Project Summary

Our project is a bank system that includes four packages including a Booking package which includes an abstract booking class which has most of the information for a booking except Branch id in BookingInfo which extends Booking and the staff member running the meeting which is an object given to booking by Staff the object name is banker. Along with an enum BookingType that stores the status of the booking (PENDING, CONFIRMED, CANCELLED)

Other Packages include Transaction, Customer and Insurance. Transaction is a package covering the information required to complete a bank transaction including an abstract transactions class including the basic information required for a transaction and then have two other class extend transaction to specific the type of transaction the Withdraw and Deposit classes which both use an interface to calculate the amount in the account after the transaction. Along with these classes, there’s a separate class Loans to include use of error handling and enums with a Negative Number Exception and LoanType (None, Small Loan, Large Loan, Mortgage)

Customer package includes an abstract Person class with is an object in every other package as each booking transaction and claim requires the customers’ information which is also the name of the class that extends person. On top of that there’s a customer id class which is an object in person and CustomerType enum (Individual, Company and Couple)

The final package is Insurance with covers making an insurance claim at the bank with an Insurance abstract class storing most of the standard policy information and the remainder of the claim is stored in claim which extends insurance class. Included with these is a Message Exception for coverage in Insurance and an enum (Life, Health, Travel)

13th November 2024

* We decided on the project idea which was a bank management system using java
* And decide to spit it down the middle and do two packages each for a total 4 packages

20th November 2024

* Started working on the UML diagram
* Decided the names and theme of the 4 packages Booking, Transaction, Customers, Insurance

27th November 2024

* Shae defines the classes for each package in the UML, and finalizes design e.g. extend arrows and associations arrows
* Matthew codes the Booking and Transactions packages and classes
* Shae codes the Customers and Insurance packages and classes

4th December 2024

* Matthew finishes the team’s document
* Matthew Changes the UML to fully match his code
* Shae Changes the UML to fully match his code
* Review the code before submission

Matthew Code Location:

* The Classes are Booking, Staff, BookingInfo in Booking & Transaction, Withdraw, Deposit, Loan in Transaction
* Withdraw and Deposit inherit from Transaction and BookingInfo inherit from Booking
* Staff is an object in Booking called banker
* Booking uses blank constructor and staff, deposit, withdraw uses a normal constructor, transaction has a constructor to allow for chaining
* Constructor chaining is used for inheritance in withdraw and deposit from Transaction
* currentAmount is public, and there’s a mix of private and protected variables used throughout both packages
* Every private variable has a get/setter as well as protected date in booking to allow it to be added even with a blank constructor
* The packages are called Bookings and Transactions
* Enum are in both Booking and Transaction Package
* Transaction and Booking class is abstract
* BookingInfo is a final class
* NegativeNumExepection is the exception class and is applied to loan if the number is less than 0
* Interface is the CalAmount class and is applied to withdraw and deposit for calculation

Shae Code Location:

* The Classes are Person, ContactINFO, CustomerID in Customer, Insurance, Claim.
* Insurance inherits Contact info
* Person, ContactInfo and CustomerID have a chain constructor while Claims has a single constructor
* There’s a mix of private and protected variables used throughout both packages
* The packages are called Insurance and Customer
* Enum are in both Insurance and Customer Package
* Person and Insurance class is abstract
* Tax is a Static
* InvalidInsuranceExepection is the exception class and is applied to Insurance if the Coverage is less than null