SCHEME OF INSTRUCTION
BE (COMPUTER SCIENCE AND ENGINEERING)
AICTE MODEL CURRICULUM
CSE - SEMESTER - V (Proposed for the academic year 2020-21)

S. No.	Course Code	Course Title	Scheme of Instruction				Scheme of Examination			Credits
			L	T	D/P	Contact Hrs/W	CIE	SEE	Duration in Hrs	
Theory Course										
1.	PC 501 CS Core-7	Software Engineering	3	1	-	4	30	70	3	3
2.	PC 502 CS Core-8	Operating Systems	3	1	-	4	30	70	3	3
3.	PC 503 CS Core-9	Automata Languages & Computation	3	1	-	4	30	70	3	3
4.	PE-I	Professional Elective-I	3	-	-	3	30	70	3	3
5.	PE-II	Professional Elective-II	3	-	-	3	30	70	3	3
6.	PE-III	Professional Elective-III	3	-	-	3	30	70	3	3
Practical/Laboratory Course										
7.	PC531 CS	Software Engineering Lab	-	-	2	2	25	50	3	1.5
8.	PC532 CS	Operating Systems Lab	-	-	2	2	25	50	3	1.5
9.	PW533 CS	Mini Project	-	-	2	2	25	50	3	1
Total			**18**	**03**	**06**	**27**	**255**	**570**		**22**

Profession Elective – I	
Course Code	**Course Title**
PE 511 CS	Artificial Intelligence
PE 512 CS	Advanced Computer Architecture
PE 513 CS	Image Processing

Profession Elective – II	
Course Code	**Course Title**
PE 527 CS	Web and Internet Technologies
PE 528 CS	Embedded Systems
PE 529 CS	Graph Theory
PE 530 CS	Data Analytics

Profession Elective – III	
Course Code	**Course Title**
PE 523 CS	Block Chain Technologies
PE 524 CS	Information Retrieval Systems
PE 525 CS	Soft Computing
PE 526 CS	Computer Graphics

Course Code	Course Title						Core/ Elective
PC 501 CS	**SOFTWARE ENGINEERING**						**CORE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	3	1	-	-	30	70	3

Course Objectives

- To introduce the basic concepts of software development processes from defining a product to shipping and maintaining
- To impart knowledge on various phases, methodologies and practices of software development
- To understand the importance of testing in software development, study various testing strategies along with its relationship with software quality and metrics

Course Outcomes

Student will be able to

- Acquired working knowledge of alternative approaches and techniques for each phase of software development
- Judge an appropriate process model(s) assessing software project attributes and analyze necessary requirements for project development eventually composing SRS
- Creation of visual models to describe (non-) algorithmic solutions for projects using various design principles.
- Acquire skills necessary as an independent or as part of a team for architecting a complete software project by identifying solutions for recurring problems exerting knowledge on patterns.
- Concede product quality through testing techniques employing appropriate metrics by understanding the practical challenges associated with the development of a significant software system.

UNIT-I

Introduction to Software Engineering:A generic view of Process: Software Engineering, Process Framework, CMM Process Patterns, Process Assessment.

Process Models: Prescriptive Models, Waterfall Model, Incremental Process Models, Evolutionary Process Models, Specialized Process Models, The Unified Models, Personal and Team Process Models, Process Technology, Product and Process.

An Agile view of Process: Introduction to Agility and Agile Process, Agile Process Models.

UNIT-II

Software Engineering Principles: SE Principles, Communication Principles, Planning Principles, Modeling Principles, Construction Principles, Deployment.

System Engineering: Computer-based Systems, The System Engineering Hierarchy, Business Process Engineering, Product Engineering, System Modeling.

Requirements Engineering: A Bridge to Design and Construction, Requirements Engineering Tasks, Initiating Requirements Engineering Process, Eliciting Requirements,

Developing Use-Cases, Building the Analysis Model, Negotiating Requirements, Validating Requirements.

UNIT-III

Building the Analysis Model: Requirements Analysis Modeling Approaches, Data Modeling Concepts, Object-Oriented Analysis, Scenario-based Modeling, Flow-oriented Modeling, Class-based Modeling, Creating a Behavioral Model.

Design Engineering: Design within the context of SE, Design Process and Design Quality, Design Concepts, The Design Model, Pattern-based Software Design.

UNIT-IV

Creating an Architectural Design: Software Architecture, Data Design, Architectural Styles and Patterns, Architectural Design.

