UNIVERSITY SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAMME STRUCTURE

M.TECH. COMPUTER SCIENCE AND ENGINEERING

SPECIALIZATION: SOFTWARE ENGINEERING

2022-2024

GAUTAM BUDDHA UNIVERSITY

GAUTAM BUDH NAGAR, GREATER NOIDA, UP, INDIA

Semester 1

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	CS521	Advanced Data Base Management System	3	0	0	3	CC1
2	CS523	Design and Architecture for Software Systems	3	0	0	3	CC2 / FC
3	CS525	Advanced Data Structure and Algorithm	3	1	0	4	CC3
4	CS527	Research Techniques in ICT	3	0	0	3	CC4
5	CS529	Java Programming	3	0	0	3	CC5
6	ES415	Energy and Environment	3	0	0	3	OE1 /AECC
7	EN531	Language, Culture and Society	3	0	0	3	OE2 /AECC
8	CS581	Advanced Database Management System Lab	0	0	3	2	CC-L1
9	CS583	Java Programming Lab	0	0	3	2	CC-L2 / SEC
10	GP	General Proficiency	Non-Credit				
		Total Hours and Credits	21	1	6	26	

Semester 2

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	CS522	Python Programming	3	0	0	3	CC6
2	CS524	Advance Software Engineering	3	0	0	3	CC7
3	CS526	Open Source Software Systems	3	0	0	3	CC8
4	CS528	Software Engineering for Data Science	3	0	0	3	CC9
5		Elective-1	3	0	0	3	E1 / DSE
6		Elective-2	3	0	0	3	E2 / DSE
7		Generic Elective	3	1	0	4	GE1
8	CS582	Python Programming lab	0	0	3	2	CC-L3 / SEC
9	CS584	Open Source Software Systems Lab	0	0	3	2	CC-L4
10	GP	General Proficiency	Non-Credit				
		Total Hours and Credits	21	1	6	26	

Semester 3

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	CS621	Artificial Intelligence Methods for Software Engineering	3	0	0	3	CC10
2	CS623	Software Engineering for Cloud Computing	3	0	0	3	CC11
3		Elective-3	3	0	0	3	E3 / DSE
4		Elective-4	3	0	0	3	E4 / DSE
5	CA681	Robotics Lab	0	0	3	2	CC-L5
6	CA683	Summer Project	0	0	8	4	SP / E
7	CA691	Dissertation Part - I	0	0	16	8	DP1 / E
8	G P	General Proficiency	Non-Credit				
		Total Hours and Credits	12	0	27	26	

Semester 4

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	CA692	Dissertation Part - II	0	0	52	26	DP2 / E
2	GP	General Proficiency	Non-Credit				
		Total Hours and Credits	0	0	52	26	

GRAND TOTAL OF CREDITS = 104

ELECTIVES FROM DCSE

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	CS530	Software Reliability Engineering	3	0	0	3	E1
2	CS532	Parallel Computation and Applications	3	0	0	3	E1
3	CS534	Web-Based Software Engineering	3	0	0	3	E1
4	CS536	Service-Oriented Computing	3	0	0	3	E1
5	CS538	Soft Computing	3	0	0	3	E2
6	CS540	Software Maintenance	3	0	0	3	E2
7	CS542	Software Measurement and Estimation	3	0	0	3	E2
8	CS544	Software Quality Assurance	3	0	0	3	E2
9	CS625	Embedded System Design	3	0	0	3	E3
10	CS627	Applied Machine Learning	3	0	0	3	E3
11	CS629	Blockchain Technology and Software Systems	3	0	0	3	E3
12	CS631	Secure Software Engineering	3	0	0	3	E3
13	CS633	AI Enabled Cyber Security	3	0	0	3	E4
14	CS635	Big Data Platforms and Analytics	3	0	0	3	E4
15	CS637	Internet of Things	3	0	0	3	E4
16	CS639	Edge Computing	3	0	0	3	E4

OPEN AND GENERIC ELECTIVES FROM OTHER SCHOOLS

17	ES415	Energy and Environment	3	0	0	3	OE1
18	EN531	Language, Culture and Society	3	0	0	3	OE2
19	MA402	Modeling and Simulation	3	1	0	4	GE1
20	MA416	Probability and Stochastic Process	3	1	0	4	GE1

CS Computer Science for Course Code

CC Core Course from USICT for Type of Course

CC-L Core Course Lab from USICT for Type of Course

FC Foundation Course

SEC Skill Enhancement Course

E Elective

GE	General Elective from related discipline of other Deptt./School	**DP1**	Dissertation Part 1
AECC	Ability Enhancement Compulsory Course	**DP2**	Dissertation Part 2
OE	Open Elective from other discipline of other Deptt./School	**SP**	Summer Project
DSE	Discipline Specific Course		

SEMESTER 1

ADVANCED DATABASE MANAGEMENT SYSTEM			
Course Code:	**CS521**	**Course Credits:**	**3**
Course Category:	**CC1**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of database design
2 A general understanding of database ,design and dependency
3 Understanding of different types of databases
4 Knowledge of databases on the internet
5 Application on enhanced database

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Basic knowledge and understanding of ER diagram and UML class diagram.
2 Ability to apply functionality and Normalization on relational database.
3 Understand and fetch data from object oriented, parallel and distributed databases.
4 Use XML and understand unstructured data
5 Implement concept and deduction of enhanced database on different applications

UNITI INTRODUCTION TO DATABASE DESIGN

Entities, Attributes, Entity Sets, Relationships, Key Constraints, Participation Constraints, Weak Entities, UML Class Diagrams, Subclasses, Superclasses, Inheritance,Specialization, Generalization, Constraints and Characteristics of Specialization and Generalization Hierarchies, Modeling of UNION Types Using Categories, Representing Specialization and Generalization In UML Class Diagrams, Data Abstraction, Knowledge Representation and Ontology Concepts.

UNIT II DATABASES DESIGN THEORY

Problems Caused by Redundancy, Decompositions, Problems Related to Decomposition, Reasoning About FD's, FIRST, SECOND, THIRD Normal Form, BCNF, Forth Normal Form, Lossless Join Decomposition, Dependency Preserving Decomposition, Schema Refinement in DataBase Design, Multi Valued Dependencies.

UNITIII OBJECT- ORIENTED, PARALLEL AND DISTRIBUTED DATABASES

Overview of Object-Oriented Concepts, Object Identity, Object Structure, Type Constructor, Encapsulation of Operations, Methods and Persistence; Architectures For Parallel Databases, Parallel Query Evaluation, Parallelizing Individual Operations, Sorting Joins, Distributed Database Concepts, Data Fragmentation, Replication and Allocation Techniques for Distributed Database Design, Query Processing in Distributed Databases, Concurrency Control and Recovery in Distributed Databases.

UNITIV DATABASES ON THE WEB AND SEMI–STRUCTURED DATA

Web interface, XML, structure of XML data, querying XML data, storage of XML data, XML applications, semi-structured data model, indexes for text data.

UNITV ENHANCED DATA MODELS FOR ADVANCED APPLICATIONS

Active database concepts, temporal database concepts, spatial databases: concept and architecture, deductive databases and query processing, mobile databases, Geographic Information Systems (GIS).

Text Books:

1. Elmasri andNavathe, Fundamentals of Database Systems,
2. Ramakrishnan and Gehrke, Database Management Systems,

References Books:

1. Korth, Silberschatz, Sudarshan, Database System Concepts,
2. Rob and Coronel, Database Systems: Design, Implementation and Management,
3. Date and Longman, Introduction to Database Systems,

DESIGN AND ARCHITECTURE OF SOFTWARE SYSTEM			
Course Code:	**CS523**	**Course Credits:**	**3**
Course Category:	**CC2**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Understand the creational and structural patterns.
2 Be capable of applying his knowledge to create an architecture for given application.
3 Be able to explain the role of analyzing architectures.
4 Understanding of software design.
5Be able to identify different structural patterns.

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Understand the architecture, creating it and moving from one to any, different structural patterns.
2 ability to analyze the architecture and build the system from the components.
3Design creational and structural patterns.
4Learn about behavioral patterns.
5Do a case study in utilizing architectural structures.

UNIT I SOFTWARE ARCHITECTURE

Foundations of software architecture, goals of software architecture limitations, role of software architect, types of architecture, qualities attributes, qualities scenario, architectural styles, common architectural design, architectural design process, key architecture principles, key design principles, functional and non-functional properties of software architectures, heterogeneous architectures, virtual machine architecture, data flow architecture, service-oriented architecture.

UNIT II DESIGN FUNDAMENTALS AND METHODOLOGIES

Nature of design process: objectives, building modules, constructs, design qualities, assessing the design, design viewpoints for software, design strategies: top down and bottom up, organizational methods and design, Jackson structural programming, Jackson system development, models for software architecture

UNIT III SOFTWARE ARCHITECTURE DESIGN

Architectural design and mapping, architecture design patterns, module architecture view,styles of themodule view type, execution architecture view, code architecture view, component-and-connector viewtype, styles of component-and-connector viewtype, allocation viewtype and styles, object-oriented architecture, user interface architecture, quantified design space, formalizing architectural description language, first class connectors, tools for architectural design: Unicon, A4; exploiting style in architectural design, architectural interconnection.

UNIT IV INTERACTION ORIENTED SOFTWARE ARCHITECTURE AND DESIGN

Model-View-Controller (MVC), Presentation-Abstraction-Control (PAC) architecture, distributed architecture: client server architecture, multi-tire, service-oriented architecture (SOA). Design principles, traditional approach to design, Structured Analysis Design Technique (SADT), Structures System Analysis and Design Method (SSADM), user interface design; human factor, human computer interaction, interface design guide lines, standards, object-oriented analysis and design.

UNIT V PATTERNS

Design patterns, creational patterns, access control patterns, service variation patterns, service extension patterns, archetypes patterns, model driven architecture with archetype patterns, literate modeling, Customer Relationship Management (CRM) archetype pattern, product archetype pattern, quantity archetype pattern, rule archetype pattern, layering, organizing domain logic, mapping to relational databases, web presentation, domain logic patterns, data source architectural patterns, object-relational behavioral patterns, objectrelational structural patterns, object-relational metadata mapping patterns, web presentation patterns, distribution patterns, offline concurrency patterns.

Text Books:

1. Software Architecture Perspectives on an Emerging Discipline, M. Shaw Prentice-Hall, 1996.
2. Software Architecture Design: Methodology and Styles, Lixin Tao, Xiang Fu and Kai Qian, Stipes Publishing L.L.C., 2006.
3. Software Architecture in Practice, Len Bass, Paul Clements, Rick Kazman, Pearson Education Asia,2003.

References Books:

4. Software Design, David Budgen, Addison-Wesley, 1994.
5. Software Engineering, Pressman R.S, McGraw Hill Inc., 1996.

-

ADVANCED DATA STRUCTURE AND ALGORITHM			
Course Code:	**CS525**	**Course Credits:**	**4**
Course Category:	**CC3**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 01**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Understand the appropriate data structures, ADT libraries, and use it to design algorithms for a specific problem.
2 Be capable of solving problems using abstraction techniques.
3 Be able to choose appropriate algorithms for a specific problem.
4 Be able to analyze algorithms in terms of their efficiency and correctness.
5 Understanding the recent developments in the area of algorithm design.

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Design and analyze programming problem statements.
2 Choose appropriate data structures and algorithms for a specific problem.
3 Understand the necessary mathematical abstraction to solve problems.
4 Come up with analysis of efficiency and proofs of correctness.
5 Comprehend and select algorithm design approaches in a problem specific manner.

UNIT I INTRODUCTION

Review of Basic Concepts: Abstract data types, Data structures, Algorithms, Big-Oh, Small-Oh, Omega, Small-Omega and Theta Notations, finding time complexity of programs, **Recurrence Relations:** Solving Recurrence Relations, Substitution Method, Master Theorem.

UNIT II

Hashing: Review of Hashing, Hash Function, Collision Resolution Techniques in Hashing, Separate Chaining, Open Addressing, Linear Probing, Quadratic Probing, Double Hashing, Rehashing, Extendible Hashing, Recent Trends in Hashing.

UNIT III TREES & GRAPH

Trees: Binary Search Trees, AVL Trees, Red Black Trees, 2-3 Trees, B-Trees, Splay Trees, Minimum Spanning Tree (MST), Kruskal's Algorithm and Prim's Algorithm, Applications to MST.

Graph: Graph, Breadth First Search, Depth First Search, Shortest path in edge-weighted case (Dijkstra's), Bellman Ford Algorithms, Topological Sorting.

