**Sprint -2 Story board**

**FTP- Server Functional Requirements.**

**Story:-The story is about application which can handle multiple clients concurrently and depending upon the accessibility client can upload, download and browse in the server.**

Accept the socket

Get IP by calling FUNCTION GET\_IP

IF IP is in BlackList THEN

Send REJECTION acknowledgment to client

ELSE

Accept the ip connection

**Function handle client**

IF User is Authenticated THEN

Change directory to authenticated user

INFINITE WHILE LOOP BEGIN

Receive command from client side

IF command is "exit" THEN

BREAK out from INFINITE WHILE LOOP

ELSE IF command is "ls" THEN

FUNCTION CALL VIEW to display the FILES

ELSE IF command is "pwd" THEN

FUNCTION CALL VIEW to display the present working directory

ELSE IF command is "get" THEN

FUNCTION CALL UPLOAD to upload the files

ELSE IF command is "put" THEN

FUNCTION CALL DOWNLOAD to download the file

ELSE IF command is "cat" THEN

FUNCTION CALL VIEW to display the data in the file

IF number\_of\_clients reached Maximum THEN

PRINT "Maximum Limit Reached"

ELSE

Run a thread to handle new client

FUNCTION CALL HANDLE\_CLIENT

Cerate socket server using Tcp/IP protocols

if socket not created.

print "Error in socket" and exit

else

print "server created successfully"

Receive message from sever

if message matches "rejected"

print "server rejected connection"

if message matches "Max"

print "Max client reached"

if user is Authenticated

get choice from use

switch(choice)

case 1:

Enter name of file to download

send "get" message to server

call download function

print "Download completed"

case 2:

Enter name of file to upload

send "put" message to server

call upload function

print "Upload successfully"

case 3:

send "ls" message to server

call Pipe\_data function

case 4:

send "pwd" data to server

call Pipe\_data function

case 5:

Enter name of file to read

send "cat" message to server

call Pipe\_data function

case 6:

send "exit" message to server

print "Quitting" and exit

close the connection

**Function upload**

Takes the location of file as input

Open the file with a file pointer

if the pointer is NULL

print "Error in open file" and exit

read each line of the file

if the line returns NULL

if send data on sever return -1

print "Error in sending file." and exit

else

send that to the server

**Function Download**

Takes the location of file as input

Create file with filename

Open with a file pointer in write mode

Receive the data from sever

if receiving data length is 0

stop receving from server.

else

Copy information in file.

**Function Pipe data**

Receive data from sever

if length of data is 0

stop receving from server.

else

print the data

**Function Auth**

Receive message from sever

print the message

get username and send to server.

receive data from server.

if data not matches Anonymous

enter password and send to server.

else

for user Anonymous change flag to 1

Finished