CSE 403
Software Engineering Professional Practice

Lecture 1
Introduction

Overview of today's lecture

- ❖Course Outline
- ❖Course Objectives
- ❖End Goal
- ❖Introduction to Professional Practice in software Engineering
  - Definition of Professional practice and software engineering
  - Role and responsibilities of a software engineer
  - Career paths in software engineering
  - Overview of the software development life cycle (SDLC)
  - Accreditation, Certification and Qualification, and Licensing
- ❖ Group Project
- ❖ Challenge

Outline

1. Introduction to Professional Practice in Software Engineering
2. Ethics and Professionalism in Software Engineering
3. Software Development Methodologies
4. Project Management and Teamwork
5. Quality Assurance and Testing
6. Risk Management
7. Software Documentation and Communication
8. Industry Standards and Best Practices
9. Career Development in Software Engineering
10. Emerging Trends and Challenges
11. Capstone or Real-World Project
12. Assessment

Objectives of the course

1. Introducing students to the ethical, legal, and professional issues that arise in software engineering practice, including issues related to privacy, intellectual property, and liability.
2. Teaching students how to work effectively in teams, communicate technical information, and collaborate with stakeholders.
3. Equipping students with the skills necessary to plan, manage, and execute software projects, including requirements analysis, project planning, and risk management.
4. Developing students; knowledge of software development methodologies, including agile development, iterative development, and waterfall development.
5. Teaching students how to write effective technical documentation, including software requirements specifications, user manuals, and system documentation.
6. Providing students with practical experience in software engineering practice through project-based assignments, case studies, and guest lectures.
7. Preparing students for careers in software engineering, project management, and related fields by exposing them to industry-standard practices and technologies.

End Goal of the course

1. Understand the importance of professional and ethical responsibilities of software engineers.
2. Apply ethical principles to software engineering practices, such as privacy, security, and intellectual property.
3. Apply project management techniques, such as planning, monitoring, and controlling software projects.
4. Develop effective communication skills to interact with technical and non- technical stakeholders.
5. Apply career development strategies, such as resume writing, interviewing, and networking.
6. Understand the role of software engineering in society and its impact on individuals and organizations.
7. Work effectively in a team environment to develop and maintain software systems.
8. Evaluate software engineering processes and tools and recommend improvements to optimize software development.
9. Apply continuous learning strategies to stay current with industry trends and technologies.

1. Introduction to Professional Practice in Software Engineering

Professional practice and software engineering

- ***Professional practice*** refers to a way of conducting services to achieve certain standards or criteria in both the process of performing a service and the end product resulting from the service.
- **Software Engineering Professional Practice knowledge area (KA)** is concerned with the knowledge, skills, and attitudes software engineers must possess to practice software engineering in a professional, responsible and ethical manner.

Responsibilities of Software Engineer

1. **Designing Software Solutions**: Create software architecture and design solutions that meet requirements, ensuring scalability, reliability, and maintainability.
2. **Coding and Development**: Write clean, efficient, and maintainable code to implement the software design using appropriate programming languages.
3. **Testing and Debugging**: Conduct testing (unit, integration, etc.) to identify and fix bugs, ensuring the software works correctly and reliably.
4. **Collaboration and Communication**: Work with teams, stakeholders, and clients to gather requirements, share updates, and review code for quality and consistency.
5. **Software Maintenance**: Continuously improve and update the software by fixing bugs, optimizing performance, and adding new features.
6. **Performance Optimization**: Optimize the software's performance by improving code efficiency, managing resources, and ensuring it scales with demand.
7. **Documentation**: Maintain clear documentation for the codebase, user manuals, and design specifications, ensuring others can understand and use the software.

Responsibilities of Software Engineer Cont.

