Mastery Project Proposal

In this document, you will identify a networked application that you would like to implement to demonstrate your mastery of the content in this course. If some of the words and phrases in this document are unfamiliar, I recommend first reviewing content from Modules 1.

Project Team

	Team Member 1	**Team Member 2**	**Team Member 3**
Name	Gali Veera Surya Bhaskar	Anudeep Uppu	Keerthana Vijaykumar
NAU Email	vg588@nau.edu	au282@nau.edu	kv582@nau.edu

Networked Application Description

1. **Provide a 2-3 paragraph description of the networked application you plan to implement. Some ideas include a web browser and proxy server; a P2P file sharing application; a music streaming application—the sky is the limit, but the application needs to be complex enough to demonstrate rigorous mastery! In your description, be specific about what type of network architecture your application will use and what roles client and server programs play in the application.**

- **Type of Architecture:**

  The project is about developing a file-sharing application using client-server architecture. It is a replica of file-sharing applications like WeTransfer and toffee-share.
- **Role of client:**

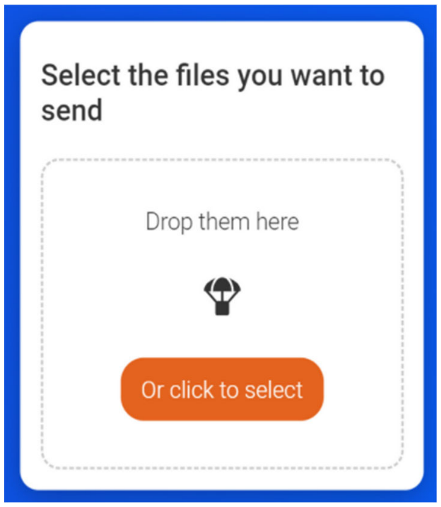
  Our application uses FTP protocol which allows clients to upload the files (like photos, videos, other file types) to the server and the receiver can perform actions such as view, download, and delete the files from the server using the storage location path.
- **Role of server:**

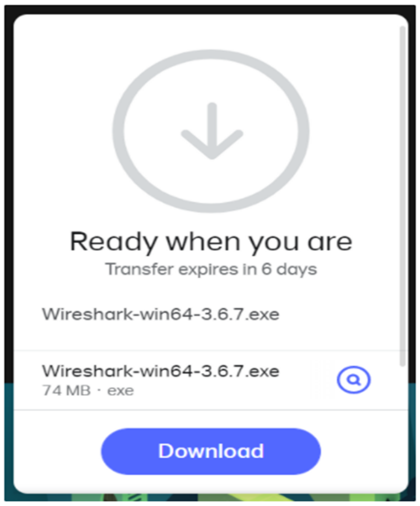
  Using FTP protocol, server responds to the client file transfer requests. Whenever client sends a request for the file upload, server uploads the files to the storage location maintained in the server and responds the client with the storage location path. Using the storage location path, client can view the list of files and can download the files on demand. Also, the storage location path which is been used will be expired after few days (based on the expiration policy). Server runs a scheduled job which automatically clean the files whose expiration time is reached and in turn free up space on the server-side.

2. **Using the description of your proposed application from Question 1, describe the application layer and transport layer protocols you plan to use in your application and why they are appropriate for use in your networked application. For the application layer, provide the IETF RFC numbers of the protocol versions you plan to use.**
   - Our application will use File-Transfer-Protocol (FTP) as the application layer protocol because it deals with file transfer between clients and the server.
   - We will use Transmission Control Protocol (TCP) as the transport layer protocol as it provides connection-oriented services where reliability is guaranteed.
   - **RFC versions:**
     - FTP - RFC 959
     - TCP - RFC 793

3. **What will the user interface for your application look like? Do you plan to implement a mobile app with a graphical user interface (GUI)? Or will you use a command line interface? Be as specific as possible and if you would like, you can provide a diagram or examples if it clarifies your vision.**

   We will develop a desktop application as a graphical user interface where it allows the users to upload files from the local system to the server and support file download from the server using the reference path which is shared with the receiver. The following screenshots are taken from the similar application which is live, and these will be taken as reference for our application development.




Ready when you are
Transfer expires in 6 days
Wireshark-win64-3.6.7.exe
Wireshark-win64-3.6.7.exe
74 MB · exe
Download

File upload experience **File download experience**

4. **Which language do you plan to use to implement the networking aspects of your project? Which packages will you use to allow you to use BSD sockets?[1]**
   Languages for the socket programming – Python. Some of the packages we are planning to use are socket, threading, os, struct, time, etc.

