Computer Science Curriculum
Grade 11 and 12

Grade: 11 and 12 | Credit hours: 5

Annual working hours: 160 | Subject code: 4281

1. Introduction

Computer Science is the study of computers and how they work to solve problems. It includes learning about programming, computer systems, and technologies like websites and databases. The Computer Science curriculum for Grades 11 and 12 in Nepal, developed by the Curriculum Development Centre under the National Curriculum Framework for School Education, 2076 teaches the basics of computers in a fun way. It covers simple coding, networking, cybersecurity, AI, and digital systems. With easy lessons and hands-on activities, it helps students learn ICT skills for future studies or tech jobs in Nepal.

Nepal needs more people with technical skills for today's digital world. The Computer Science curriculum for Grades 11 and 12 teaches students to think smart, solve problems, and understand technology. With simple lessons and fun activities, it prepares students for college or technical jobs. It stays updated to match Nepal's needs and global tech trends.

The Computer Science curriculum for Grades 11 and 12 in Nepal, is designed to build knowledge, understanding, and basic principles of Computer Science with strong connections to technology, innovation, and daily life. It helps students understand how local and global tech trends impact each other. The curriculum includes level-wise competencies, grade-wise learning outcomes, and a clear scope and sequence of topics. It also provides suggested practical/project work, teaching methods, and assessment strategies to achieve the intended learning goals. The curriculum covers 80 hours of theoretical learning and 80 hours of practical and project work to develop a strong understanding of computer concepts, programming, and digital solutions.

2. Competencies

Upon completing the course, students will demonstrate the following competencies:

a. Explain the functionality of computer systems and digital logic.
b. Utilize various software application packages and tools to create effective solutions.
c. Evaluate the impact of interactive technology and social media on society and human behaviour.

d. Design and administer databases with a focus on integrity and security.
e. Apply cyber security principles to protect information systems and address ethical considerations.
f. Employ computational thinking to solve complex problems and develop efficient algorithms.
g. Analyze the potential applications and gamification of emerging technologies.
h. Develop and maintain dynamic websites using web development technologies.
i. Implement computer network systems to enable efficient communication and data transfer.
j. Manage small-scale software projects using object-oriented programming methodologies.
k. Investigate artificial intelligence tools and their practical applications.
l. Describe software development practices that enhance project success and outcomes.

3. Grade wise Learning Outcomes

Class 11

S.N.	Area/Unit	Learning Outcome
1	**Computer System and Organization**	● Define and describe the types of computers and their basic components. ● Explain the architecture and functions of the CPU, including the control unit, ALU, and registers. ● Identify and compare different types of memory (RAM, ROM, Cache) and their roles in the memory hierarchy. ● Differentiate between primary and secondary storage, and describe the functions of hard drives, SSDs, and cloud storage. ● Classify various input and output devices, and explain the purpose of I/O ports and interfaces. ● Describe the types and components of mobile computers, including their operating systems. ● Demonstrate the process of assembling a computer and installing basic software through practical sessions.
2	**Computer Software and Application**	● Define and classify different types of software and their applications. ● Identify and describe the functions and types of operating systems, including desktop and mobile OS. ● Navigate the desktop environment, manage files, and use system utilities effectively. ● Create and format documents, insert elements, and use advanced features in word processing applications. ● Develop and analyze spreadsheets using formulas, pivot tables, and data visualization tools. ● Design and enhance effective presentations with multimedia elements, animations, and transitions.
3	**Interactive Technology and Social Media**	● Define and explain interactive technology and its applications. ● Identify and describe types of interactive technologies, such as virtual reality and augmented reality. ● Understand and apply basic concepts of image editing, including resolution and colour correction. ● Use tools and software for image manipulation, like Pixlr E, Adobe Photoshop Elements and GIMP. ● Record and edit audio using tools like Audacity, Adobe Audition enhancing quality with techniques like noise reduction. ● Create and edit videos using software like Capcut Desktop, Adobe Premiere Elements, adding transitions and special effects. ● Develop effective social media content and strategies, understanding ethical issues in digital media sharing.

4	**Web Technology**	● Explain the basic concepts of the Internet and the World Wide Web, including their history and evolution. ● Identify and describe the structure of domain names, including TLDs and cCTLDs. ● Compare and evaluate different types of web hosting services and providers, and choose appropriate hosting solutions. ● Develop and construct basic web pages using HTML, incorporating common tags, forms, and HTML5 features. ● Apply and implement JavaScript and CSS to create dynamic and interactive web content. ● Analyze and apply web security best practices to protect web applications from common vulnerabilities like SQL injection.
5	**Database Management System**	● Define DBMS and its importance. ● Explain the different types of DBMS. ● Describe the components of ER Diagrams (entities, attributes, relationships). ● Draw and interpret ER Diagrams for given scenarios. ● Identify and explain the types of relationships and their representation in ER Diagrams. ● Compare features of various database tools (MySQL, PostgreSQL, Oracle, SQL Server). ● Use SQL commands (CREATE, ALTER, DROP, INSERT, UPDATE, DELETE) to create, modify, and manage database schemas and data. ● Write and execute SQL queries to retrieve, filter, and manipulate data using SELECT statements, WHERE clauses, joins, and aggregate functions.
6	**Cyber Security and Ethical Issues**	● Describe the concept of cyber security and cybercrime. ● Explore the prevention methods for cybercrime. ● Describe the safe browsing techniques. ● Define the concept of a digital citizen. ● List good netiquette and online behaviours. ● Clarify the concept of digital footprints, privacy, and data security issues in online.
7	**Computational Thinking and Programming**	● Define computational thinking and explain its importance. ● Describe the four pillars of computational thinking. ● List common programming languages and explain the basic syntax and semantics of a chosen language. ● Break down complex problems into smaller parts, identify patterns, and create step-by-step solutions using algorithms. ● Use flowcharts to visually represent algorithms. ● Design algorithms and flowcharts using online or offline tools.

		● Identify the history, features, and applications of Python and install Python along with an IDE for development. ● Demonstrate the use of different data types along with operators for arithmetic, comparison, and assignment operations. ● Construct programs using control structures like if`, elif`, else, for loops, and while loops, and analyze the impact of loop control statements.
8	**Emerging Technology**	● Define emerging technologies and explain their significance. ● Discuss the impact of emerging technologies on society and industry. ● Describe the concept and applications of Artificial Intelligence (AI) and current trends. ● Explain the key components and practical applications of the Internet of Things (IoT). ● Define cloud computing and big data, and discuss their benefits and importance. ● Define the online education and its practices. ● Describe the concepts of e-governance and e-commerce.

Class 12

S.N.	Unit/Area	Learning Outcome
1	**Concept of Digital Logic**	• Define and explain different number systems and perform conversions between them. • Perform basic arithmetic operations in binary. • Understand and apply Boolean algebra in digital logic. • Write, simplify, and analyze Boolean expressions using Boolean laws. • Define and explain the function of basic logic gates and create truth tables.
2	**Computer Network**	• Define computer networks and explain their importance and types. • Describe various network topologies and common network protocols. • Explain the OSI and TCP/IP models and compare their layers. • Identify and explain the functions of different network devices and their configurations. • Discuss the importance of network security and management practices. • Configure and troubleshoot basic network setups and protocols.
3	**Web Development**	• Define key web technologies and client-server architecture. • Explain the basic syntax and control structures of PHP. • Demonstrate how to set up a MySQL database and connect it to PHP. • Differentiate between creating, reading, updating, and deleting records in PHP and MySQL.

