

# SHOTIQ AI Analysis - System Architecture Document

**Version:** 2.0

**Date:** December 26, 2024

**Prepared for:** Abacus AI Integration

## Table of Contents

- [1. Executive Summary](#)
- [2. System Overview](#)
- [3. Architecture Diagram](#)
- [4. Frontend Architecture](#)
- [5. Backend Architecture](#)
- [6. AI/ML Pipeline](#)
- [7. Data Flow](#)
- [8. API Endpoints](#)
- [9. Cost Analysis](#)
- [10. Deployment Architecture](#)
- [11. Future Integration Points](#)

## 1. Executive Summary

SHOTIQ is a professional-grade basketball shooting mechanics analysis platform that uses computer vision and AI to provide real-time biomechanical feedback on shooting form. The system analyzes images and videos of basketball players, detects body keypoints, identifies the basketball, calculates joint angles, and provides actionable coaching feedback.

### Key Differentiators

- **Hybrid AI System:** Combines multiple ML models for maximum accuracy
- **Cost-Effective:** Core analysis is 100% free using open-source models
- **Real-Time Processing:** Sub-second pose detection and analysis
- **Professional Output:** Annotated videos with skeleton overlays and metrics

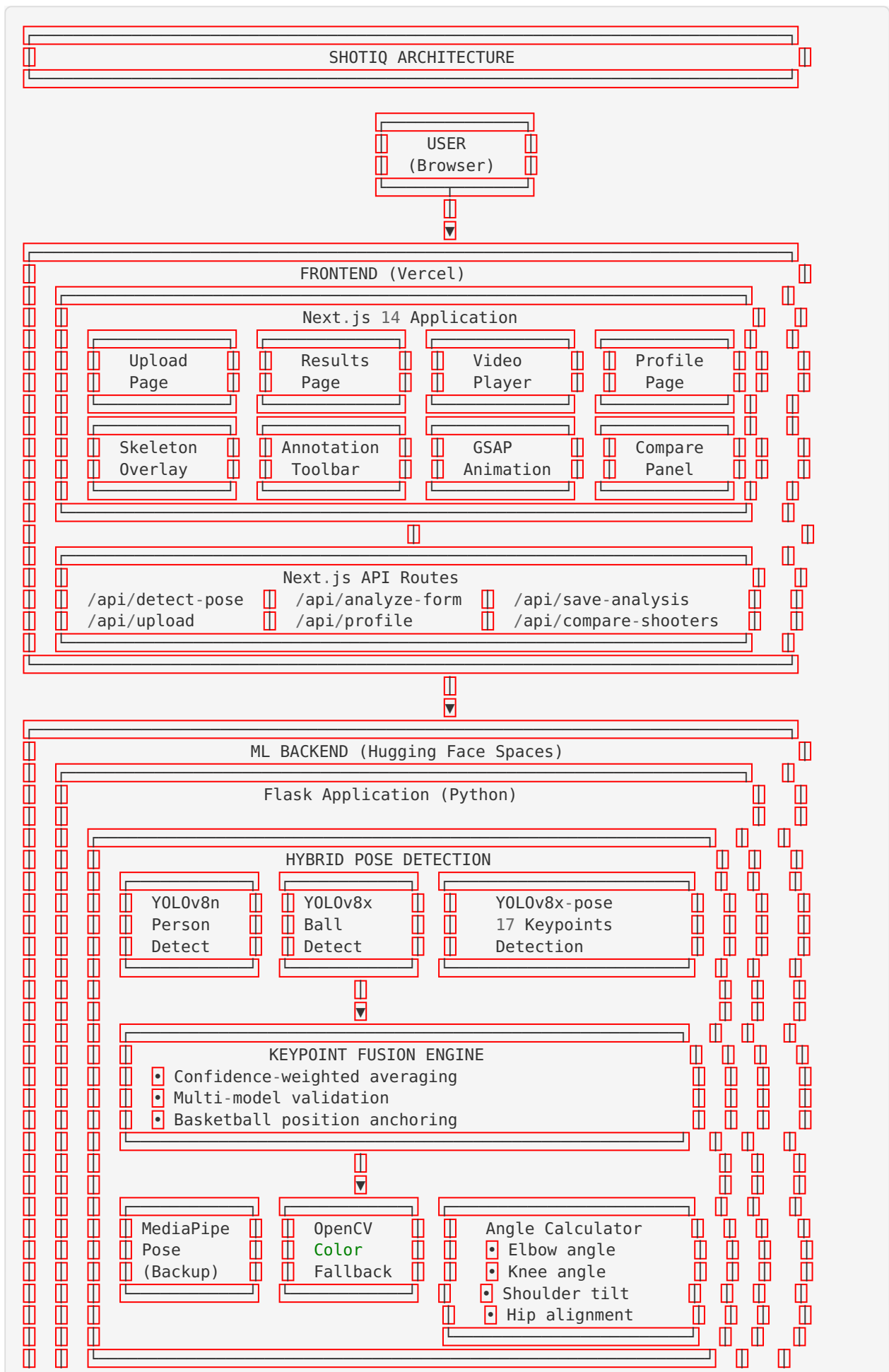
## 2. System Overview

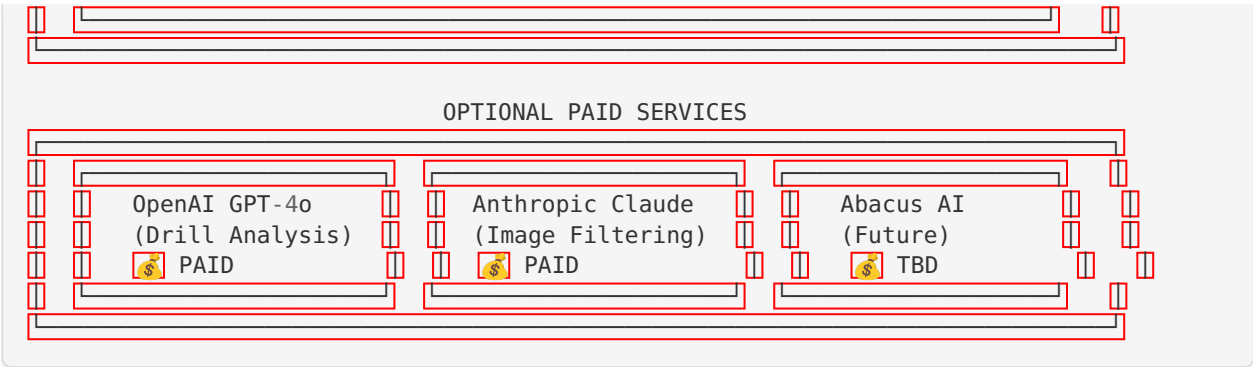
### Technology Stack

Layer	Technology	Purpose
Frontend	Next.js 14, React 18, TypeScript	Web application
Styling	Tailwind CSS, GSAP	UI/UX and animations
Backend API	Next.js API Routes	REST API endpoints
ML Backend	Flask (Python)	AI/ML processing
Pose Detection	YOLOv8x-pose, MediaPipe	Body keypoint detection
Object Detection	YOLOv8x	Basketball detection
Image Processing	OpenCV, Pillow	Image manipulation
Hosting (Frontend)	Vercel	Production deployment
Hosting (ML Backend)	Hugging Face Spaces	Free CPU tier

### 3. Architecture Diagram

---





## 4. Frontend Architecture

### 4.1 Page Structure

src/app/	
page.tsx	# Home/Upload page
results/demo/page.tsx	# Analysis results display
video-analysis/page.tsx	# Video upload & analysis
profile/page.tsx	# User profile management
elite-shooters/page.tsx	# Elite shooter comparison
badges/page.tsx	# Gamification/achievements
guide/page.tsx	# User guide
settings/page.tsx	# App settings