8. **Security and Compliance**: Write secure code, perform security testing, and ensure the software adheres to relevant regulations and industry standards.
9. **Project Management**: Estimate task timelines, manage progress, and participate in project management processes like sprint planning and task tracking.
10. **Continuous Learning:** Keep skills updated by learning new tools, programming languages, and technologies, and applying them to improve software.
11. **Mentorship and Leadership (for senior roles)**: Guide and mentor junior developers, offering support, reviewing code, and taking leadership roles in projects and teams.
12. **Client and Stakeholder Interaction**: This involves having meetings to gather client requirements, provide support, and incorporate feedback to improve or adjust the software.
13. **Innovation and Problem-Solving**: Tackle complex problems with creative and innovative solutions, and contribute new ideas or approaches to improve software.

Roles of Software Engineer

1. **Software Architect**: Responsible for designing the overall structure of a software system, choosing the right technologies, and ensuring scalability, maintainability, and performance. They often make high-level decisions about the system architecture and ensure the technical integrity of the project.
2. **Full-Stack Developer**: A developer who works on both the front-end (user interface) and back-end (server-side logic) of an application, handling both sides of the software stack.
3. **Back-End Developer**: Specializes in building and maintaining the server-side components, databases, and APIs that support the functionality of the application.
4. **Front-End Developer:** Focuses on building the user interface and experience of the application, working with technologies like HTML, CSS, JavaScript, and front-end frameworks.
5. **DevOps Engineer**: Combines software development and IT operations skills to automate, monitor, and optimize software delivery and infrastructure. They work to streamline deployment, manage servers, and ensure continuous integration and continuous deployment (CI/CD).
6. **Quality Assurance (QA) Engineer**: Focuses on testing software to ensure it meets quality standards. They write test cases, conduct manual and automated testing, and help identify and fix bugs before release.
7. **Test Automation Engineer:** A specialized QA engineer who focuses on creating automated tests to speed up the testing process and ensure the software remains bug-free as it evolves.

Roles of Software Engineer Cont.

8. **Software Security Engineer**: Focuses on ensuring that the software is secure from vulnerabilities and potential threats. They design, implement, and test security measures to protect the application and its data.
9. **Mobile Developer**: Specializes in developing applications for mobile platforms like iOS and Android, using languages like Swift, Kotlin, or cross-platform tools like React Native or Flutter.
10. **Cloud Engineer**: Specializes in developing, managing, and deploying applications and services in the cloud. They work with cloud providers like AWS, Azure, or Google Cloud.
11. **UX/UI Designer:** While not a traditional software engineering role, UX/UI designers work closely with software engineers to create user-friendly and visually appealing interfaces. They are responsible for the design and usability of the application.
12. **Site Reliability Engineer (SRE)**: A specialized role that focuses on maintaining the reliability and uptime of a system by combining engineering practices and operational tasks. SREs automate manual tasks, ensure high availability, and monitor system performance.
13. **Product Manager (PM)**: Though not an engineer, the product manager plays a critical role in a software development team. They define the product vision, requirements, and prioritize features. They work closely with engineers to bring the product to life.

