Design Decisions

ECE 651   Team #6

File Export Service
For the creation of different types of files (e.g., JSON, XML), we employed the **Factory Pattern**. This pattern is used in the ExporterFactory class to instantiate appropriate exporter objects (JSONExporter, XMLExporter, or CustomExporter) based on the required export format. The Factory Pattern allows for easy extension of export formats without modifying existing code, adhering to the Open/Closed Principle.

File Management
The design of the FIleHandler class follows the **Single Responsibility Principle**. The FileHandler class is solely responsible for file operations, including reading, writing, and managing course and student data files. This separation of concerns ensures that the file handling logic is centralized, making the system easier to maintain and extend.

Password encryption
We have independent encryption and decryption for password in the account.
original password of an account is not accessible by anything outside the account instance itself.

Account Interface
The design of account follows the **Open/Closed** Principle. Although only professors can log in to this system, we leave the option open that there will be other type of account, in case there will be other type in the future.

TextPlayer
Similar to Battleship, we create a TextPlayer to handle information display and user interaction. TextPlayer is responsible for all the functionalities inside a course, including taking attendance, changing attendance record, adding and removing student, changing student's display name, exporting record, and so on. To implement TextPlayer class, we use **Observer Pattern** to ensure that every time when we change a student's attendance record, the class EmailNotification will be triggered to send an notification to the student.

Notification Interface
The software needs multiple ways to send notifications to students and professors, here we apply **Abstract Factory Pattern** and set up the Notification Interface. That allows us to add more ways to send notifications in the future without modifying existing code.