## Computer Science: MINOR
w.e.f 2023-24 onwards

### COURSE STRUCTURE

<table>
<thead>
<tr>
<th>Year</th>
<th>Semester</th>
<th>Course</th>
<th>Title</th>
<th>No. Hrs./Week</th>
<th>No. of Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>II</td>
<td>1</td>
<td>Problem Solving using C - (T)</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Problem Solving using C- (P)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>III</td>
<td>2</td>
<td>Object Oriented Programming using Java- (T)</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Object Oriented Programming using Java- (P)</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>I</td>
<td>IV</td>
<td>3</td>
<td>Database Management System</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Database Management System</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>I</td>
<td></td>
<td>4</td>
<td>Object Oriented Software Engineering</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Object Oriented Software Engineering</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>I</td>
<td></td>
<td>5</td>
<td>Web Applications Development using PHP &amp; MYSQL</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Web Applications Development using PHP &amp; MYSQL</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>I</td>
<td></td>
<td>6</td>
<td>Internet of Things</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Internet of Things</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
Course Objectives

1. To explore basic knowledge on computers
2. Learn how to solve common types of computing problems.
3. Learn to map problems to programming features of C.
4. Learn to write good portable C programs.

Course Outcomes

Upon successful completion of the course, a student will be able to:

1. Understand the working of a digital computer and Fundamental constructs of Programming
2. Analyze and develop a solution to a given problem with suitable control structures
3. Apply the derived data types in program solutions
4. Use the ‘C’ language constructs in the right way
5. Apply the Dynamic Memory Management for effective memory utilization

UNIT-I

**Introduction to computer and programming:** Introduction, Basic block diagram and functions of various components of computer, Concepts of Hardware and software, Types of software, Compiler and interpreter, Concepts of Machine level, Assembly level and high-level programming, Flowcharts and Algorithms

**Fundamentals of C:** History of C, Features of C, C Tokens-variables and keywords and identifiers, constants and Data types, Rules for constructing variable names, Operators, Structure of C program, Input /output statements in C-Formatted and Unformatted I/O

UNIT-II

**Control statements:** Decision making statements: if, if else, else if ladder, switch statements. Loop control statements: while loop, for loop and do-while loop. Jump Control statements: break, continue and goto.
UNIT-III

Derived data types in C: Arrays: One Dimensional arrays - Declaration, Initialization and Memory representation; Two Dimensional arrays - Declaration, Initialization and Memory representation.

Strings: Declaring & Initializing string variables; String handling functions, Character handling functions

UNIT-IV


UNIT-V

Dynamic Memory Management: Introduction, Functions-malloc, calloc, realloc, free Structures: Basics of structure, structure members, accessing structure members, nested structures, array of structures, structure and functions, structures and pointers. Unions - Union definition; difference between Structures and Unions.

Text Books:

3. Computer fundamentals and programming in C, REEMA THAREJA, OXFORD UNIVERSITY PRESS
Reference Books


SUGGESTED CO-CURRICULAR ACTIVITIES & EVALUATION METHODS:

Unit 1: Activity: Quiz on computer hardware and software concepts

   Evaluation Method: Objective-based quiz assessing knowledge and understanding

Unit 2: Activity: Problem-solving using Decision-Making Statements

   Evaluation Method: Correctness of decision-making logic

Unit 3: Activity: Array and String Program Debugging

   Evaluation Method: Identification and correction of errors in code

Unit 4: Activity: Pair Programming Exercise on Functions

   Evaluation Method: Collaboration and Code Quality

Unit 5: Activity: Structured Programming Assignment

   Evaluation Method: Appropriate use of structures and nested structures

Computer Science Minor: II Semester
Course 1: Problem Solving using C
Credits -1

List of Experiments

1. A. Write a program to calculate simple & compound interest
   B. Write a C program to interchange two numbers.
2. Find the biggest of three numbers using C.
3. Write a C program to find the sum of individual digits of a positive integer.
4. A Fibonacci sequence is defined as follows: the first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence.
5. Write a C program to check whether a number is Armstrong or not.
6. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.
7. Write a C program that implements searching of given item in given list.
8. Write a C program that uses functions to perform the following: Addition of two matrices.
   Multiplication of two matrices.
9. Write a program for concatenation of two strings.
10. Write a program for length of a string with and without String Handling functions.
11. Write a program to demonstrate Call by Value and Call by Reference mechanism.
12. Write a Program to find GCD of Two numbers using Recursion.
13. Write a C program to perform various operations using pointers.
14. Write a C program to read data of 10 employees with a structure of 1. employee id2. aadhar no, 3. title, 4. joined date, 5. salary, 6. date of birth, 7. gender, 8. department.
15. Write a Program to demonstrate dynamic arrays using Dynamic Memory Management functions.
Computer Science Minpr: III Semester
Course 2: Object Oriented Programming using Java
Credits -3

Course Objectives
To introduce the fundamental concepts of Object-Oriented programming and to design & implement object-oriented programming concepts in Java.

Course Outcomes
Upon successful completion of the course, a student will be able to:

1. Understand the basic concepts of Object-Oriented Programming and Java Program Constructs
2. Implement classes and objects and analyze Inheritance and Dynamic Method Dispatch
3. Demonstrate various classes in different packages and can design own packages
4. Manage Exceptions and Apply Threads
5. Create GUI screens along with event handling

UNIT-I
OOPs Concepts and Java Programming: Introduction to Object-Oriented concepts, procedural and object-oriented programming paradigm
Java programming: An Overview of Java, Java Environment, Data types, Variables, constants, scope and life time of variables, operators, type conversion and casting, Accepting Input from the Keyboard, Reading Input with Java.util.Scanner Class, Displaying Output with System.out.printf(), Displaying Formatted Output with String.format(), Control Statements

UNIT-II
Arrays, Command Line Arguments, Strings-String Class Methods
Classes & Objects: Creating Classes, declaring objects, Methods, parameter passing, static fields and methods, Constructors, and ‘this’ keyword, overloading methods and access
Inheritance: Inheritance hierarchies, super and subclasses, member access rules, ‘super’ keyword, preventing inheritance: final classes and methods, the object class and its methods;
Polymorphism: Dynamic binding, method overriding, abstract classes and methods;

UNIT-III
Interface: Interfaces VS Abstract classes, defining an interface, implement interfaces, accessing implementations through interface references, extending interface;
**Packages:** Defining, creating and accessing a package, understanding CLASSPATH, importing packages.

**Exception Handling:** Benefits of exception handling, the classification of exceptions, exception hierarchy, checked exceptions and unchecked exceptions, usage of try, catch, throw, throws and finally, rethrowing exceptions, exception specification, built in exceptions, creating own exception sub classes.

**NIT-IV**

**Multithreading:** Differences between multiple processes and multiple threads, thread states, thread life cycle, creating threads, interrupting threads, thread priorities, synchronizing threads, inter thread communication.

**Stream based I/O (java.io)** – The Stream classes-Byte streams and Character streams, Reading console Input and Writing Console Output, File class, Reading and writing Files, The Console class, Serialization

**UNIT-V**


**Event Handling**- The Delegation event model- Events, Event sources, Event Listeners, Event classes, Handling mouse and keyboard events, Adapter classes, Inner classes, Anonymous Inner classes.