### 4.2 Key Components

Component	File	Purpose
GSAPVideoPlayer	GSAPVideoPlayer.tsx	3-stage video playback with overlays
HybridSkeletonDisplay	results/demo/page.tsx	Skeleton visualization on images
AutoScreenshots	AutoScreenshots.tsx	Key frame capture & analysis
ImageZoom	effects/image-zoom.tsx	Hover-to-zoom on images
Header	layout/Header.tsx	Navigation with SHOTIQ logo

### 4.3 Video Player Features

The GSAPVideoPlayer provides a 3-stage analysis experience:

- Stage 1: Full Speed** - Original video at normal speed
- Stage 2: Label Tutorial** - Annotated playback with metrics
- Stage 3: Slow Motion** - 0.25x speed for detailed review

**Overlay Toggles:**

- Skeleton (body keypoints & connections)

- Joints (individual keypoint markers)
- Annotations (angle measurements)
- Basketball (ball position marker)

---

## 5. Backend Architecture

---

### 5.1 Next.js API Routes

Endpoint	Method	Purpose
/api/detect-pose	POST	Proxy to ML backend
/api/analyze-form	POST	Form analysis & feedback
/api/upload	POST	Image/video upload handling
/api/save-analysis	POST	Save analysis to local storage
/api/profile	GET/POST	User profile management
/api/compare-shooters	POST	Elite shooter comparison
/api/vision-analyze	POST	GPT-4 Vision analysis (PAID)
/api/enhance-bio	POST	Bio enhancement (PAID)