UNIT IV SELECTED TOPICS
Strassen's Matrix Multiplication, Greedy method VS Dynamic Programming, Job sequencing with deadlines, Fractional Knapsack Problem, 0/1 Knapsack Problem, Travelling Salesman Problem, Huffman coding, Pre order, Post order, Inorder traversal, Postfix to infix notation, Infix to Postfix notation.

UNIT V
Linear Programming: Geometry of the feasibility region and Simplex algorithm
NP-completeness: Examples, proof of NP-hardness and NP-completeness.
Recent Trends: Recent Trends in problem solving paradigms using recent searching and sorting techniques by applying recently proposed data structures.

Text Books:

1. Introduction to Algorithms, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.
2. Algorithms Unlocked: Thomas H. Cormen.
3. The Algorithm Design Manual, Steven S. Skiena.

References Books:

4. Algorithms: Robert Sedgewick and Kevin Wayne.
5. Advanced Data Structures: Peter Brass.

RESEARCH TECHNIQUES IN ICT			
Course Code:	**CS 527**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 To get the understanding about what is research.
2 To have the knowledge about research methodology.
3 Awareness of methodology to be followed for report and paper writing.
4 Familiarization to different models and algorithms during research.
5 Conversant with various simulation and soft computing techniques.

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Acquainted with procedure for carrying out the research and its importance.
2 Can identify and apply the appropriate research techniques according to domain of research.
3 Recognize models and algorithms required for their study.
4 Know about research paper writing and publication procedure.
5 Will get the direction about different simulation packages.

UNIT I INTRODUCTION TO RESEARCH TECHNIQUES

Meaning of research, objectives of research, motivation in research, types of research, characteristics and prerequisites of research, significance of research, research process, sources of research problem, criteria of identifying the problem, necessity of defining the problem, errors in selecting research problem, technique involved in defining the problem, report and paper writing.

UNIT II DATA ANALYSIS AND STATISTICAL TECHNIQUES

Data and their analyses, quantitative methods and techniques, Measure of central tendency, measures of variation, frequency distribution, analysis of variance, methods, Correlation analysis, regression analysis, time series and forecasting, introduction to discriminant analysis, factor analysis, cluster analysis, conjoint analysis, probability distribution, binomial distribution, poisson distribution, uniform distribution, exponential distribution, and normal distribution, sampling methods, test of hypothesis.

UNIT III MATHEMATICAL MODELING

Steps of modeling, operations research models like queuing theory, stochastic processes, application of models, conceptual framework development and validation techniques, optimization techniques.

UNIT IV ALGORITHMIC RESEARCH

Algorithmic research problems, types of algorithmic research, types of solution procedure, steps of algorithm development , steps of algorithmic research, design of experiments.

UNIT V SIMULATION AND SOFT COMPUTING TECHNIQUES

Introduction to soft computing, artificial neural network, genetic algorithm, fuzzy logic and their applications, tools of soft computing, need for simulation, types of simulation, simulation language, fitting the problem to simulation study, simulation models, output analysis, data simulation packages like MATLAB, NS2, ANSYS, Cadence.

Text books:

1. Research Methodology: Methods and Techniques, C.R. Kothari

Reference Books:

1. Research Methodologies, R. Panneerselvam, Prentice Hall, 2007.
2. Research in Education, Best John V. and James V Kahn, Wiley eastern, 2005.
3. Elements of Educational Research, Sukhia, S.P., P.V. Mehrotra, and R.N. Mehrotra, PHI publication, 2003.
4. Methodology of Research Education, K. Setia, IEEE publication, 2004.
5. Research methodology, Methods and Techniques, Kothari, C.R., 2000.

JAVA PROGRAMMING			
Course Code:	**CS529**	**Course Credits:**	**3**
Course Category:	**CC5**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of basic Object-Oriented paradigm, practices and application.
2 A general understanding of class, object and methods.
3 Understanding of multithreading and applet.
4 Basic knowledge of swings and Beans with implementation.
5 Understanding of Servlet programming.

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Basic knowledge and understanding of object-oriented programming.
2 Ability to apply OOPs concept in real life problem.
3 Ability to design, develop, maintain and evaluate large-scale software systems.
4 To produce efficient, reliable, robust and cost-effective software solutions using Java.
5 Ability to perform independent research and analysis.

UNIT I OBJECT-ORIENTED PROGRAMMING

Concept of object-oriented programming (OOP), benefits of OOP, application of OOP, Java history, Java features, Java streaming, Java and Internet, Java contribution to Internet: Java applets, security, portability; Java environment, Java library, Java program structure, Java program, Java Virtual Machine (JVM) architecture, Just In Time compiler (JIT), data type, variables and arrays, operators, control statements, object-oriented paradigms; abstraction, encapsulation, inheritance, polymorphism, Java class and OOP implementation

UNIT II DATA TYPE, OPERATORS AND CONTROL STATEMENT

Data types, Java key words, identifiers, constants, variables, declaration and scope of the variable, symbolic constant, type casting, arithmetic operator, relational operator, logical operator, assignment operator, increment and decrement operator, conditional operator, bitwise operator, ?: operator, arithmetic expressions, expressions, type conversions in expressions, mathematical functions, more data types: arrays, strings, vectors, wrappers classes, program control statements: decision making and branching: if, if….else, else….if, else if ladder, switch, decision making and looping: while, do….while, for.

UNIT III CLASSES, OBJECTS AND METHODS

Java class libraries, class fundamentals, object, methods, adding variables, add methods, creating objects, accessing class members, constructors, methods overloading, static members, nesting of methods, inheritance: extending a class, overriding methods, final variables and methods, final classes, finalizer methods, abstract methods and classes, visibility control, exception handling fundamental.

UNIT IV INTERFACES AND PACKAGES

Interfaces, extending interfaces, implementing interfaces, interfaces references, accessing interface variable, creating queue interface, variable in interfaces, packages, finding a packages and classpath, package and member access, Java API package, system package, naming conventions, creating package, accessing a package, adding a class to a package, hiding classes,

UNIT V MULTITHREADING AND APPLET PROGRAMMING

Multithreading programming: creating threads, thread class and runnable interface extending the thread class, stopping and blocking a thread, life cycle of a thread, thread methods, thread exceptions, thread priority, synchronization, thread communication using notify(), wait(), and notify all(), applet programming : applet basic, applets architecture, a complete applet skeleton, building applets code, applets life cycle, creating a executable applet, designing a web page, applets tag, passing parameters to applets, applets and HTML.

Text Books:

1. Programming with JAVA, E. Balagurusawamy, Tata McGraw Hill, 1998.
2. JAVA Beginner‟s guide, Herbert Schildt, Tata McGraw Hill, 2007.
3. Java How to Program, Deitel&Deitel, Prentice-Hall, 1999.

ADVANCED DATABASE MANAGEMENT SYSTEM LAB			
Course Code:	**CS581**	**Course Credits:**	**2**
Course Category:	**CC-L1**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lab (Hrs/Week) / Total No. of Lab	**03 / 10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To explore the features of a Database Management Systems
2. To interface a database with front end tools
3. To understand the internals of a database system
4. To provide a strong foundation in advanced database concepts from an industry perspective.
5. To learn query processing and transaction management concepts for object-relational database and distributed database

COURSE OUTCOME
At the end of the course the students should be able to:
1. Develop and apply critical thinking skills.
2. Design and present Lab as well as project reports
3. Apply appropriate methods for the analysis of raw data
4. Perform logical troubleshooting as and when required.
5. Verify and implement the concepts and theory learnt in class.

1. Introduction to MySQL, PostgreSql, Microsoft Sqlsoftwares.
2. An exercise of data types in PostGresql& Data Definition Language Commands
3. Exercise on Data Manipulation Language and Transaction Control Commands using PostgreSql.
4. Exercise on Types of Data Constraints using PostgreSql.
5. Exercise on JOINS (Single-Table) Using Normalization
6. Exercise on JOINS (Multiple-Table) Using Normalization
7. Exercise on GROUP BY/ORDER BY Clause and Date Arithmetic using PostgreSql.
8. Exercise on different Functions (Aggregate, Math and String)

9. Exercise on different types of sub queries
10. Procedures, View and Triggers

JAVA PROGRAMMING LAB			
Course Code:	**CS583**	**Course Credits:**	**2**
Course Category:	**CC-L2**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Labs (Hrs/Week):	**01 (3 Hr)**		
Total No. of Labs :	**10**	**End Sem. Exam Hours:**	**3**
COURSE OBJECTIVES			
1 Knowledge of basic Object-Oriented paradigm, practices and application.			
2 A general understanding of class, object and methods.			
3 Understanding of multithreading and applets.			
4 Basic knowledge of Swings and Beans with implementation.			
5 Understanding of Servlet programming.			
COURSE OUTCOMES			
At the end of the course the students should be able to:			
1 Basic knowledge and understanding of object-oriented programming.			
2 Ability to apply OOPs concept in real life problems.			
3 Ability to design, develop, maintain and evaluate large-scale software systems.			
4 To produce efficient, reliable, robust and cost-effective software solutions using Java.			
5 Ability to perform independent research and analysis.			

1. Write a separate Java Code to implement each of the following:
   Class, Command Line Argument, how to enter value through keyboard

2. Write a separate Java Code to implement each of the following data types:
   Variable, Constant, Arrays, Strings, Vectors, Wrappers Classes, Type Casting, Operators, Decision statement, Loops statement and Branch statements and Exception handling

3. Write a separate Java Code to implement each of the following OOPs concepts:
   Abstraction, Encapsulation, Inheritance, Polymorphism,Method Overloading and Method Overriding

4. Write a separate Java Code to implement each of the following:
Class, Object, Constructors, Method, and Visibility Controls: Private, Public and Protected

5. Write a separate Java Code to implement each of the following:
Final variable, final class, final method, abstract class, abstract method and concrete method

6. Write a separate Java Code to implement each of the following:
Interface, extending and implementing interface

7. Write a separate Java Code to implement each of the following:
Multithreading: Create thread with thread class and runnable interface, thread priorities,synchronization

8. Write a separate Java Code to implement each of the following:
Swing and Beans

9. Write a separate Java Code to implement each of the following:
Swing Button: JButton, JToggleButton, CheckBoxes, Radio Button, JComboBox, Text Boxes

10. Write a separate Java Code to implement each of the following:
Servlet and JSP

Semester 2

PYTHON PROGRAMMING			
Course Code:	**CS522**	**Course Credits:**	**3**
Course Category:	**CC6**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To learn and understand Python programming basics and paradigm
2. To learn and understand python looping, control statements and string manipulations.
3. Students should be made familiar with the concepts of GUI controls and designing GUI applications.
4. To learn and know the concepts of file handling, exception handling and database connectivity.
5. To learn and understand database connectivity in the python programming language.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. To read and write simple Python programs.
2. To develop Python programs with conditionals and loops.
3. To define Python functions and to use Python data structures –- lists, tuples, dictionaries
4. To do input/output with files in Python
5. To do searching ,sorting and merging in Python

UNIT I Introduction:

The Programming Cycle for Python , Python IDE, Interacting with Python Programs , Elements of Python, Type Conversion. Basics: Expressions, Assignment Statement, Arithmetic Operators, Operator Precedence, Boolean Expression.

UNIT II Conditionals:

Conditional statement in Python (if-else statement, its working and execution), Nested-if statement and Elif statement in Python, Expression Evaluation & Float Representation. Loops: Purpose and working of loops , While loop including its working, For Loop , Nested Loops , Break and Continue.

UNIT III Function:

Parts of A Function , Execution of A Function , Keyword and Default Arguments ,Scope Rules. Strings : Length of the string and perform Concatenation and Repeat operations in it. Indexing and Slicing of Strings. Python Data Structure : Tuples , Unpacking Sequences , Lists , Mutable Sequences , List Comprehension , Sets , Dictionaries Higher Order Functions: Treat functions as first class Objects , Lambda Expressions

UNIT IV File I/O :

File input and output operations in Python Programming Exceptions and Assertions Modules : Introduction , Importing Modules , Abstract Data Types : Abstract data types and ADT interface in Python Programming. Classes : Class definition and other operations in the classes , Special Methods (such as _init_, _str_, comparison methods and Arithmetic methods etc.) , Class Example , Inheritance , Inheritance and OOP.

UNIT V Iterators & Recursion:

Recursive Fibonacci , Tower Of Hanoi Search : Simple Search and Estimating Search Time , Binary Search and Estimating Binary Search Time Sorting & Merging: Selection Sort , Merge List , Merge Sort , Higher Order Sort

Text Books:

1. Allen B. Downey, ``Think Python: How to Think Like a Computer Scientist‘‘, 2nd edition, Updated for Python 3, Shroff/O‘Reilly Publishers, 2016

2. Guido van Rossum and Fred L. Drake Jr, ―An Introduction to Python – Revised and updated for Python 3.2, Network Theory Ltd., 2011.

