**Assignment 6**

In this assignment we implemented 6 features of a File System.

1. Open:

fs\_open() takes in argument as filename and flags. This function searches for the given file name through the directory, if found, it sets it’s state to open and opens the file. We then return the file descriptor of that file. If that file is not found, then it throws a system error.

1. Close

fs\_close() takes in an argument as a file descriptor. Using the ifle descriptor, we can access the file structure and set the state of file to close. If the descriptor value is invalid, then we throw a system error.

1. Create

fs\_create() takes in argument as file name and mode of the file. File name is the name we want to give to the new file and mode is in what mode we want to create the file. We first check if any file with the same name already exists, if it does, we throw an error, if not, we then create a new file and assign an inode to it in the inode block.

1. Seek

fs\_seek() takes arguments as file descriptor, and offset. Seek will move the memory pointer to the offset it is given.

1. Read

fs\_read(). In this function, we first check if the file is closed, if file is closed then we throw an error. We cannot read a closed file. If file is open, we then check if the file descriptor is valid to make sure that the file is valid and present in the file table. We would then check if there are contents present in the file. We would start reading the file from the seek pointer until the end of the file is reached.

1. Write

fs\_write(). In this function, we first check the state of the file. If the file is closed, return error. If open, we then check the validity of the file and if it is empty or not. We then write the values into the file from the buffer. We then update the seek pointer and return the size of the file.

**Lessons Learnt:**

1. Concept of File System
2. Xinu implementation of File System
3. Understood the concept of different functions of a file system.