5. **Which packages or frameworks will you use to implement the user interfacing aspects of your project?**

   We will be designing a desktop application compatible with Windows OS using Tkinter - Python GUI designing library (reference: https://docs.python.org/3/library/tk.html) and Python Imaging Library (PIL) etc., for user interfacing aspects.

Plans for Installations 1-3

For the Mastery Project, you will be submitting incremental installations for your project. For example, if I were to implement a web browser that uses a proxy server, I might say that I would submit my implementation of the client code in Installation 1, my implementation of the server code in Installation 2, and my implementation of the browser GUI in Installation 3.

Installation 1

6. **What will you be submitting for Installation 1? Provide 1-2 paragraphs of description.**

   We would be submitting the implementation of the client-server communication in Installation 1. The connection maintained is by 3-way TCP handshake which guarantees reliability, and it involves sequence of function calls in socket programming with which the client connects with server, transfer of connection establishment messages, connection closure. This entire installation will be demonstrated using the command-line interface which will be modified into Graphical User Interface in future installations.

7. **What code will be submitted for Installation 1?**

   The code will focus on command line implementation of basic client-server communication. Using the menu options provided in the command line, client tries to make a connection with the server. This installation includes submission of client.py and server.py files which uses basic socket programming written using python. These files contain most of the function declarations (with function name and arguments) which will be defined/implemented in the future installations. Function related to connection establishment will be implemented in this installation. Some of important functions that will be included in server.py and client.py are makeServerConnection, disconnectServerConnection, fileUpload, fetchListOfFilesByStorageLocationPath, downloadFile, deleteFile, etc.

8. **In addition to code, you will be required to submit a demonstration video that provides a line-by-line walkthrough of the code submitted for the installation as well as a demonstration of the operational code. For Installation 1, what will a demo video entail (Provide 2-3 paragraphs of description)?**

   The video depicts how exactly the client connects with server using command line. It shows how client requests server for the connection and how the server serves multiple client requests at an instance i.e., it would demo the connection request initiation, listening and accepting requests, message delivery between client and server and connection closure. Using Wireshark and Netstat, we will explain how uniquely the connection can be maintained by using IP address and port number to serve the respective clients.

9. **You will be graded on Installation 1 using a rubric. While 30% of the points on the rubric are determined by Mastery Project Installation 1 Rubric the remaining 70% of points will be**

determined by a portion of the rubric that you design yourself. Specifically, based on your specific vision for what is to be implemented in Installation 1, you should provide a list of 5 or more discreet items that determine whether this installation passes or fails as a demonstration of mastery.

	Fail (0)	Pass (1)
[Item 1]	This installation fails to demonstrate mastery if server is not running on the specified address and port.	This installation succeeds at demonstrating mastery by running server at a specific address and port number.
[Item 2]	This installation fails to demonstrate mastery if server is not able to respond to the client requests.	This installation succeeds at demonstrating mastery by server accepting the incoming connections from the client.
[Item 3]	This installation fails to demonstrate mastery if server is not able to serve more than 5 clients.	This installation succeeds at demonstrating mastery by serving at least 5 clients at a time.
[Item 4]	This installation fails to demonstrate mastery if server and client are not able to run in the windows machine.	This installation succeeds at demonstrating mastery by running the client and server programs in windows OS.
[Item 5]	This installation fails to demonstrate mastery if server cannot deny the access for the clients initiating the requests from the blocked IP addresses.	This installation succeeds at demonstrating mastery by server restricting the providing access to the allowed IP addresses.

Installation 2

10. What will you be submitting for Installation 2? Provide 1-2 paragraphs of description.

We would be submitting the implementation of the file transfer(upload/download) using FTP between server and client in Installation 2. Whenever a file/list of files are uploaded to the server, it is stored at a common location in the server and server acknowledges the client with the storage location path. The storage location is a folder path where the files are stored files, and these files will be maintained in the server until the expiration time is reached. Whenever client requests for the file download using the storage location path, server responds with the list of file names stored in the given location to the client. By looking at the list of file names, user makes a specific file download request from the server using FTP. Entire communication between the client and server will be done by FTP at application layer and TCP at transport layer.