Modeling Component-Level Design: Definition of Component, Designing Class-based Components, Conducting Component-level Design, Object Constraint Language, Designing Conventional Components.

Performing User Interface Design: The Golden Rules, User Interface Analysis and Design, Interface Analysis, Interface Design Steps, Design Evaluation.

UNIT-V

Testing: Strategies: A Strategic Approach to Conventional Software Testing, Test Strategies for O-O Software. **Tactics:** Software Testing Fundamentals, Black-box and White-box Testing, Basis Path Testing, Control Structure Testing, O-O Testing Methods.

Debugging: Debugging Techniques, The Art of Debugging.

Product Metrics: A Framework for Product Metrics, Metrics for each phase of software development.

Software Quality: Definition, **Quality Assurance:** Basic Elements, Formal Approaches, Statistical Software Quality Assurance, Software Reliability, ISO9000 Quality Standards, SQA Plan.

Suggested Books:

1. Roger S. Pressman, *Software Engineering: A Practitioner's Approach* , 7th Edition, McGraw Hill, 2009
2. Ali Behforooz and Frederick J. Hudson, *Software Engineering Fundamentals*, Oxford University Press, 1996
3. PankajJalote, *An Integrated Approach to Software Engineering*, 3rd Edition, Narosa Publishing House, 2008

Course Code	Course Title						Core/ Elective
PC 502 CS	**OPERATING SYSTEMS**						**CORE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	3	1	-	-	30	70	3

Course Objectives

- To learn the fundamentals of Operating Systems.
- To learn the mechanisms of OS to handle processes and threads and their communication
- To learn the mechanisms involved in memory management in contemporary OS
- To gain knowledge on distributed operating system concepts that includes architecture, Mutual exclusion algorithms, deadlock detection
- To know the components and management aspects of concurrency management

Course Outcomes

- Identify System calls and evaluate process scheduling criteria of OS.
- Develop procedures for process synchronization of an OS.
- Demonstrate the concepts of memory management and of disk management
- Solve issues related to file system interface and implementation, I/O systems
- Describe System model for deadlock, Methods for handling deadlocks.

UNIT-I

Introduction: Concept of Operating Systems, Generations of Operating systems, Types of Operating Systems, OS Services, System Calls, Structure of an OS - Layered, Monolithic, Microkernel Operating Systems, Concept of Virtual Machine.

UNIT-II

Processes: Definition, Process Relationship, Different states of a Process, Process State transitions, Process Control Block (PCB), Context switching

Thread: Definition, Various states, Benefits of threads, Types of threads, Concept of multithreads,

Process Scheduling: Foundation and Scheduling objectives, Types of Schedulers, Scheduling Criteria, Scheduling algorithms, multiprocessor scheduling

UNIT-III

Process Synchronization: Inter-process Communication: Critical Section, Race Conditions, Mutual Exclusion, Peterson's Solution, classical problems of synchronization: The Bounded

buffer problem, Producer\Consumer Problem, reader's & writer problem, Dinning philosopher's problem. Semaphores, Event Counters, Monitors, Message Passing,

Deadlocks: Definition, Necessary and sufficient conditions for Deadlock, Methods for Handling: Deadlocks: Deadlock prevention, Deadlock Avoidance: Banker's algorithm, Deadlock detection and Recovery.

UNIT-IV

Memory Management: Basic concept, Logical and Physical address map, Memory allocation: Contiguous Memory allocation, fragmentation and Compaction; Paging: Principle of operation – Page allocation – Hardware support for paging, structure of page table, Protection and sharing, Disadvantages of paging.

Virtual Memory: Basics of Virtual Memory – Hardware and control structures – Locality of reference, Page fault, Working Set, Dirty page/Dirty bit – Demand paging, Page Replacement algorithms, Trashing

UNIT-V

I/O Hardware: I/O devices, Device controllers, Direct memory access Principles of I/O Software: Goals of Interrupt handlers, Device drivers, Device independent I/O software,

File Management: Concept of File, Access methods, File types, File operation, Directory structure, File System structure, Allocation methods, Free-space management, directory implementation, efficiency and performance.

