# PROJECT DESIGN DOCUMENTATION

## PROBLEM STATEMENT

To build a simple file system using FUSE (File System in User Space).

## OVERVIEW

The objective of the project was to implement and understand the working of a functional file system for Linux.

The important components for the design of the File System include implementing system calls to interact with the file system data structure, which is performed using the operations built for that data structure and be able to achieve persistence(i.e. retain the data stored on the file system).

## FUSE(File System in Userspace) INTERFACE

With FUSE, users are able to create their own file systems without editing kernel code. The FUSE module helps provide a "bridge" to the actual kernel interfaces. The C program which contains implementations of the system calls is now mapped to the handlers in the FUSE module. If a user now issues read/write requests for this newly mounted file system, the kernel forwards the IO-requests to the FUSE handler and then sends the handler's response back to the user.

## DESIGN OF FILE SYSTEM

What is a FILE SYSTEM? - In computing, a file system or filesystem controls how data is stored and retrieved. Without a file system, information placed in a storage medium would be one large body of data with no way to tell where one piece of information stops and the next begins. By separating the data into pieces and giving each piece a name, the information is easily isolated and identified.

The File System structure has been implemented as a Tree. The Inode structure below stores information about a file(directory file or regular file). The operations on the data structure include: initializing a node, inserting the node in the tree, searching for a node given a path and deleting a node. The file system calls are then implemented using these data structure operations. This application is developed in C, and is used to provide an abstraction for the File System.

**Inode Structure:**

```
typedef struct Inode
{
        char* path;
        char* name;
        int type;
        int no_of_children;
        int size;
        int inode_num;
        int* child_inode;
        struct Inode *parent;
        struct Inode **children;
        File file;
} Inode;
```

**File Structure:**

```
typedef struct File
{
        char* data;
        int size;
}File;
```

## FEATURES OF THE FILE SYSTEM

The File System includes implementations of system calls such as open(), read(), write(), create(), getattr(), readdir(), mkdir(), mknod(), rmdir(), unlink(), rename().The following Unix commands can be executed on a terminal to test the functioning of the File System.

```
echo “Good Morning” > testFile.txt
echo “Hello World” >> file1.txt
ls -l file1.txt
cat file1.txt
cp file1 file2
mkdir dir_name
ls dir_name
mv file1 file3
rmdir dir_name (when it is empty)
rm file1
```

## IMPLEMENTATION OF PERSISTENCE AND METADATA

For handling persistence, the following structures have been defined.

Two binary files: fsmeta and fsdata have been created which contain the meta data and actual data respectively, of each created file. When the file system is mounted, the contents of the fsmeta and fsdata file (if these files already exist) are read and the

tree structure is built. When data is written to any file or when existing file is modified, the contents of the file is updated along with the contents of the fsmeta and fsdata files. The meta_node structure values are written to the fsmeta file and the data_node structure values are written to fsdata file.

**Meta Data Node:**

```
typedef struct meta_node
{       int pathlength;
        char* path;
        int namelength;
        char* name;
        int type;
        int parent_namelength;
        char *parent_name;
        int size;
        int inode_num;
        File file;
}meta_node;
```

**Data Node:**

```
typedef struct data_node
{       int inode_num;
        int size;
        char* data;
}data_node;
```

## Writing to the binary files

- To understand the principle of writing to the binary files consider an instance where a new text file has been created. The events that occur after initializing a new Inode and inserting the Inode into the tree specific to the path, are:
    1. Initialization of the meta_node structure with the values of the root node and write it to the fsmeta file
    2. Recursively traverse through each of the root node's children and perform the following operations.
- Write the meta data of the meta_node structure into the fsmeta file.
- If the new file contains any data then initialize the fields of the data_node structure with the inode number and data of the meta_node structure and write the data_node into the fsdata file.

## Reading from the binary files

- The root is first initialised.
- When the file system is first mounted, the contents of the fsmeta file are read sequentially and inserted into the tree based on the parent.

- The content of each structure are first copied to a temporary meta_node, then a new Inode structure is initialized with the contents of the meta_node structure and finally inserted into the file system tree.