11. What code will be submitted for Installation 2?

The code would focus on how the file transfer happens between client and the server using command line. It also focuses on storage location implementation and fetching or uploading of files to/from the storage location based on the client's request. Here we will be implementing function definitions related to file transfer which are declared in the server.py and client.py in the previous installation.

12. **In addition to code, you will be required to submit a demonstration video that provides a line-by-line walkthrough of the code submitted for the installation as well as a demonstration of the operational code. For Installation 2, what will a demo video entail (Provide 2-3 paragraphs of description)?**

    The video would depict how file transfer happen in between the client and server using command line, that involves following actions:

    - Uploading the files to the server.
    - Viewing the list of files in the storage path.
    - Deleting the files in the storage path.
    - Downloading specific files from the server.

    It will also cover how storage path structure will be maintained and how the files in the storage location path will be accessed according to the client requests.

13. **You will be graded on Installation 2 using a rubric. While 30% of the points on the rubric are determined by Mastery Project Installation 2 Rubric, the remaining 70% of points will be determined by a portion of the rubric that you design yourself. Specifically, based on your specific vision for what is to be implemented in Installation 3, you should provide a list of 5 or more discreet items that determine whether this installation passes or fails as a demonstration of mastery.**

	Fail (0)	Pass (1)
[Item 1]	This installation fails to demonstrate mastery if server cannot upload the files to the server via FTP.	This installation succeeds at demonstrating mastery by uploading the files to the server via FTP.
[Item 2]	This installation fails to demonstrate mastery if server cannot respond with the storage location path after the file upload done via command line.	This installation succeeds at demonstrating mastery by responding to the client with storage location path after the file upload done via command line.
[Item 3]	This installation fails to demonstrate mastery if server cannot respond with the list of files in the storage location provided by the client.	This installation succeeds at demonstrating mastery if the server responds with the list of files in the storage location, which is provided by the client via command line.
[Item 4]	This installation fails to demonstrate mastery if server cannot delete the files in the storage location via command line.	This installation succeeds at demonstrating mastery by deleting the specific files in storage location based on the client request via command line.
[Item 5]	This installation fails to demonstrate mastery if server cannot download the requested files via FTP.	This installation succeeds at demonstrating mastery by downloading the requested files from the server via FTP.

Installation 3

14. **What will you be submitting for Installation 3? Provide 1-2 paragraphs of description.**

    We would be submitting the server and client integration with the Graphical User Interface (GUI) in Installation 3. It will be a desktop application developed using Python Tkinter and compatible with the windows OS. Our application allows client(sender) to upload the files from his personal system which needs to be transferred to the other client(receiver). It also supports the functionality of file download into the receiver's system from the application interface.

15. **What code will be submitted for Installation 3?**

    The code related to the graphical user interface will be submitted for installation 3. The function implementations related to GUI rendering and the logic related to scheduled job which automatically does the file deletion based on expiration policy will also be included in this installation.

16. **In addition to code, you will be required to submit a demonstration video that provides a line-by-line walkthrough of the code submitted for the installation as well as a demonstration of the operational code. For Installation 3, what will a demo video entail (Provide 2-3 paragraphs of description)?**

    The demo video depicts the functionality of file sharing is provided between two clients using GUI. On sender's machine, we will show how the files can be uploaded to the server and the generation of storage link. Whereas on receiver's machine, we will show how the file download happens using the storage link. The demo also explains regarding the storage link expiration policy.

17. **You will be graded on Installation 3 using a rubric. While 30% of the points on the rubric are determined by Mastery Project Installation 3 Rubric, the remaining 70% of points will be determined by a portion of the rubric that you design yourself. Specifically, based on your specific vision for what is to be implemented in Installation 3, you should provide a list of 5 or more discreet items that determine whether this installation passes or fails as a demonstration of mastery.**

	Fail (0)	Pass (1)
[Item 1]	This installation fails to demonstrate mastery if desktop application homepage is not loaded.	This installation succeeds at demonstrating mastery by loading the homepage of the desktop application.
[Item 2]	This installation fails to demonstrate mastery if server status connection is not shown on the application GUI.	This installation succeeds at demonstrating mastery by showing the active status that server is connected to the application GUI.
[Item 3]	This installation fails to demonstrate mastery if application GUI doesn't display the storage link and the list of files uploaded to the server.	This installation succeeds at demonstrating mastery by displaying the uploaded files and the storage link in the GUI once the files are uploaded to the server.