**Text Books:**

**Reference Books**

**SUGGESTED CO-CURRICULAR ACTIVITIES & EVALUATION METHODS:**

**Unit 1:** Activity: Quiz on Object-Oriented Programming Concepts and Java Constructs

**Evaluation Method:** Quiz Performance and Knowledge Retention

**Unit 2:** Activity: Object-Oriented Programming Assignment: Class Implementation

**Evaluation Method:** Assignment Completion and Correctness

**Unit 3:** Activity: Hands-on Lab Activity: Creating and Using Custom Java Packages

**Evaluation Method:** Lab Performance and Correctness of Code Implementation

**Unit 4:** Activity: Case Study Discussion on where multi-threading is crucial

**Evaluation Method:** Critical thinking, problem-solving, and presentation skills.

**Unit 5:** Activity: GUI design contest using Java Swings
**Evaluation Method:** GUI design, Visual appearance and user friendliness, usability, and adherence to event handling principles.

**Computer Science Minor: III Semester**

**Course 2: Object Oriented Programming using Java Lab**

Credits -1

**List of Experiments**

1. Write a Java program to print Fibonacci series using for loop.
2. Write a Java program to calculate multiplication of 2 matrices.
3. Create a class Rectangle. The class has attributes length and width. It should have methods that calculate the perimeter and area of the rectangle. It should have read Attributes method to read length and width from user.
4. Write a Java program that implements method overloading.
5. Write a Java program for sorting a given list of names in ascending order.
6. Write a Java program that displays the number of characters, lines and words in a text file.
7. Write a Java program to implement various types of inheritance
   i. Single  ii. Multi-Level  iii. Hierarchical  iv. Hybrid
8. Write a java program to implement runtime polymorphism.
9. Write a Java program which accepts withdraw amount from the user and throws an exception “In Sufficient Funds” when withdraw amount more than available amount.
10. Write a Java program to create three threads and that displays “good morning”, for every one second, “hello” for every 2 seconds and “welcome” for every 3 seconds by using extending Thread class.
11. Write a Java program that creates three threads. First thread displays “OOPS”, the second thread displays “Through” and the third thread Displays “JAVA” by using Runnable interface.
12. Implement a Java program for handling mouse events when the mouse entered, exited, clicked, pressed, released, dragged and moved in the client area.
13. Implement a Java program for handling key events when the key board is pressed, released, typed.
14. Write a Java swing program that reads two numbers from two separate text fields and display sum of two numbers in third text field when button “add” is pressed.

15. Write a Java program to design student registration form using Swing Controls. The form which having the following fields and button SAVE Form Fields are: Name, RNO, Mailid, Gender, Branch, Address.
Learning Objectives:

To familiarize with concepts of database design

Learning Outcomes: On successful completion of the course, students will be able to
1. Differentiate between database systems and file based systems
2. Design a database using ER model
3. Use relational model in database design
4. Use SQL commands for creating and manipulating data stored in databases.
5. Write PL/SQL programs to work with databases.

UNIT - I

UNIT - II
Entity-Relationship Model: Introduction, the building blocks of an entity relationship diagram, classification of entity sets, attribute classification, relationship degree, relationship classification, reducing ER diagram to tables, enhanced entity-relationship model (EER model), generalization and specialization, IS A relationship and attribute inheritance, multiple inheritance, constraints on specialization and generalization, advantages of ER modeling.

UNIT - III
Relational Model: Introduction, CODD Rules, relational data model, concept of key, relational integrity, relational algebra, relational algebra operations, advantages of relational algebra, limitations of relational algebra, relational calculus, tuple relational calculus, domain relational Calculus (DRC), Functional dependencies and normal forms upto 3rd normal form.
UNIT - IV

**Structured Query Language:** Introduction, Commands in SQL, Data Types in SQL, Data Definition Language, Selection Operation, Projection Operation, Aggregate functions, Data Manipulation Language, Table Modification Commands, Join Operation, Set Operations, View, Sub Query.

UNIT - V

**PL/SQL:** Introduction, Shortcomings of SQL, Structure of PL/SQL, PL/SQL Language Elements, Data Types, Operators Precedence, Control Structure, Steps to Create a PL/SQL Program, Iterative Control, Procedure, Function, Database Triggers, Types of Triggers.

