Alex Peffer
CS 3354 Project 1

Work Request System

Project Scope and Requirements:

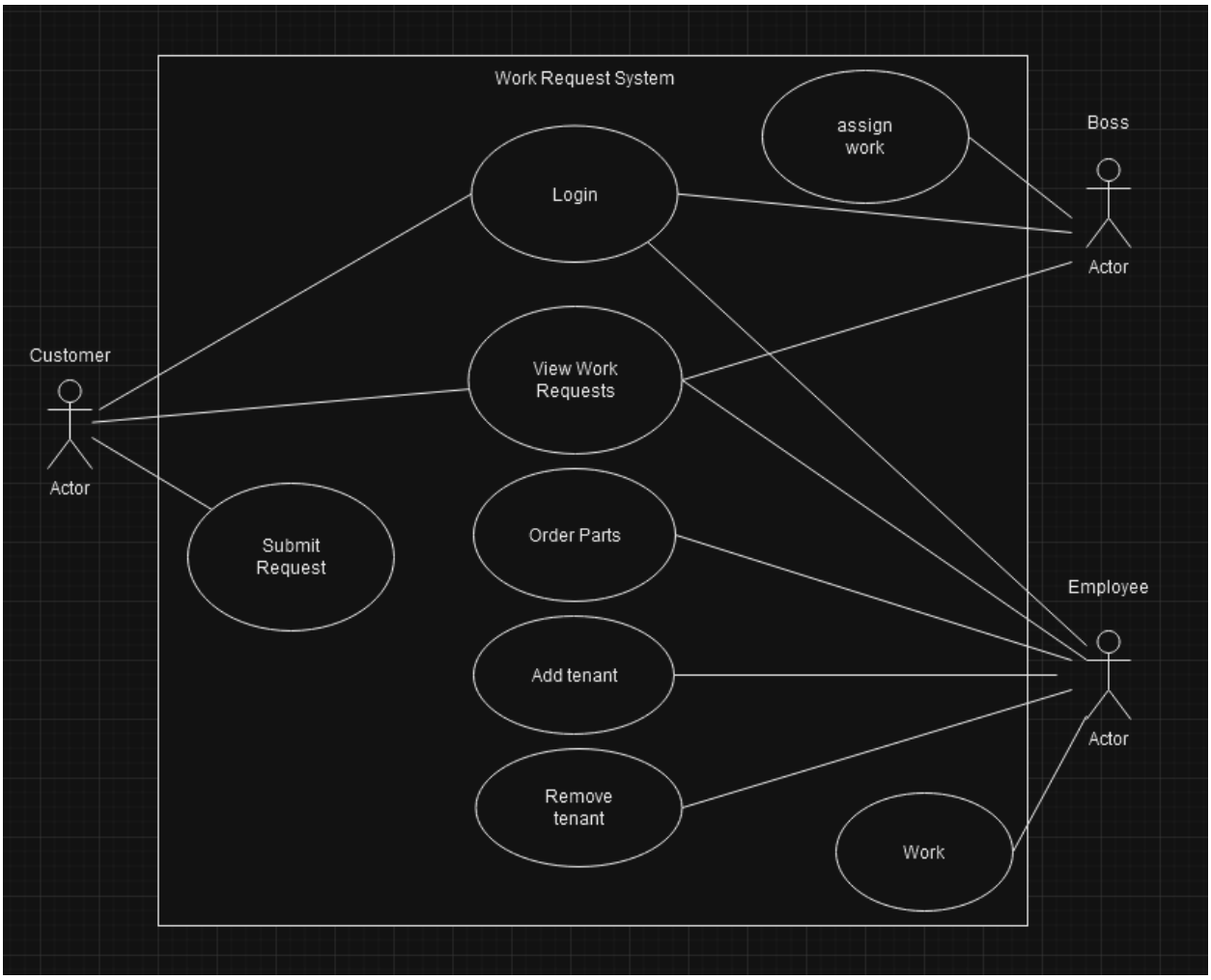
I want my application to solve the problem of the need for an apartment complex to have a work-request system.

The tenants will be able to create work requests that will require a part. They will also be able to view the status of all of their work requests

The employee will be able to work on the request and close it when work has been completed. If there are not enough parts the employee will order parts to complete the work.

The boss will also be able to manage all work requests to change their status and their priority level, but most importantly assign the work order to an employee so they can work on it.

Use Case Document



Shared Functions

View work requests:
Precondition: customer's work request list is not empty and
Postcondition: work request statuses are printed for every work request belonging to the requester

Login:
Postcondition: A menu is printed out for this person to view and interact with.

