

Streaming Service Integration

Overview

The Sports Bar TV Controller now includes comprehensive streaming service integration, allowing you to:

1. **Detect installed streaming apps** on Fire TV devices
2. **Launch streaming apps** automatically
3. **Access sports event schedules** from multiple APIs
4. **View live and upcoming games** across different services
5. **Deep link to specific events** when supported

Features

Implemented Features

Fire TV App Detection

- Automatically detects which streaming apps are installed on Fire TV devices
- Caches detection results for performance
- Supports 15+ major streaming services

App Launching

- Launch streaming apps with simple API calls
- Support for deep linking to specific content
- Handles app launch errors gracefully

API Integrations (Public APIs)




- **ESPN API:** Live scores, schedules for NFL, NBA, MLB, NHL, NCAA
- **MLB Stats API:** Comprehensive baseball data and schedules
- **NFHS Network:** Framework ready (awaiting API access)

Unified API Interface




- Consistent interface across all streaming services
- Search events by team name
- Filter by sport, date, or live status
- Get events for installed apps

Supported Streaming Services




Sports-Specific Services

1. **NFHS Network** (High School Sports)
 -  App detection
 -  App launching
 -  API integration (placeholder framework ready)




2. ESPN+

-  App detection
-  App launching
-  API integration (public API)




3. MLB.TV

-  App detection
-  App launching
-  API integration (public API)




4. NBA League Pass

-  App detection
-  App launching
-  API integration (planned)



5. Fox Sports

-  App detection
-  App launching
-  API integration (limited access)



6. NBC Sports

-  App detection
-  App launching
-  API integration (limited access)

7. FuboTV

-  App detection
-  App launching

8. DAZN

-  App detection
-  App launching

Live TV Services

- YouTube TV
- Hulu Live
- Sling TV
- Paramount+
- Peacock
- Amazon Prime Video
- Apple TV

Architecture

Core Components

```
src/
├── lib/
│   ├── streaming/
│   │   ├── streaming-apps-database.ts    # Database of streaming apps
│   │   ├── unified-streaming-api.ts      # Unified API interface
│   │   └── api-integrations/
│   │       ├── espn-api.ts               # ESPN API integration
│   │       ├── mlb-api.ts                # MLB API integration
│   │       └── nfhs-api.ts                # NFHS API framework
│   ├── services/
│   │   └── streaming-service-manager.ts  # Manages app detection/launching
│   ├── app/api/streaming/
│   │   ├── apps/detect/route.ts          # Detect apps endpoint
│   │   ├── launch/route.ts               # Launch app endpoint
│   │   ├── events/route.ts               # Get events endpoint
│   │   └── status/route.ts               # Service status endpoint
│   └── components/streaming/
│       ├── StreamingAppsPanel.tsx        # UI for managing apps
│       └── LiveEventsPanel.tsx           # UI for viewing events
```

Usage

1. Detect Installed Apps

```
// API Call
const response = await fetch('/api/streaming/apps/detect', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({
    deviceId: 'fire-tv-1',
    ipAddress: '192.168.1.100',
    port: 5555
  })
})

const data = await response.json()
console.log(data.apps) // Array of installed apps
```

2. Launch a Streaming App

```
// Launch NFHS Network app
const response = await fetch('/api/streaming/launch', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({
    deviceId: 'fire-tv-1',
    ipAddress: '192.168.1.100',
    appId: 'nfhs-network',
    port: 5555
  })
})
```

3. Launch with Deep Link

```
// Launch ESPN+ to a specific game
const response = await fetch('/api/streaming/launch', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({
    deviceId: 'fire-tv-1',
    ipAddress: '192.168.1.100',
    appId: 'espn-plus',
    deepLink: 'espn://x-callback-url/showEvent?eventId=12345',
    port: 5555
  })
})
```

4. Get Live Events

```
// Get all live sports events
const response = await fetch('/api/streaming/events?type=live')
const data = await response.json()
console.log(data.events) // Array of live events
```

5. Get Upcoming Events

```
// Get upcoming events for next 7 days
const response = await fetch('/api/streaming/events?type=upcoming&days=7')
const data = await response.json()
```

6. Search Events by Team

```
// Search for specific team
const response = await fetch('/api/streaming/events?type=search&team=Lakers')
const data = await response.json()
```

Using the Unified API

```
import { unifiedStreamingApi } from '@lib/streaming/unified-streaming-api'

// Get all live events across all services
const liveEvents = await unifiedStreamingApi.getAllLiveEvents()

// Get upcoming events for a sport
const upcomingBaseball = await unifiedStreamingApi.getUpcomingEvents('baseball', 7)

// Search for team
const lakersGames = await unifiedStreamingApi.searchEventsByTeam('Lakers', 'basketball')

// Get events for installed apps on a device
const deviceEvents = await unifiedStreamingApi.getEventsForInstalledApps(
  'fire-tv-1',
  '192.168.1.100',
  5555
)
```