Secondary-Storage Structure: Disk structure, Disk scheduling algorithms, Disk Management, RAID structure

Suggested books:

1. AviSilberschatz, Peter Galvin, Greg Gagne, *Operating System Concepts Essentials*, 9th Edition, Wiley Asia Student Edition, 2017.
2. William Stallings, *Operating Systems: Internals and Design Principles*, 5th Edition, Prentice Hall of India, 2016.
3. Maurice Bach, *Design of the Unix Operating Systems*, 8th Edition,Prentice-Hall of India, 2009.
4. Daniel P. Bovet, Marco Cesati, *Understanding the Linux Kernel*, 3rd Edition, , O'Reilly and Associates.

Course Code	Course Title						Core/ Elective
PC 503 CS	**AUTOMATA LANGUAGES & COMPUTATION**						**CORE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	3	1	-	-	30	70	3

Course Objectives

- Develop a formal notation for strings, languages and machines.
- Design finite automata to accept a set of strings of a language.
- Prove that a given language is regular and apply the closure properties of languages.
- Design context free grammars to generate strings from a context free language and Convert them into normal forms.
- Prove equivalence of languages accepted by Push down Automata and languages generated by context free grammars
- Identify the hierarchy of formal languages, grammars and machines.
- Distinguish between computability and non-computability and Decidability and undecidability.

Course Outcomes

- Write a formal notation for strings, languages and machines.
- Design finite automata to accept a set of strings of a language.
- For a given language determine whether the given language is regular or not.
- Design context free grammars to generate strings of context free languages.
- Determine equivalence of languages accepted by Pushdown Automata and languages generated by context free grammars
- Write the hierarchy of formal languages, grammars and machines.
- Distinguish between computability and non-computability and Decidability and undecidability.

UNIT-I

Introduction: Finite state automata, Non-deterministic finite state automata, FA with € - transitions, Regular expressions, Applications of FA, Properties of regular sets, Pumping Lemma, Closure properties, Myhill-Nerode Theorem, Minimization of FA.

UNIT-II

Context Free Grammars and Languages: Derivations, Parse-trees, Ambiguity in Grammars and Languages. Pushdown Automata–Definitions, The languages of PDA, Equivalence of PDAs and CFGs, Deterministic Pushdown Automata**.**

UNIT-III

Properties of CFLs: Normal forms for CFGs, Pumping Lemma, Closure properties,

Deterministic Context Free Languages, Decision properties.

UNIT-IV

Turing Machines: Introduction, Computational Languages and Functions, Techniques for construction of Turing machines. Modifications of TM, TM as enumerator, Restricted TM.

UNIT-V

Undecidability: Recursive and Recursively enumerable languages, UTM and undecidable problem, Rice Theorem, Post's correspondence problem. Chomsky's Hierarchy– Regular grammars, Unrestricted grammar, CSL, Relationship between classes of languages.

Suggested Books:

1. John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computation,* 3rd Edition, Pearson Education Asia,2007
2. John Martin, *Introduction to Languages and The Theory of Computation*, 3rd Edition,Tata McGraw Hill,2013.

Course Code	Course Title						Core/ Elective
PE 511 CS	**ARTIFICIAL INTELLIGENCE**						**ELECTIVE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	3	-	-	-	30	70	3

Course Objectives

- Understand the importance of the field of AI by discussing its history and various applications.
- Learn about one of the basic applications of A.I, search state formulations.
- Learn methods of expressing knowledge by a machine with appropriate reasoning and different mathematics involved behind it
- Learn how to reason when an agent has only uncertain information about its task.
- Know various supervised and unsupervised learning algorithms

Course Outcomes

Upon completion of the course, the students will be able to:

- Formalize a problem in the language/framework of different AI methods
- Illustrate basic principles of AI in solutions that require problem solving, search, inference
- Represent natural language/English using Predicate Logic to build knowledge through various representation mechanisms
- Demonstrate understanding of steps involved in building of intelligent agents, expert systems, Bayesian networks
- Differentiate between learning paradigms to be applied for an application

UNIT-I

Problem Solving & Search: Introduction- What is intelligence? Foundations of artificial intelligence (AI). History of AI, Structure of Agents;

Problem Solving - Formulating problems, problem types, states and operators, state space;

Search Strategies. - Informed Search Strategies- Best first search, A* algorithm, heuristic functions, Iterative deepening A*;

Adversarial Search/ Game playing - Perfect decision game, imperfect decision game, evaluation function, alpha-beta pruning;

UNIT-II

Knowledge, Reasoning & Planning : Reasoning - Knowledge based agent, Propositional Logic, Inference, Predicate logic (first order logic), Resolution

