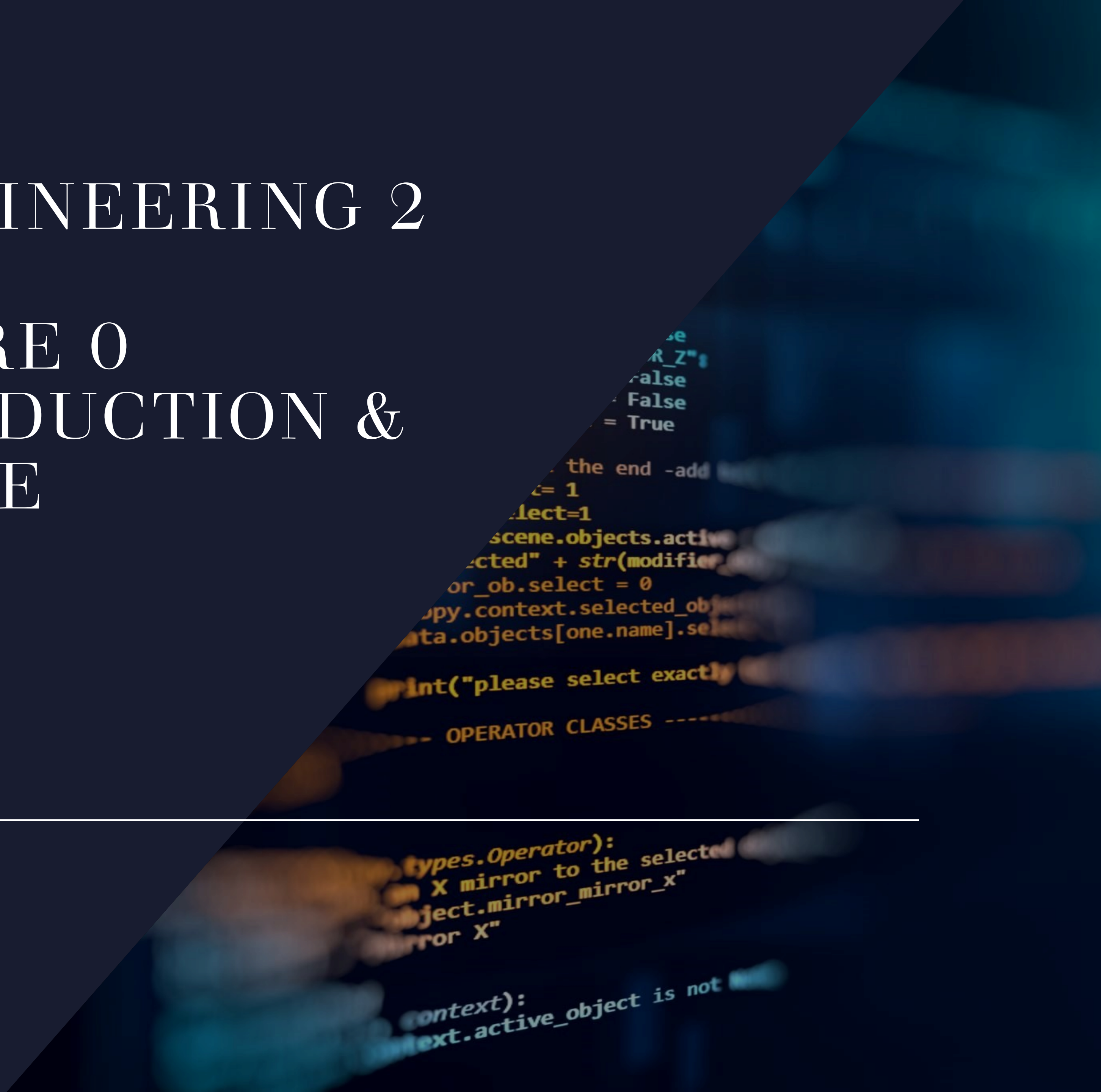
SOFTWARE ENGINEERING 2

LECTURE 0
COURSE INTRODUCTION & PLANE

Dr. Ahmed Hesham Mostafa



Course Aims and Objectives

- Writing maintainable, readable code.
- Learn the best techniques for writing understandable code.
- Learn about key principles, rules, and concepts that allow you to write clean code.
- Learn with hands-on examples and "bad to good code" transformations.
- Enrich your design skills and become an overall better programmer.

