# Mindful Champion SMS Integration Guide

## Overview

This guide documents the complete SMS functionality integrated into Mindful Champion using Twilio. The system supports SMS-based password reset, phone number verification, and two-factor authentication (2FA).

## Table of Contents

---

## Setup & Configuration

### Environment Variables

The following Twilio credentials are configured in `.env`:

```
TWILIO_ACCOUNT_SID=your_twilio_account_sid_here
TWILIO_AUTH_TOKEN=your_twilio_auth_token_here
TWILIO_PHONE_NUMBER=your_twilio_phone_number_here
```

### Database Schema

The system uses these Prisma models:

1. **User Model** - Contains SMS/2FA fields:
   - `phoneNumber`: User’s phone number (E.164 format)
   - `phoneNumberVerified`: Boolean flag for verification status
   - `phoneVerifiedAt`: Timestamp of verification
   - `twoFactorEnabled`: Boolean flag for 2FA status
   - `twoFactorSecret`: Temporary 2FA code storage
   - `twoFactorExpiry`: Code expiration time
   - `twoFactorBackupCodes`: Array of backup codes

2. **SMSVerificationCode Model** - Stores verification codes:
   - `userId`: Associated user ID
   - `phoneNumber`: Target phone number
   - `code`: 6-digit verification code
   - `type`: Verification type (PASSWORD_RESET, PHONE_VERIFICATION, TWO_FACTOR_AUTH, etc.)

  - `expiresAt`: Expiration timestamp (10 minutes)
  - `used`: Whether code was used
  - `usedAt`: When code was used
  - `attemptsCount`: Failed verification attempts

---

# Features

## 1. SMS Password Reset

**Flow:**
1. User enters their phone number on forgot password page
2. System checks if phone is verified and associated with account
3. 6-digit code sent via SMS (expires in 10 minutes)
4. User enters code and new password
5. Password is reset after successful verification

**API Endpoints:**
- `POST /api/auth/sms/request-password-reset` - Request code
- `POST /api/auth/sms/verify-and-reset` - Verify code and reset password

## 2. Phone Number Verification

**Flow:**
1. User adds/updates phone number in profile
2. Verification code sent via SMS
3. User enters code to verify ownership
4. Phone marked as verified in database

**API Endpoints:**
- `POST /api/auth/sms/send-phone-verification` - Send verification code
- `POST /api/auth/sms/verify-phone` - Verify phone number

## 3. Two-Factor Authentication (2FA)

**Enable 2FA Flow:**
1. User must have verified phone number
2. Requests 2FA enable from settings
3. Receives verification code via SMS
4. Verifies code to confirm
5. Receives 10 backup codes for safekeeping
6. 2FA is enabled

**Login with 2FA:**
1. User enters email and password
2. If 2FA enabled, SMS code is sent
3. User enters 6-digit code (or backup code)
4. Access granted upon successful verification

**API Endpoints:**
- `POST /api/auth/2fa/enable` - Start 2FA enable process
- `POST /api/auth/2fa/verify-and-enable` - Complete 2FA enable
- `POST /api/auth/2fa/disable` - Disable 2FA (requires password)

- `POST /api/auth/2fa/send-code` - Send 2FA code during login
- `POST /api/auth/2fa/verify` - Verify 2FA code

---

# API Routes

## User Routes

### Request SMS Password Reset

```
POST /api/auth/sms/request-password-reset
Body: { phoneNumber: string }
Response: { success: boolean, message: string }
```

### Verify Code and Reset Password

```
POST /api/auth/sms/verify-and-reset
Body: {
  phoneNumber: string,
  code: string,
  newPassword: string
}
Response: { success: boolean, message: string }
```

### Send Phone Verification

```
POST /api/auth/sms/send-phone-verification
Headers: Authorization (session required)
Body: { phoneNumber: string }
Response: { success: boolean, message: string }
```

### Verify Phone Number

```
POST /api/auth/sms/verify-phone
Headers: Authorization (session required)
Body: { code: string }
Response: { success: boolean, message: string }
```

### Enable 2FA

```
POST /api/auth/2fa/enable
Headers: Authorization (session required)
Response: {
  success: boolean,
  backupCodes: string[],
  requiresVerification: boolean
}
```

### Verify and Complete 2FA Enable

```
POST /api/auth/2fa/verify-and-enable
Headers: Authorization (session required)
Body: { code: string }
Response: {
  success: boolean,
  backupCodes: string[]
}
```

### Disable 2FA

```
POST /api/auth/2fa/disable
Headers: Authorization (session required)
Body: { password: string }
Response: { success: boolean, message: string }
```

### Send 2FA Code

```
POST /api/auth/2fa/send-code
Body: { email: string }
Response: {
  success: boolean,
  requires2FA: boolean
}
```

### Verify 2FA Code

```
POST /api/auth/2fa/verify
Body: {
  email: string,
  code: string,
  useBackupCode?: boolean
}
Response: {
  success: boolean,
  userId: string
}
```

## Admin Routes

### Send Admin Password Reset

```
POST /api/admin/sms/send-password-reset
Headers: Authorization (admin session required)
Body: { userId: string }
Response: { success: boolean, message: string }
```

### Manually Verify User Phone

```
POST /api/admin/sms/verify-user-phone
Headers: Authorization (admin session required)
Body: { userId: string }
Response: { success: boolean, message: string }
```

### Get SMS Logs

```
GET /api/admin/sms/sms-logs?userId={userId}&limit={limit}&offset={offset}
Headers: Authorization (admin session required)
Response: {
  success: boolean,
  data: {
    logs: SMSVerificationCode[],
    pagination: { total, limit, offset, hasMore },
    stats: { type: string, count: number }[]
  }
}
```