[Item 4]	This installation fails to demonstrate mastery if application GUI fails to handle the upload/download requests.	This installation succeeds at demonstrating mastery by fulfilling the upload/download requests raised from the application GUI.
[Item 5]	This installation fails to demonstrate mastery if server doesn't clean the files whose expiration time is reached.	This installation succeeds at demonstrating mastery by cleaning the files whose expiration time is reached.

Final Installation & Evaluation

Now that you have done 3 installations of code and a performance evaluation, all that is left is your final installation. In this installation, you will be submitting a fully integrated version of your code, meaning that you should have code libraries for client and server operations in addition to application code that uses these libraries. You should also have a well-documented user interface for this application, which could be as complex as a GUI or as simple as a CLI.

In this section, you will define the final versions of the code files that you are submitting and the documentation that you will be submitting that instructs users how to install, run, and operate the software you have developed.

In addition, you will be providing a video that demonstrates your fully integrated system in action. In the video, you will be required to demonstrate walking through the installation of the software and running the software so that it uses some of the application protocols you have developed throughout this project.

18. **Given the project you have proposed, describe the demonstration of functionality you plan to do to show that project is complete. (2-3 paragraphs)**

    The application helps in file-sharing among the users using FTP which uses client-server architecture. If a user wants to share a list of files to other users, he can jump into the application and upload all the files. Once the files are uploaded and further clicking on "generate link", the application transfers all the files to the server and responds to the sender with a storage location path.

    Now sender can share the storage location path to the receivers, where they can download the files from the server. The application provides the functionality of file cleaning when the lifetime of storage location path expires based on the expiration policy set. By default, application sets the expiration policy to 2 days, our application also supports the modification of this expiration time while uploading the files.

One of the most important parts of implementing networked applications, is developing a plan for evaluating correctness and performance. Correctness is the software's ability to do what it should do in specific circumstances. Performance is the software's ability to operate in specific circumstances. For your Final Project Submission, you will be providing the results to an evaluation plan that you create in this section.

Correctness

This section requires that you have done a deep and thorough review of the RFC of the application protocol that you will be implementing.

19. **The correct functioning of the application protocol that you implement is dictated by the RFC. Using the RFC as a guide, describe the different message types that should be generated by the server and generated by the receiver. For each message type, include all of the fields and formatting that are involved (see Figure 2.8 in the Kurose and Ross text for an example).**

    Server Message types:

    150 File status okay; about to open data connection.

    211 System status

    220 Service ready for new user.

    226 Closing data connection. Requested file action successful (for example, file transfer or file abort).

    250 Requested file action okay, completed.

    257 "PATHNAME" created.

    426 Connection closed; transfer aborted.

    450 Requested file action not taken. File unavailable (e.g., file busy).

    551 Requested action aborted: page type unknown.

    552 Requested file action aborted. Exceeded storage allocation (for current directory or dataset).

    Client Message Types:

    200 Command okay

    225 Data connection open; no transfer in progress.

    350 Requested file action pending further information

    425 Can't open data connection.

    451 Requested action aborted: local error in processing.

    452 Requested action not taken. Insufficient storage space in system.

20. **Most application messages correspond to CRUD database functions (Create, Request, Update, Delete); some also correspond to connection management. For the application protocol you are implementing, provide an example message of each type applicable from both the client and server perspective (see p. 102 and 104 in the Kurose and Ross text for examples of HTTP messages focused on Requesting data; see p. 120 in the Kurose and Ross text for examples of SMTP messages focused on connection management).**

    Making the connection between client and the server:

    S: Server running at <IP address>: <Port>………

    C: Requesting server for the connection……….

    S: 200 Client connection from the <IP address> is accepted.

    C: 211 Connection status – Active

File transfer from client to the server:

C: Upload files <File-1, File-2, ……, File-n>

S: Upload request received.

S: Files limit check -Passed

S: Files count check – Passed.

S: File upload started.

S: 250 Files uploaded successfully.

Storage location: <storage_location_path_of_uploaded_files>

File transfer from the server to the client:

C: Fetch list of file names in the storage location <storage_location_path>

S: Fetch request received.

S: Storage link expiration check – Passed.

S: List of files in given storage location.

File 1 – 20KB

File 2 – 100KB

C: File download request for File 1.

S: File download initiated.

S: 250 File downloaded successfully.

