iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IX **Month of publication:** September 2026

DOI: https://doi.org/10.22214/ijraset.2026.84902

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IX Sep 2026- Available at www.ijraset.com

Software Development

M. Manoranjani[1], T. Jannathul Firdhouse[2], M. Haripriya[3], S. Dhushyanthi[4], R. Rubali[5]
[1, 2, 3, 4]*Department of IT, Annai Women's College, Karur, Tamil Nadu, India*
[5]*Assistant Professor, Department of BCA & IT, Annai Women's College, Karur, Tamil Nadu, India*

Abstract: ***Software development is the systematic process of designing, creating, testing, deploying, and maintaining software applications and systems. It is one of the most important areas of Information Technology because software supports almost every modern activity, including education, banking, healthcare, business, communication, transportation, entertainment, and government services. A successful software product is not created through coding alone; it requires requirement analysis, planning, system design, programming, testing, deployment, security, documentation, and continuous maintenance.Modern software development has evolved from traditional sequential approaches to flexible and automated practices such as Agile and DevOps. At the same time, technologies such as cloud computing, artificial intelligence, machine learning, Internet of Things, and modern web and mobile platforms have expanded the scope of software engineering. These technologies allow developers to create applications that are scalable, intelligent, connected, secure, and accessible from many locations.This article explains the software development process, major development models, programming technologies, testing practices, security considerations, real-world applications, benefits, challenges, and future trends. It also presents online banking as a practical example to show how different stages of software development work together to deliver a useful digital service.***

Keywords: Software Development, SDLC, Programming, Software Engineering, Agile, Waterfall, DevOps, Testing, Web Development, Mobile Applications, Cloud Computing, Artificial Intelligence, Cybersecurity.

I. INTRODUCTION

Software development is a core discipline of Information Technology that converts ideas and user requirements into working digital solutions. From a simple calculator application to a large banking platform, software is built to perform tasks, process information, automate work, and support decision-making. The rapid growth of the internet and smart devices has made software an essential part of personal, educational, commercial, and public activities.A software project normally begins with a problem or requirement. Developers and stakeholders discuss what users need, what the system should accomplish, and what limitations exist. The team then plans the work, designs the solution, develops the program, tests it, releases it, and maintains it. This organized approach reduces risk and improves software quality.

II. DEFINATION AND OBJECT

Software development is also a collaborative activity. Programmers work with analysts, designers, testers, database specialists, project managers, security professionals, and operations teams. Effective communication among these roles is important because software quality depends not only on technical code but also on correct requirements, good design, usability, security, and maintainability. Software development can be defined as a systematic process of designing, programming, testing, deploying, and maintaining software according to specified requirements. The process aims to create a product that solves a real problem and provides value to its users. The main objectives of software development are to satisfy user requirements, produce reliable results, improve efficiency, reduce repetitive manual work, protect information, and make services easier to access. A good software system should also be maintainable so that developers can correct defects and introduce improvements over time. Another important objective is scalability. A small application may initially have only a few users, but a successful system may later need to support thousands or millions of users. Developers therefore consider performance, architecture, databases, security, and infrastructure during the design stage.

III. SOFTWARE DEVELOPMENT LIFE CYCLE

The Software Development Life Cycle (SDLC) provides a structured framework for developing software. Although organizations may use different names or combine activities, the common stages are requirements, planning, design, coding, testing, deployment, and maintenance.

Software Development Life Cycle

IV. REQUIREMENT ANALYSIS

Requirement analysis identifies what the software must do. Developers and analysts communicate with customers, users, managers, and other stakeholders to collect functional and non-functional requirements. Functional requirements describe the actions the system should perform. For example, a college management system may need to register students, record attendance, store marks, and generate reports. Non-functional requirements describe qualities such as performance, security, reliability, usability, and availability. Clear requirements are important because errors in requirements can become expensive to correct later. Teams often document requirements in a specification and review them with stakeholders before development begins.

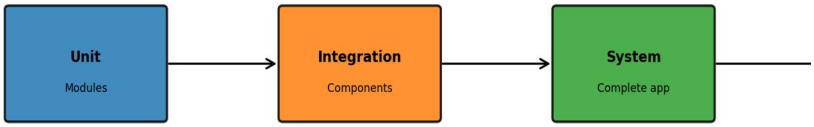
V. PLANNING AND SYSTEM DESIGN

Requirement analysis identifies what the software must do. Developers and analysts communicate with customers, users, managers, and other stakeholders to collect functional and non-functional requirements. Functional requirements describe the actions the system should perform. For example, a college management system may need to register students, record attendance, store marks, and generate reports. Non-functional requirements describe qualities such as performance, security, reliability, usability, and availability. Clear requirements are important because errors in requirements can become expensive to correct later. Teams often document requirements in a specification and review them with stakeholders before development begins.