Course Aims and Objectives

Learn to confidently write well-designed code using concepts that are widely recognized in the community.

Learn the theory of SOLID Principles.

SOLID principles of software design & Be a Better developer by learning how to write quality code.

Ensure that dependencies are well managed so that the code remains flexible, robust, and reusable.

Course Aims and Objectives

- Understand costs and trade-offs associated with object-oriented design and get a leg up in ensuring that your codebase is much cleaner.
- Gain a deep understanding of SOLID principles.
- Learn from examples of clean architecture and SOLID code.
- Apply SOLID principles in order to write quality code, as a software engineer.

Course Aims and Objectives

- Obtain a solid understanding of what design patterns are, how to implement them, and WHY you should!
- Understand the Design Pattern concepts set out by the Gang of Four.
- The students will be able to understand how to look for problems in software application development, identify the context, and choose the appropriate and elegant solution.
- Understand the concepts through a number of samples, examples, demos, hands-on labs, out-of-the-box examples, etc.
- Get an in-depth understanding of the most useful design patterns.
- Recognize and apply design patterns.

Textbooks

Text#1	Robert C. Martin, “Clean Code: A Handbook of Agile Software Craftsmanship”.
Text#2	Robert C. Martin, “Clean Architecture: A Craftsman's Guide to Software Structure and Design”.
Text#3	Rohit Joshi, “Java Design Patterns”.
Text#4	Gamma, Erich, et al. "Design Patterns: Elements of Reusable Object-Oriented Software”.
Text#5	Freeman, E., Robson, E., Bates, B., & Sierra, K., “Head First Design Patterns: Building Extensible and Maintainable Object-Oriented Software”.

Topic covered

Part -1 -Write clean Code.

Part – 2 -SOLID: The First 5 Principles of Object-Oriented Design.

Part -3 -Design Pattern.

Topic covered - Part -1 -Write clean Code from Text#1

- Chapter 2: Meaningful Names.
- Chapter 3: Functions
- Chapter 4: Comments
- Chapter 5: Formatting
- Chapter 6: Objects and Data Structures
- Chapter 7: Error Handling
- Testing (will not be covered in this course)
- Chapter 17: Smells and Heuristics

Topic covered - Part – 2 - SOLID: The First 5 Principles of Object-Oriented Design from Text#2

Chapter 7	SRP: The Single Responsibility Principle
Chapter 8	OCP: The Open-Closed Principle
Chapter 9	LSP: The Liskov Substitution Principle
Chapter 10	ISP: The Interface Segregation Principle
Chapter 11	DIP: The Dependency Inversion Principle

Topic covered - Part -3 - Design Pattern from Texts#3#4#5

Creational Design Pattern (Factory Method, Abstract Factory, Builder, Singleton, Object Pool, and Prototype).

Structural Design Pattern (Adapter, Bridge, Composite, Decorator, Facade, Flyweight, Private Class Data, and Proxy).

Behavioral Design Pattern (Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, Null Object, Observer, State, Strategy, Template method, Visitor).

There are also two types of patterns - idioms and architectural patterns. But we will not explore them in this course.

In this course, we will not study all 23 Design patterns, but we will study well-known design patterns from each type.