Structured Knowledge Representation – Frames, Semantic Nets

Planning - A Simple Planning Agent, Form Problem Solving to Planning, Basic representation of plans, partial order planning, hierarchical planning

UNIT-III

Expert Systems, Reasoning with Uncertainty: Expert System and Applications: Introduction, Phases in Building Expert Systems, Expert System Architecture, Applications;

Uncertainty - Basic probability, Bayes rule, Belief networks, Inference in BayesianNetworks**,** Fuzzy sets and fuzzy logic: Fuzzy logic system architecture, membership function;

Decision Making- Utility theory, utility functions;

UNIT-IV

Learning: Machine-Learning Paradigms: Introduction, Machine Learning Systems, Supervised and Unsupervised Learning, Inductive Learning, Learning Decision Trees

Artificial Neural Networks: Introduction, Artificial Neural Networks, Single-Layer Feed-Forward Networks, Multi-Layer Feed-Forward Networks

Reinforcement learning – Learning from rewards, Passive and Active reinforcement learning, Applications

UNIT-V

Communicating & Perceiving: Introduction to NLP- Progress & applications of NLP, Components of NLP, Grammars, Parsing

Automatic Speech Recognition (ASR) – Speech Processing, Ex: DRAGON, HARPY, **Machine Vision –** Applications, Basic Principles of Vision, Machine vision techniques: Low, Middle and High level vision

AI Today & Tomorrow - Achievements, ubiquitous AI

Suggested Readings:

1. Stuart Russell and Peter Norvig. *Artificial Intelligence – A Modern Approach*, 3rd Edition, Pearson Education Press, 2009.
2. Kevin Knight, Elaine Rich, B. Nair, *Artificial Intelligence*, 3rd Edition, McGraw Hill, 2008.
3. Nils J. Nilsson, *The Quest for Artificial Intelligence*, Cambridge University Press, 2009

Course Code	Course Title						Core/ Elective
PE 521 CS	**WEB & INTERNET TECHNOLOGY**						**ELECTIVE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
C, C++, Java, DC	3	-	-	-	30	70	-

Course Objectives

- Learn various client side technologies for developing web based applications.
- Learn the concepts of JavaScript and Angular JS for adding rich GUI.
- To Know about XML applications with DTD and Schema.
- To familiarize the concepts about Servlets and JSPs in dynamic web applications.
- To learn how to establish database connectivity in web applications.

Course Outcomes

- Understand the concepts of HTML and CSS.
- Acquire the knowledge to build AJAX based applications using Javascript.
- Understand and apply the concepts of servlet framework
- Implement JSP to build interactive web applications
- Acquire the knowledge of database connectivity in web applications

UNIT-I

A Brief Introduction to Internet, The World Wide Web, Web Browsers, Web Servers, Uniform Resource Locators, MIME, HTTP

HTML5: Evolution of HTML and XHTML, Basic Syntax, Document Structure, Links, Images, Multimedia, Lists, Tables, Creating Forms. Cascading Style sheets.

UNIT-II

JavaScript: Overview, Object Orientation and JavaScript, Syntactic Characteristics, Primitives, Operators, Expressions, Input and Output, Control Statements, Objects Creation and modification, Arrays, Functions, Constructors, Pattern Matching. Manipulating DOM, HTML DOM Events, Basics of AJAX with example.

UNIT-III

XML: Introduction to XML, Syntax, XML document structure, Document Type Definition, Name spaces, XML Schemas, Display in raw XML documents, Displaying XML documents with CSS, XPath Basics, XSLT, XML Processors.

J2EE: Exploring Enterprise architecture styles, Features of EE platform, Web servers and application servers.

Database programming with JDBC: JDBC Drivers, Exploring JDBC Processes with the java.sql Package.

UNIT-IV

Servlets Technology: Exploring the Features of Java Servlet, Exploring the Servlet API, Explaining the Servlet Life Cycle, Creating a Sample Servlet, Working with ServletConfig and ServletContext Objects, Implementing Servlet Collaboration, Exploring the Session Tracking Mechanisms.

UNIT-V

JSP Technology: Advantages of JSP over Java Servlet, Architecture of a JSP Page, Life Cycle of a JSP Page, Working with JSP Basic Tags and Implicit Objects, Working with Action Tags in JSP, Exploring EL, Exploring the Elements of Tag Extensions, Tag Extension API, Working with Simple Tag Handlers, Accessing Database from Servlet and JSP.