21. **How will you demonstrate that your software is able to correctly able to generate, handle, and respond to different messages appropriately? What tools will you use?**

We will use Wire-Shark to analyze the correctness of our application. The tool helps in analyzing the data packet transmission, latency, and retransmission of the packets etc. With this we will be able to portray how well our application is responding to the data packets.

22. **As part of your demonstration of correctness, you will be asked to create a video that shows your code running and interacting correctly over the network. This includes thoroughly demonstrating what your application protocol does in many different scenarios—including scenarios where incorrect messages are sent to both the client and the server. In this part, fill out the table of scenarios that your video demonstration will use to prove correctness. Use as many scenarios as is necessary to test your particular application protocol given the RFC.**

As an example, if I implemented SMTP (Ch. 2.3), I might test the scenario where instead of sending a correct HELO message from the client to the server, I might mis format the message sent to prove that my implementation can handle incorrect messages appropriately. Specifically, I might send "HALO" instead of "HELO." I would expect the server to respond with a message with a response code of 451 Requested action aborted: error in processing.

Note: Develop as many scenarios as is necessary to thoroughly demonstrate correctness. Add lines as needed.

Scenario	How you will implement scenario	How you will prove correctness
Server accepting the connection from the specific list of IP addresses.	We will run the server at specific address and port number in one system and try to initiate the client request from the blocked IP addresses.	If the address is in the list of allowed IP addresses, then the server establishes the connection and responds with the connection successful message. If IP is in blocked list, client receives the message stating that 'Access Denied'.
Maximum number of clients served at a time	The maximum client limit at an instance will be set as default limit in the socket programming server file. We will try to make a few more connections which exceeds the default limit.	The server will respond with the message 'Server Busy' to the new connections i.e., connections requested after reaching the max client limit.
Restricting file upload size	We will try to upload files whose size is more than the default max file size set in the server configuration file.	The server will respond with a message as "File size exceeded" when the client tries to exceed the file size limit while uploading the file.
Restricting max no of files that client can upload to the server per request.	We will try to upload more no of files than the limit specified in the server configuration file.	Server will respond by displaying the message "Maximum file count exceeded" if client tries to upload a greater number of files than the optimum limit set.
Not allowing the client to download files if storage link is expired	We will try to access the expired link from one of the clients	If the storage link is expired, server will respond to the client stating that 'Link is no longer available'.

While you do not need to do anything at this point, it is important to begin considering how you will be implementing these test scenarios. As part of your submission for Lesson 5, you will be required to submit a test version of your code that includes a test harness that allows you to manipulate messages to elicit different responses.

Performance

The next part of your evaluation plan will focus on performance. Specifically, understanding how network characteristics like bandwidth, delay, and packet loss impact the performance metrics from the perspective of throughput, latency, and jitter (if appropriate as a metric).

23. Which performance metrics will you measure as part of your performance testing? How will you measure them?

We will be measuring the average time for the client-server connection establishment, maximum no of client requests processed at a given instance of time, average time taken for a file upload to the server, average time taken to fetch the list of file names from the storage location, average time taken for a requested file download etc., as part of our performance testing, and these would be measured using Wireshark and Python time module.

24. Develop a set of network performance scenarios that you will be using to evaluate performance. Three things specifically that you will want to modify are latency, bandwidth, and packet loss rate. In addition to the scenario, describe what your network application will be doing on the network and the type of graphs you will want to use to demonstrate performance in the scenario.

Scenario	Bandwidth	Latency	Packet Loss Rate (%)	Number of tests		Network Application Actions	Graphs produced
High performance network	50 MBPS	20 ms	2	5		**x-axis:** packet count **y-axis:** time	Line graph
Long, Fat Network (high bandwidth, high latency)	100 MBPS	20 ms	5	5		**x-axis:** bandwidth **y-axis:** latency	Line graph
Low Loss Rate	5 MBPS	30 ms	1	5		**x-axis:** packet loss ratio **y-axis:** no of users	Line graph
Medium Loss Rate	10 MBPS	50 ms	2	5		**x-axis:** packet loss ratio **y-axis:** no of users	Line graph
High Loss Rate	15 MBPS	70 ms	5	5		**x-axis:** packet loss ratio **y-axis:** no of users	Line graph
Wireless network (moderate bandwidth, low sporadic loss, high latency)	40 MBPS	10 ms	3	5		**x-axis:** bandwidth **y-axis:** latency	Line graph

Note that you will be using tc, a traffic shaping utility for Linux, to help create these scenarios.