VI. CODING AND PROGRMMING

Coding is the implementation stage in which developers convert the system design into source code. The programming language and tools are selected according to the application's requirements. Python is widely used for automation, data science, artificial intelligence, and web applications. Java is used for many enterprise and application-development scenarios. C and C++ are important for systems and performance-oriented software, while JavaScript is central to modern interactive web development. Professional coding involves more than making a program work once. Developers use meaningful names, modular structures, comments where useful, version control, code review, and testing. Good coding practices make defects easier to locate and make future changes safer. Version control systems allow teams to record changes to source code and collaborate without losing earlier versions. Branching and merging workflows can help multiple developers work on different features while maintaining an organized project history.

Software Testing Levels



ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IX Sep 2026- Available at www.ijraset.com

VII. SOFTWARE TESTING AND QUALITY

A. Assurance

Testing is a critical activity used to determine whether software behaves as expected. Testers and developers identify defects, verify requirements, and evaluate performance, usability, compatibility, and security. Testing can begin early in the development process instead of waiting until the entire application is completed.Unit testing checks individual modules or functions. Integration testing verifies that components work correctly together. System testing evaluates the complete application. Acceptance testing checks whether the software satisfies business or user expectations.Quality assurance is broader than testing. It includes processes and practicesintended to prevent defects and improve the overall development process. Automated tests can be executed repeatedly and are especially useful when software is changed frequently.

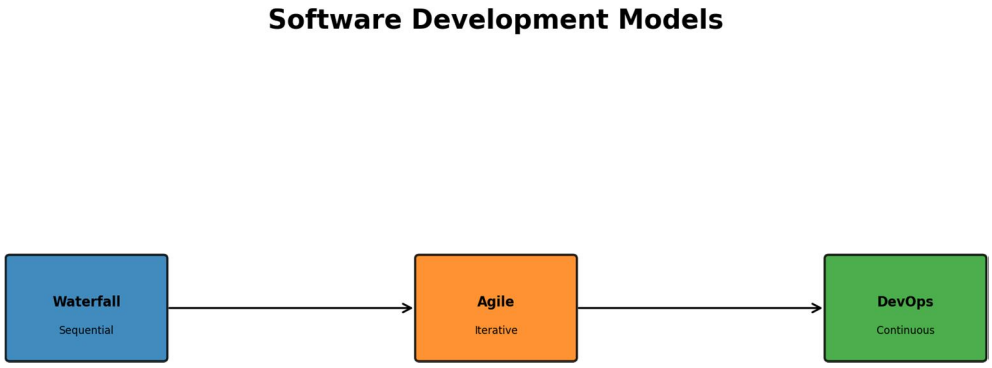


Figure 3: Common levels of software testing.

VIII. SOFTWARE DEVELOPMENT MODELS

A software development model describes how development activities are organized. The Waterfall Model follows a sequential structure in which one major phase is completed before the next. It can be easy to understand and useful when requirements are stable, but late changes may be difficult.

The Agile approach emphasizes short development cycles, frequent feedback, collaboration, and continuous improvement. Work is often divided into small increments so that useful features can be delivered and reviewed regularly.

DevOps extends collaboration across development and operations. It encourages automation, continuous integration, continuous delivery, monitoring, and rapid feedback. DevOps can reduce the time between writing code and delivering reliable software to users.

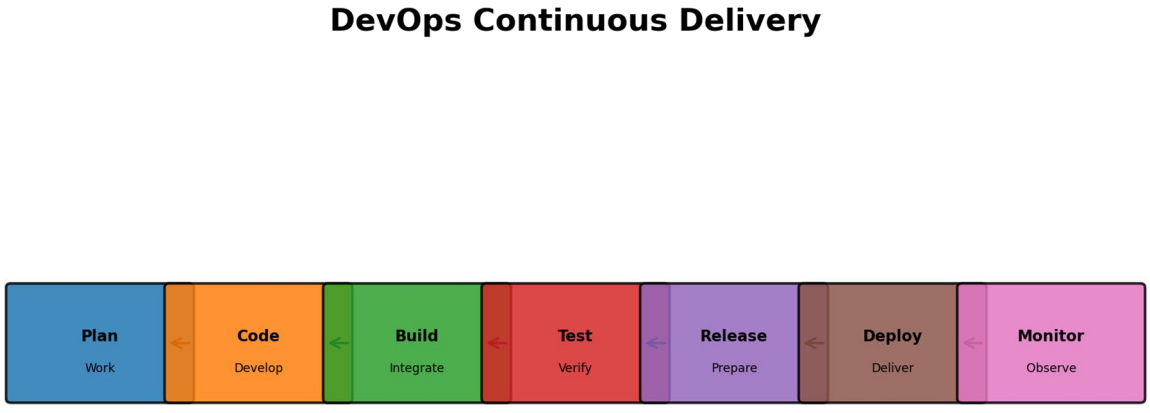


Figure 4: Three widely discussed approaches to organizing

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IX Sep 2026- Available at www.ijraset.com