3.John V Guttag, ―Introduction to Computation and Programming Using Python‘‘, Revised and expanded Edition, MIT Press , 2013

ADVANCED SOFTWARE ENGINEERING			
Course Code:	**CS524**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of basic to advanced software engineering methods.
2 A General understanding of object-orientated software engineering.
3 Understanding of component-based software engineering.
4 Understanding of aspect-oriented software engineering.
5 Understanding of software re-engineering and reverse engineering.

COURSE OUTCOME
At the end of the course the students should be able to:
1 Basic to advanced knowledge and understanding of software engineering methods.
2 Ability to apply software engineering principles and techniques.
3 Ability to design, develop, maintain and evaluate object-orientated software.
4 To produce efficient, reliable, robust and cost-effective solutions for component-based software.
5 Ability to design good software.

UNIT I SOFTWARE ENGINEERING

Introduction to software engineering, Software Development Life Cycle, software process models, requirement analysis and design, software design process, coding, software testing, implementation and maintenance, software metrics.

UNIT II OBJECT-ORIENTED SOFTWARE ENGINEERING

Object-Oriented Software Engineering, object-oriented paradigm, object modeling languages, object-oriented analysis, object-oriented design, object-oriented programming, object-oriented metrics, object-oriented case tools, object-oriented software testing.

UNIT III COMPONENT-BASED SOFTWARE ENGINEERING

Component-Based Software Engineering (CBSE), CBSE and software reuse, CBSE vs. object-oriented software engineering, CBSE processes, domain engineering, component engineering, component-based software development life cycle, component vs. object, component-oriented programming, component-oriented programming vs. object-oriented programming, component-based technology, component-based software testing, component-oriented metrics.

UNIT IV ASPECT-ORIENTED SOFTWARE ENGINEERING

Software engineering with aspects, aspects, aspect vs. object, aspect vs. component, join points and pointcuts, separation of concerns, crosscutting concerns, scattering and tangling, aspect-oriented programming, aspect-oriented software testing.

UNIT V SOFTWARE RE-ENGINEERING AND REVERSE ENGINEERING

Re-engineering concept and approaches, redevelopment vs. reengineering, reengineering process, software re-engineering techniques, reverse engineering, levels of reverse engineering: re-documentation, design recovery, specification recovery, conditions for reverse engineering, forward engineering, restructuring, re-engineering, benefits of reverse engineering.

TEXT BOOKS:

- Pankaj Jalote, An Integrated Approach to Software Engineering, Narosa Publishing House, New Delhi 1997.
- Ian Sommerville, Software Engineering, Pearson Education, 2009.

REFERENCE BOOKS:

- Pressman Roger S., Software Engineering: Practitioner's Approach, McGraw-Hill Inc., 2004.
- N. S. Gill, Software Engineering: Software Reliability, Testing and Quality Assurance, Khanna Book Publishing Co (P) Ltd., New Delhi, 2002.
- J. Rumbaugh, M. Blaha, W. Premerlani, Object-Oriented Modeling and Design, PHI, 1991.
- George T. Heineman, William T. Councill, Component-Based Software Engineering: Putting the Pieces Together, Addision Wesley, 2001.
- Robert E. Filman, Tzilla Elrad, Siobhán Clarke, Mehmet Aksit, Aspect-Oriented Software Development Addison-Wesley Professional, 2004.

OPEN SOURCE SOFTWARE SYSTEMS			
Course Code:	**CS526**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of open source software.
2 A General understanding of open-source technology.
3 Understanding of open-source language.
4 Understanding of open-source software applications and framework.
5 Understanding of open-source in the enterprise.

COURSE OUTCOME
At the end of the course the students should be able to:
1 Basic to advanced knowledge of open-source software.
2 Ability to apply open-source technology.
3 Ability to design, develop, maintain and evaluate open source software application and framework.
4 Ability to apply open source language.
5 Ability to apply open-source software in the enterprise.

UNIT I OPEN SOURCE SOFTWARE

Open Source Software (OSS), history, philosophy and ethics of open source software, Pernes' principle, open source software development methodology, open source vs. closed source, open source software vs. free software, open source software vs. source available, Windows and Linux, open source development environment, methods and models, standards, open source standards, benefits of open standards, standard setting organizations and processes, project management via open source and open standard, OSS in e-government.

UNIT II OPEN SOURCE TECHNOLOGY

Open source technology and platform, Operating system: Linux, Berkeley Software Distribution; web server: Apache; communication servers: send mail, jabber; application and messaging server: JBoss, Zope, Zend.

UNIT III OPEN SOURCE LANGUAGES

Ruby, Ruby and object-orientation, data, expressions and flow control, class, object and modules, project and libraries, developing a basic Ruby application, PHP, configure environment, PHP basic, functions, arrays, object-oriented PHP, MYSQL, PostgreSQL.

UNIT IV OPEN SOURCE SOFTWARE APPLICATIONS AND FRAMEWORK

Open source desktop applications, Wiki software, LAMP application, web server and database server application, OSS management tools: taskjuggler, dotProject.net, rapid web application development framework: Ruby on Rail, Model-View-Controller model, Don't Repeat Yourself principle.

UNIT V OPEN SOURCE IN THE ENTERPRISE

Nature of open source, leadership in open source software life cycle, comparison in the risks of commercial and open source software, measuring the maturity of open source, designing an open source strategy, open source licenses, comparison of open source licenses, open source empowerment.

Text Books:

1. Paul Kavanagh, Open Source Software: Implementation and Management, Digital Press, 2004.
2. W. Jason Gilmore, Beginning PHP and MySQL, Apress, 2010.
3. Timothy Fisher, Ruby on Rail, Apress, 2009.

Reference Books:

4. Dan Woods, Open Source for the Enterprise: Managing Risks, Reaping Rewards, O'Reilly, 2005.
5. James Lee, Brent Ware,Open Source Web Development with LAMP, Pearson Education, 2008.
6. Steven Weber, The Success of Open Source, HarvardUniversity Press, 2004.
7. Peter Cooper, Beginning Ruby, Apress, 2007.

SOFTWARE ENGINEERING FOR DATA SCIENCE			
Course Code:	**CS528**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Understanding software engineering.
2 Understanding different types of software engineering.
3 Understanding re-engineering and reverse engineering.
4 Understanding the concepts of data science.
5 Critically evaluate data visualizations based on their design and use for communicating stories from data

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Able to understand software engineering.
2 Understanding different approaches in software engineering.
3 Understanding what data science is and its importance in software engineering.
4 Explain how data is collected, managed and stored for data science.
5 Understand the key concepts in data science, including their real-world applications and the toolkit used by data scientists.

UNIT I SOFTWARE ARCHITECTURE

Introduction to software engineering, Software Development Life Cycle, software process models, requirement analysis and design, software design process, coding, testing, implementation and maintenance, software metrics, agile software engineering, clean room software engineering, empirical software engineering.

UNIT II OBJECT-BASED, COMPONENT-BASED AND ASPECT ORIENTED SOFTWARE ENGINEERING

OOSE, object-orientation paradigm, object-oriented programming languages, object modeling languages, object-oriented analysis, object-oriented design: design stereotypes and objects, design patterns, Component-Based Software Engineering (CBSE), CBSE and software reuse, CBSE vs. object-oriented software engineering, CBSE processes, component-based software development life cycle, component vs. object, component-oriented programming, component oriented programming vs. object-oriented programming, Software engineering with aspects, aspects, aspect vs. object, aspect vs. component, join points and point cuts, concepts of Aspect Oriented Programming, comparison to other programming paradigms.

UNIT III SOFTWARE REENGINEERING AND REVERSE ENGINEERING

Re-engineering concept and approaches, redevelopment vs reengineering, reengineering process, reverse engineering, levels of reverse engineering: re-documentation, design

recovery, specification recovery, conditions for reverse engineering, supporting techniques: forward engineering, restructuring, reengineering, benefits of reverse engineering, software re-engineering patterns, patterns based software reengineering, software re-engineering techniques.

UNIT IV INTRODUCTION TO DATA SCIENCE

Introduction to core concepts and technologies: Introduction, Terminology, data science process, data science toolkit, Types of data, Example applications, Data collection and management: Introduction, Sources of data, Data collection and APIs, Exploring and fixing data, Data storage and management, Using multiple data Sources, Data analysis: Introduction, Terminology and concepts, Introduction to statistics, Central tendencies and distributions, Variance, Distribution properties and arithmetic, Samples/CLT, Basic machine learning algorithms, Linear regression, SVM, Naive Bayes.

UNIT V VISUALIZATION AND RECENT TRENDS

Data visualisation: Introduction, Types of data visualisation, Data for visualisation: Data types, Data encodings, Retinal variables, Mapping variables to encodings, Visual encodings, Applications of Data Science, Technologies for visualization, Bokeh (Python), Recent trends in various data collection and analysis techniques, various visualization techniques, application development methods of used in data science.

Text Books:

1. Pankaj Jalote, An Integrated Approach to Software Engineering, Narosa Publishing House, New
Delhi 1997.
2. Ian Sommerville, Software Engineering, Pearson Education, 2009.3. Software Architecture in Practice, Len Bass, Paul Clements, Rick Kazman, Pearson Education Asia,2003.
3. Jure Leskovek, Anand Rajaraman and Jeffrey Ullman.Mining of Massive Datasets.v2.1,Cambridge University Press.

References Books:

4. Pressman Roger S., Software Engineering: Practitioner's Approach, McGraw-Hill Inc., 2004.
5. N. S. Gill, Software Engineering: Software Reliability, Testing and Quality Assurance, Khanna
Book Publishing Co (P) Ltd., New Delhi, 2002.
6. Cathy O‟‟Neil and Rachel Schutt. Doing Data Science, Straight Talk From The Frontline. O‟‟Reilly.

OPEN SOURCE SOFTWARE SYSTEM LAB			
Course Code:	**CS584**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**!P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 +0+0**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 . To expose students to a free open-source software environment and introduce them .
2. Install Red Hat and Debian based Linux distributions
3.Maintain operating system updates.
4. Interoperate between Linux and Windows, Install and configure useful application software (LAMP stack apps, SAMBA, and others);
5. Understanding of each of the following: ◦ Basics of Linux security; Major issues invce licensing solved in Open-Source.

COURSE OUTCOMES
At the end of the course the students should be able to:
1.Implement various applications using build systems.1.Implement various applications using build systems.
2.Understand the installation of various packages in open-source operating systems
3.Create simple GUI applications using Gambas
4. Understand various version control systems
5. Understand the kernel configuration and virtual environment

LAB PRACTICALS

1. Compiling from source: Learn about the various build systems used like the cmake / make / ant etc. instead of just running the commands.
2. Introduction to package management system: Given set of RPM or DEB, how to build and maintain, serve packages over http or ftp. And also, how do you configure client systems to access the package repository in any Open-Source Software Systems.
3. Install various software packages:
   (A)Install Linux and share files to windows

(B)Install Common Unix Printing System (CUPS)

4. Linux commands for operations such as redirection, pipes, filters, job control, changing ownership/permissions of files/links & directories.
5. Advanced LINUX Commands:curl, wget, ftp, ssh and grep.
6. Text Processing with Perl : Simple programs , connecting with database e.g., MYSQL
7. Running PHP: Simple applications like login forms after setting up a LAMP stack 8.Running Python : Set up the complete network interface using ifconfig command like setting gateway, DNS, IP tables, etc.
8. Kernel Configuration, Compilation and installation : Download / Access the latest kernel source code from kernel.org, compile the kernel and install it in the local system. Try to view the source code of the kernel.
9. Linux in the cloud, system & network management demos, configuration management, Linux on non-PC platforms, recompiling the kernel
10. . Virtualisation environment (e.g., xen, kqemu, virtualbox or lguest) to test applications, new kernels and isolate applications.

PYTHON PROGRAMMING LAB			
Course Code:	**CS582**	**Course Credits:**	**2**
Course Category:	**CC-L3**	**Course (U / P)**	**U / P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lab (Hrs/Week) / Total No. of Lab	**03 / 10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Interpret the use of procedural statements like assignments, conditional statements, loops and function calls.
2. Infer the supported data structures like lists, dictionaries and tuples in Python
3.Illustrate the application of matrices and regular expressions in building the Python programs.
4.Discover the use of external modules in creating excel files and navigating the file systems.
5. Describe the need for Object-oriented programming concepts in Python.

COURSE OUTCOME
At the end of the course the students should be able to:
1 To write, test, and debug simple Python programs.
2 To implement Python programs with conditionals and loops.
3 Use functions for structuring Python programs.
4 Represent compound data using Python lists, tuples, dictionaries
5 Read and write data from/to files in Python.

