Points & Rewards System Documentation

Overview

The Mindful Champion platform features a comprehensive points and rewards system that motivates users to engage with the platform through achievements and offers tangible rewards through sponsor partnerships.

Table of Contents

System Architecture

Core Components

1. **Points Tracking** - User reward points stored in database
2. **Achievement Engine** - Monitors user activities and unlocks achievements
3. **Notification System** - Real-time toast notifications for achievements
4. **Redemption System** - Allows users to redeem points for sponsor offers
5. **Tier System** - Progressive rewards tiers (Bronze, Silver, Gold, Platinum)

Data Flow

```
User Action (e.g., Video Upload)
    ↓
API Endpoint (/api/video-analysis/upload)
    ↓
Achievement Check (checkAndAwardSimpleAchievements)
    ↓
Points Awarded (awardPoints)
    ↓
Tier Unlock Check (checkTierUnlocksForUser)
    ↓
Client-side Notification (AchievementToast)
    ↓
Points Badge Update (Navigation Bar)
```

Points System

How Points are Earned

Users earn points by:
- **Uploading videos** - 10-100 points per achievement
- **Completing practice sessions** - 10-100 points per achievement
- **Maintaining streaks** - 150 points for 7-day streak
- **Account milestones** - 75 points for 30-day veteran
- **Upgrading subscription** - 200 points for PRO

Points Display Locations

1. **Navigation Bar** (Desktop & Mobile)
   - Golden badge showing current points
   - Clickable link to marketplace
   - Real-time updates every 30 seconds

2. **Home Dashboard**
   - Quick stats section showing points count
   - Trend indicator (up/down)

3. **Marketplace**
   - Prominent “Your Points” display at top
   - Points needed for each offer
   - Progress bar showing affordability

4. **Profile/Dashboard**
   - Detailed points history
   - Breakdown by category

Points Balance

- **Location in Database**: `User.rewardPoints` field (Integer)
- **Default Value**: 0
- **Updates**: Incremented when achievements unlock or tier bonuses apply
- **Deductions**: When redeeming sponsor offers

Achievement System

Achievement Types

1. Video Achievements

- **FIRST_VIDEO** - Upload first video (10 points)
- **VIDEO_MASTER_5** - Upload 5 videos (50 points)
- **VIDEO_MASTER_10** - Upload 10 videos (100 points)

2. Practice Achievements

- **FIRST_PRACTICE** - Log first practice (10 points)
- **PRACTICE_WARRIOR_5** - Log 5 practices (50 points)
- **PRACTICE_WARRIOR_10** - Log 10 practices (100 points)

- **PRACTICE_STREAK_7** - 7-day practice streak (150 points)

3. Milestone Achievements

- **ACCOUNT_VETERAN_30** - 30 days on platform (75 points)

4. Subscription Achievements

- **PRO_SUBSCRIBER** - Upgrade to PRO (200 points)

Achievement Engine

File: `/lib/achievements/check-simple-achievements.ts`

The achievement engine:
1. Monitors user activities
2. Checks achievement criteria
3. Awards points automatically
4. Sends email notifications
5. Logs achievement unlocks

Trigger Points:
- After video upload
- After practice log
- After subscription change
- Daily cron job for time-based achievements

Rewards & Redemption

Sponsor Offers

Users can redeem points for:
- **Discount codes** (e.g., 20% off equipment)
- **Free products** (e.g., free paddles)
- **Premium content access**
- **Exclusive experiences** (e.g., pro coaching sessions)

Redemption Process

1. User browses marketplace (`/marketplace`)
2. Selects an offer they can afford
3. Confirms redemption
4. Points are deducted
5. Redemption code is generated
6. Email sent with instructions

Redemption API

Endpoint: `POST /api/rewards/redeem`

```
{
  offerId: string,
  shippingInfo?: {
    fullName: string,
    address: string,
    city: string,
    state: string,
    zipCode: string
  }
}
```

