Coursework IN2018 Team Project

Team Delapa

Prashan, Akif, Danish, Andreya, Emmanuel, Levi

Contents

1.1 Description of Existing System

Bloomsbury Image processing laboratory **(BIPL)** is a photographic laboratory, which handles the work of professional photographers. They currently offer 30 standard tasks, which customers can purchase. The receptionist manually handles the inquiries made by the customers using non-computerized equipment. Examples of these services include the "use of large copy camera", "black and white film processing" and "colour film processing".

The order starts when the customer either calls the company or meets the receptionist in person, requesting specific services. The receptionist checks a logbook and judges whether the job can be taken based on the priority and the capital available. In the case that the job request is feasible the receptionist process the order and sends the relevant information to the technicians in the laboratory. A timetable is curated based on the number of available technicians, the availability of the rooms, the priority level of tasks, and the deadlines set by the customer. The technicians will carry out the jobs detailed by the timetables. Once the task is completed, the semi or fully-completed job is placed in a given location from where it can be collected for further processing or dispatched to the customer.

Once the order is completed customer can choose to pay by cash whereby the payment has to be made in full. When the full payment is received, it is recorded in a payment book and the order is marked as completed and then archived. For customers deemed to be of a greater stature, they would be given the option to pay for older purchases at the end of the month.

After analyzing the current system, we have concluded that there are seven major flaws in the current **BIPL** system. The flaws found by the Delapa.inc team are the following:

1. The current system cannot receive any card payment. This leads to a massive loss of potential customers because most customers in the current market prefer to pay by card because of ease of use and high-level security. According to Fundera.com[1], “80% of customers prefer card payment over cash payment and 10% of consumers make all of their purchases by cash”. The new system will allow a wider range of customers to interact with the company.
2. Due to possible unforeseen circumstances paper records can be destroyed or damaged with no easy form of backing them up. This is problematic because if such an event were to fruition, there could be a loss of work progress and errors in the accounting of payments, as according to the statictis providen by Recordination.com[2], “94% of companies do not fully recover from severe data loss”. This will lead to the credibility of the company being diminished. The new system will allow backups of records ensuring no loss in data.
3. Due to the use of non-computerized methods, the system as a whole is very inefficient in comparison to a computerized system. This will cause fewer orders to be processed at a given time resulting in lower potential revenue. With a computerized system, there will be time-saving features, such as "customer lookup".
4. In a world where the environmental impacts of a company are closely monitored and hence connected to its reputation, the massive amount of paper used by the company could have detrimental to both pollution levels and the reputation of the company. Surveys have shown that 76% of Americans expect companies to act against climate change and 73% of Americans would stop purchasing from a company that does not care about climate change.[3] The new system will require less usage of paper saving both money and improving the reputation of the company.

5. As much as the work process is not computerized there is high involvement of manual labor leading to a higher chance of human errors. These errors could result in an inaccurate entry of the data desired, leading to a domino effect compromising the performance of the rest of the team. The new system will have more involvement of computerized processes leading to fewer human errors.
6. There is currently little or no security regarding access to confidential information. In order to adhere to Data Protection laws, a better security system will be necessary as potential leakage of customer data could lead to legal actions taken against the company. The new system will introduce a login feature with only those who have permission to access customer files.
7. With the current system, there are many factors contributing to high running costs. Some of these factors include the constant need for paper, stationaries, and space to store current and archived orders. The new system will require less space, less usage of paper, and no requirement for stationaries.

All these major flaws listed above can be mitigated or minimized with the introduction of a new computerized system.

