YouTube Downloader - Frontend

Modern, responsive web interface for the YouTube Downloader API.

📁 Directory Structure

```
frontend/
├── index.html          # Main HTML file
├── css/
│   └── styles.css      # All styles (modern, responsive)
├── js/
│   ├── config.js       # Configuration & API settings
│   ├── api.js          # API communication module
│   ├── ui.js           # UI manipulation & helpers
│   └── app.js          # Main application logic
└── assets/             # (Optional) Images, icons, etc.
```

🚀 Quick Start

1. Start the Backend

First, make sure your Flask backend is running:

```bash
cd /path/to/backend
python app.py
```

Backend should be running on `http://localhost:5000`

2. Configure Frontend

Open `js/config.js` and update the API URL if needed:

```javascript
const CONFIG = {
    API_BASE_URL: 'http://localhost:5000',  // Change if backend is on different URL
    // ... other config
};
```

3. Serve the Frontend

Option A - Simple Python Server:

```bash
cd frontend
python -m http.server 8000
```

Then open: `http://localhost:8000`

Option B - Using Node.js:

```bash
npm install -g http-server
cd frontend
http-server -p 8000
```

Option C - VS Code Live Server:

1. Install "Live Server" extension
2. Right-click `index.html`
3. Select "Open with Live Server"

✨ Features

🎬 Download Tab

- **URL Input** - Paste any YouTube URL
- **Video Preview** - Shows thumbnail, title, uploader, views, duration
- **Download Options:**
  - Video (multiple qualities: Best, 1080p, 720p, 480p, 360p)
  - Audio (MP3 format)
- **Real-time Progress** - See download speed, ETA, progress bar
- **Auto-download** - File downloads automatically when complete

📜 History Tab

- View all downloaded files
- See file size, type, and date
- Quick re-download button
- Auto-refreshes after new downloads

📊 Stats Tab

- Total files stored
- Storage used (MB)
- Active downloads
- Completed downloads
- Manual cleanup button

🎨 UI Features

- **Responsive Design** - Works on mobile, tablet, desktop
- **Toast Notifications** - Non-intrusive feedback
- **Loading States** - Spinners and overlays
- **Error Handling** - Clear error messages
- **Modern Design** - Clean, gradient accents, smooth animations

🎨 Customization

Change Colors

Edit `css/styles.css`:

```css
:root {
    --primary: #3b82f6;       /* Main blue color */
    --primary-dark: #2563eb;  /* Darker blue */
    --success: #10b981;       /* Green for success */
    --error: #ef4444;         /* Red for errors */
}
```

Change Fonts

Update the Google Fonts import in `index.html`:

```html
<link href="https://fonts.googleapis.com/css2?family=YourFont:wght@400;600;700&display=swap" rel="stylesheet">
```

Then update in `css/styles.css`:

```css
```

```
:root {
    --font-family: 'YourFont', sans-serif;
}
```

Add More Quality Options

Edit `js/config.js`:

```javascript
VIDEO_QUALITIES: [
    { value: 'best', label: 'Best Quality' },
    { value: '2160', label: '4K (2160p)' },  // Add this
    { value: '1080', label: '1080p (Full HD)' },
    // ... more options
]
```

🔧 Configuration Options

API Settings (`js/config.js`)

```javascript
const CONFIG = {
    // Backend URL
    API_BASE_URL: 'http://localhost:5000',

    // Progress polling (ms)
    PROGRESS_POLL_INTERVAL: 1000,

    // Toast duration (ms)
    TOAST_DURATION: 3000,

    // Max URL length
    MAX_URL_LENGTH: 2048,

    // Feature flags
    FEATURES: {
        AUTO_REFRESH_STATS: true,
        SHOW_THUMBNAILS: true,
        ENABLE_HISTORY: true
    }
};
```

📱 Browser Support

- ✅ Chrome 90+
- ✅ Firefox 88+
- ✅ Safari 14+
- ✅ Edge 90+
- ✅ Mobile browsers (iOS Safari, Chrome Mobile)

