### Lesson 7: Iterators and Closures

* **Reading**: “The Rust Programming Language,” Chapter 13, on iterators, closures, and their use in Rust.
* **Assignments**: Develop a utility that applies a series of transformations and filters to a collection of items using iterators and closures, demonstrating the lazy evaluation property.
* **Preflight**: Experiment with simple iterator examples to get familiar with the methods like map, filter, collect, etc.
* **Lesson Goals**:
  + Master the use of iterators to create efficient, composable operations on collections.
  + Understand closures and their role in Rust’s functional programming features.
  + Explore the concept of lazy evaluation in iterators.
* **Motivation**: Iterators and closures are fundamental to writing idiomatic, concise, and efficient Rust code, especially in data processing and event-driven programming.
* **Lecture**:
  + Deep dive into Rust’s iterator types and the iterator trait.
  + Closure syntax, capturing environment variables, and their use cases.
  + Practical examples and patterns for combining iterators and closures.
* **Lab**: In a series of exercises, students will transform data collections through chaining iterator methods, utilizing closures for custom logic.