

We will need a table of contents at some point, (and a way to print a complete draft of what we have so far). Does that make sense? But, the TOC and print command will need to be changeable as we are far from finished at this point.

This makes complete sense. When building a dynamic project portfolio, you need a living, automated **Table of Contents (TOC)** and a single master script that allows you to print or review a complete draft of your entire project file array at any moment with a single command.

By utilizing basic Linux terminal architecture, we can design this so that as you create new guides or add new documents, the script automatically finds them, numbers them, updates the TOC, and formats a clean, readable print compilation for you instantly.

The Automated Project Assembler System

We will write a small, flexible bash script inside your [SAFE-MODE] environment. This tool scans your documentation folders, automatically maps out a fresh Table of Contents, and compiles every document into a single master plaintext file called MASTER_PROJECT_DRAFT.txt right on your Desktop, ready for a simple print command.

Step 1: Open the Assembler Script File

Run this command in your terminal to create the utility tool:

bash

```
nano ~/Downloads/assemble_project.sh
Use code with caution.
```

Step 2: Inject the Automated Compilation Logic

Copy and paste this clean script directly into your nano editor window:

bash

```
#!/bin/bash

# --- CONFIGURATION PATHS ---
PROJECT_DIR="$HOME/Documents/Sneakernet Deployment Project_v02"
GUIDES_DIR="$HOME/Documents/Popup_Guides"
OUTPUT_FILE="$HOME/Desktop/MASTER_PROJECT_DRAFT.txt"

echo "====="
```

```

echo "          AUTOMATED PROJECT DRAFT ASSEMBLER          "
echo "===== "
echo "[*] Scanning project nodes and references..."

# Initialize a fresh output file
echo
"=====
=== " > "$OUTPUT_FILE"
echo "          MASTER PROJECT INFRASTRUCTURE DRAFT
" >> "$OUTPUT_FILE"
echo "          GENERATED ON: $(date)          " >>
"$OUTPUT_FILE"
echo
"=====
=== " >> "$OUTPUT_FILE"
echo "" >> "$OUTPUT_FILE"

# 1. GENERATE DYNAMIC TABLE OF CONTENTS
echo
"=====
=== " >> "$OUTPUT_FILE"
echo "          TABLE OF CONTENTS
" >> "$OUTPUT_FILE"
echo
"=====
=== " >> "$OUTPUT_FILE"
echo "📁 MAIN PROJECT DEPLOYMENT BUNDLE:" >> "$OUTPUT_FILE"

count=1
for file in "$PROJECT_DIR"/*; do
    [ -f "$file" ] || continue
    echo " $count. $(basename "$file" .txt | tr '_' ' ')" >> "$OUTPUT_FILE"
    ((count++))
done

echo "" >> "$OUTPUT_FILE"
echo "📁 POPUP GUIDES & OPERATIONS REFERENCE LIBRARY:" >> "$OUTPUT_FILE"

gcount=1
for file in "$GUIDES_DIR"/*; do
    [ -f "$file" ] || continue
    echo " A-$gcount. $(basename "$file" .txt | tr '_' ' ')" >> "$OUTPUT_FILE"
    ((gcount++))
done

echo "" >> "$OUTPUT_FILE"
echo
"=====
=== " >> "$OUTPUT_FILE"
echo "" >> "$OUTPUT_FILE"

# 2. APPEND MAIN BUNDLE TEXT
echo "[*] Appending primary project profiles..."
for file in "$PROJECT_DIR"/*; do
    [ -f "$file" ] || continue

```

```

    echo ">>> SYSTEM NODE: $(basename "$file") <<<" >> "$OUTPUT_FILE"
    cat "$file" >> "$OUTPUT_FILE"
    echo -e "\n\n\n" >> "$OUTPUT_FILE"
done

# 3. APPEND POPUP REFERENCE GUIDES
echo "[🔗] Appending background reference popups..."
for file in "$GUIDES_DIR"/*; do
    [ -f "$file" ] || continue
    echo ">>> REFERENCE NODE: $(basename "$file") <<<" >> "$OUTPUT_FILE"
    cat "$file" >> "$OUTPUT_FILE"
    echo -e "\n\n\n" >> "$OUTPUT_FILE"
done

echo "=====
echo "[✅] Compilation Complete. Master draft built."
echo "[📄] Location: $OUTPUT_FILE"
echo "=====
Use code with caution.

```

Step 3: Save and Commit Permissions

1. Press **Ctrl + O** then hit **Enter** to commit the code to disk.
2. Press **Ctrl + X** to close out the Nano interface.
3. Elevate the script to an executable production utility by running:

bash

```
chmod +x ~/Downloads/assemble_project.sh
```

Use code with caution.

How to Use it to View and Print Your Draft

Whenever you make updates, alter text, or write a brand new popup guide file, simply type this single command to instantly assemble your entire portfolio:

bash

```
~/Downloads/assemble_project.sh
```

Use code with caution.



Printing Your Complete Master Document:

Once compiled, you can double-click **MASTER_PROJECT_DRAFT.txt** right on your Desktop to open it in LibreOffice Writer, set it to a clean monospaced font, and hit Print.

Alternatively, if you want to inspect the entire project draft text layout straight from your active terminal session without opening an app, simply run:

bash

```
less ~/Desktop/MASTER_PROJECT_DRAFT.txt
```

Use code with caution.

Proactive Next Steps

The compilation engine is completely installed and configured to adjust automatically as your files grow.

Try running `~/Downloads/assemble_project.sh` inside your **[SAFE-MODE]** window right now and let me know if it creates your master desktop file cleanly! Or, if your slate is clean, toss over your next foundational project question!