UI Components

StreamingAppsPanel

Displays installed streaming apps with ability to launch them:

```
import { StreamingAppsPanel } from '@components/streaming/StreamingAppsPanel'

<StreamingAppsPanel
  deviceId="fire-tv-1"
  ipAddress="192.168.1.100"
  port={5555}
/>
```

LiveEventsPanel

Shows live and upcoming sports events:

```
import { LiveEventsPanel } from '@components/streaming/LiveEventsPanel'

<LiveEventsPanel
  deviceId="fire-tv-1"
  ipAddress="192.168.1.100"
  port={5555}
/>
```

Environment Variables

Add to your `.env.local` :

```
# NFHS Network API (when available)
NFHS_API_KEY=your_api_key
NFHS_API_SECRET=your_api_secret

# ESPN API (public - no key needed)
# MLB API (public - no key needed)

# Other services (if you get access)
FOX_SPORTS_API_KEY=your_key
NBC_SPORTS_API_KEY=your_key
YOUTUBE_API_KEY=your_key
```

API Endpoints

POST /api/streaming/apps/detect

Detect installed streaming apps on a Fire TV device

Request:

```
{
  "deviceId": "string",
  "ipAddress": "string",
  "port": 5555,
  "forceRefresh": false
}
```

Response:

```
{
  "success": true,
  "installedCount": 5,
  "totalCount": 15,
  "apps": [...]
}
```

POST /api/streaming/launch

Launch a streaming app

Request:

```
{
  "deviceId": "string",
  "ipAddress": "string",
  "appId": "string",
  "port": 5555,
  "deepLink": "optional_deep_link",
  "activityName": "optional_activity"
}
```

GET /api/streaming/events

Get sports events

Query Parameters:

- type : live | today | upcoming | search | installed
- sport : Filter by sport name
- team : Team name for search
- days : Number of days for upcoming events (default: 7)
- deviceId : For installed app events
- ipAddress : For installed app events

GET /api/streaming/status

Get status of all streaming service integrations

Deep Link Formats

Different streaming services use different deep link formats:

```
// NFHS Network
nfhs://event/{eventId}

// ESPN+
espn://x-callback-url/showEvent?eventId={eventId}

// MLB.TV
mlb://game/{gamePk}

// NBA League Pass
nba://game/{gameId}

// Fox Sports
foxsports://video/{videoId}

// NBC Sports
nbcsports://video/{videoId}
```

NFHS API Integration

The NFHS Network API integration is ready but waiting for API access. See [NFHS_API_INTEGRATION.md](#) (./NFHS_API_INTEGRATION.md) for:

- How to request API access
- What data will be available
- Integration steps
- Code locations

Testing

Test App Detection Locally

```
# Requires actual Fire TV device on network
curl -X POST http://localhost:3000/api/streaming/apps/detect \
-H "Content-Type: application/json" \
-d '{
  "deviceId": "test-device",
  "ipAddress": "192.168.1.100",
  "port": 5555
}'
```

Test ESPN API

```
curl http://localhost:3000/api/streaming/events?type=live
```

Test MLB API

```
curl http://localhost:3000/api/streaming/events?type=upcoming&sport=baseball
```

Troubleshooting

App Detection Not Working

- Ensure Fire TV device is connected and ADB is enabled
- Check device is on same network as server
- Verify IP address and port are correct
- Try forceRefresh: true in request

App Launch Fails

- Check if app is actually installed on device
- Verify device connection is active
- Check ADB connection in logs
- Ensure package name is correct

API Returns No Events

- Check service status: GET /api/streaming/status
- Verify internet connection
- Check API service is not down
- Review console logs for errors

Deep Links Not Working

- Verify app supports deep linking
- Check deep link format is correct
- Ensure app is up to date on device
- Some apps require specific setup

Future Enhancements

Planned Features

- [] YouTube API integration for YouTube TV
- [] NBA API integration
- [] NHL API integration
- [] Automatic event scheduling/reminders
- [] Multi-device synchronization
- [] Event recommendations based on preferences
- [] Integration with TV guide overlay

When APIs Become Available

- [] NFHS Network full integration
- [] Fox Sports API integration
- [] NBC Sports API integration
- [] Hulu Live TV API
- [] FuboTV API

Performance Considerations

Caching

- App detection results cached for 5 minutes
- API responses cached where appropriate
- Fire TV connections pooled and reused

Rate Limiting

- ESPN API: No limits (public)
- MLB API: No limits (public)
- Be respectful with API calls
- Implement exponential backoff for failures

Connection Management

- Fire TV connections kept alive automatically
- Automatic reconnection on failure
- Connection pooling for efficiency
- Health monitoring built-in

Contributing

When adding new streaming service integrations:

1. Add app info to `streaming-apps-database.ts`
2. Create API client in `lib/streaming/api-integrations/`
3. Update `unified-streaming-api.ts` to include new service
4. Add conversion methods for event format
5. Update this documentation
6. Add tests

Support

For issues or questions:

- Check server logs for detailed error messages
- Review Fire TV connection status
- Verify API service status
- Check environment variables are set correctly

Last Updated: October 28, 2025

Version: 1.0.0