🔐 **Task 6: Enigma Machine**

**🧩 Welcome to the Broken Enigma Machine Challenge**

You're provided with a **simplified implementation** of the classic **Enigma cipher machine**. The code is mostly functional but contains a **bug** that affects the correctness of **encryption or decryption**.

**🛠️ Enigma Machine Overview**

The Enigma machine is a **rotor-based cipher device** historically used for secure communication. This implementation includes:

* Multiple rotors with configurable wiring and stepping
* A plugboard for letter swaps
* Reflector logic
* Basic CLI interface for encrypting and decrypting messages

📄 For a detailed explanation of how the Enigma Machine CLI works, refer to [enigma.js](https://github.com/VentionComms/AI-Challenge/blob/main/main/task_6/README.md) and [README.md.](https://ventionteamsinc-my.sharepoint.com/:t:/g/personal/mikita_sauko_ventionteams_com/EerOgYBEcytOltoeyI3IG0sBTy2g3PoGqb5Bp_u_ZARkhA?e=1yTZ6t)

**📚 Theory**

**🧠 AI Tool: Cursor IDE**

**Cursor** is an AI-powered development environment that helps you **write, analyze, and debug code** more efficiently. It integrates intelligent assistance directly into your workflow.

Key capabilities:

* Identifies bugs before execution
* Explains **why** an issue occurs
* Summarizes complex logic
* Suggests fixes and improvements
* Helps generate tests and documentation

**🔍 AI Techniques Explored**

1. **Code Comprehension**  
   Prompt the AI to explain rotors, plugboard, reflector, and the app’s structure.
2. **Bug Diagnosis & Resolution**  
   Use natural language to describe the issue and have the AI assist in debugging.
3. **Test Development**  
   Guide the AI to generate unit tests covering key cases and configurations.
4. **Documentation**  
   Let the AI help you write a short explanation of the bug and your solution.

**🎯 Task**

Your challenge is to:

* Analyze the code in enigma.js
* Identify the bug affecting encryption/decryption
* Fix the bug and ensure correct functionality

**✅ Objectives**

1. **Find and Fix the Bug**  
   Determine the cause and implement a correction.
2. **Explain the Issue**  
   Add a brief explanation via code comments or a separate write-up.
3. **Add Unit Tests**  
   Ensure core functionality is tested.  
   *Note: You may use a different programming language if preferred.*

**📌 Requirements**

* The fixed code must **correctly encrypt and decrypt** according to Enigma rules
* Unit tests must:
  + Cover all core logic
  + Achieve **at least 60% test coverage**