Proyecto 1:

El primer proyecto fue un gran reto ya que debíamos crearlo en un lenguaje nuevo. Toda la lógica de la programación orientada a objetos puede ser difícil de entender.

La decisión de tener administradores de cada elemento fue muy buena ya que permitía organizarlo mejor (esta forma no se cambió en los proyectos siguientes). El enrutador principal se encargaba.

Una mala decisión fue no llevar un conteo de ventas de cada producto del restaurante. Se debían ver las facturas e ir sumando.

La base de datos fue lo más difícil de crear. Usar csv fue una buena decisión ya que era un formato que ya se había trabajo anteriormente y no teníamos que aprender algo desde cero. La persistencia de la información está en una clase traductora de objeto a archivos. Consideramos que fue una buena decisión ya que solo esa responsabilidad.

La representación de la consola no es clara ya que solo se podían usar caracteres (aun no teníamos interfaces graficas), y se hacía difícil de entender.

Proyecto 2:

En el proyecto 2 se crearon las interfaces gráficas en vez de por consola, principalmente no hubo ningún problema con ello debido a que solo tuvimos que mover los elementos de las interfaces antiguas y modificarlas para que fueran usables en las nuevas.

Por otro lado, en términos de realizar cambios y correcciones respecto al proyecto 1 hubo algunas complicaciones con los datos, puesto que la carga de ellos o la lectura para habilitar opciones en la interfaz generaron algunos errores. Sin embargo, se logró completar con relativa satisfacción los requerimientos solicitados para un funcionamiento de la aplicación.

Principalmente las decisiones más asertivas se pueden destacar en 2, la primera es las diferentes iteraciones realizadas en el proyecto anterior que permitieron evitar errores o acoplamiento entre las diferentes clases del proyecto. Por otro lado, tenemos el haber hecho uso de diferentes clases en abundancia que, si bien puede resultar un poco engorroso explorar a través de ellas, permitió que el programa pudiera extenderse para cumplir los requerimientos sin afectar una gran cantidad de clases. No obstante, una decisión que pudo ser problemática es el hecho de elementos visuales como son los gráficos tuvieron problemas a la hora de generarse, y esto puede deberse a una falta de buena unión entre los datos con el grafico.

Principalmente en este proyecto más allá del carácter gráfico, el problema fue que muchos de los problemas que dábamos por saltados en el proyecto 1 tenían un efecto negativo a la hora de realizar cambios en este; y también que al momento de repasar entradas y/u mostrar información observamos disparidades de lo que debería ser el proyecto a lo que era. Si bien hubo esos problemas, se logró solventar para poder un programa funcional. Además, creemos que en este punto pudo ser útil el manejo de errores, el cual no había sido muy implementado, lo cual acarreo a muchas de estas situaciones. En conclusión, el proyecto pudo tener mejorías con conocimientos posteriores, sin embargo, tiene una gran calidad debido a prácticas positivas como las iteraciones además de documentación para poder arreglarlos en proyectos posteriores.

Proyecto 3:

Durante el diseño e implementación de los requerimientos del proyecto 3 nos surgieron varios retos relacionados a conceptos vistos en clase y a tareas que no habíamos implementado antes. Los principales fueron:

La carga dinámica de clases, entender el concepto y diseñar el lugar donde debía ser aplicado para optimizar la implementación de nuevas pasarelas de pagos nos tomó un tiempo considerable y requirió de iteración y re-evaluación de las decisiones que íbamos tomando respecto a la forma de procesar pagos. Principalmente el uso del método Class.forName(nombreClase), la carga de las dinámicamente y la posterior creación de instancias de esas clases fueron los elementos que nos retaron para entender en qué casos aplicados en la vida real podemos usar estas técnicas para asegurar el menor acoplamiento posible dentro de nuestros proyectos.

Crear una nueva interfaz de usuario fue algo retador también. En este punto del proyecto el reto era desarrollar una nueva aplicación con una interfaz gráfica que se ejecutara independientemente del resto del Property Management System (PMS). La complejidad aquí radicó en el diseño y desarrollo de una interfaz intuitiva, fácil de usar y que cumpla con todos los requisitos funcionales establecidos.

Como parte del uso de nuevas librerías, el proyecto menciona la posibilidad de utilizar librerías como XChart o JFreeChart para generar reportes y gráficas. El reto en este caso fue familiarizarse con estas librerías, comprender su documentación y utilizarlas correctamente en la implementación del sistema. Además, es importante garantizar la integración adecuada de estas librerías con el resto del código y asegurarse de que las gráficas generadas cumplan con los requisitos establecidos en los reportes.

Además, las pruebas automáticas no solo de integración sobre las funcionalidades relacionadas con la creación de reservas sino también de pruebas unitarias sobre las funcionalidades para la carga de archivos nos retó a diseñar pruebas efectivas, consistentes y que puedan detectar posibles errores o fallos en el sistema. Aunque estas herramientas las aplicamos en clase varias veces, no deja de ser un reto traer ese conocimiento a una base de código extensa y cambiante como lo es el PMS que hemos venido construyendo a lo largo del semestre. Otro aspecto retador fue entender los errores que nos salían en varios casos de las pruebas, no a todos nos salía lo mismo cuando lo corríamos en nuestros respectivos computadores, pero al entender mejor la implementación de las pruebas pudimos superar y dejar funcionales las clases que las contienen.

Por último, re-evaluar las decisiones que habíamos tomado en el proyecto anterior fue muy necesario a la hora de implementar las pasarelas de pago porque sabemos que a medida que se realizan cambios y se agregan nuevas funcionalidades, es importante mantener un código limpio y estructurado que sea fácil de entender, mantener y ampliar en el futuro. Así que hubo cambios que tuvimos que hacer para poder seguir buenas prácticas de programación, modularizar el código en componentes reutilizables, aplicar patrones de diseño y documentar adecuadamente el código.

En conclusión, el punto 3 del proyecto presenta diversos retos que involucran la metaprogramación o carga dinámica de clases, la creación de una nueva interfaz para el usuario, el uso de nuevas librerías y los retos asociados al testing. Superar estos retos representó para nosotros un aprendizaje sobre el análisis cuidadoso que se debe hacer antes de la implementación de cualquier parte del programa, un diseño adecuado y una implementación sólida para garantizar el correcto funcionamiento del programa.