## SYSTEM CALLS IMPLEMENTED

```
//The mapping of the inbuilt stat structure to our Inode Structure
static int sys_getattr(const char* path, struct stat *st);

//Function to read contents of a directory.
static int sys_readdir(const char *path, void *buffer, fuse_fill_dir_t filler, off_t offset, struct fuse_file_info *fi );

//Function to create a directory
static int sys_mkdir(const char * path, mode_t x);

//Function implemented to create a new file
static int sys_mknod(const char * path, mode_t x, dev_t y);

//Function to open a file. Returns inode_num of opened file
static int sys_open(const char *path, struct fuse_file_info *fi);

//Function to read contents of a file and print it on the terminal
static int sys_read( const char *path, char *buffer, size_t size, off_t offset, struct fuse_file_info *fi );

//Function to write to a new file or append to an existing file
static int sys_write(const char *path, const char *buf, size_t size, off_t offset, struct fuse_file_info *fi);

//Function to delete an empty directory
static int sys_rmdir(const char *path);

//Function to delete a file
static int sys_unlink(const char *path);
```

## DISK FUNCTIONS IMPLEMENTED

```
//Write the contents of the tree structure to disk
void write_to_disk(meta_node* disk, FILE *mfp, FILE *dfp);
```

```
//Read the contents of the file into the data of the File structure
void read_from_datafile(meta_node *disk);

//Read the disk parameters to the tree structure
void read_from_disk(meta_node* disk, FILE *mfp);
```

## PHASE I - System Call Implementation

Implement system calls for the operations that can be issued by Linux. FUSE provides a way to intercept file system calls issued by Linux programs and to redirect the program flow into a handler. Basic file system commands are defined that works only in memory.

When the file system is unmounted, the data does not persist, meaning the data is lost when unmounted. On mounting again it is seen that the file system is empty.

## PHASE II - File System Abstraction

The data structures and the necessary procedures were written to implement the file functions. The data structure is a Tree in our File System. The data part of the file is encapsulated within the node of the tree. The root is the head of the file system. The structure defined is the Inode structure where the type differentiates the file and the directory. To implement persistence, two binary files are created to act as disk (emulator). One binary file holds the metadata of the file and is called “fsmeta” and the other file holds the file contents and is called “fsdata”. When the file system is mounted the contents of these files are read to restore the previous state of the file system. This ensures persistence.

## PHASE III - SECONDARY STORAGE

Implementation of persistence to ensure that the file system is preserved across machine reboots. File system remains intact when it is remounted.

## OUTPUTS

```
Shalini@Shalini-XPS-13-9360:~/Desktop$ cd mountpoint/
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ mkdir DIR1
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ mkdir DIR2
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ ls
DIR1  DIR2
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ echo "Hello" > test1
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ cp test1 test2
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ ls
DIR1  DIR2  test1  test2
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ cp test1 DIR1/test3
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ ls DIR1
test3
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ cat test3
Hello
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$
```

```
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ echo "Goodbye" >> test1
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ cat test1
Hello
GoodbyeShalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ echo > test1
bash: test1: Function not implemented
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ rm DIR1/test3
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ ls DIR1
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ echo "Hello" > DIR1/test3
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ ls DIR1
test3
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ rmdir DIR1
rmdir: failed to remove 'DIR1': Operation not permitted
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ rm DIR1/test3
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ rmdir DIR1
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ ls
DIR2  test1  test2
```

```
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ nano
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ cat test
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 3
5 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97
98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121
122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 14
5 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 19
2 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215
216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 23
9 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262
263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 28
6 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309
310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 33
3 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356
357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 38
0 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403
404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 42
7 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450
451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 47
4 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497
498 499 500 501 502 503 504 505 506 507 508 509 510 511 512
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ du -s test
2       test
```

```
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ cat test
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 3
5 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97
98 99 100
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ du -s test
2       test
```

```
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ cat test
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 3
5 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97
98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121
122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 14
5 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 19
2 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215
216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 23
9 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262
263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 28
6 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309
310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 33
3 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356
357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 38
0 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403
404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 42
7 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450
451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 47
4 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497
498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 52
1 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544
545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 56
8 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591
592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 61
5 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638
639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 66
2 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685
686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 70
9 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732
733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 75
6 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779
780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 80
3 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826
827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 85
0 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873
874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 89
7 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920
921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 94
4 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967
968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 99
1 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011
1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ du -s test
2       test
```

```
Shalini@Shalini-XPS-13-9360:~/Desktop$ sudo umount mountpoint
Shalini@Shalini-XPS-13-9360:~/Desktop$ cd mountpoint/
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$ ls
DIR2  test
Shalini@Shalini-XPS-13-9360:~/Desktop/mountpoint$
```