🎯 Key Files Explained

index.html

- Semantic HTML5 structure
- Three main tabs: Download, History, Stats
- Toast notification system
- Loading overlay

css/styles.css

- CSS variables for easy theming
- Modern design with gradients & shadows
- Fully responsive with media queries
- Smooth animations & transitions

js/config.js

- All configuration in one place
- API endpoints
- Feature flags
- Quality options

js/api.js

- API wrapper functions
- Progress tracking class
- Error handling
- Clean async/await patterns

`js/ui.js`

- DOM manipulation helpers
- UI state management
- Formatting functions (time, filesize, etc.)
- Toast & loading controls

`js/app.js`

- Main application logic
- Event handlers
- State management
- Coordinates between API & UI

🪲 Troubleshooting

"Cannot connect to server"

- ✅ Check if backend is running on `http://localhost:5000`
- ✅ Update `API_BASE_URL` in `config.js`
- ✅ Check browser console for CORS errors

CORS Errors

- ✅ Backend has `flask-cors` installed
- ✅ Backend runs with `CORS(app)`
- ✅ Try accessing backend directly in browser

Download Button Not Working

- ✅ Click "Get Info" first to load video data
- ✅ Check browser console for errors
- ✅ Verify URL is valid YouTube link

Progress Not Updating

- ✅ Check `PROGRESS_POLL_INTERVAL` in config
- ✅ Verify backend progress endpoint works
- ✅ Check browser console for API errors

Styles Not Loading

- ✅ Check file paths are correct
- ✅ Clear browser cache (Ctrl+Shift+R)
- ✅ Verify CSS file exists in `css/styles.css`

🚀 Performance Tips

1. **Lazy Load History** - Only load when tab is active
2. **Debounce URL Input** - Wait for user to finish typing
3. **Cache Video Info** - Store in sessionStorage
4. **Optimize Images** - Use WebP thumbnails when available
5. **Service Worker** - Add for offline support (advanced)

🔒 Security Notes

- ✅ All user input is escaped (XSS prevention)
- ✅ URL validation before API calls
- ✅ No eval() or dangerous innerHTML usage
- ✅ HTTPS recommended for production
- ⚠️ Add authentication for production use

📦 Production Deployment

1. Build Optimization

- Minify CSS (`cssnano`, `clean-css`)
- Minify JavaScript (`uglify-js`, `terser`)
- Optimize images
- Enable gzip compression

2. Hosting Options

- **Static Hosting:** Netlify, Vercel, GitHub Pages
- **Traditional:** Nginx, Apache
- **Cloud:** AWS S3, Google Cloud Storage

3. Environment Variables

Create `config.prod.js`:

```javascript
const CONFIG = {
  API_BASE_URL: 'https://api.yourdomain.com',
  // ... production config
};
```

4. CDN

Use CDN for faster font/library loading:

```html
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/...">
```

🎨 Screenshots

Download Tab

- Clean URL input with instant paste detection
- Beautiful video preview cards
- Easy quality selection

History Tab

- List of all downloads
- Quick re-download buttons
- File size and timestamps

Stats Tab

- Server statistics overview
- Storage management
- Manual cleanup option

📝 License

MIT License - Feel free to use and modify!

🤝 Contributing

1. Fork the repository
2. Create feature branch (`git checkout -b feature/AmazingFeature`)
3. Commit changes (`git commit -m 'Add AmazingFeature'`)
4. Push to branch (`git push origin feature/AmazingFeature`)
5. Open Pull Request

💡 Future Enhancements

- [ ] Dark mode toggle
- [ ] Playlist support
- [ ] Batch downloads
- [ ] Download queue
- [ ] User preferences (local storage)
- [ ] PWA support (install as app)
- [ ] Subtitle download
- [ ] More video platforms
- [ ] Custom output filenames
- [ ] Thumbnail generation

📞 Support

Issues? Questions? Open an issue on GitHub or contact the maintainers.

Made with ❤️ for the community