### 5.2 Flask ML Backend

**File:** python-scraper/hybrid\_pose\_detection.py

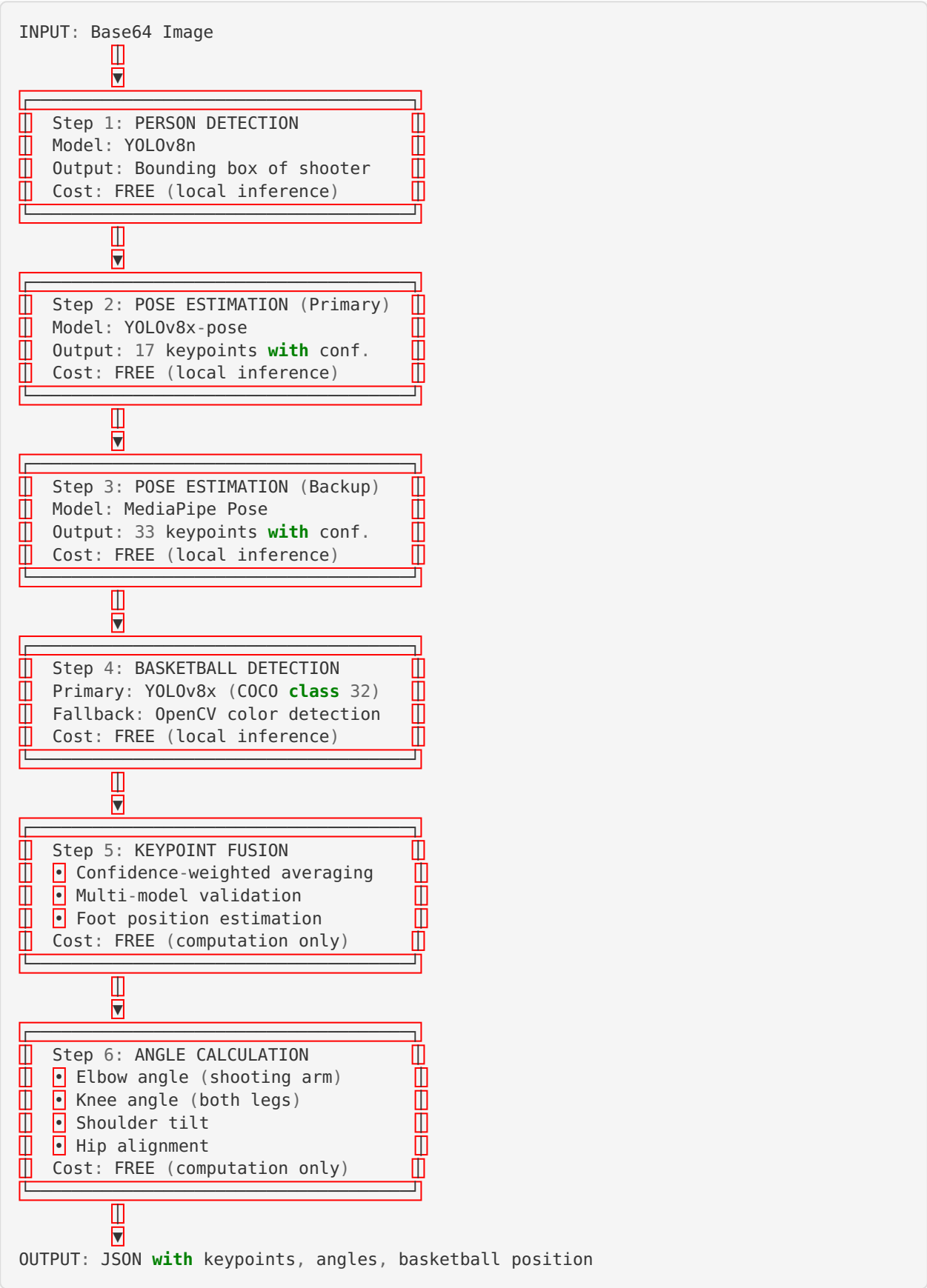
**Endpoints:**

Endpoint   Method   Purpose
----- ----- -----
/api/detect-pose   POST   Hybrid pose detection
/api/analyze-form   POST   Biomechanical analysis
/health   GET   Health check

---

## 6. AI/ML Pipeline

### 6.1 Hybrid Pose Detection Flow



## 6.2 Model Specifications

Model	Version	Parameters	Purpose	Accuracy
YOLOv8n	v8.0	3.2M	Person detection	37.3 mAP
YOLOv8x	v8.0	68.2M	Ball detection	53.9 mAP
YOLOv8x-pose	v8.0	69.4M	Pose estimation	81.0 mAP
MediaPipe Pose	v0.10	~3M	Backup pose	92% PCK

## 6.3 Keypoints Detected

### 17 YOLO Keypoints:

```
nose, left_eye, right_eye, left_ear, right_ear,  
left_shoulder, right_shoulder, left_elbow, right_elbow,  
left_wrist, right_wrist, left_hip, right_hip,  
left_knee, right_knee, left_ankle, right_ankle
```

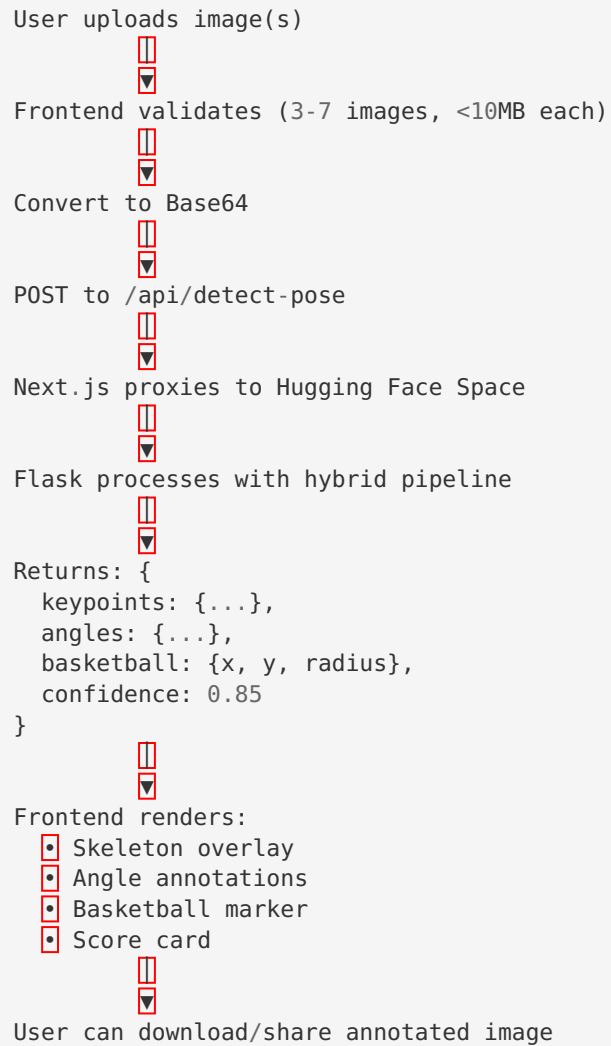
### Additional Estimated Keypoints:

```
left_foot, right_foot (derived from ankle positions)
```

