A The AlterOffice Assignment

Unity Weapon System(Elements)

1. WeaponData

- Stores static data for each weapon, including stats, audio, VFX, and icon.
- Centralizes configuration for easy modification without code changes.
- Implements ScriptableObject for flexible data management in Unity.

---

2. Weapon (Abstract Class)

- Represents the base class for all weapons, managing weapon data and attributes.
- Delegates firing and reloading behaviors to an assigned `IWeaponBehavior`.
- Provides properties and methods for firing and reloading across weapon types.

---

3. WeaponType (Enum)

- Enumerates weapon types (e.g., Fire, Ice, Electric) for easy identification.
- Facilitates weapon switching and dynamic behavior assignment in `WeaponManager`.

---

4. IWeaponBehavior (Interface)

- Defines methods for firing and reloading, ensuring consistent interfaces across behaviors.
- Encourages flexibility and expansion by allowing custom implementations for each weapon type.

---

5. FireWeapon, IceWeapon, ElectricWeapon (Classes)

- Each class represents a specific weapon type, inheriting from `Weapon`.
- Each weapon type has unique properties set by its specific `WeaponData` and `IWeaponBehavior`.

- Simplifies weapon creation by adhering to common functionality in the base `Weapon` class.

---

6. FireWeaponBehavior, IceWeaponBehavior, ElectricWeaponBehavior (Classes)

- Implements `IWeaponBehavior` interface to handle unique firing and reloading logic.
- Encapsulates distinct logic for each weapon's behavior, promoting modularity.
- Allows specific weapon types to easily expand or modify their behavior independently.

---

7. WeaponManager

- Controls initialization, switching, and firing of weapons during gameplay.
- Manages the active weapon model and VFX based on the current weapon.
- Delegates UI updates to `UIManager` for displaying weapon details.

---

8. WeaponFactory

- Factory pattern class that creates instances of weapon types (Fire, Ice, Electric).
- Determines the specific `Weapon` subclass to instantiate based on `WeaponData`.
- Enables easy addition of new weapon types without modifying existing code.

---

9. UIManager

- Manages updates to UI elements (icon, name, description) for the current weapon.
- Integrates with `WeaponManager` to display weapon data during weapon switches.
- Enhances user experience by providing real-time feedback on weapon changes.

---

10. AudioManager

- Singleton class to handle audio playback for firing, reloading, and empty magazine sounds.
- Simplifies audio management by centralizing sound effects into one location.
- Allows easy extension for future audio additions across weapons and game events.

---

11. WeaponHelper

- Static helper class to handle shared firing and reloading logic for weapons.
- Manages VFX playback and sound effects, delegating actual playback to `AudioManager`.
- Ensures firing and reloading processes are consistent across different weapon types.

---

12. CoroutineRunner

- Singleton MonoBehaviour for running coroutines from non-MonoBehaviour classes.
- Supports `WeaponHelper` by executing reload coroutines when needed.
- Ensures reload timing consistency even when called from static or non-MonoBehaviour scripts.