BIOMEDICAL INSTRUMENTATION

ENEX 325

Lecture : 3 **Year : III**
Tutorial : 2 **Part : I**
Practical : 1

Course Objectives:
The objective of this course is to provide a comprehensive understanding of the principles, methodologies, and instrumentation used in biomedical engineering. It emphasizes the design, operation, and maintenance of various biomedical instruments, along with their applications in medical diagnosis, monitoring, and therapy.

1 Fundamental of Medical Instrumentation (2 hours)

- 1.1 Biomedical engineering and scope
- 1.2 Biometrics and design consideration factors
- 1.3 Man instrument system and their objectives
- 1.4 Components of man instrument system

2 Bioelectric Signals and Electrodes (2 hours)

- 2.1 Body system and bioelectric phenomenon
- 2.2 Sources of bioelectric signals
- 2.3 Resting and action potentials
- 2.4 Electrode theory and their equivalent circuits
- 2.5 Types of biopotential electrodes
- 2.6 Application of electrodes in medical instrumentation

3 Physiological Transducers (2 hours)

- 3.1 Classification of transducers
- 3.2 Performance characteristics of transducers
- 3.3 Active transducers and their application in medical instruments
- 3.4 Passive transducers and their types used in medical instruments

4 Bioelectric Signals Measurement and Recording System (5 hours)

- 4.1 Aspects of bioelectric signals
- 4.2 Electrocardiography (ECG)
  - 4.2.1 Normal characteristics of electrocardiogram
  - 4.2.2 ECG lead configuration and recording techniques
  - 4.2.3 Computer aided electrocardiograph analysis
- 4.3 Electroencephalography (EEG)
  - 4.3.1 Electroencephalogram and evoked potential

4.3.2 EEG preamplifier design
4.3.3 EEG electrode configuration and recording techniques
4.3.4 Practical details of EEG

4.4 Electromyography (EMG)
4.4.1 Electromyography recording technique
4.4.2 Applications of EMG

5 Non- Invasive Diagnostic Instruments (10 hours)

5.1 Blood flow measurement
5.1.1 Magnetic blood flow meter
5.1.2 Ultrasonic blood flow meter
5.1.3 Blood flow measurement by radiographic method

5.2 Diagnostic medical imaging system
5.2.1 Radiographic imaging system: principle of generation of X-rays and its medical properties; Functional X-ray machine; Biological effects of X-rays
5.2.2 Ultrasonography imaging system
5.2.3 Computer tomography (CT-Scan) system
5.2.4 Magnetic resonance imaging system (MRI)
5.2.5 Nuclear medicine machine

6 Therapeutic Instruments (4 hours)

6.1 Cardiac pacemaker
6.2 Defibrillator machine
6.3 Function of kidneys
6.4 Principle of artificial kidneys
6.5 Hemodialysis machine
6.6 Lithotripter machine and its principle

7 Biomedical Telemetry and Telemedicine (2 hours)

7.1 Wireless telemetry
7.2 Single channel telemetry system
7.3 Multi channel telemetry
7.4 Telemedicine using mobile communication equipment

8 Ventilators (15 hours)

8.1 Mechanism of respiration
8.1.1 Types and classification of ventilators
8.1.2 Artificial ventilation
8.1.3 ICU ventilators
8.1.4 Types of ventilators
8.1.5 Ventilators terms
8.1.6 Modern ventilators

8.2 Anesthesia machines
- 8.2.1 Need for anesthesia
- 8.2.2 Anesthesia machine, its types and electronics

8.3 Blood gas analyzers and blood cell counters
- 8.3.1 Principle of measurements of blood gas analyzers, circuit diagram and clinical applications
- 8.3.2 Micro cell counters and its types

8.4 Patient monitoring systems
- 8.4.1 Overview and system concept
- 8.4.2 Cardiac monitor
- 8.4.3 Bed side ICU patient monitoring systems
- 8.4.4 Central monitors