Customer

Submit request:

Precondition: customer has a name and address and there exists a part type that is needed for the work request.
Postcondition: a new work request object is created using the customer's name, address, and part and added to that customer's arraylist of work requests that belong to them.

Employee

Work:
Precondition: employee has a work request assigned to them with status "open"
Scenarios: - Employee will work on the given work request
- If there is none of the part required in stock work cannot be completed, employee must order more first.
Postcondition: status of workorder changed to "closed" if completed, or "part" if awaiting parts

Add tenant:
Precondition: a tenant with the given name doesn't already exist
Postcondition: a new tenant is created with a name and apartment number

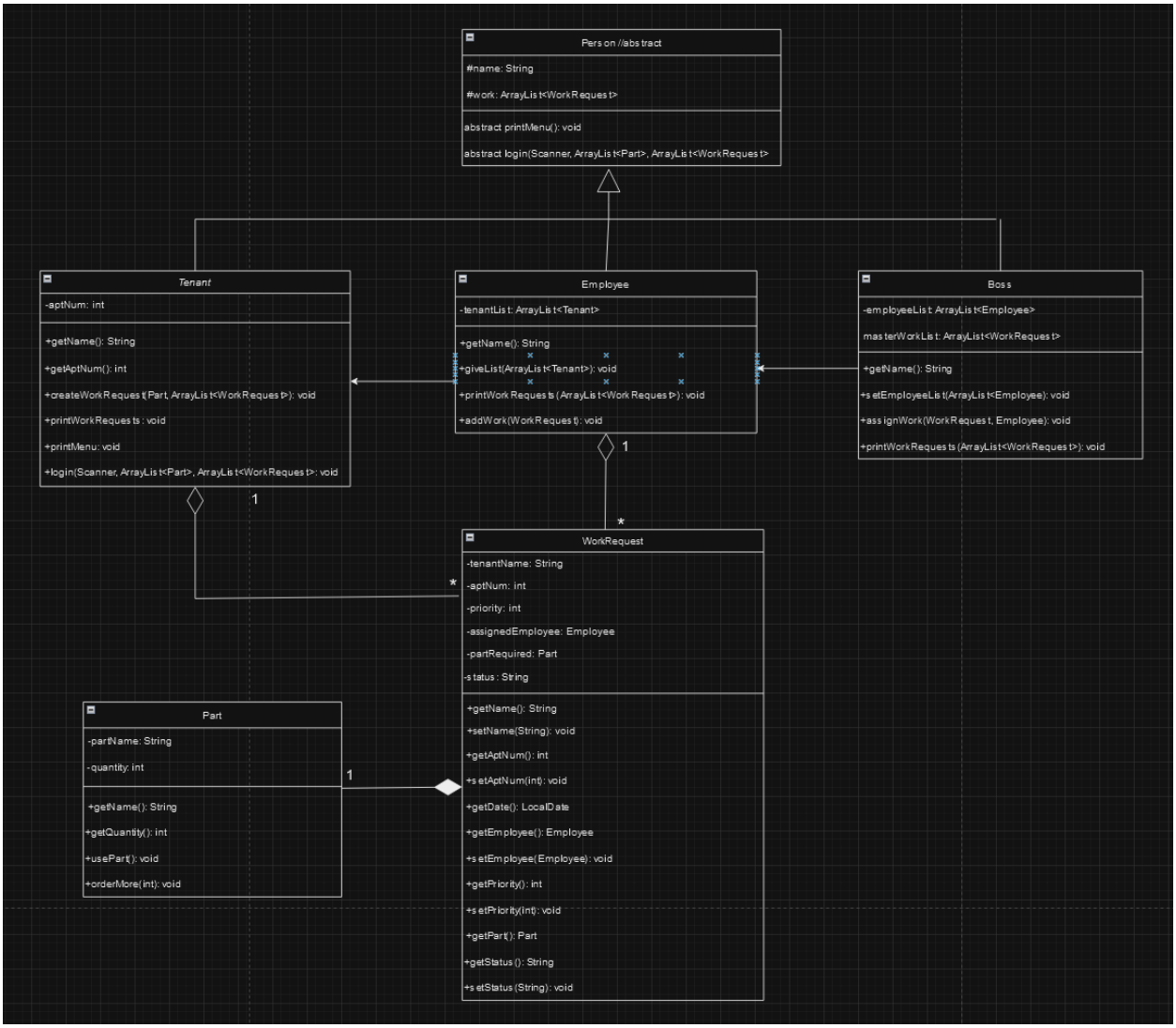
Remove tenant:
Precondition: a tenant with the given name exists
Postcondition: the given tenant is deleted from the system.