Suggested Readings :

1. Robert W.Sebesta: *Programming the World Wide Web*, 4th Edition, Pearson Education, 2009
2. Java Server Programming Java EE7 (J2EE 1.7): Black Book, (2014), Dreamtech Press
3. Porter Scobey, PawanLingras: *Web Programming and Internet Technologies an E-Commerce Approach*, 2nd Edition, Jones & Bartlett Learning, 2009.
4. Bryan Basham, Kathy Sierra, Bert Bates: Head first Servlets & JSP, 2nd edition, OREILLY, 2008.

Course Code	Course Title						Core/ Elective
PE 524 CS	**DATA ANALYTICS**						**ELECTIVE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
Any One Programming Language, Data Base & Basic Statistics	3	-	-	-	30	70	3

Course Objectives

- Overview of Data and Data analytics on huge datasets.
- Prepare Qualitative Data to perform different strategies of analytics
- Explore Data Analysis using R Software
- Able to realistically assess the application of data analytics technologies for different usage scenarios

Course Outcomes

Upon completion of the course, the students will be able to:

- Demonstrate proficiency with statistical analysis of data.
- Develop the ability to build and assess data-based models.
- Execute statistical analyses with professional statistical software.
- Demonstrate skill in data management.
- Apply data science concepts and methods to solve problems in real-world contexts and will communicate these solutions effectively

UNIT - I

Getting to Know Your Data - Data Objects and Attribute Types - Attribute, Nominal Attributes, Binary Attributes, Ordinal Attributes, Numeric Attributes, Discrete versus Continuous Attributes**. Basic Measuring Data Similarity and Dissimilarity -** Data Matrix versus Dissimilarity Matrix, Proximity Measures for Nominal Attributes, Proximity Measures for Binary Attributes, Dissimilarity of Numeric Data: Minkowski Distance, Proximity Measures for Ordinal Attributes, Dissimilarity for Attributes of Mixed Types, Cosine Similarity**.**

UNIT - II

Introduction to Data Analytics - Big Data and Data Science, Small Data, A Short Taxonomy of Data Analytics, Examples of Data Use, Breast Cancer in Wisconsin, Polish Company Insolvency Data, A Little History on Methodologies for Data Analytics.

Descriptive Statistics - Scale Types, Descriptive Univariate Analysis, Univariate Frequencies, Contents, Univariate Data Visualization, Univariate Statistics, Common Univariate Probability Distributions, Descriptive Bivariate Analysis, Two Quantitative Attributes, Two Qualitative Attributes, at Least one of them Nominal, Two Ordinal Attributes.

UNIT - III

Descriptive Multivariate Analysis - Multivariate Frequencies, Multivariate Data Visualization, Multivariate Statistics, Location Multivariate Statistics, Dispersion Multivariate Statistics.

Data Quality and Preprocessing - Data Quality, Missing Values, Redundant Data, Inconsistent Data, Noisy Data, Outliers, Converting to a Different Scale Type, Converting Nominal to Relative, Converting Ordinal to Relative or Absolute, Converting Relative or Absolute to Ordinal or Nominal, Converting to a Different Scale.

UNIT - IV

Data Analytics Lifecycle Overview - Key Roles for a Successful Analytics Project, Background and Overview of Data Analytics Lifecycle - Discovery, Data Preparation, Model Planning, Model Building, Communicate Results, Operationalize and Case Study.
Data Analytics Methods using R - Introduction to R, R Graphical User Interfaces, Data Import and Export, Attribute and Data Types, Descriptive Statistics, Exploratory Data Analysis, Visualization Before Analysis, Dirty Data Visualizing a Single Variable Examining Multiple Variables, Data Exploration Versus Presentation.

UNIT - V

Data Visualization Basics - Key Points Supported with Data, Evolution of a Graph, Common Representation Methods, How to Clean Up a Graphic, Additional Considerations.

Applications of Data Analytics on Text & Web: Working with Texts, Data Acquisition, Feature Extraction, Tokenization, Stemming, Conversion to Structured Data, Trends, Sentiment Analysis, Web Mining, & Recommender Systems.

Suggested Text Books:

1. Data Mining: Concepts and Techniques Second Edition – Jiawei Han and Micheline Kamber – Morgan KaufMan Publisher, 2011
2. A General Introduction to Data Analytics, Joao Mendes Moreira,Andre C.P.L.F.de Carvalho, Tomas Horvath, Wiley Publications., 2018.
3. David Dietrich, Barry Hiller, "Data Science & Big Data Analytics", EMC education services, Wiley publications, 2012.