9 Electrical Safety of Medical Equipment (3 hours)

9.1 Physiological effects of electricity
9.2 Leakage currents and methods of accident prevention
9.3 Micro shocks and macro shocks hazards
9.4 Electrical safety codes and standards
9.5 Special safety measures for electrical susceptible patients
9.6 Power distribution and protection system of the hospital

Tutorial (30 hours)

1. Description study of mechanism of respiration
2. Aspects of ICU ventilator and modern ventilators
3. Anesthesia machines and their electro-mechanical aspects.
4. Practical aspects and applications of life saving equipment
5. Study of diagnostic medical imaging system
6. Radiographic imaging system
7. Principle of generation of X-rays and its medical properties
8. Study of functional X-ray machine
9. Biological effects of X-rays
10. Study of Ultrasonography imaging system
11. Practical aspects and applications of us machine
12. Study of computer tomography (CT-Scan) system
13. Practical aspects and applications of ct scan machine
14. Study of magnetic resonance imaging system (MRI)
15. Practical aspects and applications of MRI machine
16. Study of nuclear medicine machine, Cath lab
17. Blood gas analyzers and blood cell counters, study of lab equipment and applications
18. Design study of 4 beds ICU patient monitoring system
19. Study of bioelectric signals measurement and recording system
20. Aspects of bioelectric signals and their recorders: Applications of electrocardiography (ECG), electroencephalography (EEG), electromyography (EMG)

Practical **(15 hours)**

1. Laboratory/ practical exercises based on portable medical devices (Sensors/transducers)
2. Practical on medical instruments Pulse Oximeter, ECG Machine, clinical based equipment in the lab
3. Practical on medical instruments patient monitor and devices, clinical based equipment in the lab
4. Observation and Inspection visit of different medical equipment to medical institution or hospitals and submission of inspection report

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapters	Hours	Marks distribution*
1, 2 and 3	6	10
4	5	6
5	10	10
6 and 7	6	10
8	15	20
9	3	4
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Webster, J.G. (1997). Medical instrumentation: Application and design (Latest edition). John Wiley & Sons.
2. Cromwell, L., Weibell, F.J., Pfeiffer, E.A. (1980). Biomedical instrumentation and measurements (Latest edition). Prentice-Hall.
3. Khandpur, R.S. (2004). Biomedical instrumentation: Technology and applications. McGraw-Hill.
4. Paul, S., Saikia, A., Majhi, V., Pandey, V.K. (2022). Introduction to biomedical instrumentation and its applications. Academic Press.

JAVA PROGRAMMING

ENCT 327

Lecture : 3 **Year : III**
Tutorial : 2 **Part : I**
Practical : 1

Course Objectives:
The objective of this course is to equip students with comprehensive knowledge and practical skills in Java programming, enabling them to design and develop robust, modern applications. Emphasis is placed on using the Spring Boot framework to build scalable, production-ready enterprise solutions.

1 Java Fundamentals (7 hours)

- 1.1 Overview of Java ecosystem (JDK, JVM, JRE)
- 1.2 Evolution of Java
- 1.3 Modern development tools (Maven, Gradle, IntelliJ, VS Code)
- 1.4 Review of core Java OOP, collections and exception handling
- 1.5 Functional programming: Lambda, expressions and streams API
- 1.6 Introduction to modern features like records and sealed classes

2 Java I/O and Data Handling (6 hours)

- 2.1 Streams and file handling
  - 2.1.1 Byte and character streams
  - 2.1.2 Reading/writing text files
  - 2.1.3 Object streams and serialization
  - 2.1.4 New I/O (NIO.2) file operations
- 2.2 JSON and data Interchange
  - 2.2.1 Handling JSON data with libraries like Jackson or Gson
  - 2.2.2 JSON vs. XML in REST APIs

3 Database and Web Communication (10 hours)

- 3.1 Java database connectivity (JDBC)
  - 3.1.1 JDBC architecture and drivers
  - 3.1.2 Connection, statement, and result set management
  - 3.1.3 Prepared statements for secure and efficient queries
- 3.2 Network programming and rest
  - 3.2.1 Client-server model and basics of socket communication
  - 3.2.2 Consuming restful APIs using the HTTPClient from Java 11+
  - 3.2.3 Overview of microservices architecture