### Get SMS Statistics

```
GET /api/admin/sms/stats
Headers: Authorization (admin session required)
Response: {
  success: boolean,
  data: {
    overview: {
      totalSMSSent: number,
      totalUsersWithPhone: number,
      totalVerifiedPhones: number,
      total2FAEnabled: number,
      smsLastHour: number,
      smsToday: number,
      successRate: string
    },
    byType: Record<string, number>
  }
}
```

---

# Testing with Twilio Trial

## Important Limitations

⚠️ **Twilio Trial Account Restrictions:**

1. **Verified Numbers Only**: In trial mode, SMS can ONLY be sent to verified phone numbers
2. **Test Phone Number**: Use `+19542348040` (954-234-8040) for testing
3. **Trial Message Prefix**: All trial SMS messages include: “Sent from your Twilio trial account -“

## Verifying Test Numbers

To test SMS functionality:

1. **Verify Your Phone in Twilio Console:**
   - Go to: https://console.twilio.com
   - Navigate to: Phone Numbers > Manage > Verified Caller IDs
   - Click “Add a new Caller ID”
   - Enter: +19542348040
   - Complete verification process

2. **Test the Integration:**

   ```
   bash
      # Test SMS password reset
      curl -X POST http://localhost:3000/api/auth/sms/request-password-reset \
        -H "Content-Type: application/json" \
        -d '{"phoneNumber": "9542348040"}'
   ```

### Common Twilio Error Codes

- **21614**: “Phone number not verified” - Add number to verified caller IDs
- **21211**: “Invalid phone number format” - Use E.164 format (+1XXXXXXXXXX)
- **21408**: “Permission denied” - Check account permissions

---

## Security Features

### Rate Limiting

1. **SMS Rate Limiting**: 5 SMS per hour per phone number
2. **Verification Attempts**: Maximum 3 attempts per code
3. **Code Expiration**: All codes expire after 10 minutes

### Code Generation

- Uses `crypto.randomBytes()` for secure code generation
- 6-digit codes (100000-999999)
- Cryptographically secure random generation

### Protection Against Attacks

1. **Phone Enumeration**: Always returns success, doesn’t reveal if phone exists
2. **Brute Force Protection**: Limited verification attempts
3. **Code Reuse Prevention**: Codes marked as used after successful verification
4. **Time-based Expiration**: Automatic code expiration

### Logging & Auditing

- All admin SMS actions logged to `SecurityLog` table
- Includes: admin ID, target user, timestamp, action type
- Failed attempts tracked in `attemptsCount`

---

## Admin Controls

### Admin Dashboard Features

Admins can:

1. **Send Password Reset SMS**: Force password reset for users
2. **Manually Verify Phones**: Verify phone numbers without SMS
3. **View SMS Logs**: See all SMS activity with filters
4. **View Statistics**: Monitor SMS usage and success rates

### Admin SMS Panel Location

Access via: `/admin/sms-management` (to be created in UI)

### Security Logs

All admin SMS actions create security log entries:

```
{
  userId: targetUser.id,
  eventType: 'PASSWORD_RESET_REQUEST',
  severity: 'MEDIUM',
  description: 'Admin action description',
  metadata: {
    adminId, adminEmail,
    targetUserId, targetUserEmail
  }
}
```

---

## Troubleshooting

### SMS Not Received

1. **Check Phone Format**: Must be E.164 (+1XXXXXXXXXX)
2. **Verify in Twilio**: Trial accounts require phone verification
3. **Check Logs**: Look for error messages in server console
4. **Rate Limiting**: Wait if 5 SMS already sent in last hour

### Invalid Code Errors

1. **Code Expired**: Codes expire after 10 minutes
2. **Too Many Attempts**: Max 3 attempts per code
3. **Already Used**: Codes are single-use only
4. **Wrong Phone**: Code must match phone number

### 2FA Issues

1. **Phone Not Verified**: Must verify phone before enabling 2FA
2. **No Backup Codes**: Save backup codes when enabling 2FA
3. **Lost Access**: Use backup codes to regain access

### Twilio Configuration Issues

1. **Check Environment Variables**: Verify `.env` has correct credentials
2. **Account Status**: Ensure Twilio account is active
3. **Phone Number Active**: Verify Twilio number is active and SMS-capable

---

# Utility Functions

## Phone Number Formatting

```
import {
  normalizePhoneNumber,
  formatPhoneNumber,
  isValidPhoneNumber
} from '@/lib/sms/twilio';

// Normalize to E.164
const normalized = normalizePhoneNumber('(954) 234-8040');
// Result: "+19542348040"

// Format for display
const formatted = formatPhoneNumber('+19542348040');
// Result: "+1 (954) 234-8040"

// Validate format
const isValid = isValidPhoneNumber('9542348040');
// Result: true
```

## Sending Custom SMS

```
import { sendSMS } from '@/lib/sms/twilio';

const result = await sendSMS(
  '+19542348040',
  'Your custom message here',
  { skipRateLimit: false } // Optional
);

if (result.success) {
  console.log('SMS sent:', result.sid);
} else {
  console.error('SMS failed:', result.error);
}
```

---

# Production Upgrade

When upgrading to a production Twilio account:

1. **Remove Verified Number Restriction**: All numbers will work
2. **Increase Rate Limits**: Adjust `MAX_SMS_PER_HOUR` as needed
3. **Enable International**: Add international phone support
4. **Setup Redis**: Replace in-memory cache with Redis
5. **Monitor Usage**: Track SMS costs and usage patterns

---

## Support

For issues or questions:
- Email: info@mindfulchampion.com
- Twilio Support: https://support.twilio.com

---

## Change Log

- **v1.0** (2025-10-24): Initial SMS integration
- SMS password reset
- Phone verification
- 2FA implementation
- Admin controls