IX. DEVOPS AND CONTINUOUS DELIVERY

DevOps brings development and operations practices closer together. Instead of treating software creation and software peration as completely separate activities, teams collaborate across the delivery lifecycle. Automation is commonly used for building, testing, releasing, deploying, and monitoring applications. Continuous integration encourages developers to integrate changes frequently and run automated checks. Continuous delivery keeps software in a releasable state so that validated changes can be delivered efficiently. Monitoring provides information about performance, errors, availability, and user experience after deployment.

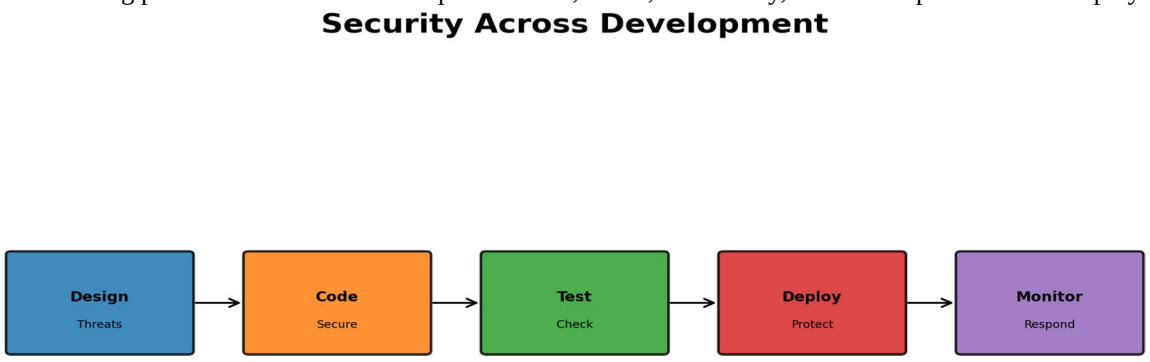


Figure 5: A simplified DevOps continuous delivery cycle.

X. CYBERSECURITY IN SOFTWARE DEVELOPMENT

Security must be considered throughout the software lifecycle. Applications may process personal information, financial records, passwords, business data, and other sensitive information. Weak security can lead to unauthorized access, data loss, service disruption, or financial damage. Secure software development includes threat analysis, secure coding, strong authentication, authorization, input validation, encryption where appropriate, dependency management, logging, security testing, and timely updates. Developers should also avoid storing sensitive information unnecessarily and should follow the security requirements relevant to their application.

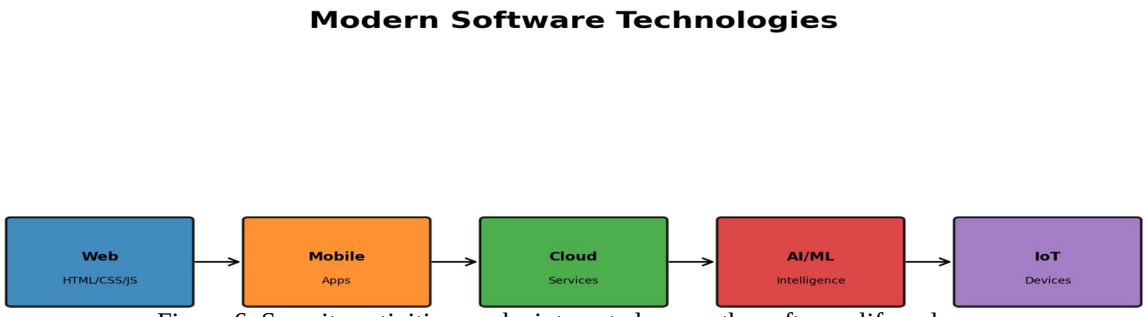


Figure 6: Security activities can be integrated across the software lifecycle.