4 Spring Boot and Modern Web Development (16 hours)

4.1 Spring Boot fundamentals; Dependency injection and inversion of control; Creating RESTful APIs with Spring Boot
4.2 Data Access with Spring
  4.2.1 Overview of ORM and JPA
  4.2.2 Building a repository layer with spring data JPA
  4.2.3 CRUD operations and transactional management
4.3 API Development and Security
  4.3.1 REST API design principles
  4.3.2 Handling exceptions in spring boot
  4.3.3 Overview of spring security for authentication and authorization
4.4 Deployment and Project Work
  4.4.1 Docker and containerization
  4.4.2 Packaging and deploying a Spring Boot application

5 Concurrency and Performance (6 hours)

5.1 Overview of multithreading
5.2 Thread lifecycle and synchronization
5.3 Executor framework and thread pools
5.4 Completable future and parallel streams
5.5 Best practices for thread-safe programming
5.6 Performance of multi-threaded program

Tutorial (30 hours)

1. Practice Lambda expressions and Stream API operations
2. Implement functional interfaces and method references
3. Perform file I/O operations and object serialization
4. Implement multithreading and synchronization mechanisms
5. Develop a Java program for CRUD operations using JDBC
6. Create and configure a Spring Boot project using Spring Initializer
7. Implement dependency injection using constructor, setter, and field injection
8. Develop restful CRUD APIs using Spring Boot and JPA
9. Create custom database queries using derived methods and the @query annotation
10. Implement centralized exception handling and API error responses
11. Configure authentication and authorization using Spring Security
12. Consume third-party rest APIs using WebClient or RestTemplate
13. Configure environment-specific properties and external database connections
14. Containerize and deploy a spring boot application using Docker
15. Write and execute unit and integration tests using JUnit 5 and MockMvc
16. Project work: Apply learned concepts by building a small, secure, full-stack REST service

Practical **(15 hours)**

1. Programs using Lambda expressions and Streams API
2. Implementation of file I/O operations
3. Development of a simple client-server application
4. Implementation of a Spring Boot application with a data access layer
5. Containerization of a Spring Boot application using Docker

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	7	8
2	6	8
3	10	14
4	16	22
5	6	8
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Bloch, J. (2017). Effective Java. Addison-Wesley Professional.
2. Schildt, H. (2024). Java: The Complete Reference. McGraw-Hill Education.
3. Goetz, B., Peierls, T., Bloch, J., Bowbeer, D., Holmes, J., Lea, D. (2006). Java Concurrency in Practice. Addison-Wesley Professional.
4. Walls, C. (2016). Spring Boot in Action. Manning Publications.

QUANTUM COMPUTING

ENCT 328

Lecture : 3 **Year : III**
Tutorial : 2 **Part : I**
Practical : 1

Course Objectives:
The objective of this course is to introduce the principles, mathematical foundations, and computational models of quantum computing. It emphasizes on algorithmic insights, and offer practical experience with quantum programming environments. By the end of the course, students will be able to represent and manipulate quantum states and gates, design and simulate basic quantum circuits, implement fundamental quantum algorithms and their speedup mechanisms, and explore applications of quantum machine learning (QML) techniques.

