CRC Cards for our Payroll System:

Table of contents:

1.0 What is a CRC?

A CRC (Class-responsibility Collaboration) is a diagramming method to represent the purpose of a class and the other systems within the project that it collaborates with. This allows software development teams working on certain aspects of a software development project to understand to a greater extent the purpose of the class they are implementing within the project.

For our outline of the CRCS for this system we will section the CRC diagrams based off the packages that contain them.

2.0 Package 1: File Handler

The file handler package (Developed by Ben Bastianelli) contains 3 classes that can be and were imported into many other classes. These 3 classes were CSVReader, CSVWriter and FolderReader.

CSVReader and CSVWriter were relatively self-explanatory being classes that allowed you to set up a reader or writer object to a csv file and then read and write from said csv file while also including useful re-usable methods to allow for specific data searches within these csv files. These classes massively reduced development time as CSV reading and writing implementation was handled by 1 class generically and didn't have to be implemented manually in each class.

FolderReader was a more niche method mainly used by the payment processing classes that allowed you to read through a specified folder and return all file names contained within it.

Below are the CRC cards for each of these classes explaining their functionality in greater detail

2.1 CSVReader:

Class Name: CSVReader	
Responsibilities	*Collaboration*
1. Reading data from a csv file at a given path name 2. Reading and returning data from a specific column of the csv file 3. Returning only unique values in that column 4. Sorting the data in a csv by a given key 5. Finding and returning a specific line in the CSV file based on a given key. 6. Reading and returning all data from the CSV file. 7. Resetting the CSV reader to allow for multiple search queries.	1. Logins 2. Employees 3. OccupationMenu 4. CLI 5. Payslip 6. PromotionManager

2.2 CSVWriter

Class Name: CSVWriter	
Responsibilities	*Collaboration*
1. Write Data to CSV Files 2. Overwrite existing data 3. Append new data	1. Login 2. CLI 3. PayslipHistory 4. Employees

2.3 FolderReader

Class Name: FolderReader	
Responsibilities	*Collaboration*
1. Read from a folder location 2. Retrieve file names from said location	1. Payslip: to get a list of payslip history files

3.0 Package 2: Occupations

The purpose of the occupations package (Developed by Ben Bastianelli) is to contain all the code to do with creating an instance of an employees occupation. This package is made up of 3 Classes and an interface: Occupation, OccupationMenu, PromotionManager and FormatManager.

The Occupation class is the template for every employees occupation. It sets a department, a job title, the current point of the user on the pay scale for that job title and their salary. It also contains a method to parse this data into a csv format for writing to the employees CSV file later.

The OccupationMenu class is a utility class for the Occupation package and is instantiated in other classes (Like the Employee class mentioned later in this document) to initialise an Occupation object. It handles all reading of data from the resources folder in the Occupation package that contains all department csv data. (Departments,JobTitles,pointValue,AssociatedSalary).

The PromotionManager class is another utility class for the Occupation package that handles the implementation of yearly point increases for each employee along with manual and automated promotions when an employee is at the top of their pay scale for 3 years.

The FormatManager interface is implemented in 2 of these above classes and has 3 usages. It contains a default method called ‘stringFixFormat’ which helps remove illegal symbols from salaries before parsing them to a double.

Below are the CRC cards for each of these classes:

3.1 Occupation:

Class Name: Occupation	
Responsibilities	*Collaboration*
1. Represents an employee's occupation 2. Formats data into a format that can be written to a csv file	1. UserTypes 2. PromotionManager 3. Employees

3.2 OccupationMenu:

Class Name: OccupationMenu	
Responsibilities	*Collaboration*
1. Guides the user through creating an occupation Object 2. Read and display occupation related data based off User inputs from CSV files	1. Uses CSVReader to read occupation data 2. Used by UserTypes to initialize occupation attribute

3.3 PromotionManager:

Class Name: PromotionManager	
Responsibilities	*Collaboration*
1. Handle employee promotions based on points and job titles 2. Handle yearly point upgrades for each employee	1. Uses CSVReader to read occupation data 2. Uses CSVWriter to update employee data 3. Used by CLI to perform promotions

4.0 Package 3: UserTypes

The UserTypes package (Developed by Jack Ryan) handles everything to do with each user of the system. As every user of the system is an employee we decided to make an abstract UserType class that represents each employee.

The UserType class contains all generic employee data and has generic methods that would be used for its 3 subclasses. Those 3 subclasses being Admin,HR and Employee. The differences in each subclass is the String userType assigned to them that changes the menu they access when that employee logs in.

There is a subclass of Employee called PartTime that generates an part-time employee object with the key difference being the fact that they have an weekly no. of hours they work and that they must sign a form before they are paid each month.

The employees class reads all data from the Employees csv file and parses the data back into employee objects to allow each part of our program to easily manipulate and change employee data on a large scale. It also overwrites any changes to the employees.csv file after program execution.

