Task 1:HR System
Task 2: Specification

Running the project allows users to first interact with an empty HR System. While running, the HR System will prompt a message saying please enter what you want to do indicating that users can type in commands that create a new worker or department head or change the worker's salary or schedule. All commands are case sensitive and the order of information needed must be the same.

When the user type exit, then the program will quit itself.

When the user type create worker followed by its name, salary, department and schedule separated by a single space, then the program will create a worker with the same attribute. Passing in the schedule needs to start with the day of the week that the work works on followed by the start time (time combined with AM or PM without space)and then the end time (same format as start time) separated by space. Here you are only allowed to enter a single day for the schedule. After the worker is created, the program will display that the worker with name, id, salary and department is created.

When the user type creates department head followed by its name, department, the program will create a department head with the same attribute. If there is already a head for the department, then the program will give a message that says there is already a department head for the department. If the department head is created successfully, then the program will display department head with name, id and department is created.

When the user type increase salary followed by the id of the worker, the id of the department head that wants to increase the salary and the percent that should be increased as a decimal. If there is no worker with the id, then the program will show a message that says no worker found with id. If the department head that is entered is not in the same department as the worker, then the program will display that the department head has no right to increase salary for the worker. If the department head id entered doesn't match any department head, then the program will display that department head not found. If everything is correct, then the program will increase the worker's salary and display the salary of worker name with id has been increased to current salary by permission from the name of the department head.

When the user type decrease salary followed by the same information as the increase salary, then the program will do similar thing as the increase salary except it is decreasing the salary. All the message for not valid input is the same as increase salary and the message for decreasing salary successfully is the same except it says the salary decreases.

When the user type change schedule followed by worker id, department head id, and the new schedule in the format mentioned above, then the program will display no worker found if the worker id doesn't match to a worker, no department head found if the department head id doesn't match to a department head, no right to change schedule when the department head has a different department as the worker if the department head is not in the same department

as the worker, schedule changed for worker name, id, in its department with current schedule on day of the week from start time to end time if the department head and worker are valid and in the same department.

The three entity classes are worker, department head and schedule with 2 use case classes change salary and change schedule. Then we have an InOutHandler which is the controller (takes in input and calls a use case class method with the input) and the presenter (displays output to UI) together (this only works since the program is still fairly small). Finally, we will have a class called HRSystem which connects the whole program together. It keeps track of workers and department heads created and generate unique ids and make sure the department entered for the department head does not have duplicates. It also display

Task 3: CRC model

Entity Class:

Class name: Worker	
Responsibilities Stores worker's information like name, id, salary and department Provide the basic setter and getter for these basic attributes	**Collaborators** Schedule changeSalary changeSchedule

Class name: departmentHead	
Responsibilities Stores department head's information like name, id, and department Provide the basic setter and getter for these attributes	**Collaborators** changeSalary changeSchedule

Class name: schedule	
Responsibilities Stores information about worker schedule like week of the day, start time and end time Provide basic getter and setter for these attributes	**Collaborators** Worker changeSchedule

Use Case Class:

Class name: changeSalary	
Responsibilities Changing salary for the worker with permission from the department head. Manipulate the salary attribute of the worker when the input is valid.	**Collaborators** Worker departmentHead InOutHandler

Class name: changeSchedule	
Responsibilities Changing the schedule for the worker with the permission from the department head. Manipulate the schedule attribute of the worker when the input is valid.	**Collaborators** Schedule Worker departmentHead InOutHandler

Class name: InOutHandler	
Responsibilities Take in input from UI and pass it into the correct use case method based on the input and display the result on the UI	**Collaborators** changeSchedule changeWorker Worker departmentHead HRSystem

Class name: HRSystem	
Responsibilities Keeps track of the id of worker and department head to make sure no duplicate id Keep track of department for department head to make sure no 2 department head are in the same department Keeps program taking user input until user enter exit Display the welcome message and message	**Collaborators** InOutHandler

that ask user to enter input and ending message	

Task 4: Scenario Walk-Through

First type create worker Bob 1000 food Monday 9AM 5PM
Then you should see worker Bob with id 0, salary 1000 and department food is created
Then type create department head Jack food
Then you should see department head Jack with id 0 and department food is created
Then type create department head Louis food
Then you should see there is already a department head for food
Then type increase salary 0 0 0.2
Then you should see the salary of Bob with id 0 has been increased to 1200 by permission from Jack
Then type increase salary 1 0 0.5
Then you should see no worker found with id 1
Then type increase salary 0 1 0.5
Then you should see the department head not found
Then type create department head Louis clothes
Then you should see department head Louis with id 1 and department clothes is created
Then type increase salary 0 1 0.3
Then you should see department head has no right to increase salary for the worker
Then type decrease salary 1 0 0.5
Then you should see no worker found with id 1
Then type decrease salary 0 2 0.5
Then you should see the department head not found
Then type decrease salary 0 1 0.3
Then you should see department head has no right to increase salary for the worker
Then type decrease salary 0 0 0.5
Then you should see the salary of Bob with id 0 has been decreased to 600 by permission from Jack
Then type change schedule 1 0 Friday 7AM 4PM
Then you should see no worker found
Then type change schedule 0 2 Friday 7AM 4PM
Then you should see no department head found
Then type change schedule 0 1 Friday 7AM 4PM
Then you should see no right to change schedule when the department head has a different department as the worker
Then type change schedule 0 0 Tuesday 11AM 9PM
Then you should see schedule changed for Bob with id 0 in food department with current schedule on Tuesday from 11AM to 9PM
Then type exit
Then you should see a thanks for using message and the program stop