Visual Feedback

Achievement Toast Notification

File: `/components/rewards/achievement-toast.tsx`

Features:
- 🎉 **Confetti animation** using canvas-confetti
- 🏆 **Animated trophy icon** with glow effect
- ⭐ **Points badge** showing earned points
- 🎬 **Sequential display** for multiple achievements
- ❌ **Dismissible** with close button or auto-advance

Navigation Points Badge

Location: Top navigation bar (desktop & mobile)

Features:
- Golden gradient background
- Award icon
- Formatted number (e.g., “1,234”)
- Hover effect with shadow
- Links to marketplace
- Real-time updates via polling

Toast Trigger Locations

1. **Video Analysis Page** (`/train/video`)
   - After successful video upload
   - Checks for video-related achievements

2. **Home Dashboard** (`/dashboard`)
   - On page load
   - Checks all achievement types

3. **Practice Log** (future)
   - After logging practice
   - Checks practice achievements

API Endpoints

Check Achievements

POST `/api/rewards/check-achievements`

Checks for new achievements and returns them for display.

Request:

```
{
  "actionType": "video" | "practice" | "subscription" | "all"
}
```

Response:

```
{
  "success": true,
  "newAchievements": [
    {
      "id": "FIRST_VIDEO",
      "name": "First Video Upload",
      "description": "Upload your first video for analysis",
      "icon": "🎥",
      "points": 10
    }
  ],
  "totalPointsEarned": 10,
  "message": "1 new achievement unlocked!"
}
```

Get Achievement Notifications

GET `/api/rewards/check-achievements`

Returns recently unlocked achievements (last 5 minutes).

Response:

```
{
  "success": true,
  "notifications": [
    {
      "id": "FIRST_VIDEO",
      "name": "First Video Upload",
      "description": "Upload your first video for analysis",
      "icon": "🎥",
      "points": 10,
      "unlockedAt": "2024-12-04T10:30:00Z"
    }
  ]
}
```

Get User Points

GET `/api/rewards/user-stats`

Returns current user points and reward tier information.

Response:

```
{
  "success": true,
  "stats": {
    "rewardPoints": 150,
    "currentTier": {
      "name": "SILVER",
      "displayName": "Silver Champion",
      "minPoints": 100,
      "maxPoints": 499
    },
    "nextTier": {
      "name": "GOLD",
      "displayName": "Gold Champion",
      "minPoints": 500
    },
    "pointsToNextTier": 350
  }
}
```

Award Points (Internal)

Function: `awardPoints(userId, points, reason)`

File: `/lib/rewards/award-points.ts`

Directly awards points to a user.

```
const result = await awardPoints(
  'user-123',
  50,
  'Completed VIDEO_MASTER_5 achievement'
);
// Returns: { success: true, newTotal: 150 }
```

Components

AchievementToast

File: `/components/rewards/achievement-toast.tsx`

Usage:

```tsx
import { AchievementToast, useAchievementNotifications } from '@/components/rewards/
achievement-toast';

function MyComponent() {
  const { achievements, isShowing, dismissAchievements, checkForAchievements } = useAc
hievementNotifications();

  // Check for achievements after an action
  const handleAction = async () => {
    // ... perform action
    await checkForAchievements('video');
  };

  return (
    <>
      {/* Your component content */}

      {isShowing && (
        <AchievementToast
          achievements={achievements}
          onDismiss={dismissAchievements}
        />
      )}
    </>
  );
}
```

Navigation Points Badge

File: `/components/navigation/main-navigation.tsx`

Automatically displays points in the navigation bar. Points are fetched and updated every 30 seconds.