Course Code	Course Title						Core/ Elective
PE 531 CS	**BLOCK CHAIN TECHNOLOGY**						**ELECTIVE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	3	-	-	-	30	70	3

Course Objectives

- Understand how block chain systems (mainly Bitcoin and Ethereum) work,
- To securely interact with them,
- Design, build, and deploy smart contracts and distributed applications,
- Integrate ideas from block chain technology into their own projects.

Course Outcomes: Upon completion of the course, the students will be able to:

- Explain design principles of Bitcoin and Ethereum.
- Explain Nakamoto consensus.
- Explain the Simplified Payment Verification protocol.
- List and describe differences between proof-of-work and proof-of-stakeconsensus.
- Interact with a block chain system by sending and reading transactions.
- Design, build, and deploy a distributed application.
- Evaluate security, privacy, and efficiency of a given block chain system.

UNIT - I

Basics: Distributed Database, Two General Problem, Byzantine General problem and Fault Tolerance, Hadoop Distributed File System, Distributed Hash Table, ASIC resistance, Turing Complete.

Cryptography: Hash function, Digital Signature - ECDSA, Memory Hard Algorithm, Zero Knowledge Proof.

UNIT - II

Blockchain: Introduction, Advantage over conventional distributed database, Block chain Network, Mining Mechanism, Distributed Consensus, Merkle Patricia Tree, Gas Limit, Transactions and Fee, Anonymity, Reward, Chain Policy, Life of Block chain application, Soft & Hard Fork, Private and Public block chain.

UNIT - III

Distributed Consensus: Nakamoto consensus, Proof of Work, Proof of Stake, Proof of Burn, Difficulty Level, Sybil Attack, Energy utilization and alternate.

Unit - IV

Cryptocurrency: History, Distributed Ledger, Bitcoin protocols - Mining strategy and rewards, Ethereum -

Construction, DAO, Smart Contract, GHOST, Vulnerability, Attacks, Sidechain, Namecoin

Unit – V

Cryptocurrency Regulation: Stakeholders, Roots of Bit coin, Legal Aspects-Crypto currency Exchange, Black Market and Global Economy.

Applications: Internet of Things, Medical Record Management System, Domain Name Service and future of Block chain.

Case study : *Naive Blockchain construction, Memory Hard algorithm - Hashcash implementation, Direct Acyclic Graph, Play with Go-ethereum, Smart Contract Construction, Toy application using Blockchain, Mining puzzles*

Suggested Readings:

1. Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller and Steven Goldfeder, Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction, Princeton University Press, 2016.
2. Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System
3. DR. Gavin Wood, "ETHEREUM: A Secure Decentralized Transaction Ledger," Yellow paper.2014.
4. Nicola Atzei, Massimo Bartoletti, and TizianaCimoli, A survey of attacks on Ethereum smart contracts

Course Code	Course Title						Core/ Elective
PE 532 CS	**INFORMATION RETRIEVAL SYSTEMS**						**ELECTIVE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	3	-	-	-	30	70	-

Course Objectives

- To understand indexing and querying in information retrieval systems
- To learn the different models for information retrieval
- To expose the students to text classification and clustering
- To learn about web searching

Course Outcomes

- Understand the algorithms and techniques for information retrieval (document indexing and retrieval, query processing)
- Quantitatively evaluate information retrieval systems
- Classify and cluster documents
- Understand the practical aspects of information retrieval such as those in web search engines.

UNIT-I

Introduction to Information Retrieval Systems: Definition of Information Retrieval System, Objectives of Information Retrieval Systems, Functional Overview, Relationship to Database Management Systems, Digital Libraries and Data Warehouses.

Boolean Retrieval: An example information, Building an inverted index, processing Boolean queries, the extended Boolean model versus ranked retrieval.
The term vocabulary and postings lists: Document delineation and character sequence decoding, determining the vocabulary of terms, Faster postings list intersection via skip pointers, Positional postings, and Phrase queries.
Dictionaries and tolerant retrieval: Search structures for dictionaries, Wildcard queries, spelling correction.