		• Assess the effectiveness of form handling and validation techniques in PHP. • Develop dynamic web applications integrating frontend and backend technologies. • Implement user authentication systems using PHP and MySQL. • Design and present a complete web application project, showcasing their understanding and skills.
4	**Computer Programming**	• Review of Python Program and its basic structure up to looping concept. • Create reusable code by defining functions with arguments and return values, and integrate custom modules and built-in libraries into Python projects. • Utilize lists and dictionaries to store and manage data, and modify data using list and dictionary. • Implement file handling for reading and writing data. • Conceptualize exceptions with try-except. • Utilize Python libraries and modules effectively, including importing and managing them with pip. • Create and manipulate NumPy arrays, Pandas DataFrames, and Matplotlib plots for data analysis and visualization. • Demonstrate the NumPy, Pandas and Matplotlib library practical applications.
5	**Object Oriented Programming**	• Explain the fundamental concepts of Object-Oriented Programming (OOP) and its key principles. • Identify and differentiate between classes and objects in Python, including their attributes and methods. • Demonstrate the use of inheritance and polymorphism in Python through practical examples.
6	**Artificial Intelligence and Application**	• Define Artificial Intelligence and describe its history and evolution. • Explain the differences between Narrow AI, General AI, and Superintelligent AI. • Demonstrate their knowledge by creating digital artwork using Generative AI tools. • Compare different machine learning algorithms and evaluate their effectiveness. • Assess the ethical implications of AI applications in various sectors. • Develop a simple machine learning model and implement an image recognition system.
7	**Software Development Process**	• Describe the purpose and applications of models such as Waterfall, Agile. • Explain techniques for gathering and evaluating requirements. • Apply basic cost estimation and scheduling techniques effectively. • Describe basics types of software testing.
8	**Software Project**	• Develop and demonstrate simple base software project.

4. Scope and Sequence

Class 11

S.N.	Area/Unit	Contents Area with teaching hours (Theory (T) + Practical (P))	Working hours
1	**Computer System and Organization**	**Theory and practical (14T +2P =16 Hrs.)** 1.1 Introduction to Computer Systems (2T) 1.1.1 Definition and types of computers 1.1.2 Basic components of a computer system 1.2 Central Processing Unit (CPU) (2T) 1.2.1 CPU architecture and functions 1.2.2 Control unit, ALU, and registers 1.2.3 CPU performance and measurement 1.3 Memory (2T) 1.3.1 Types of memory: RAM, ROM, Cache 1.3.2 Memory hierarchy and storage 1.4 Storage Devices (2T) 1.4.1 Primary vs. secondary storage 1.4.2 Hard drives, SSDs, and external storage (USB) 1.5 Input/Output (I/O) Devices (2T) 1.5.1 Types of input devices 1.5.2 Types of output devices 1.5.3 I/O ports and interfaces 1.6 Mobile Computers (2T) 1.6.1 Types of mobile computers (laptops, tablets, smartphones) 1.6.2 Mobile device components 1.6.3 Concept of mobile operating systems (Android and iOS) 1.7 Computer Components and Peripherals (2T) 1.7.1 Motherboard and its components 1.7.2 Peripheral devices (printers, scanners, webcams)	**16**

		Practical/Demo Task (2P) • Demonstrate Assembling a computer process (only show physically). • Demonstrate Installing and configuring basic software (only show physically). • Demonstrate the PC Building Simulators to show the logical structure of the PC.	
2	**Computer Software and Application**	**Theory and Practical (6T + 15P = 21 Hrs.)** 2.1 Introduction to Computer Software (2T) 2.1.1 Definition and types of software 2.1.2 Examples of software applications 2.1.3 Operating systems overview: functions, types of operating system Desktop (Windows, Linux, macOS), Mobile (iOS, Android) 2.2 Basic Operations and User Interface (2T+2P) 2.2.1 Navigating the desktop environment 2.2.2 File management: Creating, moving, and deleting files 2.2.3 Using system utilities: Disk cleanup and task manager 2.3 Working with Word Processing Application (5P) 2.3.1 Document creation and formatting • Creating new documents • Applying and modifying styles • Using templates for consistency 2.3.2 Inserting and managing elements • Adding tables, images, and charts • Using headers, footers, and page numbers • Track changes and comments for collaboration 2.3.3 Automation and efficiency • Performing mail merge for bulk letters • Recording and running macros to automate tasks	**21**

		2.4 Working with Spreadsheet (1T+2P) 2.4.1 Data management • Creating and formatting spreadsheets • Using basic formulas (SUM, IF and COUNTIF) • Sorting and filtering data • Page setup for print data 2.4.2 Data analysis tools • Creating pivot tables for data analysis • Generating charts and graphs 2.5 Effective Presentation Design (1T+2P) 2.5.1 Designing effective presentations • Creating slides with themes and templates • Adding multimedia elements: Images, videos, and audio • Using SmartArt for visual representation 2.5.2 Enhancing effective presentations • Applying animations and transitions • Setting up slide shows and rehearsing timings **Project Work (4P)** ● Write a report on a topic given by the instructor, including sections on how to track changes, leave comments, and share documents with peers. ● Create a spreadsheet on a topic given by the instructor including basic formula, data sorting, graph, pivot table. Example of sales report, payroll sheet or similar tasks. ● Create a presentation on a topic given by teacher. Utilize slide transitions, animations, and questioning techniques to make it engaging.	
3	**Interactive Technology**	**Theory and Practical (8T + 12P = 20 Hrs.)** 3.1 Basics of Interactive Technology (1T)	**20**

	and Social Media	3.1.1 Definition and applications 3.1.2 Concept of use of multimedia in interactive technologies (virtual reality, augmented reality, extended and interactive websites) 3.2 Image Editing and Manipulation (2T) 3.2.1 Basic concepts of image editing (resolution, colour correction, and layers) 3.2.2 Tools and software for image manipulation 3.2.3 Common techniques (cropping, resizing, applying filters, and retouching images) 3.3 Audio Production Tools and Techniques (1T) 3.3.1 Basics of audio production: Understanding sound recording, editing, and mixing 3.3.2 Tools for recording and editing audio: Introduction to software like Audacity, Adobe Audition etc. 3.3.3 Techniques for enhancing audio quality: Noise reduction, equalization, and adding effects 3.4 Video Production Tools and Techniques (2T) 3.4.1 Basics of video production: Understanding video recording, editing, and publishing 3.4.2 Tools for recording and editing video: Overview of software like Capcut Desktop, Adobe Premiere Elements, and iMovie 3.4.3 Techniques for creating engaging videos: Storyboarding, adding transitions, and special effects 3.5 Social Media Platforms and Their Uses (1T) 3.5.1 Overview of popular social media platforms such as Facebook, Instagram, X, LinkedIn, and YouTube 3.5.2 Uses and benefits of social media: Personal branding, networking, marketing, and information sharing	