1 Fundamentals of Quantum Concepts (6 hours)

1.1 Classical versus Quantum computation
1.2 Qubits, superposition, measurement
1.3 Postulates of quantum mechanics
1.4 Dirac notation and Bloch sphere

2 Quantum Gates and Circuits (6 hours)

2.1 Quantum gates (X, Y, Z, H, Phase, CNOT)
2.2 Unitary operations and reversibility
2.3 Multi-qubit systems and tensor products
2.4 Circuit design and visualization

3 Entanglement and Quantum Communication (5 hours)

3.1 Entangled states and Bell theorem
3.2 EPR paradox
3.3 Quantum teleportation and superdense coding
3.4 Quantum key distribution protocols: BB84, E91

4 Quantum Algorithms (10 hours)

4.1 Quantum parallelism
4.2 Deutsch and Deutsch–Jozsa algorithms
4.3 Bernstein–Vazirani
4.4 Grover’s algorithm
4.5 Shor’s algorithm

5 Quantum Hardware and NISQ Era (6 hours)

5.1 Physical realization (Ion trap, superconducting, photonic)
5.2 Noise, decoherence, and qubit metrics
5.3 NISQ model and limitations
5.4 Quantum circuit execution on IBM quantum simulators

6 Quantum Error and Correction (4 hours)

6.1 Bit-flip and phase-flip errors
6.2 Simple quantum error detection
6.3 Shor's 9-qubit code
6.4 Surface code overview

7 Quantum Annealing, Optimization Algorithms (4 hours)

7.1 Principles of quantum annealing
7.2 Relationship with adiabatic quantum computation
7.3 Optimization problem formulation (QUBO, Ising models)
7.4 Quantum annealer architecture

8 Quantum Machine Learning (4 hours)

8.1 Introduction to QML and its importance
8.2 Quantum data representation and feature encoding
8.3 Variational Quantum Circuits (VQC)
8.4 Simple quantum classifiers and QNN concepts
8.5 Hybrid quantum-classical learning models

Tutorial (30 hours)

1. Classical versus quantum computation; Qubit basics, superposition, and measurement
2. State vectors, normalization and probability amplitudes
3. Dirac notation and Bloch sphere representation with simple simulations
4. Single-qubit gates (X, Y, Z, H, Phase); Matrix representation and simulation
5. Multi-qubit systems and two-qubit gates such as CNOT; Tensor product exercises and simulation
6. Entanglement and Bell states; Measurement correlation analysis
7. Simulation of quantum key distribution protocols, including BB84
8. Quantum algorithms I: Deutsch and Deutsch–Jozsa; Algorithm formulation and coding
9. Quantum algorithms II: Bernstein–Vazirani and Grover search; Coding and simulation exercises
10. Introduction to quantum hardware; Physical qubit types, noise, and decoherence; Basic modeling
11. NISQ devices; Build and execute simple circuits on IBM Quantum or equivalent simulator and analyze results

12. Quantum error modeling including bit-flip and phase-flip; Simple error detection and correction exercises
13. Quantum annealing and optimization; Formulation of QUBO and Ising problems
14. Quantum machine learning and variational circuits; Data encoding, simple VQC construction

Practical **(15 hours)**

1. IBM qiskit / quantum composer labs
2. Implement Grover's or teleportation circuit
3. Implement Shor's algorithm
4. Applications in combinatorial optimization (E.g., scheduling, routing, network optimization)
5. Hands-on simulation using quantum annealers
6. Mini-project demonstration

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	6	8
2	6	8
3	5	7
4	10	12
5	6	8
6	4	6
7	4	5
8	4	6
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Nielsen, M. A., Chuang, I. L. (2002). Quantum computation and quantum information. Cambridge University Press.
2. Bernhardt, C. (2019). Quantum computing for everyone. MIT Press.
3. Schuld, M., Petruccione, F. (2021). Machine learning with quantum computers. Springer.
4. Sahni, V. (2007). Quantum computing. Tata McGraw-Hill Publishing Company.
5. Akama, S. (2014). Elements of quantum computing: History, theories and engineering applications. Springer International Publishing.
6. Wichert, A. (2014). Principles of quantum artificial intelligence. World Scientific Publishing Co.

PYTHON PROGRAMMING FOR DATA SCIENCE

ENCT 329

Lecture : 3 **Year : III**
Tutorial : 2 **Part : I**
Practical : 1

Course Objectives:
The objective of this course is to develop a fundamental understanding of Python programming and its applications in data science. It covers Python basics including syntax, variables, data types, and control structures, and provides hands-on experience with essential libraries such as NumPy, Pandas, Matplotlib, and Seaborn for data manipulation, analysis, and visualization. By the end of the course, students will be able to work with real-world datasets and apply Python programming to solve data-driven problems effectively.