This package also contains a EmployeeHandler interface that is implemented in the Admin and HR classes and allows an employee to be added to the system or removed from it.

Below our the CRC Cards for each of the classes

4.1: UserType:

Class Name: UserType	
Responsibilities	*Collaboration*
1. An abstract representation of every employee Type 2. Constructors for instantiating a userType object and also for instantiating one based off a line of csv 3. getters and setters 4. ToString method that prints the data to the CLI 5. A toCSV method that parses the data to csv form	1. Extended by Employee, HR and Admin 2. Uses Occupation menu to instantiate an employees occupation 3. Uses login to create a login for the user.

4.2: HR

Class Name: HR	
Responsibilities	*Collaboration*
1. A concrete subclass of UserType that represents a HR user 2. Same functionality as UserType 3. Addition of the "addEmployee" and "removeEmployee" methods from the employeeHandler interface	1. Extends userTypes 2. Implements employeeHandler 3. used by employees to manage HR officers

4.3 Admin

Class Name: Admin	
Responsibilities	*Collaboration*
1. A concrete subclass of UserType that represents an admin user 2. Same functionality as UserType 3. Addition of the "addEmployee" and "removeEmployee" methods from the employeeHandler interface	1. Extends userTypes 2. Implements employeeHandler 3. used by employees to manage HR officers

4.4: Employee

Class Name: Employee	
Responsibilities	*Collaboration*
1. Extension of the userType class that implements the basic employee userType 2. Sets the string userType variable to employee	1. Extends the UserType class 2. Used by the CLI to check requesting userType.

4.5: Employees

Class Name: Employees	
Responsibilities	*Collaboration*
1. A class that contains a list of UserType objects that are read from the employees.csv file 2. Implements methods to allow access to the users in the list and search for specific users in the list 3. implements methods to over-write existing data to the employees.csv file	1. uses UserType to create UserType objects with its string parameter constructor 2. used by the CLI and payment cycle manager to access employee data 3. uses CSVReader & CSVWriter to read and write files

4.6: PartTime

Class Name: PartTime	
Responsibilities	*Collaboration*
1. Extension of employee that generates a part time employee object 2. Confirms whether they have a part time payment form filled	1. Extends employee class

5.0 Package 4: PaymentProcess

The paymentProcess package (Developed by Amy Drew) is responsible for handling all payment processing functionality within the system. From generating payslips to processing them and then writing them to a csv file this package is responsible for all the code to do with this aspect of the system. The classes in this package are as follows:

PaymentProcess.java: This class is the core of the payment processing logic. It calculates all the necessary deductions (PRSI, USC, Income Tax, Health Insurance, Union Fees) and determines the net pay for an employee. It also provides methods to calculate gross pay and stores relevant information like salary, marital status, and union type.

Payslip.java: This class is responsible for generating and displaying the payslip for an employee. It takes in an employee object and a PaymentProcess object, and formats the data into a readable payslip output. It also handles retrieving payslip data from historical records based on user input.

PayslipHistory.java: This class manages the storage and retrieval of payslips. It maintains a list of payslips, provides methods to add new payslips to the list, and can output the entire payslip history as a string or write it to a CSV file.

PaymentCycleManager.java: This class orchestrates the payroll processing cycle. It checks if the current date requires payroll processing (e.g., if it's the 25th of the month), and if so, it generates payslips for all employees by calling the necessary methods from other classes. It also saves the generated payslips to a CSV file using the PayslipHistory class.

Below are the CRCS for each of these classes

5.1: PaymentProcess:

Class Name: PaymentProcess	
Responsibilities	*Collaboration*
1. Processes the net pay calculation for a given employee taking all necessary deductions and creating an object that stores their data. 2. The data stored in the class is then used to invoke methods to create a paymentProcess object	1. Used by paymentCycle manager to create generate a payslip for an employee 2. Used by payslip to generate a payslip for an employee

5.2 Payslip:

Class Name: Payslip	
Responsibilities	*Collaboration*
1. Generates a payslip for a given employee 2. It does this by compiling information from a userType object that represents the user being paid and a payment process object that contains the processing for this cycles pay 3. Contains a method to get all payslips for a specific employee and return the payslip they wish to view	1. Uses PaymentProcess to get the processed wage 2. Uses UserType to get the employee data 3. Uses CSVReader to read the csv data for a payslipHistory 4. uses FolderReader to read the folder that contains all csv files in the folder 5. Used by paymentCycleManager and the CLI to generate payslips

5.3 PayslipHistory:

Class Name: PayslipHistory	
Responsibilities	*Collaboration*
1. Class contains a collection of payslips for every employee in the system. 2. Writes this payslip data to a csvFile with the name of the current month and year	1. Used by PaymentCycleManager to process a cycle of wages for every employee 2. Uses CSVWriter to write the payslipHistory data to a csv file

5.4 PaymentCycleManager:

Class Name: PaymentCycleManager	
Responsibilities	*Collaboration*
1. Utility class to process the payments of a cycle 2. Includes methods to process the payments and generate payslips for every employee and generates a csv file for them to be written to	1. Uses Employees to get the list of employees. 2. Uses PayslipHistory to manage payslip history. 3. Uses Payslip to generate payslips. 4. Used by CLI to simulate time and trigger the payment cycle.