1. Write a python program find the maximum of a list of numbers.
2. Write a python program to perform Matrix Multiplication.
3. Write a python program first n prime numbers
4. Write a python program selection sort.
5. write a python program to compute the GCD of two numbers.
6. Write a python program to find the most frequent words in a text file.
7. Write a Python program to create a scientific calculator
8. Write a Python program to print all the Disarium numbers between 1 and 100.

9. Write a Python program to encrypt the text using Caesar Cipher technique. Display the encrypted text. Prompt the user for input and the shift pattern.
10. Write a Python program to construct a linked list. Prompt the user for input. Remove any duplicate numbers from the linked list.

Semester 3

ARTIFICIAL INTELLIGENCE METHODS FOR SOFTWARE ENGINEERING			
Course Code:	**CS621**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of basic software engineering and AI methods and practices and application.
2 A general knowledge of how to implement the AI in SDLC.
3 Understanding of AI and software engineering in software development.
4 Knowledge of machine learning application in software engineering
5 Understanding of software engineering for machine learning with different case studies

COURSE OUTCOME
At the end of the course the students should be able to:
1 Basic knowledge and understanding artificial intelligence in software engineering.
2 Ability to apply software engineering principles with machine learning.
3 Ability to develop, maintain and evaluate software systems with AI and Software Engineering.
4 To produce efficient, reliable, robust and cost-effective software solutions with AI.
5 Ability to perform independent research and analysis.

UNIT I INTRODUCTION OF SOFTWARE ENGINEERING AND AI

Introduction to software engineering: definitions, Principles of Software Engineering Software Development Life Cycle (SDLC), SDLC models, planning a software project, defining the problem, Software Requirement Analysis, Design, Coding, Testing and Maintenance, AI and consciousness, weak and strong AI, practical systems based on AI, development of logic, components of AI, Defining problem as state space search, analyzing and representation of the problems from AI viewpoint, Applications of AI Techniques in Software Engineering

UNIT II ARTIFICIAL INTELLIGENCE IN SOFTWARE ENGINEERING

Use of AI in Planning and Project Effort Estimation: Knowledge Based Systems, Neural Networks, Genetic Algorithms, Case Based Reasoning, Expert system development life cycle: Problem selection, Prototype construction, Formalization, Implementation, Evaluation, Knowledge acquisition: Knowledge engineer, Cognitive behavior, Acquisition techniques. Knowledge representation: Level of representation, Knowledge representation schemes, Formal logic, Inference Engine.

UNIT III SOFTWARE ENGINEERING FOR MACHINE LEARNING

Machine Learning and ML Workflow, Types of Machine Learning, Software Engineering Process with Machine Leaning, Practices with Machine Learning In Software Engineering : End-To-End Pipeline Support, Data Availability, Collection, Cleaning, and Management, Education and Training, Model Debugging and Interpretability, Model Evolution, Evaluation, and Deployment, Compliance And Varied Perceptions, Model of ML Process Maturity.

UNIT IV MACHINE LEARNING APPLICATION IN SOFTWARE ENGINEERING

ML Applications: in Prediction and Estimation, in Property and Model Discovery, in Transformation, in Generation and Synthesis, in Reuse, Requirement Acquisition, in Management of Development Knowledge

UNIT V SOFTWARE ENGINEERING FOR MACHINE LEARNING: A CASE STUDY

How several Microsoft software engineering teams work into a nine-stage workflow for integrating machine learning into application and platform development, How best practices for building applications and platforms relying on machine learning, How custom machine-learning process maturity model for assessing the progress of software teams towards excellence in building AI applications, how software engineering applies to machine-learning-centric components vs. previous application domains.

REFERENCE BOOKS:

1. Pankaj Jalote, An Integrated Approach to Software Engineering, Narosa Publishing House, New Delhi 1997.
2. Ian Sommerville, Software Engineering, Pearson Education, 2009.
3. Pressman Roger S., Software Engineering: Practitioner's Approach, McGraw-Hill Inc., 2004.
4. Software Engineering: Software Reliability, Testing and Quality Assurance, Nasib S. Gill, Khanna Book Publishing Co (P) Ltd., New Delhi, 2002.
5. Rich Elaine and Knight Kevin, Artificial Intelligence, Tata McGraw Hill.
6. Tani Moto, Introduction to AI using LISP.
7. Patterson : Artificial Intelligence and Expert Systems.
8. Balagurusamy, Artificial Intelligence & Technology.
9. Mishkoff, Henry C, Understanding Artificial Intelligence, BPB Publ.
10. Bharti & Chaitenya, Natural Language Processing, PHI

SOFTWARE ENGINEERING FOR CLOUD COMPUTING			
Course Code:	**CS623**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of basic software engineering and Cloud Computing Methods and application.
2 A General understanding of Cloud Architecture and Services.
3 Understanding of Cloud Enabling Technologies.
4Understanding of Resource Management and Security in Cloud.
5 Be able to identify different SDLC model for Cloud platform.

COURSE OUTCOMES
At the end of the course the students should be able to:
1Basic knowledge and understanding of the analysis and design of Cloud.
2Ability to apply software engineering principles and techniques.
3Ability to design, develop, maintain and evaluate Cloud Computing Services.
4To produce efficient, reliable, robust and cost-effective solutions for Cloud computing paradigm.
5Ability to perform independent research and analysis.

UNIT I INTRODUCTION

Introduction to Software Engineering: Definitions, Software Requirements, Software Testing, Software Development Life Cycle (SDLC), Verification and Validation, Agile Software Development.

Introduction to Cloud Computing – Definition of Cloud – Evolution of Cloud Computing – Underlying Principles of Parallel and Distributed Computing, Cloud Characteristics.

UNIT II CLOUD ARCHITECTURE AND SERVICES

Layered Cloud Architecture Design, NIST Cloud Computing Reference Architecture. Types of Cloud: Public, Private and Hybrid Clouds. Cloud Services: laaS, PaaS and SaaS. Architectural Design Challenges. Cloud Storage: Storage as-a-Service, advantages of Cloud Storage, Cloud Storage Providers. Service Oriented Architecture, REST and Systems of Systems, Web Services, Publish-Subscribe Model.

UNIT III CLOUD ENABLING TECHNOLOGIES

Basics of Virtualization, Types of Virtualizations, Implementation Levels of Virtualization, Virtualization Structures, Tools and Mechanisms. Virtualization of CPU, Memory and I/O Devices. Virtualization Support and Disaster Recovery**.** Cloud Technologies–Hadoop – MapReduce—Virtual Box– Federation in the Cloud – Four Levels of Federation – Federated Services and Applications.

UNIT IV RESOURCE MANAGEMENT AND SECURITY IN CLOUD

Inter Cloud Resource Management, Resource Provisioning and Resource Provisioning Methods. Global Exchange of Cloud Resources, Security Overview, Cloud Security

Challenges – Software-as-a-Service Security, Security Governance, Virtual Machine Security, IAM – Security Standards

UNIT V SOFTWARE DEVELOPMENT LIFE CYCLE MODEL FOR CLOUD PLATFORM

Cloud-Based Development using Classic Life Cycle Model, Impact of Cloud on Software Development Life Cycle.Limitations and Challenges in Cloud-Based Application Development.Impact of Cloud Paradigm on Software Engineering. Testing Perspectives for Cloud- Based Applications, Testing in the cloud: Strategies, Risk and Benefits.

REFERENCE BOOKS:

1. Burnstein, “Practical Software Testing”, Springer International Edition, ISBN 81-8128- 089-X.
2. William E. Perry, “Effective Methods for Software Testing”, John Wiley and Sons, ISBN 9971-51-345-5.
3. Judith Hurwitz, R Bloor, M. Kanfman, F. Halper “Cloud Computing for Dummies”, Wiley India Edition, First Edition.
4. Rajkumar Buyya, James Broberg, Andrzej M. Goscinski,” Cloud Computing: Principles and Paradigms”, Wiley Publication,2011.
5. Ian Sommerville, Software Engineering, Pearson Education, 2009.
6. Pressman Roger S., Software Engineering: Practitioner's Approach, McGraw-Hill Inc., 2004.

ROBOTICS LAB			
Course Code:	**CA681**	**Course Credits:**	**2**
Course Category:	**CC-L**	**Course (U / P)**	**U / P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lab (Hrs/Week) / Total No. of Lab	**03 / 10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To write programming for simple operations
2. To introduce different types of robotics and demonstrate them to identify different parts and components.
3. To introduce the concepts of robotic manufacturing system and work cells
4. Familiarize with the robot programming and control
5. To learn robot programming and simulation for industrial application.

COURSE OUTCOME
At the end of the course the students should be able to:
1. Develop the robot programming for the given application
2. perform the singularity analysis of robotics
3. Interface the vision system with robotic arm to develop the machine vision applications.
4. To learn robot programming and simulation for any industrial process
5. use of any robotic simulation software to model the different types of robots and calculate work volume for different robots

1. Design and develop the manufacturing cell using virtual robot simulator.
2. Develop a TCP and work-object for Industrial Robot using Robot simulator.
3. Develop a work-object for Industrial Robot using Robot simulator.
4. Develop robot programming for picking and place of objects.
5. Develop robot programming for material handling applications.
6. Develop robot programming for the welding process.
7. Singularity analysis using Robot simulator.
8. Interface and configure the vision system with Industrial Robot.
9. Part identification based on color & pattern and separate the components using vision system and Robot.
10. Quality control using Industrial Robot with vision system

ELECTIVES

ELECTIVES

ELECTIVES FROM DCSE

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	CS530	Software Reliability Engineering	3	0	0	3	E1
2	CS532	Parallel Computation and Applications	3	0	0	3	E1
3	CS534	Web-Based Software Engineering	3	0	0	3	E1
4	CS536	Service-Oriented Computing	3	0	0	3	E1
5	CS538	Soft Computing	3	0	0	3	E2
6	CS540	Software Maintenance	3	0	0	3	E2
7	CS542	Software Measurement and Estimation	3	0	0	3	E2
8	CS544	Software Quality Assurance	3	0	0	3	E2
9	CS625	Embedded System Design	3	0	0	3	E3
10	CS627	Applied Machine Learning	3	0	0	3	E3
11	CS629	Blockchain Technology and Software Systems	3	0	0	3	E3
12	CS631	Secure Software Engineering	3	0	0	3	E3
13	CS633	AI Enabled Cyber Security	3	0	0	3	E4
14	CS635	Big Data Platforms and Analytics	3	0	0	3	E4
15	CS637	Internet of Things	3	0	0	3	E4
16	CS639	Edge Computing	3	0	0	3	E4

OPEN AND GENERIC ELECTIVES FROM OTHER SCHOOLS

17	ES415	Energy and Environment	3	0	0	3	OE1
18	EN531	Language, Culture and Society	3	0	0	3	OE2
19	MA402	Modeling and Simulation	3	1	0	4	GE1
20	MA416	Probability and Stochastic Process	3	1	0	4	GE1

CS Computer Science for Course Code

FC Foundation Course

CC	Core Course from USICT for Type of Course	**SEC**	Skill Enhancement Course
CC-L	Core Course Lab from USICT for Type of Course	**E**	Elective
GE	General Elective from related discipline of other Deptt./School	**DP1**	Dissertation Part 1
AECC	Ability Enhancement Compulsory Course	**DP2**	Dissertation Part 2
OE	Open Elective from other discipline of other Deptt./School	**SP**	Summer Project
DSE	Discipline Specific Course		

SOFTWARE RELIABILITY ENGINEERING			
Course Code:	**CS530**	**Course Credits:**	**3**
Course Category:	**E1**	**Course (U / P)**	**U / P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of basic Software reliability.
2 A General understanding of development of reliable software.
3 Understanding of fault tolerance in hardware systems
4 Understanding of software and hardware fault tolerance.
5 Understanding of fault tolerance software.

COURSE OUTCOME
At the end of the course the students should be able to:
1 Basic to advanced knowledge and understanding of software reliability techniques.
2 Ability to apply fault tolerance techniques in hardware systems.
3 Ability to design, develop, maintain and evaluate reliable software.
4 To produce efficient, reliable, robust and cost-effective solutions for software and hardware fault tolerance problems.
5 Ability to design a good fault tolerance software.

UNIT I SOFTWARE RELIABILITY

Measures of software reliability, Mean Time To Failure (MTTF), Mean Time Between Failure (MTBF), Mean Time To Recovery (MTTR), availability, maintainability, Musa's operational profiles and type-1 uncertainty, defect removal and type-2 uncertainty, reliability stability and reliability growth, hardware reliability vs. software reliability, failure probability density function and reliability function, Reliability prediction , reliability metrics.

