Setup in GitHub

1. **Code repository**
   - All scripts (`Child_deduplication.py`, `Duplicate_Check.py`, etc.) plus helper files (`config_loader.py`, `system_config.json`, HTML pages) are stored in a GitHub repository.
   - This repo is the single source of truth: when you push updates, Render automatically redeploys them.
2. **Secrets management**
   - Sensitive values like the Bitrix webhook URL are **not stored in GitHub**.
   - Instead, the scripts use `os.getenv("B24_WEBHOOK_URL")` to read it from the environment.
3. **Version control**
   - You can use branches for development/testing and merge into `main` when stable.
   - GitHub keeps a history of changes, so you can always roll back if something breaks.

Setup in Render

1. **Web Service**
   - On Render, you deploy the repo as a **Web Service**.
   - Render automatically installs dependencies (from `requirements.txt`) and runs your Flask app that serves the scripts page.
2. **Environment Variables**
   - In Render's dashboard → Environment → add:
     - `B24_WEBHOOK_URL` → your Bitrix webhook (kept secret).
     - Optional: `CONFIG_DIR` if you want the config file stored on a persistent disk.
3. **Persistent Disk (optional)**
   - If you want edits to `system_config.json` to survive redeploys, attach a persistent disk in Render.
   - Point `CONFIG_DIR` to that disk, so your app reads/writes configs there.
4. **Auto-deploy**
   - Link Render to your GitHub repo.
   - Whenever you push changes to `main`, Render will rebuild and redeploy automatically.
5. **Running scripts**
   - Once deployed, go to your Render app's URL.
   - The `scripts.html` page shows all scripts with **Run** buttons.
   - Staff can run scripts from there without needing command line access.

✅ In short:

- **GitHub** → stores the code (but not secrets).
- **Render** → runs the app, manages secrets through environment variables, and optionally keeps configs on a persistent disk.