---

## 7. Data Flow

### 7.1 Image Analysis Flow



## 7.2 Video Analysis Flow

```

User uploads video (<10s, <50MB)
  ▮
  ▼
Frontend extracts frames (10 FPS)
  ▮
  ▼
Each frame processed through hybrid pipeline
  ▮
  ▼
Keypoints stored per frame
  ▮
  ▼
GSAPVideoPlayer renders 3-stage playback:
  Stage 1: Full speed with overlays
  Stage 2: Labels & metrics tutorial
  Stage 3: Slow motion review
  ▮
  ▼
User can download individual stages or full video
  
```

## 8. API Endpoints

### 8.1 Core Detection API

**POST** `/api/detect-pose`

Request:

```

{
  "image": "data:image/jpeg;base64,/9j/4AAQ..."
}
  
```

Response:

```
{
  "success": true,
  "keypoints": {
    "nose": {"x": 512, "y": 120, "confidence": 0.95, "source": "fused"},
    "left_shoulder": {"x": 480, "y": 200, "confidence": 0.92, "source": "yolo"},
    "right_shoulder": {"x": 544, "y": 198, "confidence": 0.91, "source": "yolo"},
    "left_elbow": {"x": 420, "y": 280, "confidence": 0.88, "source": "fused"},
    "right_elbow": {"x": 600, "y": 275, "confidence": 0.87, "source": "fused"},
    "left_wrist": {"x": 380, "y": 350, "confidence": 0.85, "source": "mediapipe"},
    "right_wrist": {"x": 640, "y": 340, "confidence": 0.84, "source": "mediapipe"},
    // ... additional keypoints
  },
  "angles": {
    "left_elbow_angle": 92.5,
    "right_elbow_angle": 88.3,
    "left_knee_angle": 145.2,
    "right_knee_angle": 148.7,
    "shoulder_tilt": 2.1,
    "hip_tilt": 1.8
  },
  "basketball": {
    "x": 620,
    "y": 320,
    "radius": 35
  },
  "bounding_box": {
    "x1": 300,
    "y1": 50,
    "x2": 700,
    "y2": 800
  },
  "confidence": 0.89,
  "image_size": {"width": 1024, "height": 768},
  "method": "hybrid"
}
```

## 8.2 Form Analysis API

**POST** /api/analyze-form

Request:

```
{
  "keypoints": {...},
  "angles": {...}
}
```

Response:

```
{
  "success": true,
  "feedback": [
    {
      "type": "success",
      "area": "elbow",
      "message": "Excellent elbow angle (92°). Perfect L-shape!"
    },
    {
      "type": "success",
      "area": "knees",
      "message": "Good knee bend (145°) for power."
    },
    {
      "type": "warning",
      "area": "alignment",
      "message": "Slight shoulder tilt detected. Work on balance."
    }
  ],
  "overall_score": 85,
  "angles": {...}
}
```

## 9. Cost Analysis

### 9.1 Free Components (Core Analysis)

Component	Service	Monthly Cost	Notes
Pose Detection	YOLOv8x-pose (local)	\$0	Open-source, runs on CPU
Person Detection	YOLOv8n (local)	\$0	Open-source, runs on CPU
Ball Detection	YOLOv8x (local)	\$0	Open-source, runs on CPU
Backup Pose	MediaPipe (local)	\$0	Google open-source
Color Detection	OpenCV (local)	\$0	Open-source
Frontend Hosting	Vercel (Hobby)	\$0	Free tier sufficient
ML Backend	Hugging Face Spaces	\$0	Free CPU tier

