Social Media Video Generation Feature

Overview

This feature enables users to generate 30-second social media optimized videos (reels/shorts) from their community posts in Writers Corner. The videos are designed for platforms like Instagram Reels, YouTube Shorts, and TikTok.

Feature Components

1. API Endpoint

Location: `app/app/api/community/posts/[id]/generate-video/route.ts`

Endpoints:
- `POST /api/community/posts/[id]/generate-video` - Initiates video generation
- `GET /api/community/posts/[id]/generate-video` - Returns video generation capabilities

Features:
- User authentication and authorization
- Post ownership verification
- Video generation job creation
- Support for multiple formats (vertical, square, horizontal)
- Configurable durations (15s, 30s, 60s)
- Multiple style options (vintage-typewriter, modern, minimalist)

2. UI Component

Location: `app/components/community/generate-video-button.tsx`

Features:
- Interactive button with loading states
- Toast notifications for user feedback
- Error handling and retry logic
- Visual feedback (icons change based on state)
- Disabled state during generation

3. Integration

Location: `app/components/community/community-overview.tsx`

The video generation button is integrated into each community post card, appearing alongside:
- Like button
- Comment button
- Download button

Technical Implementation

Current Implementation (MVP)

The current implementation provides the foundation for video generation:

1. **API Structure**: Complete REST API endpoints with proper authentication
2. **UI Components**: Fully functional button with state management
3. **User Experience**: Toast notifications and visual feedback
4. **Security**: Authorization checks to ensure users can only generate videos for their own posts

Production Integration Requirements

To make this feature fully functional in production, integrate with:

1. Video Generation Service

Options:
- **FFmpeg** (self-hosted): For video processing and rendering
- **Remotion** (React-based): For programmatic video generation
- **Cloudinary** or **Mux**: Cloud-based video processing APIs
- **Custom ML Pipeline**: For AI-powered video generation

2. Text-to-Speech (Optional)

For narration:
- Google Cloud Text-to-Speech
- Amazon Polly
- ElevenLabs
- Azure Cognitive Services

3. Background Assets

- Stock video footage APIs (Pexels, Unsplash)
- Background music libraries (royalty-free)
- Animated text overlays
- Branding elements (logo, watermark)

4. Cloud Storage

For hosting generated videos:
- AWS S3
- Google Cloud Storage
- Cloudinary
- Vercel Blob Storage

5. Job Queue System

For async processing:
- Bull (Redis-based)
- AWS SQS
- Google Cloud Tasks
- Vercel Edge Functions with background jobs

Recommended Architecture

```
User Request → API Endpoint → Job Queue → Video Generation Worker
                                              ↓
                                         Cloud Storage
                                              ↓
                                         Webhook/Polling
                                              ↓
                                         User Notification
```

Video Generation Workflow

1. **User clicks “Generate Reel” button**
2. **Frontend sends POST request** to `/api/community/posts/[id]/generate-video`
3. **Backend validates** user authentication and post ownership
4. **Job is created** and added to processing queue
5. **Worker processes job**:
   - Extract post content and metadata
   - Generate video scenes with text overlays
   - Add background music and transitions
   - Apply vintage typewriter aesthetic
   - Render final video
6. **Video is uploaded** to cloud storage
7. **User is notified** via webhook or polling
8. **Download link** is provided to user

Video Specifications

Default Settings

- **Duration**: 30 seconds
- **Format**: Vertical (9:16 for reels/shorts)
- **Resolution**: 1080x1920 (Full HD)
- **Frame Rate**: 30 fps
- **Style**: Vintage typewriter aesthetic

Customization Options

- Duration: 15s, 30s, or 60s
- Format: Vertical, Square (1:1), or Horizontal (16:9)
- Style: Vintage-typewriter, Modern, or Minimalist

Testing Instructions

Local Testing

1. **Install dependencies**:

   ```
   bash
     cd app
     yarn install
   ```

2. **Set up environment variables**:

   ```
   bash
      # .env.local
      DATABASE_URL="your_database_url"
      NEXTAUTH_SECRET="your_secret"
      NEXTAUTH_URL="http://localhost:3000"
   ```

3. **Run database migrations**:

   ```
   bash
      npx prisma generate
      npx prisma db push
   ```

4. **Start development server**:

   ```
   bash
      yarn dev
   ```

5. **Test the feature**:
   - Navigate to http://localhost:3000/community
   - Sign in with a test account
   - Find a post you own
   - Click “Generate Reel” button
   - Verify toast notification appears
   - Check browser console for API response
   - Verify API endpoint returns expected JSON structure

API Testing with cURL

```
# Get video generation capabilities
curl -X GET http://localhost:3000/api/community/posts/[POST_ID]/generate-video \
  -H "Cookie: next-auth.session-token=YOUR_SESSION_TOKEN"

# Initiate video generation
curl -X POST http://localhost:3000/api/community/posts/[POST_ID]/generate-video \
  -H "Cookie: next-auth.session-token=YOUR_SESSION_TOKEN" \
  -H "Content-Type: application/json"
```

Future Enhancements

1. **Video Templates**: Pre-designed templates for different writing genres
2. **Custom Branding**: User-configurable colors, fonts, and logos
3. **AI Narration**: Automatic text-to-speech narration of the post
4. **Music Selection**: User choice of background music
5. **Batch Generation**: Generate videos for multiple posts at once
6. **Analytics**: Track video views and engagement
7. **Direct Sharing**: One-click sharing to social media platforms
8. **Video Editor**: In-app video editing capabilities
9. **Captions**: Auto-generated captions for accessibility
10. **Thumbnail Generation**: Custom thumbnail creation

Security Considerations

- ✅ User authentication required
- ✅ Post ownership verification
- ✅ Rate limiting (recommended for production)
- ✅ Input validation and sanitization
- ✅ Error handling and logging
- ⚠️ Add CSRF protection in production
- ⚠️ Implement video generation quotas per user
- ⚠️ Add content moderation for generated videos

Performance Considerations

- Video generation should be asynchronous (job queue)
- Implement caching for frequently generated videos
- Use CDN for video delivery
- Optimize video file sizes for faster loading
- Consider progressive video loading
- Implement retry logic for failed generations

Cost Estimation (Production)

Approximate costs per 1000 videos:
- Video processing: $5-20 (depending on service)
- Cloud storage: $0.50-2
- CDN bandwidth: $1-5
- Text-to-speech (optional): $4-16
- Total: ~$10-43 per 1000 videos

Support and Maintenance

- Monitor video generation success rates
- Track average generation time
- Log errors and failures for debugging
- Collect user feedback on video quality
- Regular updates to video templates and styles

License

This feature is part of The Writer’s Corner project and follows the same MIT license.