1 Introduction (8 hours)

1.1 Overview of Python and its role in data science
1.2 Setting up Python environment (Anaconda, Jupyter Notebook)
1.3 Basic syntax, variables, keywords, and expressions
1.4 Data types: Numeric, string, boolean, and type conversion
1.5 Input/output operations and basic file handling

2 Control Structures and Functions (8 hours)

2.1 Conditional statements (if, elif, else)
2.2 Looping constructs (for, while, nested loops)
2.3 Loop control statements (break, continue, pass)
2.4 Functions – Definition, arguments, return values, lambda functions
2.5 Scope and lifetime of variables, error, and exception handling

3 Data Structures and Collections (8 hours)

3.1 Lists, tuples, sets, and dictionaries – Creation and manipulation
3.2 List comprehensions and dictionary comprehensions
3.3 Working with nested data structures
3.4 Introduction to Python modules and importing libraries

4 NumPy for Numerical Computing (5 hours)

4.1 Introduction to NumPy and array creation
4.2 Indexing, slicing, and reshaping arrays
4.3 Array operations and broadcasting

4.4 Mathematical, statistical, and aggregate functions
4.5 Practical applications in scientific computing

5 Data Manipulation with Pandas (8 hours)

5.1 Introduction to Series and DataFrame
5.2 Data import/export (CSV, Excel, JSON)
5.3 Data cleaning: handling missing values, filtering, sorting
5.4 Data transformation: grouping, merging, joining, and aggregation
5.5 Descriptive statistics and data summarization

6 Data Visualization with Matplotlib and Seaborn (8 hours)

6.1 Importance of visualization in data science
6.2 Plotting basics using Matplotlib (Line, bar, scatter, histogram)
6.3 Customizing plots: Titles, labels, legends, colors, subplots
6.4 Seaborn for statistical visualization: Boxplot, heatmap, pairplot, distplot
6.5 Creating visual stories using combined plots

Tutorial (30 hours)

1. Practice on Python syntax, variables, data types, and expressions
2. Exercises on conditional statements, loops, and writing small programs
3. Practice creating and manipulating lists, tuples, sets, and dictionaries
4. Problem-solving using functions, arguments, return values, and lambda functions
5. Exercises on file input/output operations and handling exceptions
6. Numerical computation problems using NumPy arrays and vectorized operations
7. Data manipulation problems using Pandas: Filtering, grouping, merging, and summarization
8. Visualization exercises using Matplotlib and Seaborn for different chart types
9. Discussion and review of mini-project design and dataset selection
10. Mini project: Complete a small end-to-end data analysis task: Importing a dataset, cleaning and processing it using Pandas, performing numerical analysis with NumPy, and visualizing insights using Matplotlib and Seaborn

Practical (15 hours)

1. Environment setup: Installation and configuration of Python, Anaconda, and Jupyter Notebook
2. Basic programming tasks: Writing simple Python programs involving variables, expressions, loops, and conditional statements
3. Functions and data structures: Implementing Python functions, list and dictionary operations, and comprehension techniques
4. File handling: Reading and writing text and CSV files using Python's built-in libraries

5. NumPy practice: Creating, reshaping, and manipulating NumPy arrays; Performing vectorized mathematical operations
6. Data manipulation with pandas: Loading real datasets, handling missing values, filtering, sorting, and aggregating data
7. Data summarization and descriptive statistics: Using Pandas to compute mean, median, standard deviation, correlation, and other statistics
8. Visualization using Matplotlib: Plotting line graphs, bar charts, histograms, and scatter plots; Customizing labels, titles, and legends
9. Statistical visualization using Seaborn: Creating advanced visualizations such as boxplots, heatmaps, and pairplots

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	8	10
2	8	12
3	8	10
4	5	8
5	8	10
6	8	10
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. VanderPlas, J. (2016). Python data science handbook. O'Reilly Media.
2. McKinney, W. (2018). Python for data analysis. O'Reilly Media.
3. Gaddis, T. (2018). Starting out with Python. Pearson Education.
4. Matthes, E. (2023). Python crash course. No Starch Press.