Total Core Analysis Cost: \$0/month

9.2 Paid Components (Optional Features)


Feature	Service	Cost	Usage
Drill Video Analysis	OpenAI GPT-4o Vision	~\$0.01/image	Only if enabled
Bio Enhancement	OpenAI GPT-4o	~\$0.003/request	Only if enabled
Image Filtering	Anthropic Claude	~\$0.008/image	Training data only
Custom Domain	Vercel Pro	\$20/month	Optional
GPU Acceleration	Hugging Face GPU	\$0.60/hour	Optional

9.3 Estimated Monthly Costs by Usage

Usage Level	Core Analysis	Optional AI	Hosting	Total
Free Tier (1000 analyses)	\$0	\$0	\$0	\$0
Light (5000 analyses)	\$0	\$0	\$0	\$0
Medium (10000 analyses)	\$0	\$50 (if GPT enabled)	\$0	\$0-50
Heavy (50000 analyses)	\$0	\$250 (if GPT enabled)	\$20	\$20-270

9.4 Hugging Face Rate Limits

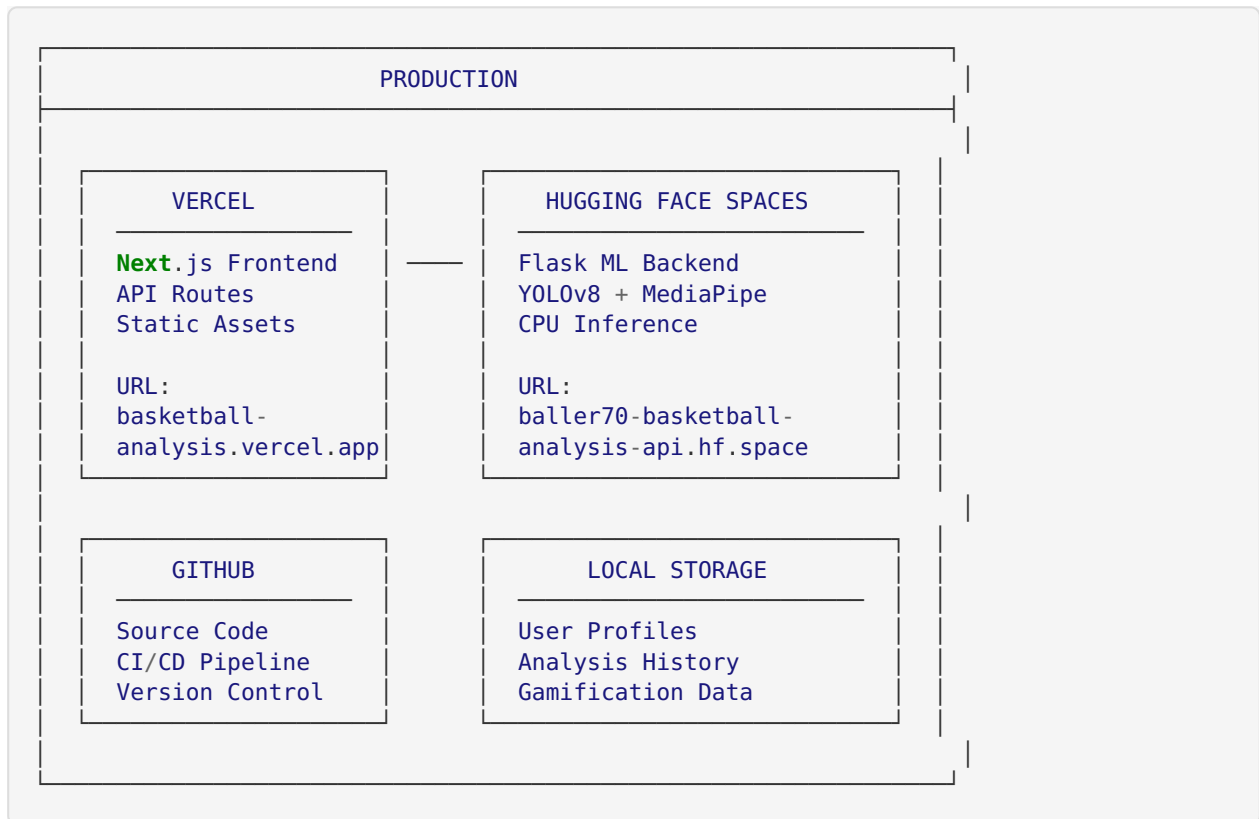
Tier	Requests/min	Requests/day	Cost
Free	10	1000	\$0
Pro	100	10000	\$9/month
Enterprise	Unlimited	Unlimited	Custom