6.0 Package 5: Login

The login package (Developed by Evelyn Sadlier) is the class that handles everything to do with generating logins, authenticating logins and creating new logins from the system. It also allows access to the CLI which is the main user interface of the system.

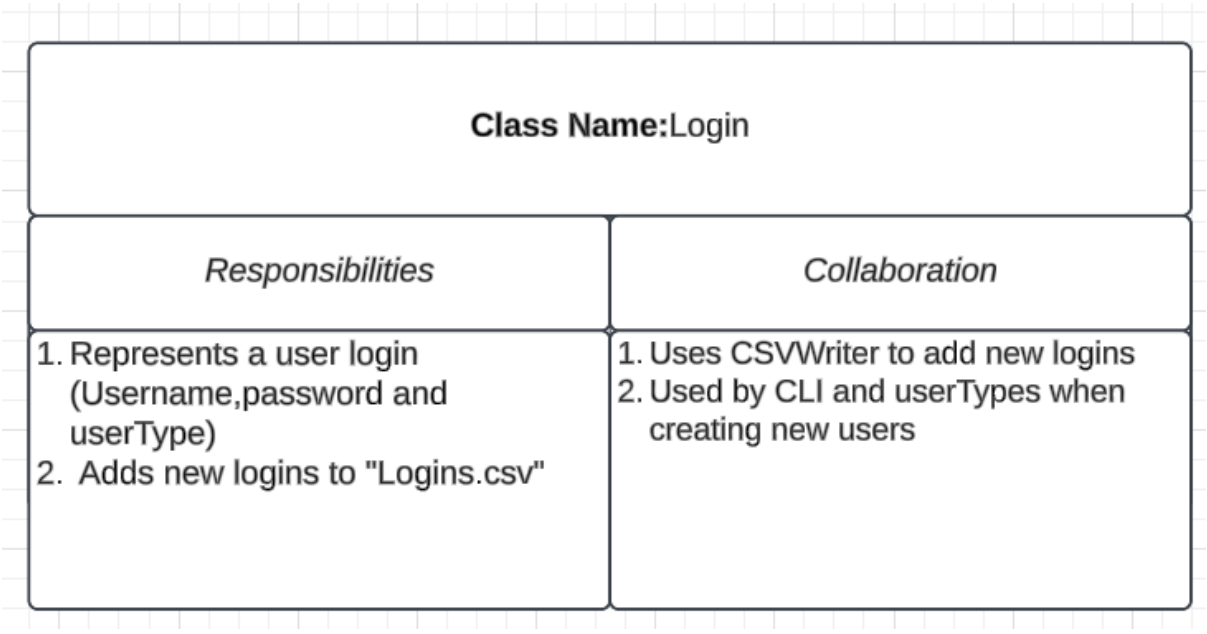
Login.java: This class represents a single login credential entry. It stores the username, password, and user type for a user. It provides methods to add new login credentials to the system and handles the case when a login attempt fails.

Logins.java: This class manages the authentication process. It reads login credentials from a CSV file and provides a method to verify if a given username and password match an existing entry. It also stores the type of the currently logged-in user (Admin, HR, Employee) for access control.

CLI.java: This class provides the command-line interface for the application. It handles user login, displays menus based on user type (Admin, HR, Employee), and executes commands according to user input. It interacts with other classes to perform actions like viewing employee details, reviewing payslip history, adding or deleting employees, and simulating time for payroll processing.

Below are the CRC's for each class:

6.1 Login:



6.2 Logins:

Class Name: Logins	
Responsibilities	*Collaboration*
1. Authenticate user login credentials against Logins.csv 2. if the login is correct allow access to the CLI 3. if the login is incorrect prompt the user to try again or quit the application	1. Uses CSVReader to read login data. 2. Used by CLI for user authentication

6.3 CLI :

Class Name:CLI	
Responsibilities	*Collaboration*
1. Provides a command-line interface for user interaction 2. Handles user commands based on UserTypes 3. Acccess' methods from other packages and correctly uses them to integrate system functionality 4. Implementation of time simulation methods to mimic time passing	1. Uses Logins for authentication 2. Uses Employees to manage Employees 3. Uses Payslip to display PayslipHistory 4. Uses PromotionManager to handle promotions