		3.5.3 Strategies for effective social media presence: Content creation, engagement tactics, and analytics 3.6 Ethical Issues in Digital Media Sharing (1T) 3.6.1 Understanding digital ethics: Privacy, consent, and digital footprint 3.6.2 Privacy concerns and copyright issues: Protecting personal information and respecting intellectual property 3.6.3 Responsible sharing and publishing practices: Best practices for ethical behavior online **Practical Sessions (12 Hours)** ● Image Editing Practice (2P) o Using software such as Pixlr E, Adobe Photoshop Elements, GIMP, and other online editors or online tools: Hands-on practice with basic tools o Editing and manipulating images: Applying techniques like cropping, resizing, and color correction o Applying filters and effects: Experimenting with different filters and effects to enhance images ● Creating an Audio Clip (2P) o Recording audio using tools like audacity or adobe audition: Setting up and using a microphone o Editing and enhancing audio quality: Cutting, splicing, and applying noise reduction o Adding background music and effects: Incorporating music tracks and sound effects ● Producing a Short Video (2P) o Recording video using a smartphone or camera: Basic filming techniques	

		○ Editing video using software like Capcut desktop, Adobe premiere elements: cutting, adding transitions, and effects ○ Adding titles and credits: Creating professional-looking videos ● Social Media Content Creation (2P) ○ Creating posts for different platforms: Crafting engaging content for Facebook, Instagram, TikTok and X etc. ○ Using hashtags and tagging: Strategies for increasing reach and engagement ○ Scheduling posts using tools like Facebook pages: Planning and automating social media activity ● Image and Video Sharing (2P) ○ Uploading images and videos to social media ○ Engaging with audience through comments and shares ○ Analyzing Engagement Metrics: Using analytics to measure success ● Collaborative Students Project (2P) ○ Students prepare a simple Multimedia Project: Combining image, audio, and video elements and share it on social media as given instruction to the class teacher. ○ Presenting the project and receiving feedback: Sharing the final product and discussing improvements	
4	**Web Technology**	**Theory and Practical (12T + 16P = 28 Hrs.)** 4.1 Introduction to Web Technology (3T) 4.1.1 Overview of the Internet and the World Wide Web 4.1.2 History and evolution of web technologies 4.1.3 Basic web architecture (client-server model) 4.2 Domain Names and Web Hosting (3T) 4.2.1 Concept of domain name	28

		4.2.2 Structure of domain names (TLDs, cCTLDs) 4.2.3 Concept of Web Hosting and web hosting providers 4.3 Web Development Basics (3T) 4.3.1 Introduction to HTML, CSS, and JavaScript 4.3.2 Role of front-end and back-end development 4.3.3 Overview of web development tools and development environments 4.4 Web Security and Practices (3T) 4.4.1 Basic web security concepts (SSL, HTTPS) 4.4.2 Introduction to common web vulnerabilities **Practical Sessions for HTML (6P)** • HTML Basics (1P) o Basic structure of an HTML document o Common HTML tags (headings, paragraphs, links, images) • Creating a Simple Web Page (1P) o Building a basic web page with HTML o Adding text, images, and links o Structuring content with lists and tables • Forms and Inputs (2P) o Creating forms in HTML o Different types of input fields o Form validation basics • HTML5 Features (2P) o Introduction to HTML5 o New semantic elements (header, footer, article, section) o Multimedia elements (audio, video) **Practical Sessions for JavaScript (4 Hours)** • JavaScript Basics (2P) o Introduction to JavaScript o Variables, data types, and operators	

		○ Basic syntax and structure • Functions and Events in JavaScript (2P) ○ Defining and calling functions ○ Event handling (click, mouseover, etc.) ○ Using JavaScript to manipulate HTML elements **Practical Sessions for CSS (4 Hours)** • CSS Integrated with HTML (4P) ○ Style a Heading and Paragraph ○ Create a Colored Box ○ Make a Simple Button ○ Layout with Flexbox ○ Responsive Image **Project Work (2P)** ○ Building a small interactive web application ○ Applying JavaScript and CSS to create dynamic content	
5	**Database Management System**	**Theory and Practical (10T + 15P = 25 Hrs.)** 5.1 Introduction to DBMS (2T) 5.1.1 Concept of Database Management Systems (DBMS) 5.1.2 Importance and applications of DBMS 5.1.3 Types of DBMS (hierarchical, network, relational, object-oriented) 5.1.4 Concept of NoSQL 5.2 Entity-Relationship (ER) Diagrams (3T) 5.2.1 Introduction to ER Diagrams 5.2.2 Components of ER Diagrams (entities, attributes, relationships) 5.2.3 Creating ER Diagrams 5.3 Relationships in Databases (3T) 5.3.1 Types of relationships (one-to-one, one-to-many, many-to-many) 5.3.2 Representing relationships in ER Diagrams 5.3.3 Integrity Constraints: Entity integrity and Referential integrity	**25**

		5.4 Database Tools and Software (2T) 5.4.1 Overview of database tools (MySQL, PostgreSQL, Oracle, SQL Server, MongoDB) 5.4.2 Concept of database design and management tools (phpMyAdmin, pgAdmin, SQL Server Management Studio) **Practical Sessions for DBMS (15 Hours)** • Data Definition Language (DDL) (2P) ○ Creating and modifying database schemas ○ Using SQL commands: CREATE, ALTER, DROP • Data Manipulation Language (DML) (2P) ○ Inserting, updating, and deleting data ○ Using SQL commands: INSERT, UPDATE, DELETE • Basic SQL Queries (4P) ○ Writing basic SQL queries ○ Retrieving data using SELECT statements ○ Filtering data with WHERE clause ○ Using aggregate functions (COUNT, SUM, AVG, MAX, MIN) ○ Grouping data with GROUP BY ○ Filtering groups with HAVING clause • Join Statements **(3P)** ○ Introduction to JOIN operations ○ Writing simple queries using joins two or more tables • Project Work (4P) ○ Designing and implementing a small database project ○ Applying DDL, DML include basic concept of join statement	
6	**Cyber Security and Ethical Issues**	**Theory and Practical (10T + 6P = 16 Hrs.)** 6.1 Introduction to Cyber Security (2T) 6.1.1 Cyber security and its importance	**16**

		6.1.2 Basic principles of cyber security 6.2 Types of Cyber Threats and Attacks (2T) 6.2.1 Different types of cyber threats (malware, phishing, cyber bullying, ransomware, and social engineering) 6.2.2 Various cyber attacks (DDoS, SQL injection, man-in-the-middle, etc.) 6.3 Cyber Security Measures and Tools (3T) 6.3.1 Preventive measures (firewalls, antivirus software, and encryption) 6.3.2 Safe online surfing techniques 6.4 Concept of Ethical Hacking (1T) 6.5 Legal and Ethical Issues in Cyber Security (2T) 6.5.1 Laws and regulations in Nepal 6.5.2 Ethical Issues and Netiquitee 6.5.3 Concepts of digital citizenship and digital footprints **Practical Sessions (6P)** • Setting Up Operating System's Firewalls: Configure and test firewall settings in Window OS or Mobile OS • Using Antivirus Software: Install and run antivirus scans • Encryption: Demonstrate using end to end encryption in messaging applications such as Viber, WhatsApp, https vs http use in browser • Configure Multi-factor authentication in email or social media	
7	**Computational Thinking and Programming**	**Theory and Practical (12T + 14P = 26 Hrs.)** 7.1 Introduction to Computational Thinking (2T) 7.1.1 Define computational thinking and its importance 7.1.2 Concept of pillars of computational thinking: decomposition, pattern recognition, abstraction, and algorithms 7.2 Problem-Solving Strategies (3T) 7.2.1 Concept of decomposition: process and examples 7.2.2 Concept of pattern recognition: process and examples	**26**