UNIT II DEVELOPMENT OF RELIABLE SOFTWARE

Reliable software, defect prevention, detection and removal, design for robustness, verification & validation, stabilization of requirements, design, code and test artifacts, active and passive fault detection, fault handling and correction, exceptions, survivability, reliability models, software availability model.

UNIT III FAULT TOLERANCE IN HARDWARE SYSTEMS

Fault classification, fault tolerance attributes and system structure, fault prevention, anticipated and unanticipated fault, test generation for digital systems, combinational logic

network, Boolean difference method, test generation for sequential circuits, fault simulation, application of hardware fault tolerance in developing fault tolerant software systems.

UNIT IV SOFTWARE AND HARDWARE FAULT TOLERANCE

Software and hardware faults, failure intensity function, characterization of fault injection, detection and correction, techniques for prediction of remaining faults and fault injection, classification tree analysis, code coverage, coding technique, fault tolerant & self checking, fail safe circuits, synchronous and asynchronous fail safe circuits.

UNIT V FAULT TOLERANT SOFTWARE

Concept of N-version programming (NVP) and methods, recovery block, acceptance tests, fault trees, validation of fault tolerant systems, security, fault tolerance in wireless/mobile networks and Internet.

Reference Books:

1. Software Reliability Engineering, John D. Musa, Tata McGRAW Hill, 2005.
2. Fault-Tolerant Computer System Design, D.K. Pradhan, 2003.
3. Design and Analysis of Fault-Tolerant Digital Systems, B. W. Johnson, Addison-Wesley, 1989.
4. Fault-Tolerant Computing, Theory & Techniques, D.K. Pradhan, Prentice Hall, 1986.
5. Reliable Computer Systems: Design and Evaluation, D. P. Siewiorek and R. S. Swartz, Digital Press, 1992.
6. Probability and Statistics with Reliability, Queueing and Computer Science application, K.S.Trivedi, Prentice Hall, 1982.
7. Fault Tolerant Principles and Practice, Anderson and Lee, PHI, 1989.

PARALLEL COMPUTING AND APPLICATION			
Course Code:	**CS532**	**Course Credits:**	**3**
Course Category:	**E1**	**Course (U / P)**	**U / P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**
COURSE OBJECTIVES			
1 Knowledge of basic parallel computing.			
2 A General understanding of development of reliable software.			
3 Understanding of fault tolerance in hardware systems			
4 Understanding of software and hardware fault tolerance.			
5 Understanding of fault tolerance software.			
COURSE OUTCOME			
At the end of the course the students should be able to:			
1 Basic to advanced knowledge and understanding of parallel computing techniques.			
2 Ability to apply fault tolerance techniques in hardware systems.			
3 Ability to design, develop, maintain and evaluate reliable software.			

4 To produce efficient, reliable, robust and cost-effective solutions for software and hardware fault tolerance problems.
5 Ability to design a good fault tolerance software.

UNIT-1 INTRODUCTION

parallel computing,Shared memory and distributed memory parallelism, Amdahl's law, speedup and efficiency, supercomputers.Principles of parallel algorithm design: decomposition techniques, mapping & scheduling computation, template

UNIT-2 MESSAGE PASSING

MPI basics, point-to-point communication, collective communication, synchronous/asynchronous send/recv, algorithms for gather, scatter, broadcast, reduce.

Network topologies, network evaluation metrics, communication cost, routing in interconnection networks, static and adaptive routing, process-to-processor mapping.

UNIT-3 PERFORMANCE

Scalability, benchmarking, performance modeling, impact of network topologies, parallel code analysis and profiling.Domain decomposition, communication-to-computation ratio, load balancing, adaptivity, case studies: weather and material simulation codes.

UNIT-4

MPI I/O algorithms, contemporary large-scale I/O architecture, I/O bottlenecks. Job scheduling, RDMA, one-sided communication, NVM, extreme scale computing: issues and trends.

UNIT-5

Analytical modeling of program performance -speedup, efficiency, scalability, cost optimality efficiency Collective communication

Synchronization, Non-numerical algorithms-sorting,graphs.Numerical algorithms-dense matrix algorithms,sparse matrix algorithms. Performance measurement and analysis of parallel programs ,GPU Programming

TEXT BOOKS

1.Network and Parallel Computing
2.15th IFIP WG 10.3 International Conference, NPC 2018, Muroran, Japan, November 29 – December 1, 2018, Proceedings

3.INTRODUCTION TO PARALLEL COMPUTING RAJU K. NIRANJAN N. CHIPLUNKAR
4.Load Balancing in Parallel Models using Optimization & Virtualization
Optimization of single programme multiple data virtualization in parallel environment
Rajesh Tiwari, Manisha Sharma, Kamal K. Mehta

WEB BASED SOFTWARE ENGINEERING			
Course Code:	**CS 534**	**Course Credits:**	**3**
Course Category:	**E1**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 To learn the concept of how to learn patterns and concept of software engineering
2 Recognize the need for, and have the ability to engage in independent and life-long learning.
3 Ability to analyze the requirements, design, develop software programs. Also evaluate and recognize potential risks and provide creative solutions
4 Understand and commit to professional ethics and responsibilities and norms of engineering practice.
5 To explore Perceptrons working for web based software engineering

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Able to understand how to extract patterns from the data.
2 Students will be able to decompose the given project in various phases of a lifecycle.
3 Students will be able to choose an appropriate process model depending on the user requirements.
4Students will be able to know various processes used in all the phases of the product..
5 Students will be able to choose an appropriate process model depending on the user requirements.

UNIT I
Introduction, analysis, architectural design, design patterns, formulation, interface design, navigation, design, project, management, quality, attributes, structures, testing, WebApp, attributes, WebApp categories, WebE process, WebE team.

UNIT II
Attributes of web based applications, quality attributes, technologies, web application quality, the web process, framework for the web, web process model

UNIT III
Formulating and analyzing web based systems, design for web based applications, testing

web based applications, management issues, project management.

UNIT IV
Reviews general architectures for web application and technology-aware application designs, discusses the concepts and techniques for engineering and evaluating user interfaces appropriate for a web application's intended audience.

UNIT V
Explores the interaction between users and the application's user interface, special attention will be paid to web technologies and standards available for audiences with special needs.

Related research papers reading as suggested by the subject Teacher and their analysis.

TEXT BOOKS:
1. Software engineering A practitioner's Approach, Roger S Pressman, sixth edition McGraw Hill International Edition.
2. Software Engineering, Ian Sommerville, seventh edition, Pearson education.

REFERENCE BOOKS:
1. Software Engineering, A Precise Approach, Pankaj Jalote, Wiley India, 2010.
2. Software Engineering: A Primer, Waman S Jawadekar, Tata McGraw-Hill, 2008
3. Fundamentals of Software Engineering, Rajib Mall, PHI, 2005
4. Software Engineering, Principles and Practices, Deepak Jain, Oxford University Press.
5. Software Engineering1: Abstraction and modeling, Diner Bjorner, Springer International edition, 2006.
6. Software Engineering2: Specification of systems and languages, Diner Bjorner, Springer International edition 2006.
7. Software Engineering Foundations, Yingxu Wang, Auerbach Publications, 2008.
8. Software Engineering Principles and Practice, Hans Van Vliet, 3rd edition, John Wiley &Sons Ltd.
9. Software Engineering 3: Domains, Requirements, and Software Design, D. Bjorner, Springer International Edition.
10. Introduction to Software Engineering, R. J. Leach, CRC Press.

SERVICE -ORIENTED COMPUTING			
Course Code:	**CS536**	**Course Credits:**	**3**
Course Category:	**E1**	**Course (U / P)**	**U**
Course Year (U / P):	**3U**	**Course Semester (U / P):**	**2U**
No. of Lectures + Tutorials (Hrs/Week):	**03 +0+0**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**30**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To gain understanding of the web services architectures and motivation for composition.
2. To learn service basic concept of SOAP, WSDL and UDDI
3. To learn technology underlying the service design
4. To learn advanced concepts such as service composition, orchestration and Choreography.
5. To learn about collaboration, Agents, Multi agents system, Agent communication.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. After the completion of this course students will be able to Understand primary concepts of SOA .
2. Know the integration of SOA technological points with Web Services.
3. Implement SOA in the development cycle of Web Services.
4. Study and implement the development cycle of Web Services.
5. To learn agents,multi agents systems.

UNIT I Introduction to distributed Computing

Brief history of information technology, Challenges for composition, Web Services Architectures and Standards. Computing with services, Visions for web, Semantic web, Peer to Peer Computing, Processes and Protocols. Pragmatic web, Open environments.Directory services, SOAP, REST WSDL, UDDI

UNIT II Enterprise architectures and Service Oriented Computing

Integration versus interoperation, J2EE, .NET, Model Driven Architecture, Legacy systems. Use cases: Intra-enterprise and Inter-enterprise Interoperation, Application, Configuration, Dynamic Selection, Software Fault Tolerance, Grid, and, Utility Computing, Elements of

ServiceOriented Architectures, RPC versus Document, Orientation, Composing Services. .

UNIT III Description: Modeling and representation

XML primer, Conceptual modeling, Ontology and knowledge sharing, Relevant standards: RDF, RDFS, and OWL, Differencing and tools, Matchmaking. Engagement: Execution Models: Messaging, CORBA, Peer to peer computing, Jini, Grid Computing, Transactions: ACID Properties, Schedules, Locking, Distributed Transactions, Transactions over Composed Services: Architecture, Properties, Compositional Serializability, Process specification: Processes, Workflows.

UNIT IV Business Process Management:

Introduction of Business process Management, Process Specification Language, Relevant standards: BPEL4WS, WSCI, WS-C, ebXML, Relaxed transactions, Exception handling.

UNIT V Collaboration

Describing collaborations, Agents, Multiagent systems, Agent communication, languages, Protocols, Commitments and contracts, Planning, Consistency maintenance, Relevant standards: FIPA, OWL-S, Economic models, Organizational models

Reference Books:

1. Sandeep Chatterjee, James Webber, “Developing Enterprise Web Services, An Architect’s Guide”, Pearson Education.
2. Newcomer, Lomow, “Understanding SOA with Web Services”, Pearson Education.
3. Thomas Erl, “Service-Oriented Architecture: Concepts, Technology, and Design”,Pearson Education.
4. Dan Woods and Thomas Mattern, “Enterprise SOA Designing IT for Business Innovation” O’REILLY

SOFT COMPUTING TECHNIQUES			
Course Code:	**CS538**	**Course Credits:**	**3**
Course Category:	**E2**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of basic to advanced soft computing techniques.
2 A General understanding of fuzzy sets and fuzzy logic.
3 Understanding of fuzzy relations and fuzzy systems.
4 Understanding of neural networks
5 Understanding of evolutionary algorithms.

COURSE OUTCOME
At the end of the course the students should be able to:
1 Basic to advanced knowledge and understanding of soft computing.
2 Ability to apply principles and techniques of fuzzy sets and fuzzy logic.
3 Ability to design, develop, maintain and evaluate neural networks.
4 To produce efficient, reliable, robust and cost-effective solutions for soft computing techniques.
5 Ability to design a good evolutionary algorithm.

UNIT I INTRODUCTION

Introduction to Soft Computing; Definition, requirement, necessity and adequacy; various dialects of soft computing – Evolutionary Algorithms, Fuzzy Sets and Fuzzy Logic, Artificial Neural Networks - their suitability in Searching, optimization, decision matching and pattern related problems; potential areas of applications.

UNIT II FUZZY SETS AND FUZZY LOGIC

Introduction to fuzzy sets and fuzzy logic; difference between classical and fuzzy sets; chance vs fuzziness; limitations of fuzzy systems; typical shapes of membership functions

and their usage; operations on fuzzy sets: complement, intersection, union; combinations on operations, aggregation operation.

UNIT III FUZZY RELATIONS AND FUZZY SYSTEMS

Cartesian Product; Classical Relations and Fuzzy Relations; Cardinality, operations and properties of crisp and fuzzy relations; Composition of operations, Fuzzy cartesian product; The linguistic variables, Reasoning in fuzzy logic, Fuzzification and defuzzification; Mamdani and Sugeno Fuzzy Inference Systems.

UNIT IV NEURAL NETWORK

Overview of biological neurons; McCulloch-Pitts model, Rosenblatt's Perceptron model, difference, capabilities and limitations; Model of generic computational neuron; Basic activation functions; Basic Learning laws of neurons; Single layer and multilayer architectures; Feedforward and feedback networks.