Database Schema

User Model

```prisma
model User {
  id                String              @id @default(cuid())
  email             String              @unique
  name              String?
  rewardPoints      Int                 @default(0)  // <-- Points balance
  // ... other fields
}
```

Achievement Model

```
model Achievement {
  id              String              @id @default(cuid())
  achievementId   String              @unique
  name            String
  description     String
  icon            String
  points          Int
  category        AchievementCategory
  tier            AchievementTier
  isActive        Boolean             @default(true)
  // ... other fields
}
```

AchievementProgress Model

```
model AchievementProgress {
  id              String       @id @default(cuid())
  userId          String
  achievementId   String
  currentValue    Int          @default(0)
  targetValue     Int
  percentage      Int          @default(0)
  user            User         @relation(fields: [userId], references: [id])
  achievement     Achievement  @relation(fields: [achievementId], references: [id])

  @@unique([userId, achievementId])
}
```

TierUnlock Model

```
model TierUnlock {
  id                  String       @id @default(cuid())
  userId              String
  tierId              String
  pointsAtUnlock      Int
  emailSent           Boolean      @default(false)
  celebrationShown    Boolean      @default(false)
  // ... other fields
}
```

Integration Guide

Adding Achievement Checks to New Features

1. **Import the achievement checker**:

```
import { checkAndAwardSimpleAchievements } from '@/lib/achievements/check-simple-achievements';
```

1. **Call after user action**:

```
// After successful action
await checkAndAwardSimpleAchievements(userId, 'video');
```

1. **Add achievement toast to component**:

```
import { AchievementToast, useAchievementNotifications } from '@/components/rewards/
achievement-toast';

// In component
const { achievements, isShowing, dismissAchievements, checkForAchievements } = useAchi
evementNotifications();

// After action
await checkForAchievements('video');

// In render
{isShowing && (
  <AchievementToast
    achievements={achievements}
    onDismiss={dismissAchievements}
  />
)}
```

Creating New Achievements

Add to `/lib/achievements/check-simple-achievements.ts`:

```
{
  id: 'MY_NEW_ACHIEVEMENT',
  name: 'Achievement Name',
  description: 'Achievement description',
  icon: '🏆',
  points: 50,
  category: 'video',
  checkFunction: async (userId: string) => {
    // Check if criteria is met
    const count = await prisma.someModel.count({
      where: { userId }
    });
    return count >= 10;
  }
}
```

Troubleshooting

Points Not Showing Up

1. **Check database**: Query `User.rewardPoints` to verify points are saved
2. **Check achievement unlock**: Look for achievement progress records
3. **Check logs**: Search for “Awarded X points” in server logs
4. **Refresh**: Wait 30 seconds for navigation badge to update

Achievement Not Triggering

1. **Verify criteria**: Double-check the checkFunction logic
2. **Check if already unlocked**: Query AchievementProgress for percentage = 100
3. **Test manually**: Call `/api/rewards/check-achievements` endpoint
4. **Check logs**: Look for achievement check errors

Toast Not Appearing

1. **Check component integration**: Verify AchievementToast is in render
2. **Check isShowing state**: Console log the isShowing value
3. **Verify API response**: Check network tab for check-achievements call
4. **Check z-index**: Toast has z-index: 9999

Performance Considerations

1. **Polling**: Navigation badge polls every 30 seconds (reasonable for real-time feel)
2. **Achievement checks**: Run asynchronously to not block user actions
3. **Database queries**: Optimized with proper indexes on userId fields
4. **Toast animations**: Uses framer-motion for smooth, performant animations

---

Future Enhancements

- [ ] Push notifications for achievement unlocks
- [ ] Leaderboard showing top point earners
- [ ] Weekly/monthly point challenges
- [ ] Referral bonuses (earn points for inviting friends)
- [ ] Social sharing of achievements
- [ ] Achievement badges on user profiles
- [ ] Points history/transaction log
- [ ] Bonus point multipliers during special events

Support

For questions or issues with the points system:
- Check this documentation first
- Review server logs for errors
- Test API endpoints directly
- Verify database state

Last Updated: December 4, 2024
Version: 1.0.0