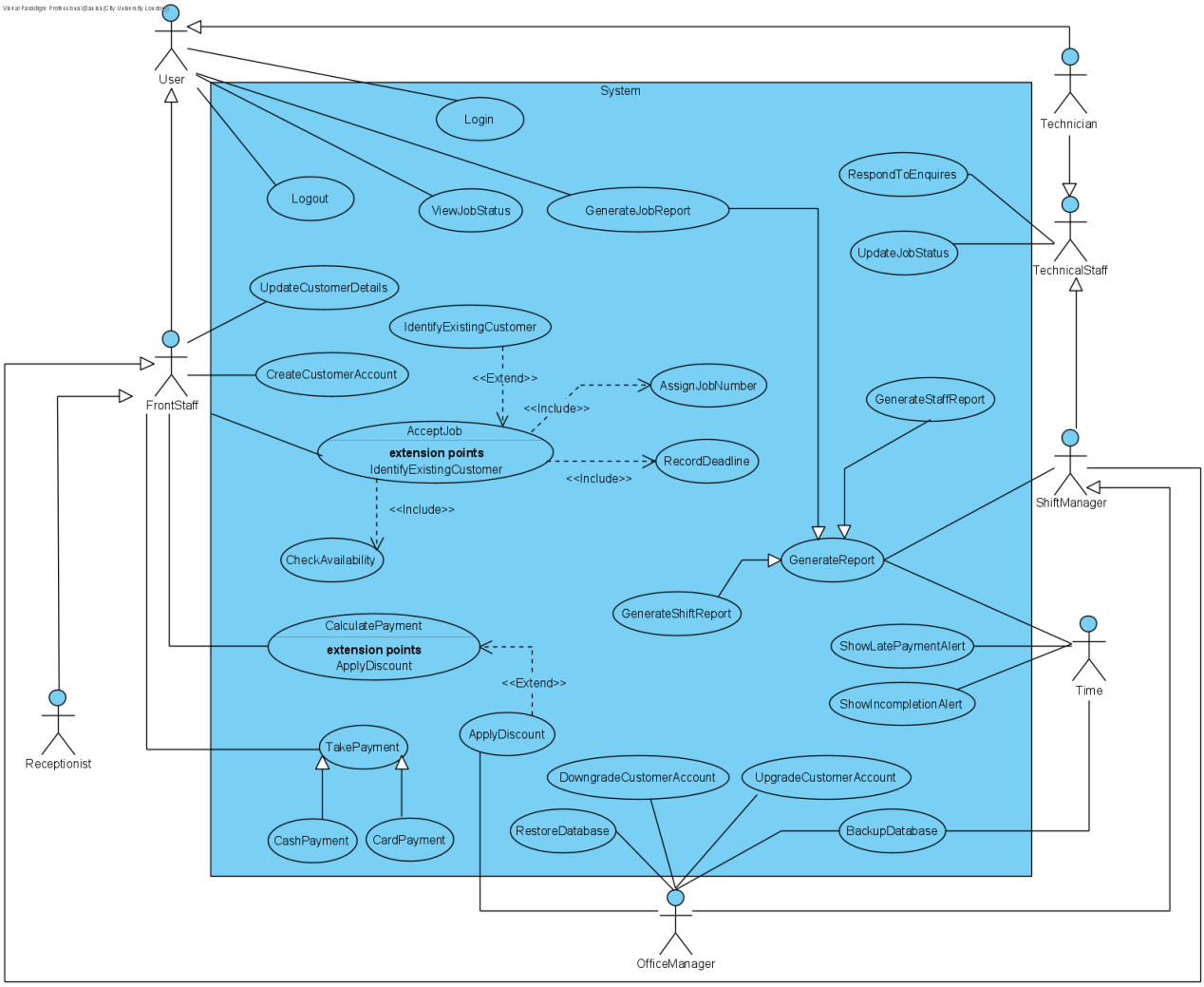
[1] https://www.fundera.com/resources/cash-vs-credit-card-spending-statistics

[2] https://www.recordnations.com/2018/11/risks-skipping-data-backup/

[3] https://online.maryville.edu/blog/importance-of-environmental-awareness-when-running-a-business/

2.1 Use Case Diagram:

Presented below is the full use case diagram, created by Team Delapa based on the specifications provided by BAPERS for the new computerised system. In summary, all users of the system will be able to login to the new system which will improve digital security and allow delegation of functions based on the role of the staff.



Case Diagram V2.2

2.2 Use Case Specifications

Below you can find the most important use case specifications written up for the new BAPERS system. It shows the flow of events for a given function in the system along with possible alternative flows if the expected requirements are not met.