UNIT-II

Index construction: Hardware basics, blocked sort-based indexing, Single-pass in-memory indexing, Distributed indexing, Dynamic indexing, Other types of indexes.
Index compression: Statistical properties of terms in information retrieval, Dictionary compression, Postings filecompression.
Cataloging and Indexing: History and Objectives of Indexing, Indexing Process, Automatic Indexing, Information Extraction.

Scoring, term weighting and the vector space model: Parametric and zone indexes, Term frequency and weighting, the vector space model for scoring, and Variant tf-idf functions.

UNIT-III

Evaluation in information retrieval: Information retrieval system evaluation, Standard test collections, Evaluation of unranked retrieval sets, Evaluation of ranked retrieval results, Assessing relevance.

Relevance feedback and query expansion: Relevance feedback and pseudo relevance feedback, Global methods for query reformulation.

Probabilistic information retrieval: Basic probability theory, The Probability Ranking Principle, The Binary Independence Model.

Language models for information retrieval: Language models, The query likelihood model.

UNIT-IV

Text classification and Naive Bayes: The text classification problem, Naive Bayes text classification, The Bernoulli model, Properties of Naive Bayes, and Feature selection.

Vector space classification: Document representations and measures of relatedness in vector spaces, Rocchio classification, k- nearest neighbour, Linear versus nonlinear classifiers.

Flat clustering: Clustering in information retrieval, Problem statement, Evaluation of clustering, k-means. **Hierarchical clustering:** Hierarchical agglomerative clustering, Single-link and complete-link clustering, Group-average agglomerative clustering, Centroid clustering, Divisive clustering.

UNIT-V

Matrix decompositions and Latent semantic indexing: Linear algebra review, Term-document matrices and singular value decompositions, Low-rank approximations, Latent semantic indexing.

Web search basics: Background and history, Web characteristics, Advertising as the economic model, The search user experience, Index size and estimation, Near-duplicates and shingling.

Web crawling and Indexes: Overview, Crawling, Distributing indexes, Connectivity servers.

Link analysis: The Web as a graph, Page Rank, Hubs and Authorities.

Suggested Readings:

1. Christopher D. Manning, PrabhakarRaghavan, HinrichSchütze, *An Introduction to Information Retrieval*, Cambridge University Press, Cambridge, England,2008
2. David A. Grossman, OphirFrieder, *Information Retrieval–Algorithms and Heuristics*, Springer, 2nd Edition (Distributed by Universities Press),2004.
3. Gerald J Kowalski, Mark T Maybury. *Information Storage and Retrieval Systems*, Springer,2000
4. SoumenChakrabarti, *Mining the Web: Discovering Knowledge from Hypertext Data*, Morgan- Kaufmann Publishers,2002.

Course Code	Course Title						Core/ Elective
PC 531 CS	**SOFTWARE ENGINEERING LAB**						**CORE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	-	-	-	2	25	50	2

Course Objectives

- To understand the software engineering methodologies for project development.
- To gain knowledge about open source tools for Computer Aided Software Engineering (CASE).
- To develop test plans and test cases to perform various testing.

Course Outcomes

Student will be able to:

- Analyze and design software requirements in an efficient manner.
- Use open source case tools to develop software
- Implement the design , debug and test the code

I. FORWARD ENGINEERING

Students have to form a team with a batch size of two or three and take up a **case study based project** to analyze, plan, design UML models and create a prototypical model (identifying deliverables) by coding the developed designs and finally documenting considering any one example of the following domains:-

1. Academics (Course Registration System, Student marks analyzing system)
2. Health Care (Expert system to prescribe medicines for given symptoms, Remote Diagnostics, Patient/Hospital Management System)
3. Finance (Banking:ATM/NetBanking, UPI:PayTM/PhonePay, Stocks:Zerodha)
4. E-Commerce (various online shopping portals like FlipKart/Amazon/Myntra)
5. Logistics (Postal/Courier:IndiaPost/DTDC/UPS/FedEx, Freight:Maersk)
6. Hospitality (Tourism Management:Telangana Tourism/Incredible India, Event Management: MeraEvents/BookMyShow/Explara/EventBrite)
7. Social Networking (LinkedIn, FaceBook, Shaadi.com, BharatMatrimony, Tinder)
8. Customer Support (Banking Ombudsman,Indian Consumer Complaints Forum)
9. Booking/Ticketing(Food:Zomato/Swiggy/BigBasket/Grofers/JioMart, Hotel:OYO/Trivago or Travel: {Cars:Uber/OLA/Zoom, Railways:IRCTC, Buses:OnlineTSRTC/RedBus/AbhiBus, Flights:MakeMyTrip/Goibibo, Ships:Lakport})

II. REVERSE ENGINEERING: Students have to refer any project repository:GitLab/GitHub, execute the code in order to observe its functionalities/features/requirements and by the help of any tool derive the designs from the code for understanding the relationships among various subsystems/classes/components and

if the tool partially generates models then identify by associating elements to judge/mark the appropriate relationships.

III. TESTING: Prepare Test Plan and develop Test Case Hierarchy to monitor or uncover/report errors using manual/automated testing tools

Software Required: *StarUML/Umbrello, NetBeans/Eclipse IDE, XAMPP/MEAN stack, JUnit, JMeter, Selenium, Bugzilla*

Course Code	Course Title						Core/ Elective
PC 532 CS	**OPERATING SYSTEMS LAB**						**CORE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	-	-	-	2	25	50	2

Course Objectives

- Learn different types of CPU scheduling algorithms
- Demonstrate the usage of semaphores for solving synchronization problem
- Understand memory management techniques and different types of fragmentation that occur in them and various page replacement policies
- Understand Banker's algorithm used for deadlock avoidance
- Learn various disk scheduling algorithms.

Course Outcomes

- Evaluate the performance of different types of CPU scheduling algorithms
- Implement producer-consumer problem, reader-writers problem, Dining philosopher's problem
- Simulate Banker's algorithm for deadlock avoidance
- Implement paging replacement and disk scheduling techniques
- Use different system calls for writing application programs.

I. CASE STUDY

Perform a case study by installing and exploring various types of operating systems on a physical or logical (virtual) machine

II. List of Experiments (preferred programming language is C)

1. Write a C programs to implement UNIX system calls and file management
2. Write C programs to demonstrate various process related concepts.
3. Write C programs to demonstrate various thread related concepts.
4. Write C programs to simulate CPU scheduling algorithms: FCFS, SJF, Round Robin
5. Write C programs to simulate Intra & Inter-Process Communication (IPC) techniques: Pipes, Messages Queues, Shared Memory.
6. Write C programs to simulate solutions to Classical Process Synchronization Problems: Dining Philosophers, Producer-Consumer, Readers-Writers
7. Write a C program to simulate Bankers Algorithm for Deadlock Avoidance.
8. Write C programs to simulate Page Replacement Algorithms: FIFO, LRU
9. Write C programs to simulate implementation of Disk Scheduling Algorithms: FCFS, SSTF

Course Code	Course Title						Core/ Elective
PW 533 CS	**MINI PROJECT**						**CORE**
Prerequisite	Contact Hours Per Week				CIE	SEE	Credits
	L	T	D	P			
-	-	-	-	2	50	-	

Course Objectives: To prepare the students

- To enhance practical and professional skills.
- To familiarize tools and techniques of systematic literature survey and documentation
- To expose the students to industry practices and team work.
- To encourage students to work with innovative and entrepreneurial ideas

Course Outcomes

- Demonstrate the ability to synthesize and apply the knowledge and skills acquired in the academic program to the real-world problems.
- Evaluate different solutions based on economic and technical feasibility
- Effectively plan a project and confidently perform all aspects of project management
- Demonstrate effective coding, written, presentation and oral communication skills

The students are required to carry out mini projects in any of the areas such as Data Structures, Microprocessors and Interfacing, Database Management Systems, Operating Systems, Design and Analysis of Algorithms, Software Engineering, Data Communications, Web Programming & Services, Computer Networks, Compiler Construction, and Object Oriented System Development.

Problems Statements are suggested to be taken from Smart India Hackathon (SIH) Portal invited from the Ministries / PSUs / MNCs / NGOs to be worked out through.

The project could be classified as hardware, software, modeling, simulation etc. The project should involve one or many elements of techniques such as analysis, design, and synthesis.

The department will appoint a project coordinator who will coordinate the following:

1. Grouping of students (maximum of 3 students in a group)
2. Allotment of projects and project guides.
3. All projects allotment is to be completed by the 4th week of the semester so that the students get sufficient time for completion of the project.
4. Disseminate guidelines given by monitoring committee comprising of senior faculty members to the students and their guides.

Sessional marks are to be awarded by the monitoring committee.

Common norms will be established for the final presentation and documentation of the project report by the respective departments.

Students are required to submit a presentation and report on the mini project at the end of the semester.