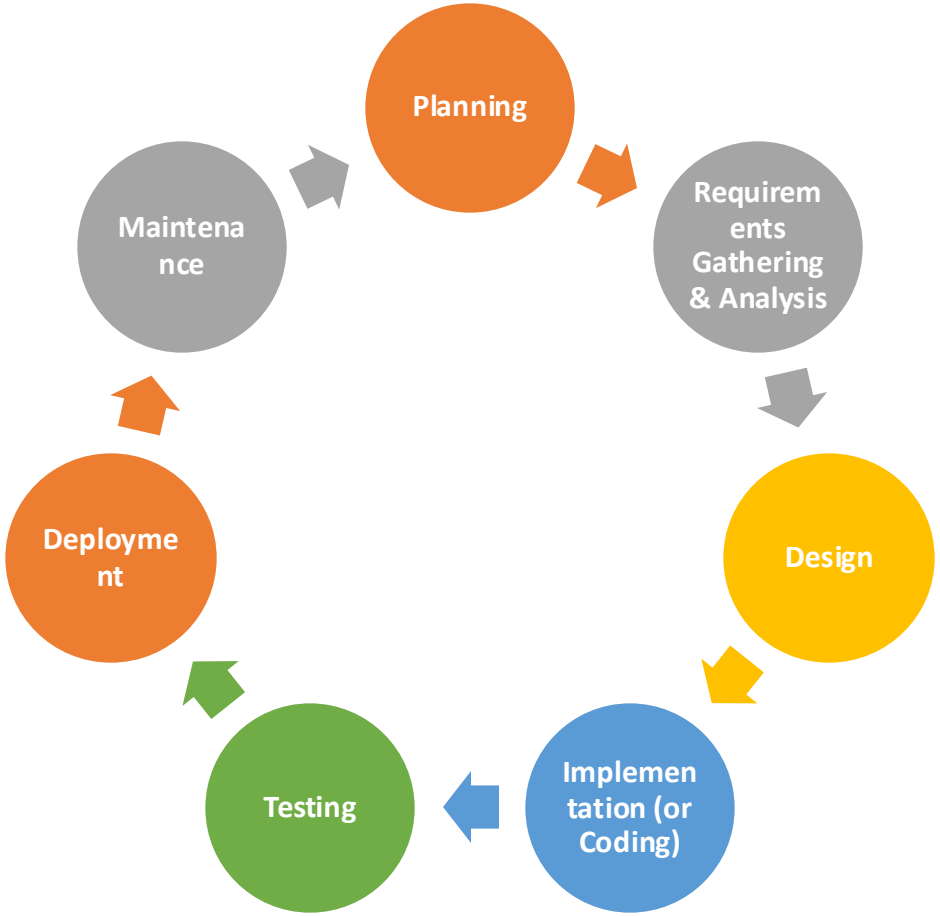
Overview of the software development life cycle (SDLC)

- The **software development life cycle (SDLC)** is a step-by-step process that development teams use to create high-quality, cost-effective and secure software

- The Software Development Life Cycle (SDLC) is a structured process used for developing software applications. The SDLC ensures that software is developed efficiently, meets user requirements, and maintains quality. It also helps manage costs, timelines, and resources throughout the development process. It encompasses several phases that guide the development team from the initial concept to the final deployment and maintenance of the software. Below is a detailed overview of the stages:

-

SDLC


Planning
Requirements Gathering & Analysis
Design
Implementation (or Coding)
Testing
Deployment
Maintenance

SDLC

1. **Planning**: Define the project scope, resources, timeline, and potential risks.
2. **Requirements Gathering & Analysis**: Collect and analyse what the software needs to do and how it should perform.
3. **Design**: Create the system architecture and detailed design for how the software will be built.
4. **Implementation (Coding):** Developers write the actual code based on the design specifications.
5. **Testing**: The software is tested to find and fix bugs, and ensure it meets all requirements.
6. **Deployment**: The completed software is released to the live environment for user access.
7. **Maintenance**: Ongoing updates, bug fixes, and improvements are made after deployment.

Accreditation, Certification, Qualification and Licensing

Accreditation:

Accreditation is a voluntary process through which an external organization assesses and recognizes the quality and standards of educational institutions, programs, or services. It involves an evaluation of factors such as curriculum, faculty qualifications, student outcomes, facilities, and resources.

Certification:

Certification is a process by which an individual's knowledge, skills, or competencies are evaluated and validated against established standards or criteria set by a certifying body. It often involves passing an examination or completing specific requirements.

Licensing:

Licensing is a legal process by which a government authority grants permission to an individual or organization to engage in a regulated profession, occupation, or activity. It typically involves meeting specific requirements and adhering to regulations set by the licensing authority.

Qualification

a condition that must be fulfilled before a right can be acquired; an official requirement

Group Project

See download link below to download resources for this course including assignments.

https://drive.google.com/drive/folders/1uA0lTL9VQIfCDQVejd6HY_P4peBeu1p4?usp=share_link

Your Challenge this week

- You should be able to describe "professional practice" in software engineering, and why is it important for ensuring software quality and societal well-being?
- Be able to describe at least 6 responsibilities of a software engineer within a team setting and explain how each contributes to the success of a software project.
- You should be able to explain the roles of accreditation, certification, and licensing in the professional development of a software engineer.
- Also, be able to Identify and describe 5 distinct career paths in software engineering, highlighting the key responsibilities of each.

END OF LECTURE 1