**Text Books:**

**Reference Books**
1. Database Management Systems by Raghu Ramakrishnan, McGrawhill
2. Principles of Database Systems by J. D. Ullman
3. Fundamentals of Database Systems by R. Elmasri and S. Navathe

**SUGGESTED CO-CURRICULAR ACTIVITIES & EVALUATION METHODS:**

**Unit 1:** Activity: Seminar Presentation on Database Management Systems

**Evaluation Method:** Depth of research, clarity of explanations, ability to address questions and engage the audience.

**Unit 2:** Activity: Case Study on EER model

**Evaluation Method:** Identification of inheritance relationships, effective use of generalization and specialization, and adherence to constraints.

**Unit 3:** Activity: Exercise on Normalization: Assign students a set of unnormalized tables and have them normalize the tables to third normal form

**Evaluation Method:** Normalized table designs, identification of functional dependencies, adherence to normalization rules, and elimination of anomalies.

**Unit 4:** Activity: Competition on SQL Query Writing

**Evaluation Method:** Query correctness, efficiency, proper use of SQL commands, ability to handle complex scenarios, and creativity in query formulation.

**Unit 5:** Activity: Peer Review of PL/SQL code

**Evaluation Method:** Peer evaluation of code quality, adherence to coding standards, proper use of language elements, and logic.
List of Experiments:

1. Draw ER diagram for hospital administration
2. Creation of college database and establish relationships between tables
3. Relational database schema of a company is given in the following figure.

Relational Database Schema - COMPANY

Questions to be performed on above schema

1. Create above tables with relevant Primary Key, Foreign Key and other constraints
2. Populate the tables with data
3. Display all the details of all employees working in the company.
4. Display ssn, lname, fname, address of employees who work in department no 7.
5. Retrieve the Birthdate and Address of the employee whose name is 'Franklin T. Wong'
6. Retrieve the name and salary of every employee
7. Retrieve all distinct salary values
8. Retrieve all employee names whose address is in ‘Bellaire’
9. Retrieve all employees who were born during the 1950s
10. Retrieve all employees in department 5 whose salary is between 50,000 and 60,000(inclusive)
11. Retrieve the names of all employees who do not have supervisors
12. Retrieve SSN and department name for all employees
13. Retrieve the name and address of all employees who work for the 'Research' department
14. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birth date.
15. For each employee, retrieve the employee's name, and the name of his or her immediate supervisor.
16. Retrieve all combinations of Employee Name and Department Name
17. Make a list of all project numbers for projects that involve an employee whose last name is 'Narayan' either as a worker or as a manager of the department that controls the project.
18. Increase the salary of all employees working on the 'Product X' project by 15%. Retrieve employee name and increased salary of these employees.
19. Retrieve a list of employees and the project name each works in, ordered by the employee's department, and within each department ordered alphabetically by employee first name.
20. Select the names of employees whose salary does not match with salary of any employee in department 10.
21. Retrieve the employee numbers of all employees who work on project located in Bellaire, Houston, or Stafford.
22. Find the sum of the salaries of all employees, the maximum salary, the minimum salary, and the average salary. Display with proper headings.
23. Find the sum of the salaries and number of employees of all employees of the 'Marketing' department, as well as the maximum salary, the minimum salary, and the average salary in this department.
24. Select the names of employees whose salary is greater than the average salary of all employees in department 10.
25. Delete all dependents of employee whose ssn is ‘123456789’.
26. Perform a query using alter command to drop/add field and a constraint in Employee table.
Course Objective:
To introduce Object-oriented software engineering (OOSE) - which is a popular technical approach to analyzing, designing an application, system, or business by applying the object- oriented paradigm and visual modeling.

Course Outcomes:
Upon successful completion of the course, a student will be able to:
1. Understand and apply the fundamental principles of Object-Oriented Programming (OOP) concepts and Unified Modeling Language (UML) basics, in the development of software solutions.
2. Analyze and specify software requirements, develop use cases and scenarios, apply object-oriented analysis and design (OOAD) principles
3. Familiar with the concept of test-driven development (TDD) and its practical implementation
4. Analyze and Evaluate Software Maintenance and Evolution Strategies
5. Apply Advanced Object-Oriented Software Engineering Concepts

UNIT-I
Introduction to Object-Oriented Programming: Overview of software engineering, Introduction to Object-Oriented Programming (OOP) concepts (classes, objects, inheritance, polymorphism), Unified Modelling Language (UML) basics, Introduction to software development process and software development life cycle (SDLC).

UNIT-II
Requirements Analysis and Design: Requirements analysis and specification, Use cases and scenarios, Object-oriented analysis and design (OOAD), Design patterns, UML modelling techniques (class diagrams, sequence diagrams, state machine diagrams, activity diagrams)

UNIT-III
Software Construction and Testing: Software construction basics, Object-oriented design principles, Object-oriented programming languages (Java, C++, Python), Software testing basics (unit testing, integration testing, system testing), Test-driven development (TDD)

UNIT-IV
Software Maintenance and Evolution: Software maintenance basics, refactoring techniques Software version control, Code review and inspection, Software evolution and reengineering
UNIT-V
Advanced Topics in Object-Oriented Software Engineering: Model-driven engineering (MDE), Aspect-oriented programming (AOP), Component-based software engineering (CBSE), Service-oriented architecture (SOA), Agile software development and Scrum methodologies.

Text Book(s)
2. Programming in Java by Sachin Malhotra, Oxford University Press

Reference Books
2. Design Patterns, E.Gamma, R. Helm, R. Johnson, and J. Vlissides

SUGGESTED CO-CURRICULAR ACTIVITIES & EVALUATION METHODS:
Unit 1: Activity: Group Activity: Design and implement a small OOP project
    Evaluation Method: Presentation evaluation rubric, Project evaluation based on OOP principles.
Unit 2: Activity: Use Case Scenario Presentation & Peer Activity: Review and provide feedback on each other's use case diagrams
    Evaluation Method: Presentation evaluation rubric, Peer feedback assessment.
Unit 3: Activity: Poster Presentation: Illustrate TDD principles and benefits
    Evaluation Method: Poster presentation evaluation
Unit 4: Activity: Peer Activity: Analyze and discuss different maintenance strategies
    Evaluation Method: Peer discussion participation evaluation
Unit 5: Activity: Seminar on Design Patterns
    Evaluation Method: Depth of research, clarity of explanations, ability to address questions and engage the audience.
Suggested Software Tools: StarUML/UMLGraph/Topcased/Umberollo/ArgoUML/ Eclipse IDE, Visual Paradigm for UML/Rational Software Architect/Any other Open Source Tool

List of Experiments:
Select domain of interest (e.g. College Management System) and identify multi-tier software application to work on (e.g. Online Fee Collection). Analyze, design and develop this application using OOSE approach:

1. Develop an IEEE standard SRS document. Also develop risk management and project plan (Gantt chart).
2. Understanding of System modeling: Data model i.e. ER – Diagram and draw the ER Diagram with generalization, specialization and aggregation of specified problem statement
3. Understanding of System modeling: Functional modeling: DFD level 0 i.e. ContextDiagram and draw it
4. Understanding of System modeling: Functional modeling: DFD level 1 and DFD level 2 and draw it.
5. Identify use cases and develop the use case model.
6. Identify the business activities and develop an UML Activity diagram.
7. Identity the conceptual classes and develop a domain model with UML Class diagram.
8. Using the identified scenarios find the interaction between objects and represent them using UML Interaction diagrams.
9. Draw the state chart diagram.
10. Identify the user interface, domain objects, and technical services. Draw the partial layered, logical architecture diagram with UML package diagram notation.
11. Implement the technical services layer.
12. Implement the domain objects layer.
13. Implement the user interface layer.
Learning Objectives:
To enable students to understand open-source tools to create dynamic web pages, implement user interactivity, and gain proficiency in developing web sites

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

1. Write simple programs in PHP.
2. Understand how to use regular expressions, handle exceptions, and validate data using PHP.
3. Apply In-Built functions and Create User defined functions in PHP programming.
4. Write PHP scripts to handle HTML forms.
5. Know how to use PHP with a MySQL database and can write database driven web pages.

