**Project Overview**

(Anthony and Owyn)

This Employee\_Mangement\_System project is meant to be a simple Employee system. It has simple functionalities like creating a employee, setting up salarys, bounses and, hours.

**Class Descriptions**

AbstractEmployee(Abstract and implments Employee):

* Attributes
  + String Name
  + Double baseSalary
  + JobType jobType
* Methods
  + DisplayDetails(): Prints out the employees information
  + GetName(): returns Name

Employee(Interface):

* Methods signatures
  + CalculateSalary():
  + DisplayDetails():

FullTimeEmployee(Extends AbstractEmployee):

* Attributes
  + Double Bonus
* Methods
  + CalculateSalary(): Returns an updated salary
  + SetBonus(double bonus): sets a bonus

InternEmployee(Extends AbstractEmployee):

* Methods
  + CalculateSalary(): Returns an updated salary

PartTimeEmployee(Extends AbstractEmployee):

* Attributes
  + Double hourly\_wage
  + Double hours
* Methods
  + CalculateSalary(): Returns an updated salary
  + SetHours(double hours): sets hours then updates salary

JobType(ENUM):

* Attributes
  + FULL\_TIME
  + PART\_TIME
  + INTERN

Main():

* Attributes
  + ArrayList employees
  + Scanner Scanner
  + Int Choice
  + Int SubChoice
  + String name
* Main():
  + Runs all of the Menu and options for the users to manage and create employees

**Implementation Details**

The central class in this project is the AbstractEmployee class. Inside of the AbstractEmployee class is the attributes used to make other employees in it becasue it is the super class to the Specific type of employee classes. The other Employee class use entend off of the abstractemployee class with some have their own unique methods like FullTimeEmployee having a setbonus method which the other employee methods don't have. All of theess classes are brought together in the main class where it creates a menu for the user to be able to create employees and manage them with ease.

**Test Cases**

|  |  |
| --- | --- |
| **Action** | **Expected Output** |
| Create a full-time employee with a bonus | Salary = base + bonus |
| Create a part-time employee with hourly pay | Salary = hours × rate |
| Create an intern with a fixed stipend | Salary = stipend amount |
| Store multiple employees in a list | List holds all employees |
| Retrieve and display employee details | Name, Job Type, Salary appear correctly |
| Attempt to set a negative salary | Sets to minimum valid value |
| Remove an employee from the list | Employee is deleted successfully |
| Handle incorrect input for job type | Displays an error message |

**Challenges and Solutions**

One of the challenges we encountered during the lab was differentiating the different classes of employee when managing them. We needed this because different types of employees had unique things that could be changed such as bonuses, or hours worked. Originally when we tried to use getClass() we would get an error for comparing an Object with a String but after a couple minutes of messing around we learned we could use getClass().getSimpleName() to convert to a String allowing us to check what type of employee someone was.