ID: 1	**Use Case:** AcceptJob
Brief Description: The system accepts a job that is provided by the customer via the front staff to which the job is assigned to a customer account, given a job number and have its deadline recorded.	
Primary Actors: FrontStaff	
Secondary Actors: None	
Pre-conditions: 1. The actor triggering the function is logged in. 2. There must be an available slot for the job to be accepted (CheckAvailability)	
Main Flow: 1. This use case starts after the availability for the job is checked *include(CheckAvailability)* 2. If the customer has no customer account; 2.1. The job cannot be accepted. 3. If the customer does have a customer account; 3.1. The customer account is retrieved. *extension point: identifyExistingCustomer.* 4. A valid customer account is retrieved; whether a new one is created or an existing one is retrieved from the database 5. Based on the job given by the customer, the receptionist assigns a job number to the job *include(AssignJobNumber)* 6. Derived from the customers' requirements, the deadline is also recorded *include(RecordDeadline)* 7. The account retrieved/created earlier is linked to the job by the system	
Post-conditions: 1. The job is recorded to the system and linked to a relevant customer account	
Alternative Flows: None.	

ID: 1.1	**Alternative Flow:** AcceptJob:CustomerNotFound
Brief Description: Customer has no account when trying to accept the job	
Primary Actor: FrontStaff	
Secondary Actor: None	
Pre-Conditions: A valid customer is not found	
Alternative Flow: 1. The system prompts the actor that a valid customer does not exist 2. The FrontStaff accepts the notification.	
Postconditions: None	

ID: 1.2	**Use Case:** identifyExistingCustomer
Primary Actors: FrontStaff	
Secondary Actors: None	
Segment 1 Pre-conditions (extension point: identifyExistingCustomer)**:** 1. Customer already has an account	
Segment 1 Flow: 1. This use case starts as part of 3.1 of use case ID 1 2. The receptionist enters either name or ID of customer 3. A valid customer account is retrieved and its details are sent back to the main use case	
Post-conditions: 1. An existing customer account is retrieved	
Alternative Flows: None.	

ID: 2	**Use case:** GenerateReport
Brief description: A use case that can be manually triggered by a user or by time. Multiple different types of reports can also be generated.	
Primary actors: ShiftManager, Time	
Secondary actors: User	
Preconditions: 1. BAPERS is operational, the user is logged in.	
Main Flow: 1. If the actor triggering the use case is Time 1.1. Generate a report automatically based on pre-configured settings. 2. If the actor triggering the use case is User or ShiftManager 2.1. The actor selects the type of report they want to generate. 3. The system generates a report.	
Postconditions: 1. Report is successfully generated.	
Alternative Flows: None	

ID: 3	**Use case:** UpgradeCustomerAccount
Brief description: Customer accounts can be upgraded by Office Manager in which they become a valued customer enabling them to receive extra discounts on further purchases.	
Primary actors: Office Manager	
Secondary actors: None	
Preconditions: 1. The Office Manager logs in the system.	
Main Flow: 1. The Office Manager selects a valid customer using the system. 2. They then upgrade the selected customer's account.	
Postconditions: The customer account is successfully upgraded.	
Alternative flow: None	

ID: 4	**Use case:** ShowLatePaymentAlert
Brief description: Payment can only be accepted once the job they have placed has been fully completed. Payments deemed to be late should be shown to an office manager to warn them a specific payment is now late.	
Primary actors: Time	
Secondary Actors: None	
Preconditions: 1. The BAPERS system is functional	
Main Flow: 1. A payment is acknowledged as being late by the system. 2. An alert is displayed on screen to an office manager. 3. The alert will reappear in 15-minute intervals. 4. The Office Manager understands and accepts the warning alert.	
Postconditions: 1. Late payment alert has been successfully sent out	
Alternative Flows: None	

ID: 5	**Use case:** ShowIncompletionAlert
Brief description: It is possible that a job is not completed on time due to other events (not likely to be completed on time) hence the system must notify the relevant person.	
Primary actors: Time	
Secondary Actors: None	
Preconditions: 1. The BAPERS system is functional	
Main Flow: 1. An incomplete Job is detected by the system. 2. The system deems it unlikely to be finished. 3. An alert message is displayed to a shift or office manager.	
Postconditions: 1. Incomplete job alert has been successfully sent out	
Alternative Flows: None	

ID: 6	**Use Case:** CalculatePayment
Brief description: This will calculate a payment amount for the customer based on the tasks within the job. Also allows the customer to pay for job(s) once they have been completed, offering them to pay by card or cash.	
Primary actors: Front Staff	
Secondary Actors: None	
Preconditions: 1. The system is functional 2. The requested job/s has been completed.	
Main Flow: 1. The use case starts when one of the front staff has “Logged in” 2. The system will calculate a price using the information available. Extension point: ApplyDiscount	
Postconditions: 1. The transaction is stored in the database. 2. The payment is made in full.	
Alternative Flows: None	

ID: 6.1	**Use Case:** ApplyDiscount
Primary Actors: OfficeManager	
Secondary Actors: None	
Segment 1 Preconditions (extension point: ApplyDiscount)**:** 1. The Customer is a valued customer.	
Segment 1 Flow: 1. The discount for the associated customer is applied. 2. The system calculates a new total based on the discount.	
Postconditions: 1. The job is updated with the new price	
Alternative Flows: None	

ID: 7	**Use case:** UpdateJobStatus
Brief description: Staff members who have the classification "technical staff" will have the option to update the status of job requests currently in the system. The system will allow TechnicalStaff to change status from "processing" to "completed", "delivered", and "archived".	
Primary actors: TechnicalStaff	
Secondary actors: None	
Preconditions: 1. BAPERS is operational. 2. The staff member must be a TechnicalStaff and be logged in. 3. There needs to be a valid job request in the database system	
Flow of events: 1. A TechnicalStaff finds job by searching it up in the system. 2. The job is selected. 3. The status of the job is changed to reflect the jobs current status of the job.	
Postconditions: 1. The system has recorded the new status onto the database	
Alternative Flows: None	

ID: 8	**Use case:** UpdateCustomerDetails
Brief description: Front desk staff members can change the details of customers. Customers can ask to change their details.	
Primary actors: FrontStaff	
Secondary actors: None	
Preconditions: 1. BAPERS is operational. 2. The staff member needs to be a FrontStaff and logged in. 3. The customer be already registered on the system	
Flow of events: 1. A customer requests a front staff to change their details. 2. The front staff looks up the customer on the database system. 3. The customer is found and selected. 4. The front staff changes their details on the system.	
Postconditions: 1. The system records the updated details on the database system	
Alternative Flows: CustomerNotFound	

ID: 8.1	**Alternative Flow:** UpdateCustomerDetails:CustomerNotFound
Brief Description: Customer details are not found during the update process	
Primary Actor: FrontStaff	
Secondary Actor: None	
Pre-Conditions: 1. A valid customer is not found	
Alternative Flow: 1. The system prompts the actor that a valid customer does not exist 2. The FrontStaff accepts the notification.	
Postconditions: None	

ID: 9	**Use Case:** Login
Brief Description: Each member of the BAPERS team will be required to login to the system.	
Primary Actors Users	
Secondary Actors None	
Preconditions 1. BAPERS is operational. 2. The primary actor has entered their login details	
Main Flow: 1. When the Users entered their login details the system checks if the login details are correct 2. If the login details are correct, then the users will have access to the system. 3. Once logged into the system the users can view job status or generate a job report. Extension points: ViewJobStatus Extension points: GenerateJobReport 4. Once the users finished with what they needed to do then they just need to logout 1. Extension points: Logout	
Postconditions: 1. The user is logged out of the system	
Alternative Flow LoginFailure	

ID: 9.1	**Alternative Flow:** Login:LoginFailure
Brief Description: Login details are invalid	
Primary Actor: User	
Secondary Actor: None	
Pre-Conditions: The User does not enter valid login details	
Alternative Flow: 1. The system prompts the User that the login details entered are invalid. 2. The User accepts the notification.	
Postconditions: None	