UNIT-I
The building blocks of PHP: Variables, Data Types, Operators and Expressions, Constants. Flow Control Functions in PHP: Switching Flow, Loops, Code Blocks and Browser Output. Working with Functions: Creating functions, Calling functions, Returning the values from User-Defined Functions, Variable Scope, Saving state between Function calls with the static statement, arguments of functions

UNIT-II
Working with Arrays: Creating Arrays, Some Array-Related Functions.
Working with Objects: Creating Objects, Accessing Object Instances, Working with Strings, Dates and Time: Formatting strings with PHP, Manipulating Strings with PHP, Using Date and Time Functions in PHP.

UNIT-III
Working with Forms: Creating Forms, Accessing Form Input with User defined Arrays, Combining HTML and PHP code on a single Page, Using Hidden Fields to save state, Redirecting the user, Sending Mail on Form Submission, and Working with File Uploads, Managing files on server, Exception handling.

UNIT-IV
Working with Cookies and User Sessions: Introducing Cookies, setting a Cookie with PHP, Session Function Overview, starting a Session, working with session variables, passing session IDs in the Query String, Destroying Sessions and Unsetting Variables, Using Sessions in an Environment with Registered Users.

UNIT-V
Interacting with MySQL using PHP: MySQL Versus MySQLi Functions, connecting to MySQL with PHP, Working with MySQL Data. Planning and Creating Database Tables, Creating Menu, Creating Record Addition Mechanism, Viewing Records, Creating the Record Deletion Mechanism.

Text Book(s)

Reference Books

**SUGGESTED CO-CURRICULAR ACTIVITIES & EVALUATION METHODS:**

**Unit 1:** Activity: Infographic explanation of client-server architecture and different server-side scripting languages.

**Evaluation Method:** Assess the accuracy, visual design, clarity, creativity, use of visual elements, presentation of the infographic explaining the benefits of server-side scripting languages.

**Unit 2:** Activity: Presentation on various open-source frameworks available in LAMP model

**Evaluation Method:** Content knowledge, organization, clarity, presentation skills, visual aids, audience engagement

**Unit 3:** Activity: Code snippets Challenge.

**Evaluation Method:** Accuracy, functionality, efficiency, code readability, and problem-solving approach of the PHP code snippets

**Unit 4:** Activity: Group discussion on Session Management in PHP

**Evaluation Method:** Active participation, knowledge sharing, critical thinking, and demonstration of Session Management

**Unit 5:** Activity: Hands-on Lab Session on MYSQL Queries

**Evaluation Method:** Lab Performance and Correctness of solution Implementation

---

**Computer Science Minor: V Semester**

**Course 4:** Web Applications Development using PHP & MYSQL

Credits -1

**List of Experiments:**

1. Write a PHP program to Display “Hello”
2. Write a PHP Program to display the today’s date.
3. Write a PHP program to display Fibonacci series.
4. Write a PHP Program to read the employee details.
5. Write a PHP program to prepare the student marks list.
6. Create student registration form using text box, check box, radio button, select, submit button. And display user inserted value in new PHP page.
7. Create Website Registration Form using text box, check box, radio button, select, submit button. And display user inserted value in new PHP page.
8. Write PHP script to demonstrate passing variables with cookies.
9. Write a PHP script to connect MySQL server from your website.
10. Write a program to keep track of how many times a visitor has loaded the page.
11. Write a PHP application to perform CRUD (Create, Read, Update and Delete) operations on a database table.

