Mathes Online Utilities - Project Documentation

Project Overview

A web-based system for managing Excel spreadsheets and other utilities from anywhere via browser. The system uses Flask backends running on a Windows 11 machine (always-on Plex server), exposed to the internet via Cloudflare Tunnel, with static frontend pages hosted on GitHub Pages.

Architecture

Core Components

1. **Windows 11 Machine** (192.168.178.21)
   - Always running (Plex server)
   - Python 3.13.7 installed at: `D:\Misc\Python313\python.exe`
   - Flask backends run on multiple ports (5000, 5001, etc.)
   - Auto-start via Windows Task Scheduler
2. **Cloudflare Tunnel**
   - Single tunnel: `excel-backend` (Tunnel ID stored in config)
   - Multiple subdomains route to different local ports
   - Config location: `C:\Users\[USERNAME]\.cloudflared\config.yml`
   - Credentials: `C:\Users\[USERNAME]\.cloudflared\[TUNNEL-ID].json`
3. **GitHub Pages**
   - Repository: `dennyrgood/excel-web-interface`
   - Live at: `https://dennyrgood.github.io/`
   - Contains static HTML forms and landing page
4. **Domain**
   - `ldmathes.cc` (registered with Cloudflare)
   - Subdomains managed via Cloudflare DNS

Current Applications

Application 1: Excel Quick-Add Form

Purpose: Simple form to add rows to Movies & Shows Excel spreadsheet

Backend:

- File: `D:\OneDrive\MS\excel_backend.py`
- Port: `5000`
- Tunnel: `https://api.ldmathes.cc`
- Endpoints:
  - `GET /api/health` - Health check
  - `POST /api/submit` - Add new row

Frontend:

- Path: `/excel-web-interface/`
- URL: `https://dennyrgood.github.io/excel-web-interface/`
- Features: Multi-project dropdown (Movies/Shows active, others placeholder)

Excel File:

- Location: `D:\OneDrive\MS\MoviesShows.xlsx`
- Columns: A (Code), F-M (Title, Season, Watch, Made, Date, Website, Notes, Synopsis)
- Formula columns: B, C, D, E (auto-calculated, preserved on row insert)

Application 2: Movies & Shows Full CRUD Editor

Purpose: Full add/edit/delete interface for Movies & Shows spreadsheet

Backend:

- File: `D:\OneDrive\MS\excel_backend_full_edit.py`
- Port: `5001`
- Tunnel: `https://api-edit.ldmathes.cc`
- Endpoints:
  - `GET /api/health` - Health check
  - `GET /api/data` - Fetch all rows
  - `POST /api/add` - Add new row
  - `POST /api/update` - Update existing row (requires `row_index` parameter)
  - `POST /api/delete` - Delete row (requires `row_index` parameter)

Frontend:

- Path: `/MoviesShowsFullEdit/`
- URL: `https://dennyrgood.github.io/MoviesShowsFullEdit/`
- Features:
  - Tab 1: Add new entries
  - Tab 2: Edit existing (dropdown selector, pre-populated form)
  - Tab 3: Delete entries (table view, two-click confirmation)

Excel File: Same as Application 1

Hub Landing Page

Purpose: Central dashboard linking to all utilities

Location:

- Path: `/` (root)
- URL: `https://dennyrgood.github.io/`

Features:

- Title: "Mathes Online Utilities"
- Backend health check (shows online/offline status)
- Project cards (grid layout)
- Link to USDZ version: `index_usdz.html`

System Configuration

Cloudflare Tunnel Configuration

Current Config (`C:\Users\[USERNAME]\.cloudflared\config.yml`):

```yaml
tunnel: [TUNNEL-ID]
credentials-file: C:\Users\[USERNAME]\.cloudflared\[TUNNEL-ID].json

ingress:
  - hostname: api.ldmathes.cc
    service: http://localhost:5000
  - hostname: api-edit.ldmathes.cc
    service: http://localhost:5001
  - service: http_status:404
```

Active Routes:

Subdomain	Local Port	Purpose
`api.ldmathes.cc`	5000	Excel Quick-Add backend
`api-edit.ldmathes.cc`	5001	Excel Full Editor backend

Windows Task Scheduler Setup

Task 1: Cloudflared Tunnel

- Name: `Cloudflared Tunnel`
- Trigger: At startup
- Program: `D:\OneDrive\MS\cloudflared.exe`
- Arguments: `tunnel run excel-backend`
- Start in: `D:\OneDrive\MS\`
- Settings: ✅Run whether logged in or not, ✅Highest privileges, ✅Auto-restart on failure

Task 2: Flask Backend (Port 5000)

- Name: `Flask Excel Backend`
- Trigger: At startup
- Program: `D:\Misc\Python313\python.exe`
- Arguments: `D:\OneDrive\MS\excel_backend.py`
- Start in: `D:\OneDrive\MS\`
- Settings: ✅Run whether logged in or not, ✅Highest privileges, ✅Auto-restart on failure

Task 3: Flask Backend (Port 5001)

- Name: `Flask Full Edit Backend`
- Trigger: At startup
- Program: `D:\Misc\Python313\python.exe`
- Arguments: `D:\OneDrive\MS\excel_backend_full_edit.py`
- Start in: `D:\OneDrive\MS\`
- Settings: ✅Run whether logged in or not, ✅Highest privileges, ✅Auto-restart on failure

---

How to Add New Applications

Option A: Add New Tunneled Backend Application

Use this when you want to run a Flask/Python backend on the Windows machine and expose it via Cloudflare Tunnel.

Step 1: Create Backend

1. Create new Python file (e.g., `D:\OneDrive\MS\my_new_app.py`)
2. Choose a unique port (e.g., `5002`, `5003`, etc.)
3. Ensure Flask runs on that port:

```python
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5002, debug=False)
```

4. Test locally: `python my_new_app.py`
5. Verify: `http://localhost:5002/api/health` (or your endpoint)

Step 2: Add Cloudflare Tunnel Route

1. Edit tunnel config: `C:\Users\[USERNAME]\.cloudflared\config.yml`
2. Add new ingress entry BEFORE the `http_status:404` line:

```yaml
ingress:
  - hostname: api.ldmathes.cc
    service: http://localhost:5000
  - hostname: api-edit.ldmathes.cc
    service: http://localhost:5001
  - hostname: my-new-app.ldmathes.cc    # NEW
    service: http://localhost:5002       # NEW
  - service: http_status:404
```

3. Add DNS route:

```powershell
cloudflared tunnel route dns excel-backend my-new-app.ldmathes.cc
```

4. Restart tunnel:

```powershell
Stop-ScheduledTask -TaskName "Cloudflared Tunnel"
Start-ScheduledTask -TaskName "Cloudflared Tunnel"
```

5. Test: `https://my-new-app.ldmathes.cc/api/health`

Step 3: Add Task Scheduler Auto-Start

1. Open **Task Scheduler**
2. **Create Basic Task**
3. Name: `Flask My New App`
4. Trigger: At startup
5. Action: Start a program
   - Program: `D:\Misc\Python313\python.exe`
   - Arguments: `D:\OneDrive\MS\my_new_app.py`
   - Start in: `D:\OneDrive\MS\`
6. Properties: ✅Run whether logged in or not, ✅Highest privileges, ✅Auto-restart

Step 4: Create Frontend

1. In GitHub repo: `D:\excel-web-interface`
2. Create new folder: `mkdir MyNewApp`
3. Create HTML file: `MyNewApp\index.html`
4. Set backend URL in JavaScript:

```javascript
const BACKEND_URL = 'https://my-new-app.ldmathes.cc';
```

5. Commit and push:

```powershell
git add MyNewApp\
git commit -m "Add new app frontend"
git push origin main
```

6. Access at: `https://dennyrgood.github.io/MyNewApp/`

Step 5: Add Card to Landing Page

1. Edit `D:\excel-web-interface\index.html`
2. Find the `<div class="projects-grid">` section
3. Add new card (copy existing card structure):

```html
<div class="project-card">
    <div class="project-icon">🔧</div>
    <h2>My New App</h2>
    <p>Description of what this app does.</p>
    <div class="project-links">
        <a href="https://dennyrgood.github.io/MyNewApp/" class="btn btn-primary">Open App</a>
    </div>
</div>
```

4. Commit and push:

```powershell
git add index.html
git commit -m "Add My New App to landing page"
git push origin main
```

Option B: Add Static GitHub Pages Site

Use this when you don't need a backend—just static HTML/JS/CSS.

Step 1: Create HTML Files

1. In GitHub repo: `D:\excel-web-interface`
2. Create new folder: `mkdir MyStaticPage`
3. Create HTML file: `MyStaticPage\index.html`
4. Add your static content (HTML/CSS/JS)

Step 2: Commit and Push

```powershell
git add MyStaticPage\
git commit -m "Add static page"
git push origin main
```

Access at: `https://dennyrgood.github.io/MyStaticPage/`

Step 3: Add Card to Landing Page

Same as Step 5 in Option A, but link directly to the static page:

```html
<div class="project-card">
  <div class="project-icon">📄</div>
  <h2>My Static Page</h2>
  <p>Description of static content.</p>
  <div class="project-links">
    <a href="https://dennyrgood.github.io/MyStaticPage/" class="btn btn-primary">View Page</a>
  </div>
</div>
```

Landing Page Card Customization

Card Structure

Each card in the landing page follows this HTML pattern:

```html
<div class="project-card">
  <div class="project-icon">[EMOJI]</div>
  <h2>[Title]</h2>
  <p>[Description - can be multiple sentences]</p>
  <div class="project-links">
    <a href="[URL]" class="btn btn-primary">[Button Text]</a>
    <a href="[URL2]" class="btn btn-secondary">[Button 2 Text]</a>
  </div>
</div>
```

Available Button Styles

- `btn-primary`: Purple gradient (main action)
- `btn-secondary`: Gray (secondary action)

Icon Options

Use any emoji for the icon, examples:

- 🎬Movies/Entertainment
- 📊Data/Spreadsheets
- 🔧Tools/Utilities
- 📄Documents/Static pages
- 🎯Quick actions
- 🔄Sync/Transfer
- 🌐Web apps
- 💾Storage/Files

Troubleshooting

Backend Not Responding (502 Bad Gateway)

1. Check if backend is running:

```powershell
netstat -ano | findstr :[PORT]
```

2. Check Task Scheduler tasks are running
3. Manually start to see errors:

```powershell
python D:\OneDrive\MS\[backend_file].py
```

4. Check logs in PowerShell output

CORS Errors in Frontend

1. Verify `BACKEND_URL` in HTML points to correct subdomain
2. Check Flask backend has `CORS(app)` enabled
3. Hard refresh browser: `Ctrl + F5`

404 on GitHub Pages

1. Verify folder/file exists in repo on GitHub.com
2. Check case sensitivity in URL (match exact folder name)
3. Wait 1-2 minutes after push for GitHub Pages to deploy
4. Verify GitHub Pages is enabled: Settings → Pages

Tunnel Not Working

1. Check tunnel is running:

```powershell
Get-ScheduledTask -TaskName "Cloudflared Tunnel"
```

2. Restart tunnel:

```powershell
Stop-ScheduledTask -TaskName "Cloudflared Tunnel"
Start-ScheduledTask -TaskName "Cloudflared Tunnel"
```

3. Test manual run:

```powershell
D:\OneDrive\MS\cloudflared.exe tunnel run excel-backend
```

4. Check config file syntax: `C:\Users\[USERNAME]\.cloudflared\config.yml`

Excel File Errors

1. Verify file path in backend: `EXCEL_FILE = "D:/OneDrive/MS/MoviesShows.xlsx"`
2. Check file isn't open in Excel (locks file)
3. Verify OneDrive sync is complete
4. Check backup files were created (in same folder with timestamp)

Key Files and Locations

Windows Machine

Path	Description
`D:\Misc\Python313\python.exe`	Python interpreter
`D:\OneDrive\MS\`	All backend scripts and Excel files
`D:\OneDrive\MS\excel_backend.py`	Port 5000 backend
`D:\OneDrive\MS\excel_backend_full_edit.py`	Port 5001 backend
`D:\OneDrive\MS\MoviesShows.xlsx`	Excel data file
`D:\OneDrive\MS\cloudflared.exe`	Cloudflare Tunnel executable
`C:\Users\[USERNAME]\.cloudflared\config.yml`	Tunnel configuration
`C:\Users\[USERNAME]\.cloudflared\[TUNNEL-ID].json`	Tunnel credentials

GitHub Repository

Path	Description
`D:\excel-web-interface\`	Local repo clone
`index.html`	Landing page (hub)
`excel-web-interface\index.html`	Quick-add form (if exists)
`MoviesShowsFullEdit\index.html`	Full CRUD editor
`index_usdz.html`	USDZ version (if exists)

Live URLs

URL	Purpose
`https://dennyrgood.github.io/`	Landing page hub
`https://dennyrgood.github.io/excel-web-interface/`	Quick-add form
`https://dennyrgood.github.io/MoviesShowsFullEdit/`	Full CRUD editor
`https://api.ldmathes.cc/api/health`	Port 5000 backend health
`https://api-edit.ldmathes.cc/api/health`	Port 5001 backend health

Testing Checklist

After adding a new application or making changes:

- [ ] Backend runs locally on assigned port
- [ ] Backend accessible via `localhost:[PORT]/api/health`
- [ ] Cloudflare tunnel routes correctly to subdomain
- [ ] Subdomain accessible: `https://[subdomain].ldmathes.cc/api/health`
- [ ] Task Scheduler task created and running
- [ ] Frontend folder created in GitHub repo
- [ ] Frontend pushed to GitHub and accessible
- [ ] Landing page card added and links work
- [ ] Hard refresh browser to clear cache
- [ ] Test full workflow (add/edit/delete if applicable)

Future Expansion Ideas

- Add authentication/login system
- Multi-user support with permissions
- Additional Excel files (Transfers, etc.)
- File upload capabilities
- Real-time collaboration features
- Mobile-responsive improvements
- Database integration for non-Excel data
- API documentation page
- Admin dashboard for backend monitoring

Session Handoff Prompt

When starting a new Claude session to continue this project, paste this documentation and say:

> "I'm working on expanding my Mathes Online Utilities project. This is a multi-application web system with Flask backends running on my Windows machine (exposed via Cloudflare Tunnel) and static frontends on GitHub Pages.
> Here's the complete documentation [paste this document].
> I want to add a new [application/feature/card]. Can you help me with [specific task]?"

Include specific context about what you're building:

- New backend application (describe purpose, endpoints needed)
- New static page (describe content/functionality)
- Modification to existing app (what needs to change)
- New card on landing page (describe new utility)

Version History

- **v1.0** (October 2025): Initial setup with Excel Quick-Add and Full CRUD Editor
- Current tunnel: `excel-backend`
- Current applications: 2 (Quick-Add, Full Editor)
- Domain: `ldmathes.cc`