ID: 10	**Use case:** BackupDatabase
Brief description: The reason to implement a data backup is to save files in the event of system failure.	
Primary actors: OfficeManager	
Secondary actors: Time	
Preconditions: None.	
Main Flow: 1. The Office Manager prompts the database to backup. 2. The system creates a new backup of the relevant data	
Postconditions: A successful backup is made.	
Alternative flows: BackupFailed	

ID: 10.1	**Alternative Flow:** BackupDatabase:BackupFailed
Brief Description: The system backup feature has run into an error and the backup has failed as a result	
Primary Actor: OfficeManager	
Secondary Actor: Time	
Pre-Conditions: Backup has failed	
Alternative Flow: 1. The system prompts the Office Manager the backup has failed. 2. The Office Manager accepts the notification.	
Postconditions: None	

ID: 11	**Use case:** RestoreDatabase
Brief description: The database may be restored from a previous backup in the event that the system has failed.	
Primary actors: OfficeManager	
Secondary actors: None.	
Preconditions: There is a previous backup to restore from	
Main Flow: 1. The Office Manager selects the backup file to restore from 2. The system verifies that the backup file is valid. 3. Backup procedure begins	
Postconditions: The system has been fully restored to a previous state	
Alternative flows: RestoreFailed	

ID: 11.1	**Alternative Flow:** RestoreDatabase:RestoreFailed
Brief Description: The system restore feature has run into an error and the restore has failed as a result	
Primary Actor: OfficeManager	
Secondary Actor: None.	
Pre-Conditions: Restore has failed	
Alternative Flow: 1. The system prompts the Office Manager the restore has failed. 2. The Office Manager accepts the notification.	
Postconditions: None	

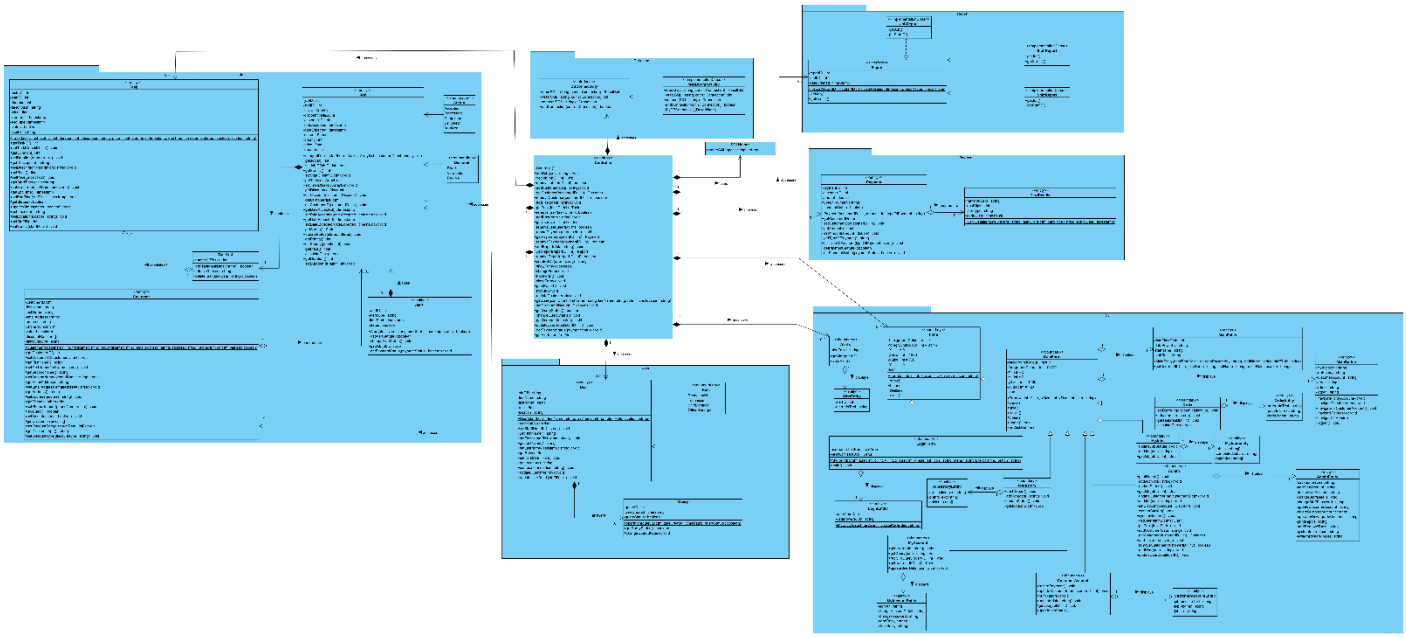
2.3 Indexed Priority List of Use Case Specifications