		7.2.3 Concept of algorithms: process and examples 7.3 Introduction to Algorithms and Flowcharts (2T) 7.3.1 Algorithms: Definition and importance 7.3.2 Flowcharts: Use, symbols and examples 7.4 Overview of Python: History, features, and applications (1T) 7.4.1 Python Installation: Setting up Python, and installing an IDE (IDLE, Jupyter Notebook, VS Code) 7.4.2 Basic Syntax: Understanding Python syntax, indentation, and comments 7.4.3 Running Python Code: Running scripts and command-line basics 7.4.4 Basic I/O Operations: Using input () and print () functions 7.5 Data Types and Operators (2T) 7.5.1 Primitive Data Types: Integers, floats, strings, Booleans 7.5.2 Data Type Conversion: Type Casting and implicit conversion 7.5.3 String Manipulation: Common string methods, slicing, and formatting 7.5.4 Arithmetic Operators: Basic math operations 7.5.5 Comparison and Logical Operators: Conditions and truth values 7.5.6 Assignment Operators: Compound assignments (+=, -=, etc.) 7.6 Control Structures (2T) 7.6.1 Conditional Statements: if, elif, and else structures 7.6.2 Loops: Understanding for and while loops 7.6.3 Loop Control Statements: break, continue, and pass **Practical Sessions** **Practical Task 1: (4P)** • Designing Algorithms and Flowcharts using online or offline tools o Design an algorithm and flowchart for a simple sequence of steps o Design an algorithm and flowchart that includes conditional statements.	

		○ Design an algorithm and flowchart that includes loops • **Practical Task 2: (2P)** ○ Setting up Python and an IDE on individual systems ○ Writing and running a simple Python program that prints a message to the screen ○ Practicing basic I/O operations: prompts, reading input, and displaying output ○ Exploring different ways to run Python scripts in the chosen IDE • **Practical Task 3: (4P)** ○ Exercises on data types: defining variables, practicing type conversion ○ Writing small programs to manipulate strings ○ Practicing arithmetic and logical operations on different data types ○ Creating expressions to evaluate using assignment operators ○ Comprehensive exercise combining different operators in one program • **Practical Task 4: (4P)** ○ Writing basic programs with conditional statements ○ Exploring nested conditions and creating a small program ○ Practicing for and while loops to handle repetitive tasks ○ Writing programs that utilize break, continue, and pass within loops	

8	**Emerging Technology and Issues**	**Theory (8T+0P = 8 Hrs.)** 8.1 Introduction to Emerging Technologies (1T) 8.1.1 Definition and significance of emerging technologies 8.1.2 Impact on society and industry 8.2 Artificial Intelligence (AI) (1T) 8.2.1 Definition and applications 8.2.2 Current trends (Generative AI and Large Language Model (LLM)) 8.2.3 Concept of Prompt Engineering 8.3 Internet of Things (IoT) (1T) 8.3.1 Concept and key components (sensors, connectivity, data processing) 8.3.2 Practical applications (smart homes, healthcare, industrial automation etc.) 8.4 Cloud Computing and Big Data (2T) 8.4.1 Cloud Computing: Concept and benefits (scalability, cost-efficiency, accessibility) 8.4.2 Big Data: Definition and importance 8.5 Extended Reality (XR) and 3D Printing (1T) 8.5.1 Extended Reality (XR): Definition and components (virtual reality, augmented reality, mixed reality) 8.5.2 Applications of XR in gaming, education, and training 8.5.3 Basics of 3D printing technology and its applications in manufacturing, healthcare, and prototyping 8.6 Online Education (1T) 8.6.1 Definition and importance 8.6.2 Online Vs blended learning and examples 8.6.3 Massive Open Online Course and examples 8.7 E-Governance and E-Commerce (1T) 8.7.1 E-Governance: Definition and role in improving public services and transparency 8.7.2 E-Commerce: Concept and impact on traditional retail and consumer behaviour	**8**
Theory 80 hours +Practical 80 hours			

Class 12

S.N.	Unit/Area	Contents Area with teaching hours (Theory (T) + Practical (P))	Working hours
1	**Concept of Digital Logic**	**Theory and Practical (12T+ 2P = 14 Hrs)** 1.1 Number System Concepts (2T) 1.1.1 Introduction to number Systems 1.1.2 Different number systems (binary, decimal, octal, hexadecimal) 1.1.3 Importance of number system computing 1.2 Binary Arithmetic (2T) 1.2.1 Explanation of basic arithmetic operations (addition, subtraction, 1s and 2s Complement) in binary 1.3 Boolean Logic (4T) 1.3.1 Introduction to Boolean Algebra: Define Boolean algebra and its significance in digital logic, Basic Boolean operations (AND, OR, NOT) 1.3.2 Boolean Expressions and Boolean laws: Laws of Boolean algebra (Boolean identities, Complement Laws, Identity, Commutative, Associative and Distributive), Statement and verification of Laws of Boolean algebra using truth table 1.4 Logic Gates Concepts (4T) 1.4.1 Definition and function of basic logic gates (AND, OR, NOT, NAND, NOR, XOR, XNOR) 1.4.2 Truth tables for each gate **Practical (2P)** • Logic Gates simulation using tools such as Logic.ly	**14**
2	**Computer Network**	**Theory and Practical (14T + 8P = 22 Hrs.)** 2.1 Data Communication Methods (2T) 2.1.1 Basic elements of data communication 2.1.2 Analog vs. digital transmission 2.1.3 Periodic vs. non-periodic signals 2.2 Introduction to Computer Networks (2T) 2.2.1 Definition and importance of computer networks 2.2.2 Types of Networks (LAN, MAN, WAN,PAN, SDWAN) 2.2.3 Network Architectures (Peer-to-Peer, Client-Server)	**22**

		2.3 Network Topologies (2T) 2.3.1 Common Network Topologies (Bus, Ring, Star, Mesh, Hybrid) 2.3.2 Advantages and disadvantages of each Topology 2.4 Network Protocols (2T) 2.4.1 Basic Network Protocols (TCP/IP, HTTP, FTP) 2.4.2 Protocol Functions and Uses 2.5 Network Devices and Their Functions (2T) 2.5.1 Common Network Devices (Router, Switch, Modem) 2.5.2 Functions and uses of each device 2.5.3 Setting up and configuring network devices 2.6 OSI and TCP/IP Models (2T) 2.6.1 OSI Model Layers and functions 2.6.2 TCP/IP Model Layers and functions 2.6.3 Comparison between OSI and TCP/IP models 2.7 WiFi Networking and Network Security (2T) 2.7.1 Basics of WiFi Networking 2.7.2 Setting Up a secure WiFi network 2.7.3 Concept of network security principles (Encryption, Firewalls, IDS) **Practical Sessions (8 hours)** • Setting up a local network (4P) o Configuring a LAN using simulation tools o Establishing a wireless network using a simulator, using tools like cisco packet tracer o Use Simulator to design LAN concept • Network Troubleshooting and Security (4P) o Identifying and resolving basic network issues in LAN and Wi-Fi. o Apply basic network security measures	
3	**Web Development**	**Theory and Practical (12T + 18P = 30 Hrs.)** 3.1 Introduction to Web Development (2T) 3.1.1 Overview of web technologies 3.1.2 Client-server architecture in web technologies 3.1.3 Overviews of HTML and its common tags 3.2 PHP Basics and Syntax (2T)	30