WEB PROGRAMMING

ENCT 330

Lecture : 3 **Year : III**
Tutorial : 2 **Part : I**
Practical : 1

Course Objectives:
The objective of this course is to provide comprehensive understanding of web technologies for developing responsive and interactive web applications. The course emphasizes the use of modern front-end frameworks and tools, as well as techniques for integrating front-end interfaces with back-end APIs and databases. By the end of the course, students will be able to design, implement and evaluate contemporary applications while exploring current trends and future directions in web development.

1 Introduction (6 hours)

1.1 Overview of web applications and evolution of web architecture
1.2 Client-server architecture, HTTP, HTTPS, URLs, DNS, web browsers
1.3 Html basics: Syntax, tags, attributes, forms and inputs, tables, lists, multimedia, semantic HTML5 elements
1.4 CSS basics: Selectors, properties, values, box model
1.5 CSS framework: Bootstrap

2 JavaScript and Client-Side Programming (12 hours)

2.1 JavaScript essentials
  2.1.1 Data types, variables, control structures, functions
  2.1.2 Dom manipulation, events, and form validation
  2.1.3 Local storage and session storage
  2.1.4 GUI interactions
  2.1.5 JavaScript library: jQuery
2.2 Modern JavaScript (ES6+)
  2.2.1 Arrow functions, destructuring, spread/rest operators
  2.2.2 Callbacks, promises, async/await
  2.2.3 Modules and imports
2.3 Client-side applications
  2.3.1 Client-side web application development using React JS library
  2.3.2 Traditional multi-page apps vs. single-page apps (SPAs)
  2.3.3 Component-based UI and props
  2.3.4 State management and data flow
  2.3.5 Client-side routing and navigation without reloads
  2.3.6 Fetch API: Fetching, displaying data, handling errors and loading states

2.3.7 Comparison of modern JavaScript libraries: React, Angular, Vue

3 Server-Side Web Programming (9 hours)

3.1 MVC architecture in web development
3.2 Role of backend in web applications
3.3 Backend web framework: Django
3.4 Handling requests and responses
3.5 Form data handling and sessions
3.6 Routing, middleware, templating concepts
3.7 Overview and comparison of backend frameworks: Django, flask, fastapi, dot NET MVC framework, ruby on rails, java spring boot, node.js
3.8 Database integration: Relational versus NoSQL, CRUD operations, ORM concept
3.9 Authentication and authorization: Cookies, sessions, JWT
3.10 Middleware for logging, error handling, and security

4 Web Services and APIs (7 hours)

4.1 API basics: Role in web applications
4.2 REST principles and design, RESTful APIs
4.3 JSON versus XML data exchange
4.4 Data validation and serialization
4.5 Microservices

5 Web Application Security (6 hours)

5.1 Common vulnerabilities: XSS, SQL injection, CSRF
5.2 Security best practices: Input validation, sanitization, https, secure cookies, env variables
5.3 Authentication practices and token handling
5.4 Security in full-stack apps: CORS, safe sessions

6 Web Application Deployment and Modern Trends (5 hours)

6.1 Full-stack development
6.2 Testing and QA
6.3 DevOps, continuous integration (CI) and continuous delivery (CD)
6.4 Progressive web apps (PWAs), responsive design and usability

Tutorial (30 hours)

1. Walkthrough of client-server request flow (DNS→HTTP/HTTPS→ Browser)
2. Guided coding: HTML page with semantic tags, styled with CSS grid/flexbox
3. JavaScript and client-side programming
4. Hands-on exercises with DOM manipulation, event handling
5. Form validation and local storage demo

