Commands

- Command for api testing

```
curl -L -A "Mozilla/5.0" "api22-normal-c-useast1a.tiktokv.com" -o video.mp4
```

- Command for tunneling through cloudflare

```
cloudflared tunnel --url http://localhost:3000
```

- Command to test the endpoint with token **[PowerShell]**

```
$h = @{ "x-api-token"="my_strong_token"; "Content-Type"="application/json" }
Invoke-RestMethod -Uri "http://localhost:3000/api/download" -Method POST -
Headers $h -Body '{"url":"https://example.com/video"}'
```

- Command set a new value into env token **[PowerShell]**

```
setx API_TOKEN "my_token"
```

- Install PM2 globally

```
npm install -g pm2
```

- Command to start backend from server.js file

```
pm2 start server.js --name ttd-backend
```

- Command to restart a service and set a new env value for **pm2**

```
npx pm2 restart ttd-backend --update-env
```

PM2 Production Commands

- Start with ecosystem file (development)

```
pm2 start ecosystem.config.cjs
```

- Start with ecosystem file (production)

```
pm2 start ecosystem.config.cjs --env production
```

- Reload (graceful restart, zero downtime)

```
pm2 reload ttd-backend
```

- Stop all PM2 services

```
pm2 stop all
```

- Delete process from PM2

```
pm2 delete ttd-backend
```

- Save current PM2 state (for autorestart on reboot)

```
pm2 save
pm2 startup
```

- Monitor logs in real-time

```
pm2 logs ttd-backend
```

- Show PM2 status

```
pm2 status
```

Expo Go Commands

```

```

```
npx create-expo-app tiktok-mobile
cd tiktok-mobile
npx expo install expo-file-system expo-sharing
npx expo start
```

USB testing, Run ADB Reverse(This tells the phone: "If the app looks for localhost:3000, send it to the PC's port 3000"):

```
adb reverse tcp:3000 tcp:3000
```

Build App via Electron

Windows/Linux/macOS commands for deploy

```
npx electron-builder --win
electron-builder --linux AppImage
electron-builder --mac
```

Used information

- Used api: https://api22-normal-c-alisg.tiktokv.com/aweme/v1/feed/?aweme_id=...
- Tiktok video for demo: https://www.tiktok.com/@arkhamsorigin/video/7462239608529521950?q=destroy%20arasaka&t=175760085...
- github [Xlinka TikTok-Downloader C#]:
- Old api: https://api16-normal-c-useast1a.tiktokv.com/aweme/v1/play/?video_id=...
- Cloudflare Tunnel

Deployment on Railway

Requirements

- GitHub account
- Railway account (free tier available)

Steps

1. Go to railway.app create Railway account
2. Click "New Project" → "Deploy from GitHub"
3. Select `tiktok-video-downloader` repository
4. Railway auto-detects Node.js and deploys
5. In Railway Dashboard → Environment Variables:

    - Add `API_TOKEN=your_secure_token`
    - Add `PORT=3000` (optional, Railway sets this automatically)
6. Auto Deploy: Push to `main` branch = instant deployment
7. Your app URL: `https://your-project-name.railway.app`

Features for Railway deployment

- Video is streamed directly to browser (no file storage needed)
- Automatic SSL/HTTPS
- Auto-restart on crashes
- GitHub auto-deploy (push to main = instant deploy)
- Free tier: $5/month credits

Modules Overview

`downloader.js`

Handles direct downloads:

- Saves video or photo post files locally.
- Supports resuming (Range headers).
- Ensures unique filenames.

`hlsParser.js`

Parses `.m3u8` HLS playlists:

- Detects multiple quality variants.
- Lets users choose resolution/bitrate.
- Returns direct stream URLs.

`audioExtractor.js`

Uses FFmpeg to:

- Extract audio track (`.mp3`) from video.
- Save in chosen folder.

`frameExtractor.js`

Uses FFmpeg to:

- Split video into frames (`frame-%d.jpg`).
- Extract specific photos (from TikTok "photo mode" posts).

`utils.js`

Helper functions:

- Path management.

- Filename generation.
- Logging and error handling.