		3.2.1 Introduction to PHP 3.2.2 PHP syntax and variables 3.2.3 Control structures (if, else, switch) 3.2.4 Functions and arrays 3.3 Database Connectivity with MySQL (2T) 3.3.1 Connecting PHP to MySQL 3.3.2 Executing SQL Queries from PHP 3.4 CRUD Operations in PHP and MySQL (3T) 3.4.1 Creating records 3.4.2 Reading records 3.4.3 Updating records 3.4.4 Deleting records 3.5 Building Dynamic Web Applications (3T) 3.5.1 Form handling in PHP 3.5.2 Session management 3.5.3 User authentication **Practical Sessions (18 hours)** • Setting up the development environment (2P) o Setting up a local server o Creating a project directory • Basic PHP Scripting (2P) o Writing simple PHP scripts o Using PHP variables and control structures o Creating and using functions • Implementing CRUD Operations in PHP (3P) o Creating a PHP script to add records o Displaying records in a web page o Updating and deleting records • Form Handling and Validation (3P) o Creating HTML forms o Handling form data with PHP o Validating user input	

		• User Authentication (3P) o Creating a registration system o Implementing login and logout functionality o Managing user sessions • Web Development Project (5P) o Plan, design, develop, test, debug a simple dynamic web application in PHP and MySQL.	
4	**Computer Programming**	**Theory and Practical (14T + 22P = 36 Hrs.)** 4.1 Review of Python Programming (3T) 4.2 Functions and Modules (3T) 4.2.1 Creating functions: def, naming rules 4.2.2 Inputs (Parameters & Arguments): Required, *args, **kwargs. 4.2.3 Returning Values: return vs print 4.2.4 Variable Scope: Local vs global variables 4.2.5 Built-in Modules: import, examples (math, random). 4.2.6 Custom Modules: Save & reuse your own .py files 4.3 Working with Lists and Dictionaries (2T) 4.3.1 Lists: Creating, accessing, modifying, and common methods (append, extend, etc.). 4.3.2 Dictionaries: Creating, accessing, and manipulating key-value pairs. 4.4 Concept of Exception Handling (1T) 4.5 Concept of File Handling: Reading and writing text files, CSV basics. (1T) 4.6 Handling Libraries and Modules (4T) 4.6.1 Introduction to Python libraries and modules 4.6.2 Importing and using standard libraries 4.6.3 Creating and managing custom modules 4.6.4 Package management with pip 4.6.5 Concept of NumPy, Pandas and Matplotlib **Practical Session** • Practical Task 1: Review of Python: (4P) o IDE setup and writing and running a simple Python program that prints a message to the screen	**36**

		○ Writing basic programs with conditional statements, loop statements • Practical Task 2: Function and Modules (4P) ○ Writing simple functions and practicing with arguments and parameters ○ Exercises to understand return values and when to use them ○ Experimenting with scope: local vs. global variables in different functions ○ Importing and using built-in modules like math and random ○ Writing and organizing custom modules for code reuse • Practical Task 3: List and dictionaries (3P) ○ Exercises on creating and manipulating lists and dictionaries ○ Practicing list methods and dictionary methods (get keys, values) ○ Working on nested data structures, building and accessing complex data ○ Applying list and dictionary comprehensions in practical examples • Practical Task 4: Read file (3P) ○ Read and display a text file ○ Write user input to a file ○ Handle CSV data (Basic) • Practical Task 5: Libraries and Modules (8P) ○ Importing and utilizing standard Libraries ○ Creating custom modules and packages ○ Managing pip and external packages ○ Create and manipulate NumPy arrays ○ Create and manipulate Pandas DataFrames ○ Create and customize plots using Matplotlib	
5	**Object Oriented Programming**	**Theory and Practical (6T + 12P = 18 Hrs.)** 5.1 Introduction to Object-Oriented Programming (OOP) (2T) 5.1.1 Concept of OOP 5.1.2 Key Principles of OOP: Encapsulation, Abstraction, Inheritance, and Polymorphism 5.1.3 Benefits of OOP over Procedural Programming	**18**

		5.2 Classes and Objects in Python (2T) 5.2.1 Classes and Creating Objects 5.2.2 Class Attributes and Methods 5.2.3 The __init__ Method and Constructors 5.2.4 Instance vs. Class Variables 5.3 Inheritance and Polymorphism (2T) 5.3.1 Understanding Inheritance: Single, Multiple, and Multilevel 5.3.2 Polymorphism: Method Overloading and Overriding 5.3.3 Abstract Classes and Interfaces **Practical Session** (6P) • Create class and object: Make a basic class and create objects. • Use methods and constructor: Add methods and use __init__ to set values. • Inheritance Basics: Use one class inside another using inheritance. • Polymorphism and Abstract Class: Try method overriding and use abstract classes. **Project Work:** (6P) • Develop, test, and debug a simple application/system using libraries, user defined functions and visualization tool in Python.	
6	**Artificial Intelligence and Application**	**Theory and Practical (14T + 8P = 22 Hrs.)** 6.1 Introduction to Artificial Intelligence (4 T) 6.1.1 Overview of AI: Definition, history, and evolution 6.1.2 Types of AI: Narrow AI, General AI, and Superintelligent AI 6.1.3 AI vs. Human Intelligence: Key differences and similarities 6.1.4 Current Trends in AI: Recent advancements and future prospects 6.2 Machine Learning Basics (4 T) 6.2.1 Introduction to Machine Learning: Definition and types (supervised, unsupervised, and reinforcement learning) 6.2.2 Key Algorithms: Decision trees 6.2.3 Training and Testing Models: Concepts of training data, testing data, and validation 6.3 Neural Networks and Deep Learning (2 T) 6.3.1 Basics of Neural Networks: Structure and function of neurons, layers, and activation functions	**22**

		6.3.2 Deep Learning: Introduction to deep learning and its importance 6.3.3 Applications: Image recognition, natural language processing, and more 6.4 IoT and Robotics (2T) 6.4.1 Introduction to IoT: Definition, components, and applications 6.4.2 Role of AI in IoT: Enhancing IoT with AI for smart solutions 6.4.3 Robotics Basics: Types of robots and their applications 6.5 AI Applications and Ethical Issues (2T) 6.5.1 Application of Generative AI, LLM and AI Agents 6.5.2 Real-world Applications: Healthcare, finance, automotive, and more 6.5.3 Future of AI: Emerging trends and potential future applications 6.5.4 Ethical Considerations: Addressing ethical issues in AI development and deployment **Practical Sessions (8P)** **Practical Task: Handling AI-related Apps** ● Practices on prompt Engineering in Generative AI tools ● Build a simple machine learning model using online tools ● Apply chatbot using NLP tools like language translators ● Implement an image/face recognition system by using Teachable Machines ● Implement the online robotics tools and applications such as Tinkercad or Roboblocky	
7	**Software Development Process**	**Theory and Practical (8T + 0P = 8 Hrs.)** 7.1 Software Development Models (2T) 7.1.1 Introduction to software development models 7.1.2 Waterfall Model 7.1.3 Agile Model: SCRUM, XP 7.2 Requirement Collection and Analysis (2T) 7.2.1 Introduction to requirements 7.2.2 Requirement gathering techniques 7.2.3 Requirement analysis	**8**