UNIT V LEARNING FUNDAMENTALS

Learning paradigms, supervised and unsupervised learning, reinforced learning; back propagation algorithm; Radial basis neurons, Generalized Regression Neural network, Probabilistic Neural Networks; Competitive learning; Self Organizing Features Map, Hopfield networks, associative memories, applications of artificial neural networks. Elasticity vs plasticity dilemma, preprocessing, post processing, early stopping.

UNIT VI EVOLUTIONARY ALGORITHMS

Problems suitable and not suitable for applying evolutionary algorithms; Various dialects of evolutionary Algorithms; Terminology of Genetic Algorithms; Canonical Genetic Algorithm; Common representations and related reproduction operators; premature convergence, schema theorem, minimal deceptive problem and Royal Road function; fitness function, Roulette wheel selection, Rank selection, Tournament Selection; termination criteria, survivor selection, population models; parallel implementations.

Text Books:

1. Artificial Neural Networks: An introduction to ANN Theory and Practice, Peteus J. Braspenning, PHI publication, 2005.
2. Fuzzy Logic: A spectrum of Theoretical and Practical issues, Paul P. Wang, pearson publication 2004.
3. An Introduction to Genetic Algorithms, Milanie Mitchell, MIT Press 1998.
4. A Genetic Algorithm Tutorial, Darrell Whitley.

Reference Books:

5. Fuzzy Sets, Fuzzy logic, and Fuzzy Systems: Selected Papers- Lotfi Asker Zadeh, George J. Kilr, Bo yuan, 2005.
6. Foundations of Fuzzy logic and Soft Computing: 12th International Fuzzy conference proceeding, 2005.
7. Neural Networks Theory, Particia Melin, Oxford University press, 2003
8. Neural Networks Theory and Application, Oscar Castillo, Wiley Eastern publication
9. Genetic Algorithms in Search, Optimization and Machine Learning, David E

Goldberg, Eddison-Wesley, 1988.

SOFTWARE MAINTENANCE			
Course Code:	**CS540**	**Course Credits:**	**3**
Course Category:	**E2**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
Following objectives are:
1.Have a solid understanding of fundamental concepts of software maintenance & evolution
2.Understand some of the state-of-the-art techniques used in maintaining and evolving legacy systems
3. Learn reverse engineering and reengineering for program comprehension techniques
4.Understand software reuse
5. Learn legacy system management
COURSE OUTCOME
At the end of the course the students should be able to:
1. be in the capacity of systematically approaching changes in large code bases
2. understand theories, models, tools and processes related to the maintenance and evolution of large software systems.
3. be able to apply state-of-the-art techniques when maintaining and/or evolving large software systems.
4.be familiar with a few tools that support software maintenance
5.be familiar with a few tools that support software quality

UNIT I INTRODUCTION

Introduction to Software maintenance: Definitions, types of maintenance,cost of maintenance,real world affecting maintenance cost, software end factor affecting

maintenance,maintenance activities, Need for maintenance ,causes of software maintenance problem:lack of traceability,lack of code comment,obsolete legacy system.

UNIT II SOFTWARE MAINTENANCE PROCESS

Program understanding, generating a particular maintenance problem, ripple effect,modified program testing, maintainability, software maintenance cost factor, technical factor ,non technical factor: application domain, staff stability, program lifetime, dependence on external environment ,Hardware stability

UNIT III TYPES OF SOFTWARE MAINTENANCE

Types of software maintenance : Corrective maintenance , adaptive maintenanace, perfective maintenance, preventive maintenance

UNIT IV SOFTWARE REENGINEERING

Software reengineering definition, reengineering process, reverse engineering,program reconstructing,forward engineering, component reusability, reuse process , case tools

UNIT V SOFTWARE METRICS

software metrics, classification of software metrics, types of metrics,advantage of software metrics,disadvantage of software metrics, size oriented metrics, loc metrics, functional point analysis, differentiate between functional point analysis and loc, extended functional point analysis

TEXT BOOK:

1. Software Requirements and Estimation by Rajesh Naik and Swapna Kishore, Tata Mc Graw Hill.

REFERENCES:

1. Software Requirements by Karl E. Weigers, Microsoft Press.
2. Managing Software Requirements, Dean Leffingwell & Don Widrig, Pearson Education, 2003.
3. Mastering the requirements process, second edition, Suzanne Robertson & James Robertson, Pearson Education, 2006.
4. Estimating Software Costs, Second edition, Capers Jones, TMH, 2007.
5. Practical Software Estimation, M.A. Parthasarathy, Pearson Education, 2007. 6. Measuring the software process, William A. Florac & Anita D. Carleton,Pearson Education, 1999.

.

SOFTWARE MEASUREMENT AND ESTIMATION			
Course Code:	**CS542**	**Course Credits:**	**3**
Course Category:	**E2**	**Course (U / P)**	**U**
Course Year (U / P):	**3U**	**Course Semester (U / P):**	**2U**
No. of Lectures + Tutorials (Hrs/Week):	**03 +0+0**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**30**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1.Students will demonstrate knowledge of the distinction between critical and non- critical systems.
2. Students will demonstrate the ability to manage a project including planning, scheduling and risk assessment/management.
3. Students will demonstrate an understanding of the proper contents of a software requirements document.
4. Students will demonstrate proficiency in rapid software development techniques.
5. Students will author a software testing plan.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. To develop an understanding of software requirements and assess their nature.
2. To analyze software requirement management
3.To be able to draw conclusions on effort, schedule and cost estimation
4. Survey tools for requirements management, software estimation tools.
5. To be able to estimate the cost of software development by understanding various methods

UNIT I Software Requirements:

What and Why Essential Software requirement, Good practices for requirements engineering, Improving requirements processes, Software requirements and risk management Software

Requirements Engineering Requirements elicitation, requirements analysis documentation, review, elicitation techniques, analysis models, Software quality attributes, risk reduction through prototyping, setting requirements priorities, verifying requirements quality.

UNIT II Software Requirements Management

Requirements management Principles and practices, Requirements attributes, Change Management Process, Requirements Traceability Matrix, Links in requirements chain Software Requirements Modeling Use Case Modeling, Analysis Models, Dataflow diagram, state transition diagram, class diagrams, Object analysis, Problem Frames

UNIT III Software Estimation

Components of Software Estimations, Estimation methods, Problems associated with estimation, Key project factors that influence estimation Size Estimation Two views of sizing, Function Point Analysis, Mark II FPA, Full Function Points, LOC Estimation, Conversion between size measures.

UNIT IV Effort, Schedule and Cost Estimation

What is Productivity? Estimation Factors, Approaches to Effort and Schedule Estimation, COCOMO II, Putnam Estimation Model, Algorithmic models, Cost Estimation

UNIT V Tools for Requirements Management and Estimation Requirements Management Tools:

Benefits of using a requirements management tool, commercial requirements management tool, Rational Requisite pro, Caliber – RM, implementing requirements management automation, Software Estimation Tools: Desirable features in software estimation tools, IFPUG, USC's COCOMO II, SLIM (Software Life Cycle Management) Tools

TEXT BOOK:

1. Software Requirements and Estimation by Rajesh Naik and Swapna Kishore, Tata Mc Graw Hill.

REFERENCES:

1. Software Requirements by Karl E. Weigers, Microsoft Press.

2. Managing Software Requirements, Dean Leffingwell & Don Widrig, Pearson Education, 2003.
3. Mastering the requirements process, second edition, Suzanne Robertson & James Robertson, Pearson Education, 2006.
4. Estimating Software Costs, Second edition, Capers Jones, TMH, 2007.
5. Practical Software Estimation, M.A. Parthasarathy, Pearson Education, 2007. 6. Measuring the software process, William A. Florac & Anita D. Carleton,Pearson Education, 1999.

SOFTWARE QUALITY ASSURANCE AND ENGINEERING			
Course Code:	**CS625**	**Course Credits:**	**3**
Course Category:	**E2**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 consists of a set of reporting and auditing functions.
2.It ensures management of data which is important for product quality.
3 It also ensures that software which is developed, does it meet and complies with standard quality assurance.
4 It ensures that the end result or product meets and satisfies user and business requirements.
5.It simply finds or identifies defects or bugs, and reduces the effect of these defects.

COURSE OUTCOMES
At the end of the course the students should be able to understand:
1what SQA is and why SQA is not SQC(testing).
2 Reasons for SQA failures and factors critical to success of SQA in IS development.
3 Proactive methods for more effectively reviewing requirements and designs.
4 To produce efficient, reliable, robust and cost-effective software solutions.
5 How to measure system quality and SQA/Testing .

UNIT I SOFTWARE QUALITY AND ENGINEERING

Quality concepts and productivity relationship, software quality factors, software quality costs, Total Quality Management (TQM), continuous improvement cycle: Plan, Do, Check and Act (PDCA), quality policy, cost of quality, quality engineering, quality planning: goal

setting and strategy formation, assessment and improvement.

UNIT II SOFTWARE QUALITY ASSURANCE (SQA)

Components of SQA, classification, defect detection, defect prevention, defect reduction, defect containment, QA activities in software processes, verification and validation, software review, inspection, formal verification, statistical software quality approach.

UNIT III COMPONENTS MEASUREMENT WITH REFERENCE TO SQA

Metrics, product quality metrics, process quality metrics, metrics for software maintenance, quality tools for quality control, test management and organizational structures, Capability Maturity Model(CMM), Capability Maturity Model Integration (CMMI), ISO 9000, quality and quality management metrics, Deming's Principle, SQA team formation

UNIT IV QUALITY MANAGEMENT MODEL

Integrating quality activities in project life cycle, reviews, software testing, strategies and implementation, Computer-Aided Software Engineering (CASE) tools, The Rayleigh model framework, code integration pattern, Problem Tracking Report (PTR), reliability growth model,Service Quality, Kano Model, Customer retention, continuous process improvement, Juran's Trilogy,TQM principles, Kaizen Technique, Statistical Quality Assurance, Mc call quality factors

UNIT V SOFTWARE QUALITY ASSURANCE BEYOND TESTING

Defect prevention and process improvement, root cause analysis for defect prevention, software inspection, inspection related activities, fault tolerance and failure containment, comparing quality assurance techniques and activities.

REFERENCE BOOKS:

1. Metrics and Models in Software Quality Engineering, Stephan H. Kan, Pearson Education, 2007.
2. An Integrated Approach to Software Engineering, Pankej Jalote, Narosa Publishing House, New Delhi 1997.
3. Making Sense of Software Quality Assurance, Raghav J. Nandyal, Tata McGRAW Hill,
   2007.
4. Software Quality Assurance: A Practitioner Approach, Kaman Malik, Praveen Chaudhary, Tata McGRAW Hill, 2008.

EMBEDDED SYSTEM DESIGN			
Course Code:	**CS625**	**Course Credits:**	**3**
Course Category:	**E3**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Building Blocks of Embedded System
2. Various Embedded Development Strategies
3. Bus Communication in processors, Input/output interfacing.
4. Various processor scheduling algorithms.
5. Basics of Real time operating system and example tutorials to discuss on one real time operating system too.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. understand and analyze Embedded systems.
2. study about bus Communication in processors.
3. operate various Embedded Development Strategies
4. acquire knowledge on various processor scheduling algorithm
5. suggest an embedded system for a given application

UNIT I INTRODUCTION TO EMBEDDED SYSTEMS

Introduction to Embedded Systems –Structural units in Embedded processor, selection of processor & memory devices- DMA – Memory management methods- Timer and Counting devices, Watchdog Timer, Real Time Clock, In circuit emulator, Target Hardware Debugging.

UNIT II EMBEDDED NETWORKING

Embedded Networking: Introduction, I/O Device Ports & Buses– Serial Bus communication protocols RS232 standard – RS422 – RS 485 - CAN Bus -Serial
Peripheral Interface (SPI) – Inter Integrated Circuits (I2C) –need for device drivers.

UNIT III EMBEDDED FIRMWARE DEVELOPMENT ENVIRONMENT

Embedded Product Development Life Cycle- objectives, different phases of EDLC,Modeling of EDLC; issues in Hardware-software Co-design, Data Flow Graph, state machine model, STUDENTSFOCUS.COM Sequential Program Model, concurrent Model, object-oriented Model.

UNIT IV RTOS BASED EMBEDDED SYSTEM DESIGN

Introduction to basic concepts of RTOS- Task, process & threads, interrupt routines in RTOS, Multiprocessing and Multitasking, Pre-emptive and non-pre-emptive scheduling,Task communication shared memory, message passing-, Inter process Communication synchronization between processes-semaphores, Mailbox, pipes, priority inversion,
priority inheritance.