Order part:
Precondition: A part exists, and a stockroom array exists.
Postcondition: the quantity of the ordered part is increased by 3.

Boss

Assign work:
Precondition: a work request exists and an employee exists
Postcondition: work request is now assigned to the employee

Class Diagram



Person: this class was designed to be used so that a person could be downcasted to become a tenant or an employee. An employee would be hired by the boss, and an employee would sign new apartment tenants. (this idea was scrapped as the program became big and the deadline approached)

protected String name – name of the person.
protected String ArrayList<WorkRequest> work – an arraylist of work that belongs to this person. Boss = all work. Employee = assigned work. Tenant = work that the tenant requested.

abstract void printMenu() – every person will have a menu that displays the actions that that person can perform.

abstract void login(Scanner scnr, ArrayList<Part> partList, ArrayList<WorkRequest> masterWorkList) – every person should have a login that handles input and actions they perform.

Boss extends **Person:** this class should only have one instance. The default name is Admin for login purposes. The boss's role is to assign work requests to an employee that will work on them, and to set the priority of the work request.

private ArrayList<Employee> employeeList – a list of all of the employees given to the boss for the purpose of assigning work.
ArrayList<WorkRequest> masterWorkList – a list of all of the work requests so the boss can assign work to an employee and change the priority.

public String getName() – returns the boss's name.
public void setEmployeeList(ArrayList<Employee> employeeList – gives the boss access to the employee list.
public void printWorkRequests(ArrayList<WorkRequest> masterWorkList) – prints out the work requests.

Employee extends **Person:** the employees will be assigned work that requires a part. They can work on the work to "close" the work request, but if they do not have enough of the part, they will need to order more first. While waiting for the part, the work request status will change to "part". An employee can also add new tenants to the apartment or remove tenants.
private ArrayList<Tenant> tenantList – a list of all the tenants.
public String getName() – returns the name of the employee
public void giveList(ArrayList<Tenant> tenantList) – used to assign the tenantlist to the the employee from main.
public void printWorkRequests(ArrayList<WorkRequest> masterWorkList) – prints the list of the work assigned to this employee.
public void addWork(WorkRequest request) – assigns work request to this employees list of work

Tenant extends **Person:** the tenant's role is to create work requests for Parts, they can make requests for light bulbs, air filters, or paint. (3 of the parts added).
private int aptNum – the tenant's assigned apartment number.

public String getName() – returns this tenant's name
public String getAptNum - returns this tenant's apartment number.
public void createWorkRequest(Part partNeeded, ArrayList<WorkRequest> masterWorkList) – creates a work request for apart, assigns it to the tenant's work request list and the masterWorkList.
public void printWorkRequests() – prints this tenant's created work requests.

Part: this class is used to create parts, for the use of this project the 3 available parts are "light bulb", "air filter" and "paint". Parts have a quantity in the stock room and when a part is out of stock, the employee must order more before completing work requests.

private String partName – the part's name
private int quantity – the part's quantity

public String getName() – returns the name of the Part
public int getQuantity() – returns the current quantity of this Part
public void usePart() – uses the particular part, reduces it's quantity
public void orderMore(int amount) – orders more of this part (can take any number but for this program I set it so the employee will always order 2).

WorkRequest: this class is for creating work requests. It contains all of the functions and data of the work request objects.

private String tenantName – the tenant's name assigned to this work request.
private int aptNum – the apartment number assigned to this work request.
private LocalDate date – the date assigned to this work request. (initialized to the day of the creation of the work request)
private int priority – the priority level assigned to this work request.
private Employee assignedEmployee – the tenant's name assigned to this work request. (0,1,2,3)
private Part partRequired – the part assigned to this work request.
private String status – the status assigned to this work request. (open, part, closed)

public getName()/setName(String name) get/sets the tenant's name assigned to this work request
public getAptNum()setAptNum(int aptNum) get/sets the tenant's apartment number assigned to this work request.
public getEmployee/setEmplyee(Employee employee) get/sets the employee assigned to this work request
public LocalDate getDate() – returns the date this work request was created.
public getPriority/setPriority(int prioLevel) - get/sets the priority level of this work request
public Part getPart() – returns the part required for working on this work request.
public getStatus/setStatus(String status) - get/sets the status of this work request

Sequence Diagram