		7.3 Time Scheduling (2T) 7.3.1 Scheduling techniques (Gantt charts) 7.4 Software Testing and Quality (2T) 7.4.1 Introduction to software testing and quality 7.4.2 Concept of unit, integration and system testing	
8	**Software Project**	**Project Work (0T + 10P = 10 Hrs.)** 8.1 Develop Real Problem based simple software project in group. 8.1.1 Define the project requirements 8.1.2 Set up the development environment 8.1.3 Design the database (if applicable) 8.1.4 Develop the application/software/website 8.1.5 Test and debug the Project 8.1.6 Create simple report	**10**
		Theory 80 hours +Practical 80 hours	**160 hours**

Possible teaching and learning activities and evaluation process and methods
Grade 11

S.N.	Unit/Area	Possible model learning activities (Methods, Techniques, and Activities)	Evaluation techniques	Working hours
1	**Computer System and Organization**	● Flipped Classroom Activity: Assign a video on how CPUs and computer systems work (e.g., Branch Education on YouTube) and have students build an Input-Process-Output model in class. ● Project-Based Learning Task: Show a detailed video on building a custom PC and discuss component selection using like PCPartPicker. ● Tech Exploration Assignment: Ask students to analyze the hardware and software of their smartphones and present their findings using like GSMArena. ● Virtual Lab Visit: Arrange a virtual tour of a data center using videos like Google Data Center Tour or Microsoft's interactive tour.	Quizzes and Tests, Peer Evaluation, Presentation, Rubrics Prefer to use digital tools	**16**
2	**Computer Software and Application**	● Lecture-Based Teaching: Deliver a short presentation on the history of operating systems and have students take notes for a follow-up discussion. ● Inquiry-Based Learning: Assign students to research different types of software applications and present their findings using slides. ● Collaborative Learning: Organize group projects where students create a report using Word, Excel, and PowerPoint to demonstrate software use. ● Flipped Classroom: Share a video tutorial on advanced Excel functions (YouTube Link) for homework and analyze data in	Quizzes and Tests, Presentations, Group Project Assessment Practical Tasks, Gamified Scoring Lab Sheet and Score	**21**

		class. ● Hands-On Learning: Conduct lab sessions where students practice formatting documents, building spreadsheets, and designing presentations. ● Gamification: Create point-based challenges for tasks in Word, Excel, and PowerPoint to make learning interactive and fun.		
3	**Interactive Technology and Social Media**	● Flipped Classroom: Assign a Photoshop tutorial for homework and have students edit images in the lab. ● Project-Based Learning: Guide students to create a multimedia presentation using Word for scripting, Excel for planning, and PowerPoint for integration. ● Interactive Simulations: Use an Audacity tutorial to help students record and edit a podcast episode. ● Collaborative Learning: Organize group video projects and edit them using photo editing tools in the lab. ● Hands-On Learning: Conduct workshops on video recording and editing using PowerPoint or similar tools. ● Gamification: Create challenges where students earn points for editing images, audio, and video using Canva or similar tools.	Evaluation of developed game and logics Quizzes and tests Presentation Develop project	**20**
4	**Web Technology**	● Flipped Classroom: Assign HTML video tutorials for homework; in the lab, students build a simple webpage. ● Project-Based Learning: Guide students to create a dynamic website using HTML and JavaScript in a project-based format. ● Interactive Simulations: Use Codecademy to simulate JavaScript coding; in the lab, students build interactive web elements.	Presentation of web pages Quiz and test Rubrics Demonstration Reflection	**28**

		● Collaborative Learning: Organize group projects where students collaborate on a web application using like GitHub for version control. ● Hands-On Learning: Conduct workshops on JavaScript basics; students practise DOM manipulation in the lab. ● Gamification: Create coding challenges with points for tasks in HTML, CSS, and JavaScript using Visual Studio Code. ● Guest Lectures and Webinars: Invite web developers for guest lectures; students apply insights to improve their web projects. ● Peer Works: Pair students to share coding techniques and solve HTML and JavaScript problems together in the lab.		
5	**Data Base Management System**	● Flipped Classroom: Assign video tutorials on SQL basics and DDL commands for homework; in the lab, students create and modify database schemas using MySQL. ● Project-Based Learning: Students design and implement a database for a small business using SQL to create tables, insert data, and run queries in MySQL. ● Interactive Simulations: Use platforms like SQLZoo or W3Schools SQL Editor to practise SQL queries; in the lab, students write and test DDL and DML commands in MySQL. ● Collaborative Learning: Organize group projects where students develop a database application, collaborating on schema design and SQL queries using MySQL. ● Hands-On Learning: Conduct workshops on advanced SQL queries and optimization; in the lab, students write complex queries and improve performance using indexing and joins. ● Gamification: Create SQL challenges where students earn points for tasks like creating tables, inserting data, and running queries using MySQL. ● Guest Lectures and Webinars: Invite database professionals to discuss best practices and real-world applications; students apply insights to enhance their database projects.	Quizzes and Tests Peer evaluation Presentation Rubrics	**25**

		● Peer Learning: Have students share SQL techniques and work in pairs or small groups to solve SQL problems together in the lab using MySQL.		
6	**Cyber Security and Ethical Issues**	● Flipped Classroom: Assign video tutorials on basic cyber security concepts and threat prevention; in the lab, students install and configure antivirus software. ● Project-Based Learning: Students identify potential threats, propose prevention strategies, and implement antivirus solutions during lab sessions. ● Collaborative Learning: Organize group projects to analyze recent cyber security breaches; students present case studies on causes, impacts, and prevention in the lab. ● Hands-On Learning: Conduct workshops on ethical hacking and penetration testing; students use tools like Kali Linux or free online tools to assess vulnerabilities and secure systems. ● Gamification: Create challenges where students earn points for tasks like spotting phishing emails and configuring firewalls using tools any Antivirus. ● Guest Lectures and Webinars: Invite cyber security professionals to discuss current threats and best practices; students apply insights to improve their lab projects and simulations.	Quizzes and Tests Peer evaluation Presentation mentoring	**16**
7	**Computational Thinking and Programming**	● Flipped Classroom: Assign video tutorials on computational thinking and flowchart creation for homework. In the lab, students create flowcharts for simple algorithms using tools like Lucidchart. ● Project-Based Learning: Have students develop a small program that solves a real-world problem. They use flowcharts to plan their algorithms and write code in Python during lab sessions.	Quizzes and Tests Peer evaluation Presentation, Demo Rubrics Rating Scale Observation form	**26**

		• Interactive Simulations: Use online platforms like Code.org to simulate programming concepts. In the lab, students practice writing and debugging code that includes sequences, iterations, and loops. • Collaborative Learning: Organize group projects where students design and implement algorithms. They collaborate to create flowcharts and write programs in Scratch or Python in the lab. • Hands-On Learning: Conduct workshops on basic programming concepts. In the lab, students write simple programs that use loops and conditionals in a language like JavaScript. • Gamification: Create coding challenges where students earn points for completing tasks like writing loops, creating algorithms, and debugging code. In the lab, they use tools like Repl.it to solve these challenges.		
8	**Emerging Technology**	• Flipped Classroom: Assign video tutorials on AI and IoT basics from platforms like Coursera or YouTube; in the lab, students explore AI tools and IoT applications. • Collaborative Learning: Organize group research on e-governance using SWAYAM or FutureLearn; students present findings using Google Slides in the lab. • Case Studies: Analyze company use of cloud services and simulate scenarios in the lab using cloud and data visualization tools.	Quizzes and Tests Peer evaluation Presentation, Demo	**8**
				160 hours