6. Guided task: Small SPA with component-based UI and fetch API integration
7. Discuss request/response cycle with server framework
8. Guided coding: Simple CRUD using Django/Flask (e.g., library system)
9. Session handling and authentication example
10. REST principles with examples
11. Hands-on exercise: Build a simple REST API endpoint and test with fetch/postman
12. Demonstration of XSS and SQL Injection vulnerabilities
13. Guided exercise: Secure login form with input validation and session/token handling
14. Discussion on PWAs, serverless, microservices
15. Develop a comprehensive project that encompasses the complete process of front-end and back-end web application development and deployment. The project should be carried out over at least six to seven lab sessions, covering the following stages: Project proposal, front-end UI/UX design, back-end and database design, API design and integration, testing and deployment, and culminating in a final presentation and demonstration

Practical **(15 hours)**

1. Build a responsive personal portfolio page using semantic HTML5, CSS grid/flexbox, and media queries
2. Create a dynamic to-do list app with DOM events, local storage, and JavaScript event handling (using jQuery)
3. Develop a CRUD application (e.g., library system) using a server-side framework (Django/Flask/FastAPI)
4. Create a frontend in React that fetches data from your backend API and displays it dynamically.
5. Extend the CRUD app with user authentication (Login/logout), form validation, and secure token/session handling

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	6	8
2	12	16
3	9	12
4	7	9
5	6	8
6	5	7
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Copes, F. (2019). The JavaScript Handbook. (https://flaviocopes.com/the-javascript-handbook-2019-edition/)
2. Nixon, R. (2025). Learning PHP, MySQL & JavaScript: A Step-by-Step Guide to Creating Dynamic Websites. O'Reilly Media.
3. Vincent, W. S. (2025). Django for Beginners: Build websites with Python and Django.
4. Richardson, L., & Amundsen, M. (2013). RESTful Web APIs: Services for a Changing World. O'Reilly Media.
5. Hoffman, A. (2024). Web Application Security: Exploitation and Countermeasures for Modern Web Applications (2nd ed.). O'Reilly Media.

OPERATING SYSTEM (OS)

ENCT 331

Lecture : 3 **Year : III**
Tutorial : 2 **Part : I**
Practical : 1

Course Objectives:
The objective of this course is to provide a comprehensive understanding of the principles, design, and functionality of operating systems. It focuses on core concepts such as process management, inter-process communication and synchronization, input/output management and file system organization. The course also introduces essential aspects of system security, administration, and virtualization, equipping students with both theoretical knowledge and practical skills in modern operating system design.

1 Introduction (6 hours)

1.1 History of operating systems (OS)
1.2 OS as an extended machine and resource manager
1.3 Type of operating system: Mainframe, server, personal, smartphone and handheld, IoT and embedded, real-time, smart-card
1.4 Operating system components: Kernel, shell, utilities, applications
1.5 Types of OS kernel: Monolithic, micro, nano, layered, hybrid, Exo Kernel
1.6 System calls, shell commands, shell programming
1.7 POSIX standard
1.8 Bootloader, MBR/GPT, UEFI and legacy boot

2 Process Management (7 hours)

2.1 Introduction to process
  2.1.1 Process description
  2.1.2 Process states
  2.1.3 Process control
2.2 Scheduling algorithms
  2.2.1 First come first serve (FCFS)
  2.2.2 Shortest job first (SJF)
  2.2.3 Shortest remaining time (SRT)
  2.2.4 Round robin (RR)
  2.2.5 Highest response ratio next (HRNN)
  2.2.6 Completely fair scheduler (CFS) used in Linux
2.3 Threads and thread scheduling

3 Process Communication and Synchronization (10 hours)

- 3.1 Principles of concurrency, race condition, critical region
- 3.2 Mutual exclusion, semaphores, and Mutex
- 3.3 Message passing and monitors
- 3.4 Classical problems of synchronization: Readers-writers problem, producer-consumer problem, dining philosopher problem
- 3.5 Deadlock: Prevention, ignorance, avoidance, detection, and recovery

