2. Privacy Impact Assessment: Students can conduct a privacy impact assessment (PIA) of a new technology or system to identify potential privacy risks and develop strategies to mitigate them.

3. Regulation Compliance: Students can explore the requirements of the Data Protection Regulations and develop a plan for ensuring compliance with the regulation.

4. Cryptography: Students can learn about different cryptographic techniques and tools, such as encryption, hashing, and digital signatures, and implement them in practice.

5. Anonymization Techniques: Students can learn about data anonymization techniques, such as k-anonymity, differential privacy, and data masking, and apply them to a real-world dataset.

6. Privacy Policy Analysis: Students can analyze the privacy policies of different companies and identify gaps or areas for improvement.

7. Privacy-Enhancing Technologies: Students can explore privacy-enhancing technologies (PETs), such as virtual private networks (VPNs), Tor, and secure messaging apps, and evaluate their effectiveness in protecting privacy.

8. Privacy Breach Response Plan: Students can develop a privacy breach response plan for a company or organization, including steps to take in the event of a data breach and strategies for communicating with affected parties.

9. Ethical Considerations: Students can explore ethical considerations in data privacy, such as the balance between privacy and security, the impact of data collection and analysis on marginalized communities, and the role of data ethics in technology development.

10. Case Studies: Students can analyze case studies of privacy breaches or successful privacy protection strategies, and identify key lessons and takeaways.

DISCIPLINE SPECIFIC ELECTIVE COURSE: Unix Network Programming

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	
		Lecture	Tutorial	Practical/ Practice		
Unix Network Programming	4	3	0	1	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b: Java Programming

Learning Objectives

This course introduces the concepts of Internet protocols, ports used during communication, Client/Server concepts and various transport protocols used in computer network applications and services. The objective is to equip the students with technical knowledge of it comprises of the study of the sockets used with TCP and UDP.

Learning outcomes

On successful completion of the course, students will be able to:

- Describe and analyze the various Internet Transport layer protocols used in TCP AND UDP.
- Comprehend the concepts and structures of both TCP based connection-oriented and UDP based connectionless client server applications.
- Write various real-life client-server applications using socket programming.
- Modify, maintain and extend the present internet client-server applications and write any new type of internet applications to suit the current needs of Internet users.

SYLLABUS OF DSE

Unit 1 (6 hours)

Introduction Basics of Client Server applications, Example of day time client server, concurrent servers, protocols, sockets, port numbers.

Unit 2 (17 hours)

Connection-oriented Socket Applications: Elementary TCP sockets – Socket, connect, bind, listen, accept, fork and exec function, close function, Socket Address Structures, Byte Ordering and Manipulation Functions, TCP Client and Server for Echo, Signal Handling in case of crashing and rebooting of server, Shutdown process function. Socket Options: Getsockopt and stockpot functions, Socket states, Generic socket option.

Unit 3 (15 hours)

Connectionless Socket Applications: TCP-oriented basic concurrent client server applications, UDP oriented Echo client and server application, Handling of errors like lost datagram, Lack of flow control with UDP, determining outgoing interface with UDP.

Unit 4 (7 hours)

Elementary name and Address conversions: Domain Name System, socket functions like gethostbyname, gethostbyname2, gethostbyaddr function, uname function, gethostname function, getservbyname and getservbyport functions.

Essential/recommended readings

1. W. Richard Stevens, Bill Fenner, Andrew M. Rudoff, *Unix Network Programming*, The sockets Networking API, Vol. 1, 3rd Edition, PHI.
2. B. A. Forouzan: *Data Communications and Networking*, THM Publishing Company Ltd
3. R. Stevens, *Unix Network Programming*, PHI 2nd Edition

Suggested Practical List : (30 Hours)

Practical exercises such as

1. Implement TCP Echo client and TCP Echo server (Iterative).
2. Implement TCP Echo client and TCP Echo server (Concurrent).
3. Implement TCP daytime client and TCP daytime server (Iterative).
4. Implement TCP daytime client and TCP daytime server (concurrent).
5. Implement UDP Echo Client and UDP Echo Server.
6. Implement UDP daytime Client and UDP daytime server.
7. Implement TCP client and server (concurrent) where client gets input from the user and sends it to server. Server displays it on the screen. Server then gets another input from the user and sends it to client. Client displays it on the screen. The process continues till server or client sends "bye" to the other party.
8. Implement TCP client and server (concurrent) where client requests server to transfer a file. Assume file is smaller than 1K size. If the file is present on the server, it is sent to the client otherwise an error message is sent to client. Client copies the file on the hard disk and disconnects.
9. Implement UDP client and UDP server where server displays the IP address and port number of the client sending the datagram. Client sends a datagram (size 64 bytes) three times to the same server. Server sends the message back to client. Client reports the time elapsed in sending and receiving of the message. Use connected UDP sockets.
10. Write a program to
    a) display name of the host
    b) all IP addresses of the host.
    c) Check whether FTP and HTTP services are running on the system.
    d) Display the name of the service running on port number specified by user.