Design Issues

Initially in our project we had bloaters code smell, which implies that the classes were too large and there were data clumps. We had dispensable code smells which were lazy classes and dead code. Lastly, we had couplers code smell which means middleman class implementations.

To fix the bloating code smell, we divided our GUI and data into separate classes, each with one functionality. To get rid of dispensable and couplers code smells, we removed unnecessary classes and combined features such as SQL and GUI into one class.

Previously our code had unlinked classes which only provided functionality from the developer's perspective. They're existed lazy classes which had functionality that could not be implemented.

Some of the refracting techniques used were:

1- Moving features between objects
2- Organizing Data
3- Simplifying method calls.

Currently our program only contains linked classes that provide functionality to both the client and the developer. The lazy classes were removed. There is linking between the backend classes and the GUI to ensure the data can be displayed.