utils/compiler.py

import subprocess

import os

import logging

from .helpers import run\_subprocess

logger = logging.getLogger(\_\_name\_\_)

def compile\_code(toolchain\_path, source\_path, binary\_path):

riscv\_gcc = os.path.join(os.path.expanduser(toolchain\_path), 'bin', 'riscv32-unknown-elf-gcc')

if not os.path.isfile(riscv\_gcc):

logger.error(f"RISC-V GCC compiler not found at {riscv\_gcc}")

raise FileNotFoundError(f"RISC-V GCC compiler not found at {riscv\_gcc}")

logger.info("Compiling the generated pipeline code...")

compile\_command = [

riscv\_gcc,

'-o',

binary\_path,

source\_path

]

try:

run\_subprocess(compile\_command)

logger.info(f"Compilation successful. Executable at {binary\_path}")

except Exception as e:

# Clean up resources here if necessary

if os.path.exists(binary\_path):

os.remove(binary\_path)

logger.error(f"Compilation failed: {e}")

raise

```

In this corrected version, I added a check to remove the binary file if it exists when an exception occurs during compilation. This ensures that the binary file is not left behind in an incomplete state.