```
import os
import shutil
import subprocess
import sys
import tarfile

# Define constants
VERSION_NUMBER = "5.3"
HOME_DIR = os.getenv('HOME', '')
CONFIG_DIR = os.path.join(HOME_DIR, '.config', 'amdb')
LOGS_DIR = os.path.join(HOME_DIR, '.local', 'share', 'amdb', 'logs')
ASSETS_DIR = os.path.join(HOME_DIR, '.local', 'share', 'amdb', 'assets')
TAR_GZ_PATH = f"AMDb{VERSION_NUMBER}.tar.gz"

def create_directories():
    os.makedirs(CONFIG_DIR, exist_ok=True)
    os.makedirs(LOGS_DIR, exist_ok=True)
    os.makedirs(ASSETS_DIR, exist_ok=True)

def extract_files():
    with tarfile.open(TAR_GZ_PATH, 'r:gz') as tar:
        tar.extractall(ASSETS_DIR)

def run_key_generation():
    key_gen_script_path = os.path.join(ASSETS_DIR, 'key_gen.py')
    encryption_utils_path = os.path.join(ASSETS_DIR, 'utils', 'encryption_utils.py')
    result = subprocess.run([sys.executable, key_gen_script_path, encryption_utils_path],
capture_output=True, text=True)
    if result.returncode != 0:
        raise Exception(f"Key generation failed: {result.stderr}")

def compile_binary():
    pyinstaller_command = [
        sys.executable, '-m', 'PyInstaller',
        '--onefile',
        '--distpath', ASSETS_DIR,
        os.path.join(ASSETS_DIR, 'AMDb.py')
    ]
    result = subprocess.run(pyinstaller_command, capture_output=True, text=True)
    if result.returncode != 0:
        raise Exception(f"Binary compilation failed: {result.stderr}")
    shutil.move(os.path.join(ASSETS_DIR, 'AMDb'), os.path.join(ASSETS_DIR, 'AMDb'))

def cleanup_files():
    for root, dirs, files in os.walk(ASSETS_DIR):
        for file in files:
            file_path = os.path.join(root, file)
            if file not in ['AMDb', 'amdb.ico', 'movies.db']:
                os.remove(file_path)
        for del_dir in dirs:
            shutil.rmtree(os.path.join(root, del_dir))
```

```
def create_shortcut():
    desktop_entry_dir = os.path.join(HOME_DIR, '.local', 'share', 'applications')
    os.makedirs(desktop_entry_dir, exist_ok=True)
    desktop_entry_path = os.path.join(desktop_entry_dir, "amdb.desktop")
    with open(desktop_entry_path, 'w') as desktop_entry_file:
        desktop_entry_file.write("[Desktop Entry]\n")
        desktop_entry_file.write("Name=AMDb\n")
        desktop_entry_file.write("Comment=Andy's Movie Database\n")
        desktop_entry_file.write(f"Exec={os.path.join(ASSETS_DIR, 'AMDb')}\n")
        desktop_entry_file.write(f"Icon={os.path.join(ASSETS_DIR, 'amdb.ico')}\n")
        desktop_entry_file.write("Terminal=false\n")
        desktop_entry_file.write("Type=Application\n")
        desktop_entry_file.write("Categories=Application;Utility;\n")

    desktop_shortcut_path = os.path.join(HOME_DIR, "Desktop", "AMDb.desktop")
    shutil.copy(desktop_entry_path, desktop_shortcut_path)
    os.chmod(desktop_shortcut_path, 0o755)

def main():
    try:
        print("Creating directories...")
        create_directories()

        print("Extracting files...")
        extract_files()

        print("Running key generation...")
        run_key_generation()

        print("Compiling binary...")
        compile_binary()

        print("Cleaning up files...")
        cleanup_files()

        print("Creating shortcut...")
        create_shortcut()

        print("Installation complete!")
    except Exception as e:
        print(f"Installation failed: {e}")

if __name__ == "__main__":
    main()
```