UNIT V EMBEDDED SYSTEM APPLICATION AND DEVELOPMENT

Case Study of Washing Machine- Automotive Application- Smart card System Application-ATM machine –Digital camera

TEXTBOOKS:

1. Peckol, “Embedded system Design”, John Wiley & Sons,2010
2. Lyla B Das,” Embedded Systems-An Integrated Approach”, Pearson, 2013
3. Shibu. K.V, “Introduction to Embedded Systems”, 2e, Mc graw Hill, 2017.

REFERENCES

1. Raj Kamal, ‘Embedded System-Architecture, Programming, Design’, Mc Graw Hill, 2013.
2. C.R.Sarma, “Embedded Systems Engineering”, University Press (India) Pvt. Ltd, 2013.
3. Tammy Noergaard, “Embedded Systems Architecture”, Elsevier, 2006.
4. Han-Way Huang, “Embedded system Design Using C8051”, Cengage Learning, 2009.
5. Rajib Mall “Real-Time systems Theory and Practice” Pearson Education, 2007.

APPLIED MACHINE LEARNING			
Course Code:	**CS 635**	**Course Credits:**	**3**
Course Category:	**E3**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**
COURSE OBJECTIVES			
1 To learn the concept of how to learn patterns and concepts from data without being explicitly programmed.			
2 To design and analyze various machine learning algorithms and techniques with a modern outlook focusing on recent advances.			
3 Explore supervised and unsupervised learning paradigms of machine learning.			
4 To explore Deep learning techniques and various feature extraction strategies.			
5 To explore Perceptrons working for deep learning			
COURSE OUTCOMES			
At the end of the course the students should be able to:			
1 Able to understand how to extract patterns from the data.			

2 To compare and contrast pros and cons of various machine learning techniques and to get an insight of when to apply a particular machine learning approach.
3 To mathematically analyze various machine learning approaches and paradigms.
4 Able to utilize Machine Learning in practical applications.
5 Able to apply classification tasks using perceptrons.

UNIT I Introduction
What is Machine Learning, Types of Machine Learning, Supervised Learning, Unsupervised Learning, Reinforcement Learning, Applications of Machine Learning – Stock Price Prediction, Face Recognition, Handwriting Recognition, Image Recognition, Virtual Personal Assistants, Medical Diagnosis, Online Fraud Detection.

UNIT II Supervised Learning
Basic methods: Distance-based methods, Nearest-Neighbours, Decision Trees, Naive Bayes
Linear models: Linear Regression, Logistic Regression, Generalized Linear Models, Support Vector Machines, Nonlinearity and Kernel Methods.

UNIT III Unsupervised Learning
Clustering: K-means/Kernel K-means, Self-Organizing Maps.
Dimensionality Reduction: PCA and kernel PCA Matrix Factorization and Matrix Completion Generative Models (mixture models and latent factor models).

UNIT IV Artificial Neural Network
Biological Neurons and Biological Neural Networks, Artificial Neural Network, Types of Neural Network, Perceptron, History behind Perceptron, Importance of Perceptron, Working of Perceptron, Perceptron Learning, Perceptron Learning Rule, Perceptron Learning of AND & OR gate, XOR gate, Activation functions, Binary Activation function, ReLU, Sigmoid, Hyperbolic, Softmax Activation function, Multilayer Perceptrons, Back propagation Neural Networks, and Feed-Forward Neural Networks, Applications and Future of Neural Networks.

UNIT V Selected Topics
Ensemble Methods (Boosting, Bagging, Random Forests), Sparse Modeling and Estimation, Modeling Sequence/Time-Series Data, Deep Learning and Feature Representation Learning, Recent trends in various learning techniques of machine learning and classification methods, Case studies in interdisciplinary domains.

Text Books:
1. Kevin Murphy, Machine Learning: A Probabilistic Perspective, MIT Press, 2012.
2. Trevor Hastie, Robert Tibshirani, Jerome Friedman, The Elements of Statistical Learning, Springer 2009.
3. Christopher Bishop, Pattern Recognition and Machine Learning, Springer, 2007.

References Books:
4. Artificial Intelligence: A Modern Approach, Stuart J. Russell and Peter Norvig
5. Machine Learning, Tom M. Mitchell

Blockchain Technology and Software Systems			
Course Code:	**CS629**	**Course Credits:**	**3**
Course Category:	**E3**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To understand the technology behind blockchain
2. Explain distributed Consensus, and Consensus in Bitcoin
3. Discuss Permissioned Blockchain, and Hyperledger Fabric
4. To comprehend the issues related to blockchain
5. To study the real-world applications of blockchain

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Describe the basic concept of Blockchain, Crypto Primitives, Bitcoin Basics
2. Identify the area in which they can apply permission or permission less blockchain.
3. Apply Block chaining concept in various applications.
4. Design and implement new ways of using blockchain for applications other than cryptocurrency
5. Recognize the underlying technology of transactions, blocks, proof-of-work, and consensus building

UNIT I Introduction to Blockchain

What is Blockchain, Public Ledgers, Blockchain as Public Ledgers, Bitcoin, Blockchain 2.0, Smart Contracts, Block in a Blockchain, Transactions, Distributed Consensus, The Chain and the Longest Chain, Cryptocurrency to Blockchain 2.0, Permissioned Model of Blockchain

UNIT II Basic Crypto Primitives

Cryptographic Hash Function, Properties of a hash function, Hash pointer and Merkle tree, Digital Signature, Public Key Cryptography, A basic cryptocurrency. Bitcoin Basics: Creation of coins, Payments and double spending, FORTH – the precursor for Bitcoin scripting, Bitcoin Scripts, Bitcoin P2P Network, Transaction in Bitcoin Network, Block Mining, Block propagation and block relay.

UNIT III Distributed Consensus

Why Consensus, Distributed consensus in open environments, Consensus in a Bitcoin

network. Consensus in Bitcoin: Bitcoin Consensus, Proof of Work (PoW) – basic introduction, Hashcash PoW, Bitcoin PoW, Attacks on PoW and the monopoly problem, Proof of Stake, Proof of Burn and Proof of Elapsed Time. The life of a Bitcoin Miner, Mining Difficulty, Mining Pool.

UNIT IV Permissioned Blockchain

Permissioned model and use cases, Design issues for Permissioned blockchains, Execute contracts, State machine replication, Consensus models for permissioned blockchain, Distributed consensus in closed environment, Paxos, RAFT Consensus, Byzantine general problem. Blockchain Components and Concepts: Actors in a Blockchain, Components in Blockchain design, Ledger in Blockchain.

UNIT V Hyperledger Fabric Transaction Flow

Fabric Architecture, Transaction flow in Fabric. Hyperledger Fabric Details:Ordering Services, Channels in Fabric, Fabric Peer and Certificate Authority. Fabric – Membership and Identity Management: Organization and Consortium Network, Membership Service Provide, Transaction Signing.

Text Books

1. Nitin Gaur, Luc Desrosiers, Venkatraman Ramakrishna, Petr Novotny, Salman Baset, Anthony O'Dowd.Hands-On Blockchain with Hyperledger: Building decentralized applications with Hyperledger Fabric and Composer. Packt Publishing Ltd.
2. Bellaj Badr, Richard Horrocks, Xun (Brian) Wu. Blockchain By Example: A developer's guide to creating decentralized applications using Bitcoin, Ethereum, and Hyperledger. Packt Publishing Ltd, 2018.

Reference Books

1. Vikram Dhillon, David Metcalf, Max Hooper. Blockchain Enabled Applications: Understand the Blockchain Ecosystem and How to Make it Work for You. Apress.
2. Mayukh Mukhopadhyay Ethereum Smart Contract Development: Build blockchain-based decentralized applications using solidity. Packt Publishing Ltd.

SECURE SOFTWARE ENGINEERING			
Course Code:	**CS631**	**Course Credits:**	**3**
Course Category:	**E3**	**Course (U / P)**	**U**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**4P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Knowledge of basic SW engineering methods and practices and application.
2 A general understanding of software process models.
3 Understanding of software requirements and the SRS documents.
4 Understanding of software design process.
5 Understanding of software coding, testing and maintenance.

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Basic knowledge and understanding of the analysis and design of complex systems.
2 Ability to apply software engineering principles and techniques.
3 Ability to design, develop, maintain and evaluate large-scale software systems.
4 To produce efficient, reliable, robust and cost-effective software solutions.
5 Ability to perform independent research and analysis.

UNIT I WHY IS SECURITY A SOFTWARE ISSUE?

Introduction, The Problem, Software Assurance and Software Security, Threats to Software Security, Sources of Software Insecurity, The Benefits of Detecting Software Security Defects Early, Managing Secure Software Development What Makes Software Secure? Introduction, Defining Properties of Secure Software, How to Influence the Security Properties of Software, How to Assert and Specify Desired Security Properties

UNIT II SOFTWARE LIFE CYCLE MODELS

Software Development Life Cycle (SDLC), SDLC models, waterfall model and its variations, prototype model, iterative enhancement model, spiral model, RAD model, comparison of these models, software development teams, software development environments, validation and traceability, maintenance, prototyping requirements, Software project management.

UNIT III SECURE SOFTWARE ARCHITECTURE AND DESIGN

Introduction, Software Security Practices for Architecture and Design: Architectural Risk Analysis, Software Security Knowledge for Architecture and Design: Security Principles, Security Guidelines, and Attack Patterns Considerations for Secure Coding and Testing:

Introduction, Code Analysis, Coding Practices, Software Security Testing, Security Testing Considerations throughout the SDLC

UNIT IV SECURITY AND COMPLEXITY

System Assembly Challenges: Introduction, Security Failures, Functional and Attacker Perspectives for Security Analysis: Two Examples, System Complexity Drivers and Security, Deep Technical Problem Complexity

UNIT V SOFTWARE CODING, TESTING AND MAINTENANCE

Coding, Testing Methods: unit testing, integration testing, system testing, acceptance testing, testing techniques: white box testing, black box testing, thread testing, regression testing, alpha testing, beta testing, static testing, dynamic testing, Evolution of software products, economics of maintenance, category of software maintenance, Role of product development life cycle, deployment model, adaptive maintenance, corrective maintenance, perfective maintenance, enhancement request, proactive defect prevention, problem reporting, problem resolution, software maintenance from customers' perspective, maintenance standard: IEEE-1219, ISO-12207.

REFERENCE BOOKS:

1. Pankaj Jalote, An Integrated Approach to Software Engineering, Narosa Publishing House, New Delhi 1997.
2. Ian Sommerville, Software Engineering, Pearson Education, 2009.
3. Pressman Roger S., Software Engineering: Practitioner's Approach, McGraw-Hill Inc., 2004.
4. Software Engineering: Software Reliability, Testing and Quality Assurance, Nasib S. Gill, Khanna Book Publishing Co (P) Ltd., New Delhi, 2002.
5. Howard , M and Lipner,S : The Security Development Lifecycle , Microsoft Press, 2006
6. Swiderski, F and Snyder W. :, Threat Modeling, Microsoft Press, 2004.
7. Viega, J and MCGraw G., : Building Secure Software: How to avoid Security Problems in the Right Way, Addison-Wesley,2001
8. The Open Web Application Security Project: A Guide to Building Secure Web Applications and Web Services", 2.0 Black Hat Edition, 2005

AI ENABLED CYBER SECURITY			
Course Code:	**CS633**	**Course Credits:**	**3**
Course Category:	**E4**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To Analyze and improving cybersecurity posture
2. To understand AI & CS systems are being trained to identify malware, execute pattern recognition
3. To study better Vulnerability Management
4. To understand the concepts of AI by Adversaries
5. AI enabled cyber security.

COURSE OUTCOMES
At the end of the course the students should be able to:
1.Insight into the main methods used in artificial intelligence (AI) and Cyber security
2.Students are able to gain knowledge AI and CS important for cybersecurity
3.Students will be able to use CS/ML Security in today's lives.
4.Students will be able to explain CS/ML Used in Security.
5. knowledge of the historical development of the field of Cyber Security enabled AI.

UNIT I Fundamentals of Cyber Security & AI

Identity, authentication, confidentiality, privacy, anonymity, availability and integrity,deep learning fundamentals from a security perspective.fundamentals of AI and how AI can solve problems in the cyber security space,Examples of companies using AI for Security, such as Cylance and FireEye.

UNIT II Secure Web

Security using AI techniques for injection using regular expressions,identifying patterns and matching with existing scores,Vulnerability measures,statistical patterns and Bayesian statistics,statistical distributions

UNIT III Cyber Security Threat

Web Application Security**,**Injection,Broken authentication,Sensitive data exposure,XML External Entities (XXE),Broken access control,Security misconfigurationCross-Site Scripting (XSS),Insecure deserialization,Using components with known vulnerabilitiesInsufficient logging and monitoring.

