#include "merge.h"

#include "commit\_utils.h"

#include <map>

#include <iostream>

#include <fstream>

#include <sstream> // Required for stringstream

using namespace std;

// Finds the lowest common ancestor commit between two commits.

CommitData findBaseCommit(CommitData one, CommitData two) {

map<string, bool> seen;

while (!one.commitHash.empty()) {

seen[one.commitHash] = true;

one = readCommitFromFile(one.parentHash);

}

while (!two.commitHash.empty()) {

if (seen[two.commitHash]) {

return readCommitFromFile(two.commitHash);

}

two = readCommitFromFile(two.parentHash);

}

return CommitData{};

}

// Merges changes from another branch, handling conflicts if necessary.

void mergeFromBranch(string otherBranch) {

string currentHash = getCurrentHead();

CommitData current = readCommitFromFile(currentHash);

ifstream file(".minigit/refs/" + otherBranch);

string otherHash;

if (!file.is\_open()) {

cerr << "Error: Could not open branch file: " << otherBranch << endl;

return;

}

getline(file, otherHash);

CommitData other = readCommitFromFile(otherHash);

CommitData base = findBaseCommit(current, other);

map<string, string> baseMap, currentMap, otherMap, result;

bool conflict = false;

for (auto f : base.fileList) baseMap[f.fileName] = f.fileHash;

for (auto f : current.fileList) currentMap[f.fileName] = f.fileHash;

for (auto f : other.fileList) otherMap[f.fileName] = f.fileHash;

for (auto [fname, oHash] : otherMap) {

string cHash = currentMap[fname];

string bHash = baseMap[fname];

if (cHash != oHash && bHash != cHash && bHash != oHash) {

cout << "CONFLICT: File changed in both: " << fname << "\n";

conflict = true;

// Read file contents

string currentContent = readFileFromObjects(fname, cHash);

if (currentContent.empty()) {

cerr << "Error reading current version of file: " << fname << endl;

continue;

}

string otherContent = readFileFromObjects(fname, oHash);

if (otherContent.empty()) {

cerr << "Error reading other version of file: " << fname << endl;

continue;

}

// Create a conflict marker

string conflictMarker = "<<<<<<< HEAD\n" + currentContent +

"=======\n" + otherContent +

">>>>>>> " + otherBranch + "\n";

writeFileToWorkingDirectory(fname, conflictMarker);

}

result[fname] = oHash;

}

for (auto [fname, cHash] : currentMap) {

if (result.find(fname) == result.end()) {

result[fname] = cHash;

}

}

if (!conflict) {

CommitData newCommit;

newCommit.parentHash = current.commitHash;

newCommit.secondParentHash = other.commitHash;

newCommit.message = "Merge from branch " + otherBranch;

newCommit.time = getCurrentTime();

newCommit.commitHash = current.commitHash.substr(0, 4) + other.commitHash.substr(0, 4);

for (auto [fname, fhash] : result) {

FileBlob f;

f.fileName = fname;

f.fileHash = fhash;

newCommit.fileList.push\_back(f);

}

saveCommitToFile(newCommit);

changeHead(newCommit.commitHash);

cout << "Merge successful. New commit is: " << newCommit.commitHash << "\n";

} else {

cout << "There were conflicts. Fix them and commit manually.\n";

}

}