**Bonus Assignment**

**Group 13 - Tara Vulakovich, Kolby Sarson, Anthony Azar, Brandon Ferrari, Xavier Nunes**

Research and identify bad coding examples that illustrate violation to one or more design principles we covered in the class today.

|  |  |
| --- | --- |
| **Single Responsibility Principle** | **Defintion:**  An entity (component, service, module) should only focus on delivering a specific function or cohesive functions.  Each class should be responsible for a specific function or service. |
| class Book {  function getTitle() {  return "A Great Book";  }  function getAuthor() {  return "John Doe";  }  function turnPage() {  // pointer to next page  }  function getCurrentPage() {  return "current page content";  }  **function getLocation() {**  // returns the position in the library  // ie. shelf number & room number  }  } | This example is for usage within a library, searching and examining books.  Function getLocation is not relative specifically to the Book.  Another class to locate books in the library should be used.  Each function is relative to the book objects, whereas the location is used throughout the entire library.  From:  https://code.tutsplus.com/tutorials/solid-part-1-the-single-responsibility-principle--net-36074 |
| **Do Not Repeat Yourself Principle** | **Defintion:**  Duplication in logic should be eliminated via abstraction. Concepts related to business logic should not be duplicated.  Avoid adding additional and unnecessary code to a codebase. |
| if (isEmpty()) {   if (canReload()) {   reload()   ***spawnProjectile(***)   } else {   playEmptySound()   }  } else {    ***spawnProjectile()***  } | This code is used for shooting gun, where the if statements can be easily condensed without calling spawnProjectile() twice.  if (isEmpty() && !canReload() {   playEmptySound()  } else {   if (isEmpty() && canReload()) {   reload()   }   spawnProjectile()  }  From https://www.reddit.com/r/gamedev/comments/3f16u8/an\_important\_game\_programming\_principle\_dont/ |