UNIT IV Securing Infrastructure

Security issues in systems,secure software design, secure programming, and security testing, covering security analysis as well as the secure development of software-based systems both on architectural level and system level

UNIT V Impact of AI on Cyber Security

Threat hunting in memory, file system and network data,cyber threat hunting and digital investigation, introductory analysis of malicious programs.,KNN (K - Nearest Neighbours) for threat visualisers,Isolation forest for anomaly detection,LSTM for multi-vector correlation DBSCAN for riskware detection and fraudLSTM (Autoencoder) for endpoint protection

Text Books:

1. Artificial Intelligence for Cybersecurity: Implement smart AI systems for preventing cyber attacks and detecting threats and network anomalies
Alessandro Parisi , 1st edition (2 August 2019)

2. AI-Enabled Threat Detection and Security Analysis,by Hadis Karimipour (Editor), Farnaz Derakhshan 3 August 2021

Reference Books:

1.Cyber Security Incident Response Guide. – Alan J White,

2. Cybersecurity – Attack and Defense Strategies: Counter Modern Threats and Employ State-of-the Art Tools and Techniques to Protect Your Organization Against Cybercriminals. – Yuri Diogenes, Erdal Ozkaya,August 2019

3. Cybersecurity: An Essential Guide to Computer and Cyber Security for Beginners, Including Ethical Hacking, Risk Assessment, Social Engineering, Attack and Defense Strategies, and Cyberwarfare. – Latest Edition

BIG DATA PLATFORMS AND ANALYTICS				
Course Code:	**CS635**	**Course Credits:**		**3**
Course Category:	**E4**	**Course (U / P):**		**P**
Course Year (U / P):	**2P**	**Course Semester :**		**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**		**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**		**3**

COURSE OBJECTIVES
1. To prepare the Big Data for analysis.
2. To extract meaningful data from unstructured Big Data.
3. To develop Data Visualizations skills.
4. To apply various tools for analysis of structured data.
5. To apply various tools for analysis of unstructured Big Data.
COURSE OUTCOMES
At the end of the course the students should be able to:
1. Able to analyze the identification of Big Data problem
2. Able to extract the structured data from unstructured data.
3. Use Hadoop related tools such as JAQL.
4. Use Spark, Pig and Hive for structured Data analytics
5. Use Spark, Pig and Hive for unstructured Big Data analytics.

UNIT I Big Data Introduction

What is big data, why big data, convergence of key trends, unstructured data, industry examples of big data, web analytics, big data and marketing, fraud and big data, risk and big data, big data and healthcare, big data in medicine, advertising and big data, big data technologies, open source technologies, cloud and big data, mobile business intelligence, Crowd sourcing analytics, inter and trans firewall analytics. Data Gathering and Preparation: Data formats, parsing and transformation, Scalability and real-time issues.

UNIT II Data Cleaning

Consistency checking, Heterogeneous and missing data, Data Transformation and segmentation. Visualization: Descriptive and comparative statistics, Designing visualizations, Time series, Geo-located data, Correlations and connections, Hierarchies and networks, interactivity.

UNIT III Big Data Technology
Big Data Architecture, Big Data Warehouse, Functional Vs. Procedural Programming Models for Big Data NoSQL: Introduction to NoSQL, aggregate data models, key-value and document data models.

UNIT IV Big Data Tools
Introduction to Hadoop Ecosystem, Hadoop, Requirement of Hadoop Framework, Design principle of Hadoop, Comparison with other system, Hadoop Components, Hadoop 1 vs Hadoop 2, Hadoop Daemon's, HDFS Commands, Map Reduce Programming: I/O formats, Map side join, Reduce Side Join, Secondary sorting, Pipelining Map Reduce jobs

UNIT V Advance Big Data Tools
Spark, PIG, JAQL, Understanding Text Analytics and Big Data, Predictive Analysis of Big Data, Role of Data Analyst.

Text and References Books:
1. EMC Education Services, Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data, John WIley & Sons.
2. Anil Maheshwari, Data Analytics Make Accesible, Orilley Publications.
3. Croll and B. Yoskovitz Lean Analytics: Use Data to Build a Better Startup Faster, Oreilley Publications.**3.** Croll and B. Yoskovitz Lean Analytics: Use Data to Build a Better Startup Faster, Oreilley Publications.

INTERNET OF THINGS			
Course Code:	**CS637**	**Course Credits:**	**3**
Course Category:	**E4**	**Course (U / P)**	**U**
Course Year (U / P):	**4U**	**Course Semester (U / P):**	**7U**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Students will be explored to the interconnection and integration of the physical world in IoT.
2. Learning of networking concepts in the IoT environment.
3. Understanding of various wireless networks, topologies, and IoT protocols.
4. Understanding of the importance of security issues in IoT.
5. Implementation of IoT in real life with learning of tools like MATLAB.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Understand all concepts of the Internet of Things.
2. Understand the building blocks of the Internet of Things and its characteristics.
3. Learn application protocols for IoT.
4. Able to understand the application areas of IoT.
5. Able to realize the revolution of the Internet in Mobile Devices, Cloud & Sensor Networks.

UNIT I INTRODUCTION TO IOT

Genesis of IoT, IoT and Digitization, IoT Impact, Convergence of IT and OT, IoT Challenges, Drivers Behind New Network Architectures: Scale, Security, Constrained Devices and Networks, Data, Legacy Device Support.

UNIT II IOT NETWORK ARCHITECTURE AND DESIGN

Comparing IoT Architectures: The one M2M IoT Standardized Architecture, The IoT World Forum (IoTWF) Standardized Architecture, Additional IoT Reference Models, A Simplified IoT Architecture, The Core IoT Functional Stack- Layer 1: Things: Sensors and Actuators Layer, Layer 2: Communications Network Layer, Layer 3: Applications and Analytics Layer,

IoT Data Management and Compute Stack :Fog Computing , Edge Computing, The Hierarchy of Edge, Fog, and Cloud.

UNIT III NETWORK AND APPLICATION PROTOCOLS FOR IOT

Wireless Communication Technologies: ZigBee, ESP8266, Introduction to sensors and modules - concept, layout, working, applications, Introduction of IoT Development Boards-Node MCU, Arduino, IoT Access Technologies 107IEEE 802.15.4, IEEE 802.15.4g and 802.15.4e, IEEE 1901.2a, IEEE 802.11ah, LoRaWAN, Constrained Devices, Constrained-Node Networks, Optimizing IP for IoT :From 6LoWPAN to 6Lo, Header Compression, Fragmentation, Mesh Addressing, Mesh-Under Versus Mesh-Over Routing, Authentication and Encryption on Constrained Nodes , Application Protocols for IoT: CoAP, Message Queuing Telemetry Transport (MQTT) .

UNIT IV DATA ANALYTICS AND SECURITY OF IOT

An Introduction to Data Analytics for IoT, Structured Versus Unstructured Data, Data in Motion Versus Data at Rest, IoT Data Analytics Overview, IoT Data Analytics Challenges, Machine Learning : Machine Learning Overview Supervised Learning, Unsupervised Learning, Neural Networks, Securing IoT : Common Challenges in IoT Security, Device Insecurity, Network Characteristics Impacting Security, Security Priorities: Integrity, Availability, and Confidentiality, Formal Risk Analysis Structures: IAS OCTAVE, Top Vulnerabilities of Iot.

UNIT V. IMPLEMENTING IoT IN REAL LIFE

Interfacing sensors with development boards, communication modules with sensors, communication modules with development boards, MATLAB and Arduino Interfacing, Hands-on in IoT - various real life projects involving different boards, sensors, modules and communication technologies .

Text Books :

1. IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things
   by Rob Barton, Gonzalo Salgueiro, David Hanes
2. Vijay Madisetti and Arshdeep Bahga, “Internet of Things (A Hands-on-Approach)”, 1stEdition,
   VPT, 2014.
3. Francis daCosta, “Rethinking the Internet of Things: A Scalable Approach to Connecting Everything”, 1st Edition, Apress Publications, 2013

EDGE COMPUTING			
Course Code:	**CS639**	**Course Credits:**	**3**
Course Category:	**E4**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1.This course will explore research, frameworks, and applications in Edge Computing
2.The class will begin with a review of current big data analytics frameworks for Cloud Computing.
3.We will then explore frameworks for computing over edge devices and cloud.
4.We will study algorithms for distributed data analytics over edge devices.
5.Understanding of this content through class presentations and tutorials.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Explore the need for new computing paradigms..
2. Explain major components of fog and edge computing architectures.
3. Identify potential technical challenges of the transition process and suggest solutions.
4. Analyze data and application requirements and pertaining issues.
5. Design and model infrastructures

UNIT-1 Introduction to Edge Computing

New computing paradigm, Middleware Infrastructures, Edge cloud architectures , Lightweight Edge Clouds Fog Computing: A Platform for Internet of Things and Analytics , The Emergence of Edge Computing

UNIT-2 Big Data Analytics in the Cloud

The Google File System,The Hadoop Distributed File System, MapReduce: Simplified Data Processing on Large Clusters, Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing, Discretized Streams: Fault-Tolerant Streaming Computation at

Scale , Storm @Twitter, Apache Flink: Stream and Batch Processing in a Single Engine ,Kafka: a Distributed Messaging System for Log Processing, Cassandra: a decentralized structured storage system

UNIT-3 Geo-Distributed Computing

Low Latency Geo-distributed Data Analytics,Trading Timeliness and Accuracy in Geo-Distributed Streaming Analytics, Nebula: Distributed Edge Cloud for Data Intensive Computing

UNIT-4 Edge Architecture

CloudPath: A Multi-Tier Cloud Computing Framework, Cloud4Home -- Enhancing Data Services with @Home Clouds, Femto Clouds: Leveraging Mobile Devices to Provide Cloud Service at the Edge, Fast, Scalable and Secure Onloading of Edge Functions Using AirBox

UNIT-5 Sensor Networks

Algorithmic Models for Sensor Networks ,GPSR: greedy perimeter stateless routing for wireless networks, TAG: a Tiny AGgregation service for ad-hoc sensor networks, Approximate aggregation techniques for sensor databases, Gossip-Based Computation of Aggregate Information, Optimal aggregation algorithms for middleware, Efficient top-K query calculation in distributed networks

TEXT BOOK

1. Buyya R., Srirama S.N., “Fog and Edge Computing”, Wiley, 2019.
2. Instructor’s notes.

REFERENCE BOOKS

1. Taheri J. & Deng S. (eds.): “Edge Computing: Models, technologies and applications”, IET, 2020
2. Sabella D., Reznik A., Frazao R., “Multi-access Edge Computing in Action”, 1st edition, Kindle, 2019
3. Al-Turjman F. (ed.): “Edge Computing: from hype to reality”, Springer, 2019.

WEB-BASED SOFTWARE ENGINEERING			
Course Code:	**CS534**	**Course Credits:**	**3**
Course Category:	**E4**	**Course (U / P)**	**P**
Course Year (U / P):	**3P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 +0+0**	**Mid Sem. Exam Hours:**	**1**
Total No. of Lectures (L + T):	**30+0+0**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Understanding of learning the process of technology planning.
2. Basic to advanced knowledge and understanding of software engineering methods.
3. To produce efficient, reliable, robust and cost-effective solutions for component-based software
4. Understanding of aspect-oriented software engineering.
5. A General understanding of object-oriented software engineering

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Understand principles of Web page design and about types of websites.
2. Explain web development Strategies and Protocols governing Web.
3. Helps in computational thinking.
4. Understanding of paradigms of web page coding.
5. Understanding of networking fundamentals.

UNIT I
Introduction, analysis, architectural design, design patterns, formulation, interface design, navigation, design, project, management, quality, attributes, structures, testing, WebApp, attributes, WebApp categories, WebE process, WebE team.

UNIT II
Attributes of web based applications, quality attributes, technologies, web application quality, the web process, framework for the web, web process model

UNIT III
Formulating and analyzing web based systems, design for web based applications, testing web based applications, management issues, project management.

UNIT IV
Reviews general architectures for web application and technology-aware application designs, discusses the concepts and techniques for engineering and evaluating user interfaces appropriate for a web application's intended audience.

UNIT V

Explores the interaction between users and the application's user interface, special attention will be paid to web technologies and standards available for audiences with special needs.

Related research papers reading as suggested by the subject Teacher and their analysis.

Reference Books:

1. Web Engineering: A Practitioner's Approach by Pressman and Lowewhich considers the Web engineering process in its entirety.
2. Web Engineering: Principles and Techniques [Paperback] by Woojong Suh
3. Roger Pressman.S., " Software Engineering : A Practitioner's Approach ", (3rd Edition), McGraw Hill,