12. Create a web site using any open-source framework built on PHP and MySQL – It is a team activity wherein students are divided into multiple groups and each group comes up with their own website with basic features.
Learning Objectives:
To enable students to understand basic IoT constructs, create IoT solutions to real world problems using IoT

Learning Outcomes: On successful completion of the course, students will be able to
1. Understand various concepts, terminologies and applications of IoT
2. Learn how to build IoT devices with development boards
3. Understand various Wireless protocols for IoT
4. Learn how to use various sensors and actuators & develop IoT solutions using Arduino
5. Develop and Connect IoT with Cloud Platforms.

UNIT - I
Applications of IoT: Home Automation, Smart Cities, Energy, Retail Management, Logistics, Agriculture, Health and Lifestyle, Industrial IoT, Legal challenges, IoT design Ethics, IoT in Environmental Protection.

UNIT - II

Unit - III
Wireless Technologies for IoT: WPAN Technologies for IoT: IEEE 802.15.4, Zigbee, HART, NFC, Z-Wave, BLE, Bacnet and Modbus.
IP Based Protocols for IoT: IPv6, 6LowPAN, LoRA, RPL, REST, AMPQ, CoAP, MQTT. Edge connectivity and protocols.
Unit - IV

**Arduino Simulation Environment:** Arduino Uno Architecture, Setting up the IDE, Writing Arduino Software, Arduino Libraries, Basics of Embedded C programming for Arduino, Interfacing LED, push button and buzzer with Arduino, Interfacing Arduino with LCD.

**Sensor & Actuators with Arduino:** Overview of Sensors working, Analog and Digital Sensors, Interfacing of Temperature, Humidity, Motion, Light and Gas Sensors with Arduino, Interfacing of Actuators with Arduino, Interfacing of Relay Switch and Servo Motor with Arduino.

Unit - V

**Developing IOT’s:** Implementation of IoT with Arduino, Connecting and using various IoT Cloud Based Platforms such as Blynk, Thingspeak, AWS IoT, Google Cloud IoT Core etc. Cloud Computing, Fog Computing, Privacy and Security Issues in IoT.

**Text Book(s)**

3. Internet of Things- Dr Surya Durbha & Dr Jyoti Joglekar, Oxford University Press

**Reference Books**


**SUGGESTED CO-CURRICULAR ACTIVITIES & EVALUATION METHODS:**

**Unit 1: Activity:** Seminar on various applications of IoT through PPT

**Evaluation Method:** Content knowledge, organization, clarity, presentation skills, visualaids, audience engagement

**Unit 2: Activity:** Hands-on Lab activity on Arduino Development

**Evaluation Method:** Lab Performance and Correctness of Circuit Implementation

**Unit 3: Activity:** Group discussion on Future Wireless Technologies.
Evaluation Method: Active participation, knowledge sharing, critical thinking, and demonstration of different wireless technologies for IoT

Unit 4: Activity: Peer activity on different types of Sensors
   Evaluation Method: Peer evaluation of working principle of Sensor, use-cases of sensors.

Unit 5: Activity: Guest Lecture or Expert talk on Cloud based IoT platforms
   Evaluation Method: Active Participation, Post Talk report presentation

Computer Science Minor: V Semester
   Course 6: Internet of Things
   Credits -1

List of Experiments:

1. Understanding Arduino UNO Board and Components
2. Installing and work with Arduino IDE
3. Blinking LED sketch with Arduino
4. Simulation of 4-Way Traffic Light with Arduino
5. Using Pulse Width Modulation
6. LED Fade Sketch and Button Sketch
7. Analog Input Sketch (Bar Graph with LEDs and Potentiometre)
8. Digital Read Serial Sketch (Working with DHT/IR/Gas or Any other Sensor)
9. Working with Adafruit Libraries in Arduino
10. Spinning a DC Motor and Motor Speed Control Sketch
11. Working with Shields
12. Design APP using Blink App or Things peak API and connect it LED bulb.
13. Design APP Using Blynk App and Connect to Temperature, magnetic Sensors