Current Status: Operating within Free tier limits 

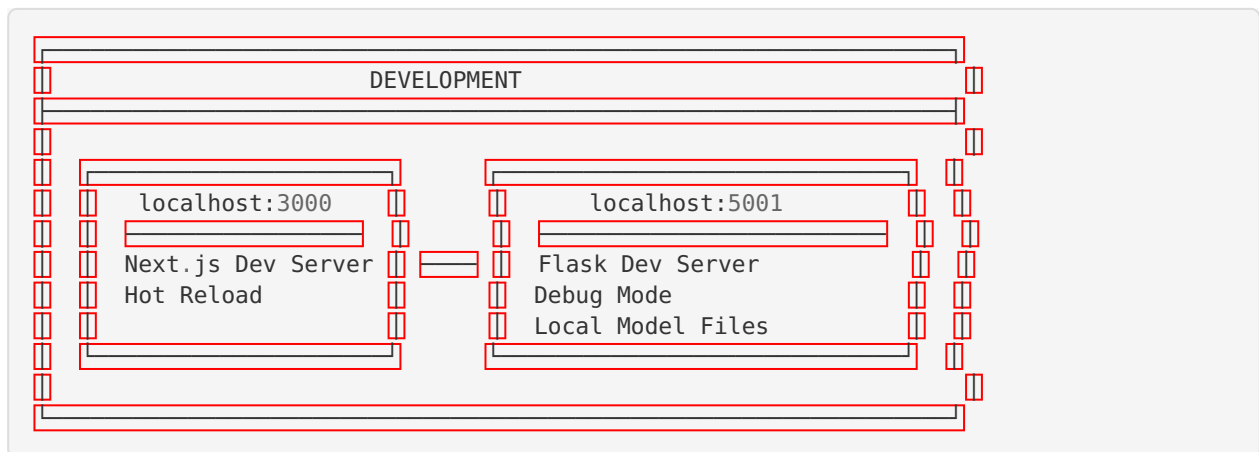
---

## 10. Deployment Architecture

### 10.1 Production Environment



### 10.2 Development Environment



## 11. Future Integration Points

### 11.1 Abacus AI Integration Opportunities

Feature	Integration Type	Potential Use
Enhanced Coaching	LLM API	Natural language feedback
Shot Prediction	Custom Model	Success probability
Player Comparison	Vector DB	Find similar shooting styles
Training Plans	LLM + RAG	Personalized drills
Video Summarization	Vision API	Key moment extraction

### 11.2 Recommended Abacus AI Features

- 1. Conversational Coaching**
  - Use Abacus LLM to provide interactive Q&A about shooting form
  - RAG system with professional coaching knowledge base
- 2. Advanced Analytics**
  - Time-series analysis of shooting improvement
  - Predictive modeling for shot success
- 3. Multi-modal Analysis**
  - Combine pose data with audio (coach instructions)
  - Integration with wearable sensor data

### 11.3 API Integration Points

```
# Suggested Abacus AI integration points

# 1. Enhanced feedback generation
@app.route('/api/ai-feedback', methods=['POST'])
def ai_feedback():
    # Send pose data to Abacus LLM for detailed coaching
    pass

# 2. Shot success prediction
@app.route('/api/predict-shot', methods=['POST'])
def predict_shot():
    # Use Abacus custom model for prediction
    pass

# 3. Similarity search
@app.route('/api/find-similar', methods=['POST'])
def find_similar():
    # Use Abacus vector DB to find similar shooting forms
    pass
```