Grade 12

S.N.	Unit/Area	Possible model learning activities (Methods, Techniques, Activities)	Evaluation techniques	Working hours
1	**Concept of Digital Logic**	• Flipped Classroom: Assign video tutorials on binary operations and logic gates for homework. In the lab, students use simulation tools like Logisim to design and test digital circuits. • Project-Based Learning: Have students develop a project that involves using a simple digital calculator. They use Boolean algebra to design the logic and implement it using logic gates in the online lab. • Interactive Simulations: Use online platforms like Tinkercad to simulate digital logic circuits. In the lab, students practice building and testing circuits that perform basic binary operations. • Peer Teaching: Have students teach each other specific concepts in digital logic. In the lab, they work in pairs or small groups to solve logic problems together using tools like Digital Works.	Quizzes and Tests Peer evaluation Presentation	**14**
2	**Computer Network**	• Lecture Method: Deliver structured lectures on key networking concepts such as TCP/IP etc. In the lab, students apply these concepts by configuring and testing networks using Cisco Packet Tracer. • Flipped Classroom: Assign video tutorials on the basics of data communication and networking for homework. In the lab, students use Cisco Packet Tracer to simulate network setups and configurations. • Interactive Simulations: Use online platforms like GNS3 to simulate complex network environments. In the lab, students practise configuring routers, switches, and other network devices. • Hands-On Learning: Conduct workshops on network troubleshooting and maintenance. In the lab, students use tools like Wireshark to analyze network traffic and diagnose issues.	Quizzes and Tests Peer evaluation Presentation	**22**

3	**Web Development**	• Lecture Method: Deliver structured lectures on key concepts such as PHP syntax, MySQL queries, and CRUD operations. In the lab, students apply these concepts by developing a small web application that performs CRUD operations using PHP and MySQL. • Flipped Classroom: Assign video tutorials on PHP basics and MySQL connections for homework. In the lab, students create a simple PHP script to connect to a MySQL database and perform basic CRUD operations. • Project-Based Learning: Have students develop a web application that includes user registration and management. They use PHP for server-side scripting and MySQL for the database, implementing CRUD operations in the lab. • Interactive Simulations: Use online platforms like Codecademy to simulate PHP and MySQL coding exercises. In the lab, students practise writing and testing PHP scripts that interact with a MySQL database. • Collaborative Learning: Organize group projects where students build a Content Management System (CMS). • Hands-On Learning: Conduct workshops on setting up a local development environment using tools like XAMPP. In the lab, students configure their environments and write PHP scripts to interact with MySQL databases.	Quizzes and Tests, Peer evaluation Presentation	**30**
4	**Computer Programming**	• Lecture Method: Deliver structured lectures on key Python concepts such as control structures and list operations. In the lab, students apply these concepts by writing scripts that perform various tasks. • Guided Lab: Begin each topic with a short demo by the teacher, followed by hands-on coding tasks such as writing Python functions, manipulating lists and dictionaries, or handling files. Teachers provide starter code and students modify or extend it.	Quizzes and Tests, Peer evaluation, Presentation	**36**

		• Pair Programming Sessions: Students work in pairs to solve coding challenges, such as building a simple game/apps using loops and conditionals or debugging a script with exception handling. This encourages peer learning and collaboration. • Live Coding Demonstrations: Teachers code live in class using Jupyter Notebook or PyCharm, showing how to use built-in modules, create custom packages, or visualize data. Students follow along and replicate the process. • Interactive Coding Challenges: Weekly challenges are posted (e.g., "Build a program that reads a file and summarizes its content"), and students compete to solve them using efficient code. Leaderboards and badges can be used for motivation. • Use of Online Coding Platforms: Students practice syntax and logic on platforms like Replit, Codecademy, or Google Colab. These platforms offer instant feedback and help reinforce concepts learned in class. • Data-Driven Exercises: Students work with real or simulated datasets (e.g., student scores, weather data) to apply NumPy and Pandas for analysis and Matplotlib for visualization.		
5	**Object Oriented Programming**	• Lecture Method: Deliver structured lectures on key OOP concepts and basic operations. In the lab, students apply these concepts by writing scripts that perform data analysis tasks. • Flipped Classroom: Assign video tutorials on OOP concepts for homework. In the lab, students create object-oriented concept and operations. • Project-Based Learning: Have students develop a data analysis project. They use OOP to structure their code and libraries. • Interactive Simulations: Use online platforms like Replit, Codecademy, or Google Colab.	Quizzes and Tests Peer evaluation Presentation	**18**

		• Hands-On Learning: Conduct lab session on creating and using Python classes. In the lab, students write programs that define classes.		
6.	**Artificial Intelligence and Application**	• Lecture Method: Deliver structured lectures on key AI concepts such as supervised learning, unsupervised learning, and reinforcement learning. In the lab, students apply these concepts by writing and testing AI algorithms. • Flipped Classroom: Assign video tutorials on AI basics and machine learning concepts for homework. In the lab, students use tools like Google Colab to implement simple machine learning models. • Hands-On Learning: Conduct workshops on setting up and using AI tools. In the lab, students practice using TensorFlow and Keras to build and train neural networks. • Gamification: Create AI challenges where students earn points for completing tasks like training models, tuning hyperparameters, and deploying AI solutions. In the lab, they use tools like Kaggle to participate in competitions. • Case Studies: Analyze real-world case studies of AI applications. In the lab, students simulate these scenarios by writing Python scripts that replicate the functionality using AI libraries. • Hackathons: Organize hackathons focused on solving problems using AI and machine learning. In the lab, students collaborate to develop innovative solutions using AI techniques and tools. • Interactive Quizzes: Use tools like Kahoot! to create quizzes on AI, machine learning, and ethical issues. In the lab, students answer questions and write small scripts to reinforce their understanding.		**22**
7.	**Software Development Process**	• Selecting Project Topics: Teachers provide a list of potential project topics related to real-world problems or allow students to propose their own, ensuring relevance and feasibility.		**8**

		• Teacher Facilitation: Teachers guide students through the project development process, offering regular feedback, conducting check-ins, and providing resources and support as needed. • Flipped Classroom: Students watch video lectures at home and engage in interactive activities in class and software development best practices.		
8.	**Software Project**	• Collaborative Coding Sessions: Students work in groups (not more than four member) to develop a web application using PHP and MySQL, using Visual Studio Code for coding and GitHub for version control. • Agile Methodology Workshops: Implement Agile practices in the classroom, conducting daily stand-up meetings. • Database Design and Implementation: Students design and implement a database using MySQL Workbench for database design and phpMyAdmin for database management, performing CRUD operations. • Plan, Desing, Development, testing python based project like application development, data visualization performing CRUD operations or data handling operation. • Final Project Presentation and Report: Each group presents their project using PowerPoint and submits a 15-20 page report written in Microsoft Word or Google Docs, detailing the development process and functionality. • Encourage students to use tools like GitHub, Trello, draw.io, and VS Code across projects.		**10**
				160 hours

