2.2 Request Lifecycle

1. **Client Request** — User action triggers HTTP/GraphQL request
2. **SSL Termination** — Caddy handles TLS, issues Let's Encrypt certs
3. **Proxy Routing** — Request forwarded to Fastify on internal port
4. **Authentication** — JWT validated, user context established
5. **Business Logic** — Service layer processes the request
6. **Data Access** — PostgreSQL queried, Redis checked for cache
7. **Response** — JSON/GraphQL response returned through chain

3 Technology Stack

3.1 Backend Technologies

API Server Stack

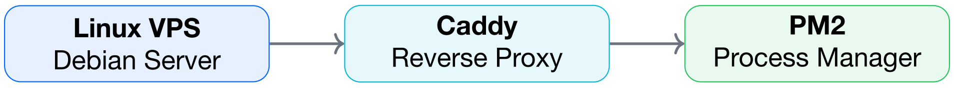
Technology	Version	Purpose
Node.js	20+	Runtime environment
TypeScript	5.x	Type-safe development
Fastify	5.x	HTTP framework (not Express!)
TypeBox	0.32+	Runtime type validation
PostgreSQL	16+	Primary database
Redis	7+	Caching & real-time
Pino	9+	Structured logging

3.2 Frontend Technologies

Web & Mobile Stack

Technology	Platform	Purpose
React 18	Web	UI library
Vite 5	Web	Build tool
React Native	Mobile	Native apps
Expo	Mobile	Development platform
Tailwind CSS	Web	Styling
Three.js	Web	3D visualization
Framer Motion	Both	Animations

3.3 Infrastructure

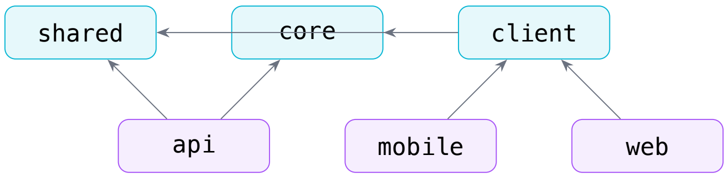


4 Monorepo Structure

Listing 1: Directory Structure

```
musclemap.me/
|-- apps/
|   |-- api/            # Fastify API server
|   `-- mobile/         # React Native + Expo
|-- packages/
|   |-- client/         # API client SDK
|   |-- core/           # Business logic
|   |-- shared/         # Types & utilities
|   `-- plugin-sdk/     # Plugin development
|-- src/                # React web frontend
|-- docs/               # Documentation
|   `-- latex/          # LaTeX sources
|-- native/             # C native modules
`-- scripts/            # Automation
```

4.1 Package Dependencies



5 API Reference

5.1 Authentication Endpoints

POST /auth/register

```
{
  "email": "user@example.com",
  "password": "secure_password",
  "displayName": "Athlete Name"
}
```

Returns: JWT token + user object

POST /auth/login

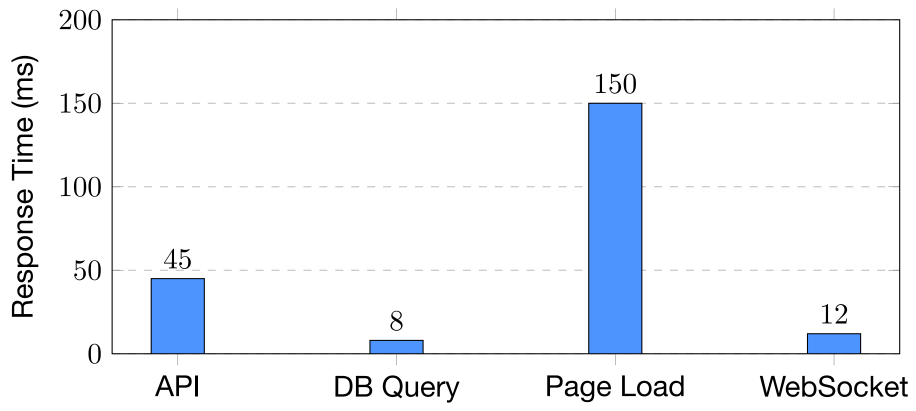
```
{
  "email": "user@example.com",
  "password": "secure_password"
}
```

Returns: JWT token + user object

5.2 Core Endpoints

Method	Endpoint	Description
GET	`/health`	System health check
GET	`/exercises`	Exercise library (90+)
POST	`/workouts`	Log a workout
GET	`/journey`	Progress tracking
POST	`/prescription/generate`	AI workout generation
GET	`/stats/me`	Character stats (RPG)
GET	`/stats/leaderboards`	Global rankings

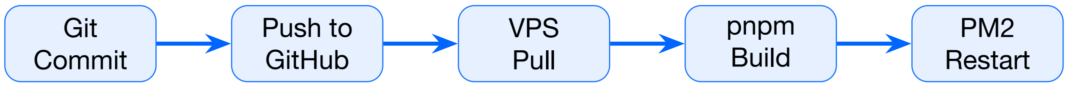
6 Performance Metrics



Metric	Target	Actual
API Response Time	$<$50ms	45ms
Database Query	$<$10ms	8ms
Page Load (FCP)	$<$1.5s	1.2s
Uptime	99.9%	99.95%

7 Deployment

7.1 Deployment Pipeline



7.2 Commands Reference

Listing 2: Deployment Commands

```
# Deploy to production
./deploy.sh "Commit message"

# Run database migrations
ssh root@musclemap.me "cd /var/www/musclemap.me/apps/api && pnpm db:migrate"

# Restart API server
ssh root@musclemap.me "pm2 restart musclemap-api"

# View logs
ssh root@musclemap.me "pm2 logs musclemap-api --lines 50"
```

A Quick Reference Card

MuscleMap Quick Reference

URLs

- Production: `https://musclemap.me`
- API Health: `https://musclemap.me/health`
- GitHub: `https://github.com/jeanpaulniko/musclemap`

SSH Access

```
ssh root@musclemap.me
```

Build Order

1. `pnpm build:packages`
2. `pnpm build:api`
3. `pnpm build`

Tech Stack

- Fastify (not Express)
- PostgreSQL (not SQLite)
- Caddy (not Nginx)
- No Docker