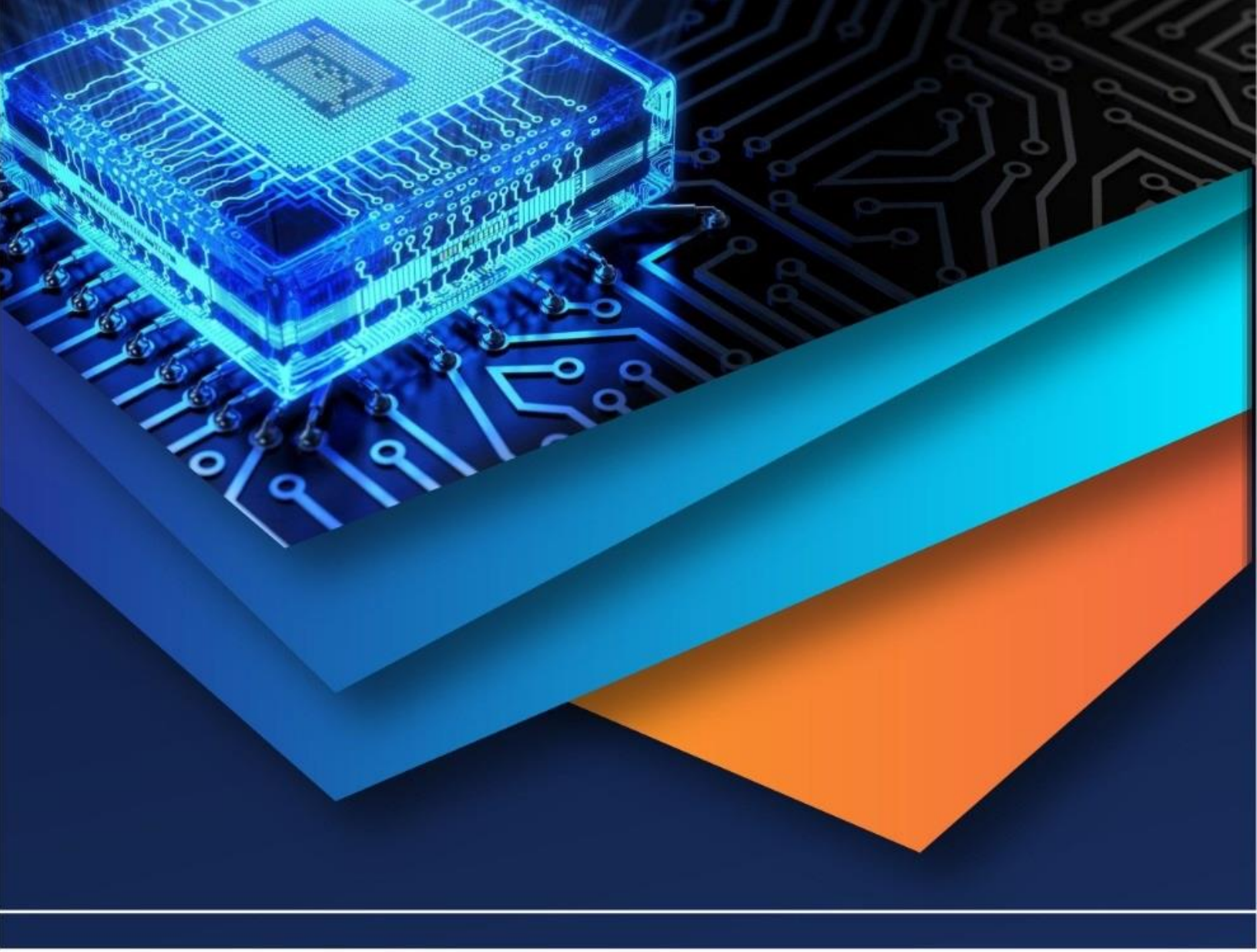
Security is most effective when it is built into the development process rather than added at the final stage. Regular testing and monitoring help teams identify new risks as technologies and threats change.

XI. CONCLUSION

Software development is a fundamental part of modern Information Technology. It is a complete engineering process rather than simply writing code. Requirement analysis, planning, design, coding, testing, deployment, security, and maintenance all contribute to the success of a software product. Different development approaches such as Waterfall, Agile, and DevOps provide teams with different ways to organize work. Modern technologies such as cloud computing, artificial intelligence, machine learning, mobile platforms, and the Internet of Things are expanding the possibilities of software development. Software applications have transformed education, banking, healthcare, business, transportation, communication, and entertainment. As digital services continue to grow, the demand for reliable, secure, efficient, and user-friendly software will also increase. Therefore, software development is an important and continuously evolving field. Students and professionals who understand programming, problem solving, software engineering principles, testing, security, and modern technologies can contribute to the development of useful solutions for real-world problems.

REFERENCES/SUGGESTED READING

[1] Software Engineering principles and Software Development Life Cycle (SDLC) concepts.
[2] Agile software development and iterative project management concepts.
[3] DevOps, continuous integration, continuous delivery, and software operations concepts.
[4] General programming language and database development documentation.
[5] General cybersecurity and secure software development practices.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY