README: Asst3 WTF

Usage

Server: ./WTFserver <IP> <PORT>

Client: ./WTF <command> <projectname> |file|

./WTF configure <IP> <PORT>

./WTF rollback <projectname> <version>

Design

Thread Synchronization

Threads:

A new thread is created for each client that connects to the server

A new threadNode for the thread is added to a list of

```
struct threadNode{
    pthread_t thread; //the thread
    int sockfd; //the socket fd of the client-server connection
    struct threadNode* next;
};
```

When the client command has finished executing, the sockfd is closed and the threadNode is removed from the list of threadNodes.

Mutexes:

When the server is start up, a mutex is created for every project on the server.

Every mutexNode created is added to the list of

```
struct mutexNode{
    char* projectname; //mutex is for this project
    pthread_mutex_t mutex; //the mutex
    struct mutexNode* next;
};
```

A mutex is also created and added to the list on successful execution of ./WTF create command

The mutexNode for the project is removed from the list on successful execution of a ./WTF destroy command

When a client executes a command that requires locking of project repository:

The mutex for the project is retrieved from the list.

The mutex is locked.

The mutex is unlocked when the command successfully finishes executing.

If there's an error with execution of the command, the mutex is unlocked before the server sends an error message to the client and ends execution of the command.

Client-Server Communication

Sending end:

char* data; //this is the data being send

char* compressedData; //data is compressed into this using zlib

-compressedData is sent to the receiving end

Receiving end:

compressedData is received.

compressedData is decompressed using zlib

char* data; //this is the decompressed data
data is parsed into a list of
struct node{
char* nodeType; //what the node contains- command, dataType, project, numFile, fileName, fileContent
char* name; //path to the file, command, num of file, etc (depending on nodeType)
char* content; //if fileContent, this is the content of the file
struct node* next;
};

Project Structure:
Each project on the server has the corresponding directories:
<projectname>
.<projectname>
Each project directory on the server also contains the following data files and directories for the project:
.archive/
.commit/
.manifest
.history

Storage of Archive files:
When a project is pushed to the server. The existing project and it's filed are compressed into a .gz file using zlib. All compressed files of previous versions of the project is stored in the .<projectname>/.archive on the server.

WTFtest
Executables:
WTFserver - Server Side
WTF - Client Side
WTFTest - Runs test cases

make test: generates the following file structure
|--Server
|--WTFserver
|--Client1
|-WTF
|--Client2
|--WTF
|--WTFtest

./WTFtest: executes WTFtest
|--Server
|--WTFserver
|-- (Projects on server's end will be here)
|--Client1
|-WTF
|-- (Projects on client1's end will be here)
|--Client2
|--WTF
|-- (Projects on client2's end will be here)
|--WTFtest