Course Objective: 1) To understand the principles and practices of secure coding.
2)To learn how to identify and mitigate common security vulnerabilities in software applications.
3) To study secure software development methodologies and tools.
4) To gain hands-on experience in writing secure code across different programming languages.

S. NO	Course Outcomes (CO)
CO1	Understand and apply secure coding principles in software development.
CO2	Identify and mitigate common security vulnerabilities in software applications.
CO3	Use secure software development tools and techniques effectively.
CO4	Contribute to the creation of secure software that meets industry standards.

S. NO	Contents	Contact Hours
UNIT 1	**Introduction to Secure Coding** Overview of Software Security:Importance of software security in modern applications. Common types of software vulnerabilities. The cost of insecure software. Principles of Secure Coding: Least Privilege, Defense in Depth, Secure by Default. Security policies and secure coding standards (e.g., CERT). Secure Software Development Life Cycle (SSDLC): Stages of SSDLC and their importance. Threat modeling and risk assessment.ecure design principles.	10
UNIT 2	**Common Security Vulnerabilities** Buffer Overflows: Stack-based and heap-based buffer overflows. Prevention techniques: Bounds checking, use of safe libraries. Injection Flaws: SQL Injection, Command Injection, Cross-Site Scripting (XSS).Input validation and sanitation techniques. Cross-Site Request Forgery (CSRF) and Cross-Site Scripting (XSS): Understanding CSRF and XSS attacks. Methods to prevent CSRF and XSS vulnerabilities.	12
UNIT 3	**Unit III: Secure Coding in Various Programming Languages** Secure Coding in C/C++:Handling pointers, memory management, and input validation.Avoiding common pitfalls like null pointers, integer overflows. Secure Coding in Java: Security Manager and sandboxing. Avoiding common vulnerabilities in Java: Deserialization, improper resource shutdown. Secure Coding in Web Languages (JavaScript, PHP): Protecting against XSS, CSRF, and other web vulnerabilities. Safe handling of user inputs and data sanitization.	10

UNIT 4	**Unit IV: Cryptography and Secure Data Handling** Basics of Cryptography: Symmetric and Asymmetric encryption. Key management and Public Key Infrastructure (PKI). Hashing and Digital Signatures: Importance of hashing in security. Implementing and using digital signatures. Secure Data Storage:Encrypting data at rest and in transit. Secure use of cookies and session management.	**10**
UNIT 5	**Unit V: Secure Software Testing and Auditing** Security Testing Techniques: Static and dynamic analysis tools. Fuzz testing, Penetration testing, and Code reviews. Vulnerability Assessment and Ethical Hacking: Tools and methodologies for vulnerability scanning. Reporting and mitigating identified vulnerabilities. Case Studies and Best Practices: Real-world case studies of security breaches. Industry best practices for secure software development.	**10**
	TOTAL	**42**

REFERENCES

S.No.	**Name of Books/Authors/Publishers**	**Year of Publication / Reprint**
1	"Software Security: Building Security In" by Gary McGraw, Addison-Wesley	2006
2	"The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities" by Mark Dowd, John McDonald, Justin Schuh, Addison-Wesley	2006