4 I/O and Memory Management (9 hours)

- 4.1 I/O management
  - 4.1.1 Principles of I/O hardware and software
  - 4.1.2 I/O software layer
  - 4.1.3 Disk technologies: magnetic disk, SSD, NVMe storage
  - 4.1.4 RAID
  - 4.1.5 Concept of stable storage, cost per bit comparison
- 4.2 Memory management
  - 4.2.1 Memory address, swapping, and managing free memory space
  - 4.2.2 Virtual memory management; Paging; Segmentation
  - 4.2.3 Page replacement algorithms (FIFO, LRU, LFU, optimal), page fault, and hit ratio
  - 4.2.4 Allocation of frames
  - 4.2.5 Thrashing

5 File Systems (3 hours)

- 5.1 File concepts: Name, structure, types, access, attributes, operations
- 5.2 Directory structures: Paths and hierarchies (Linux/Windows)
- 5.3 File system implementation: Inodes, allocation methods (Contiguous, linked, indexed)
- 5.4 File system performance: Factors affecting efficiency
- 5.5 Example file systems: NTFS, EXT4, FAT32, NFS

6 Security and System Administration Tools (3 hours)

- 6.1 OS security: Cryptography, multi-factor authentication (MFA), secure boot, and sandboxing
- 6.2 Access control: Policies, lists, and OS support
- 6.3 System administration: User management, environment setup, and tools (AWK, shell scripts, Make)

7 Hypervisors and Virtual Systems (4 hours)

- 7.1 Hypervisors: Type 1 and type 2
- 7.2 Virtual machines: Creating a virtual machine in Qemu/VirtualBox/VMWare
- 7.3 Container virtualization: Docker and Kubernetes

7.4 Powershell and windows subsystem for Linux(WSL)
7.5 Performance optimization and security in virtualized environments

8 Recent Development (3 hours)

8.1 Role of operating systems in embedded and communication systems
8.2 Synchronization in real-time communication and IoT systems
8.3 Memory and I/O management in embedded communication devices
8.4 File systems used in mobile, IoT, and communication devices (YAFFS, F2FS)
8.5 Securing communication-oriented OS and networked embedded devices
8.6 Application of virtualization in IoT gateways and communication systems

Tutorial (30 hours)

1. Introduction to operating systems and kernel types
2. Unix shell programs to perform a variety of tasks
3. Problems on process states and scheduling algorithms (FCFS, SJF, RR, HRRN, CFS)
4. Thread and concurrency concepts with numerical and conceptual problems
5. Problems on semaphores and classical synchronization (Readers–writers, producer–consumer, dining philosophers)
6. Problems on deadlock prevention, avoidance, and recovery
7. Study on I/O hardware–software interface and stable storage
8. Comprehensive study on storage technologies with cost/bit and speed comparisons
9. Problems on memory allocation, fragmentation, and swapping
10. Problems on paging and segmentation
11. Problems on page replacement algorithms (FIFO, LRU, LFU, optimal)
12. Study on file system implementation and directory structures
13. Study on OS security concepts (MFA, secure boot, sandboxing)
14. Study on virtualization and containerization (Hypervisors, Docker, WSL)
15. Study on OS components and performance in IoT and embedded systems

Practical (15 hours)

1. Basic Unix commands and shell programming
2. Implementation of the ls and grep commands
3. Programs using the I/O system calls of the Unix operating system
4. Implementation of scheduling algorithms (FCFS, round robin) and the producer-consumer problem using semaphores
5. Implementation of some memory management schemes, such as paging and segmentation
6. Project work

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	6	8
2	7	10
3	10	13
4	9	12
5	3	4
6	3	4
7	4	5
8	3	4
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Tanenbaum, A. S., Bos, H. (2009). Modern operating systems. PHI Learning.
2. Stallings, W. (2012). Operating systems: Internals and design principles. Pearson Education.
3. Chaturvedi, A., Rai, B. L. (2011). Unix and shell programming. University Science Press.