Grade Policy

Final Exam 60%

Midterm Exam 20%

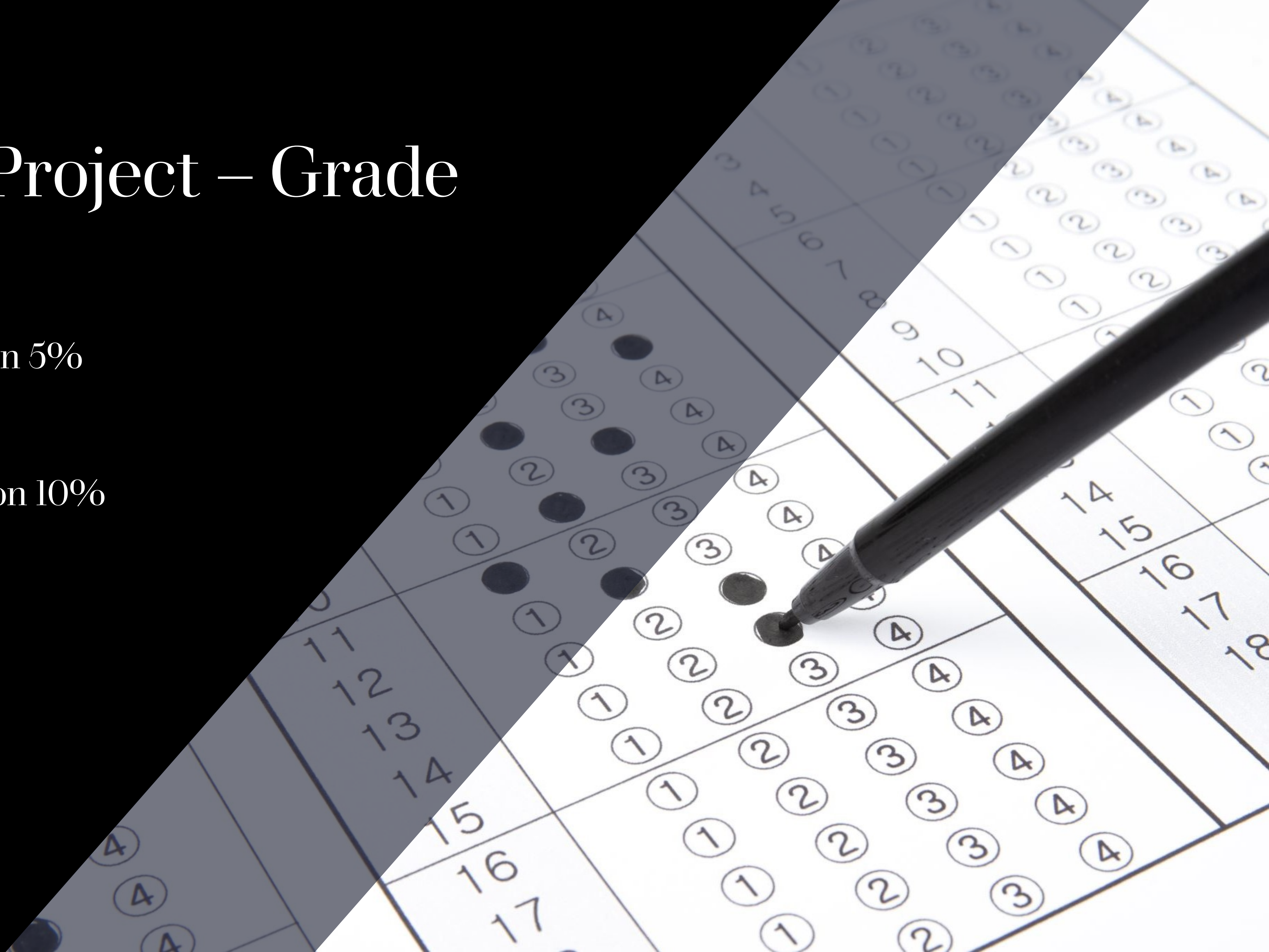
Project 20%

Course Project

- Each Team should register their project idea before Week 4.
- Project must be implemented via any OOP languages (Java, C#, Python(OOP), PHP(OOP)).
- Each Project should contain three separated modules (Model(DB), View(GUI), Controller (Logic)).
- Project can be based on Desktop, Web, or Mobile Application.
- Each Team should submit project documentation based on (Software Engineering l).
- Each team must submit the Project via GitHub.

Course Project – Grade Policy

- Documentation 5%
- GitHub 5%
- Implementation 10%



THANKS
SEE U NEXT LECTURE