5. Learning Facilitation process and methods

During the delivery process of computer science teaching in classes 11 and 12, the following basic approaches will be adopted:

- **Project-Based Learning (PBL):** Introduce real-world computer science problems, encouraging students to investigate, plan, design, and complete small projects. Provide a structured framework, regular check-ins, and peer reviews.
- **Practical Coding Labs:** Implement practical coding labs, starting with simple exercises and gradually increasing complexity. Encourage experimentation, mistakes, and learning through trial and error, with guidance from teachers and problem-solving challenges.
- **Collaborative Work:** Involve students in coding tasks, assigning roles, showcasing project outcomes through presentations, and evaluating individual and group contributions.
- **Peer Mentorship:** Identify experienced students and pair them with less experienced ones. Conduct code reviews, provide guidance, and promote regular interactions.
- **Gamification and Coding Challenges:** Create coding challenges, contests, and competitions with rewards. Incorporate gamification elements like leaderboards, badges, and achievements. Create coding clubs, encourage friendly competition, and provide students with opportunities to showcase their accomplishments.
- **Flipped Classroom:** Assign video lectures on software development topics for homework and use class time for interactive activities, discussions, and hands-on practice.
- **Case Studies:** Analyze real-world software projects to understand development models, requirement collection, and testing. Use collaborative tools like Google Docs and Miro for analysis.
- **Workshops:** Conduct hands-on workshops on specific topics like requirement gathering techniques or testing strategies. Utilize tools like open sources tools such as draw.io for creating diagrams and Selenium for automated testing.
- **Interactive Quizzes and Polls**: Use interactive quizzes and polls during lectures to engage students and assess their understanding in real time. Tools like Kahoot! and Mentimeter can make learning fun and interactive.

6. Evaluation

Evaluation is an integral part of the learning process. Both formative and summative modes of evaluation are emphasized. Formative evaluation will be conducted so as to provide regular feedback for students, teachers, and parents/guardians about student learning. Class tests, unit tests, oral question-answers, reflective writing, project work, practical works, home assignments, etc. are some ways of formative evaluation.

There will be separate evaluations of theoretical and practical learning. Summative evaluation embraces theoretical examination, practical examination, and evaluation of research work or innovative work.

(a) Internal Evaluation

Internal evaluation covers 50 percent weightage. Internal evaluation consists of (a) classroom participation (5 percent) (b) practical activities (Practical works and project works) (35 percent), (c) marks from trimester examinations (10 percent). Practical work should be based on the list of activities mentioned in this curriculum. Project works should be based on the mentioned lists or created by teachers. Marks distribution for internal evaluation (practical work and project work) will be as follows:

S.N.		Main activities	Activities in detail	Percent
1		Participation	● Participation in classroom attendance	2
			● Participation in homework, classwork, project works, practical works ● Be very curious in learning, have a thorough frequent interaction in discussion, present own creative views and ideas in all activities, complete the entire task oneself	3
2	Practical and Project Work	Practical work	Conduction and presentation of practical work activities	20
			Record keeping of practical work activities	5
		Project work	Conduction and presentation of project work activities	5
			Record keeping of project work activities	5
3		**Terminal exam**		
		Terminal exam	Terminal exams should be based on specification grid.	10
			Total	**50**

Note:

i. Practical assessment will be conducted in the presence of teacher. Evaluation of practical and project work will focus both the product of work and skills competencies of student in using computer.
ii. Project work assessment is the internal assessment of reports and presentation of their project works either individually or group basis. In case of group presentation, every member of the group should submit a short reflection on the presented report in their own language. Records of project works must be attested by the head teacher.
iii. Two terminal examinations should be taken in a year. Each trimester test should be conducted in 50 full marks and converted to 10 marks. For annual trimester test, the average of the two terminal examination marks can be calculated

(b) External-Evaluation

External/final evaluation of the students will be based on the written examination. It carries 50 percent of the total weightage. Questions for the external examination will be based on the specification grid developed by the Curriculum Development Centre. Examination question paper will be developed using various levels of revised Bloom's taxonomy including remembering level, understanding level, application level and higher ability (analyzing, evaluating, and creating).

Remembering	Understanding	Applying	Higher Ability (analyzing, evaluating, creating)
15%	30%	30%	25%

Specification Grid

Subject: Computer Science

Grade 11

Full marks: 50 **Time: 2 hours**

Item plan and marks distribution																		
S. N.	**Unit/Area**	**Working Hour**	**Full marks**	**Knowledge**			**Understanding**			**Application**			**Higher Ability**			**Total marks**		
				MCQ	**S**	**L**	**MCQ**	**S**	**L**	**MCQ**	**S**	**L**	**MCQ**	**S**	**L**	**MCQ**	**S**	**L**
1	Computer System and Organization	14				1				1	1			1		1	2	1
2	Computer Software and Application	6		1				1			1					1	2	
3	Interactive Technology and social media	8		1								1				1		1
4	Web Technology	12						1			1		1	1	1	1	3	1
5	Database Management System	10		1							1			1		1	2	0
6	Cyber Security and Ethical Issues	10							1	1						1	0	1
7	Computational Thinking and Programming	12					1		1	1			1			3	0	1
8	Emerging Technology	8		1				1								1	1	
	Total	**80**		**4**		**1**	**1**	**3**	**2**	**3**	**4**	**1**	**2**	**3**	**1**	**10**	**10**	**5**

Grade 12

Item plan and marks distribution																		
S. N.	**Area/Unit**	**Working Hour**	**Full marks**	**Knowledge**			**Understanding**			**Application**			**Higher Ability**			**Total marks**		
				MCQ	**S**	**L**	**MCQ**	**S**	**L**	**MCQ**	**S**	**L**	**MCQ**	**S**	**L**	**MCQ**	**S**	**L**
1	Concept of Digital Logic	12		1	1			1			1				1	1	3	1
2	Computer Network	14					1				1			1	1	1	2	1
3	Web Development	12		1						1	1	1				2	1	1
4	Computer Programming	14					1	1							1	1	1	1
5	Object Oriented Programming	6			1					1			1			2	1	
6	Artificial Intelligence and Application	14		1			1					1		1		2	1	1
7	Software Development Process	8		1				1								1	1	
8	Software Project	**0**																
	Total	80		4	2	0	3	3	0	2	3	2	1	2	3	10	10	5

Types and number of questions, marks and time allocation

Types of question	Number of questions	Marks	Time allocation
Multiple Choice Question (MCQ)	10 x 1 =10	10	2 hours
Short question (SQ)	10 x 2 =20	20	
Long Question (LQ)	5 x 4 =20	20	
Total	25	50	

Remarks:

- The item format in composites should be met as per the specification grid.
- The designated weightage of the units/content areas should be met.
- In the case of SQ and LQ, these should ensure that 1 mark will be assigned per element expected as the correct response.
- The distribution of cognitive domains of questions should be nearly 15% knowledge/remembering, 30% understanding, 30% applying, and 25% higher ability level. The designated percentage will me made ±2 marks variation.
- SQ and LQ can be structured (have two or more sub-items). SQ and LQ can be distributed to two or more cognitive behaviours.