In this section you can find the indexed priority list of the most important use case specifications along with their justification.

Use Cases	Priority	Justification
AcceptJob	High	This is an integral function of the business aspect of the BAPERS system. For revenue to be made this must be included.
GenerateReport	Medium	The system can function without this use case, however, it is still an important feature to maintain the system.
UpgradeCustomerAccount	Low	This feature is likely to be used less than the other use cases therefore it is does not hold much importance.
ShowLatePayment	Medium	As late payments/incomplete jobs can be handled manually the alerts are not essential.
ShowIncompleteJob	Medium	
CalculatePayment	High	This must be implemented otherwise the business will be at a loss and can cause customer dissatisfaction.
UpdateStatus	High	This is an important requirement of the system to help manage and improve the workflow efficiency.
UpdateCustomerDetails	Low	It is unlikely customers will need to change their details
Login	High	This is an integral function of the system as it is the first line of digital security.
BackupDatabase	High	To protect the integrity of the data being held frequent backups are necessary to ensure sensitive data is not lost.
RestoreSystem	High	A backup is less valuable if it can not be restored hence this will also be essential.

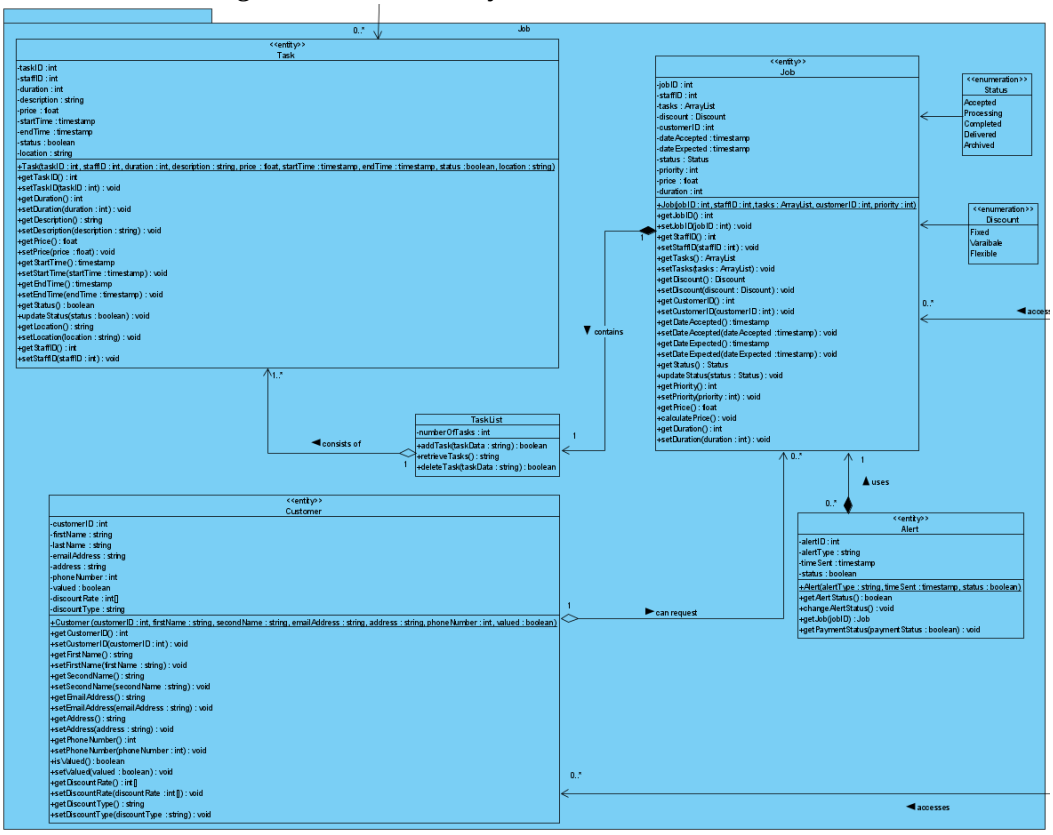
3.1 Design Class Diagram

The following is the design class diagram which models the new BAPERS system. It will model the specifications of the system in as much detail as possible. This is the link to a higher quality version of the image: https://i.imgur.com/iiFZFOj.png.

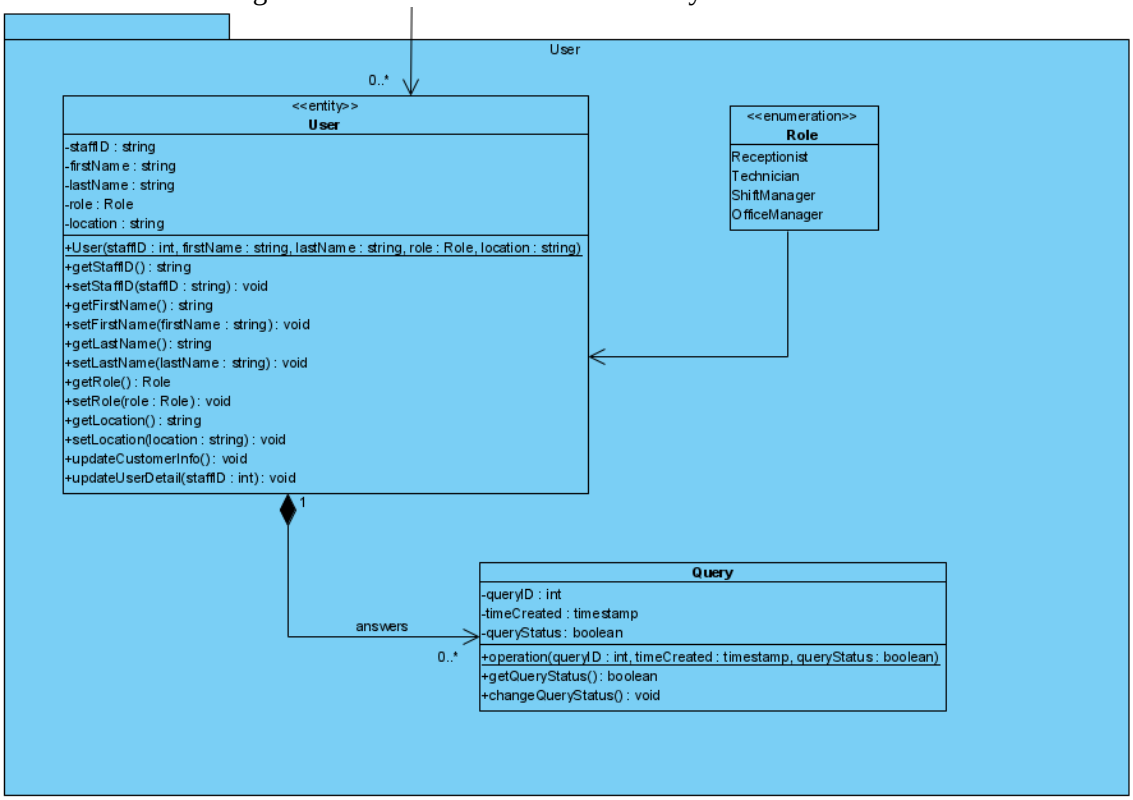


Case Diagram V3.1

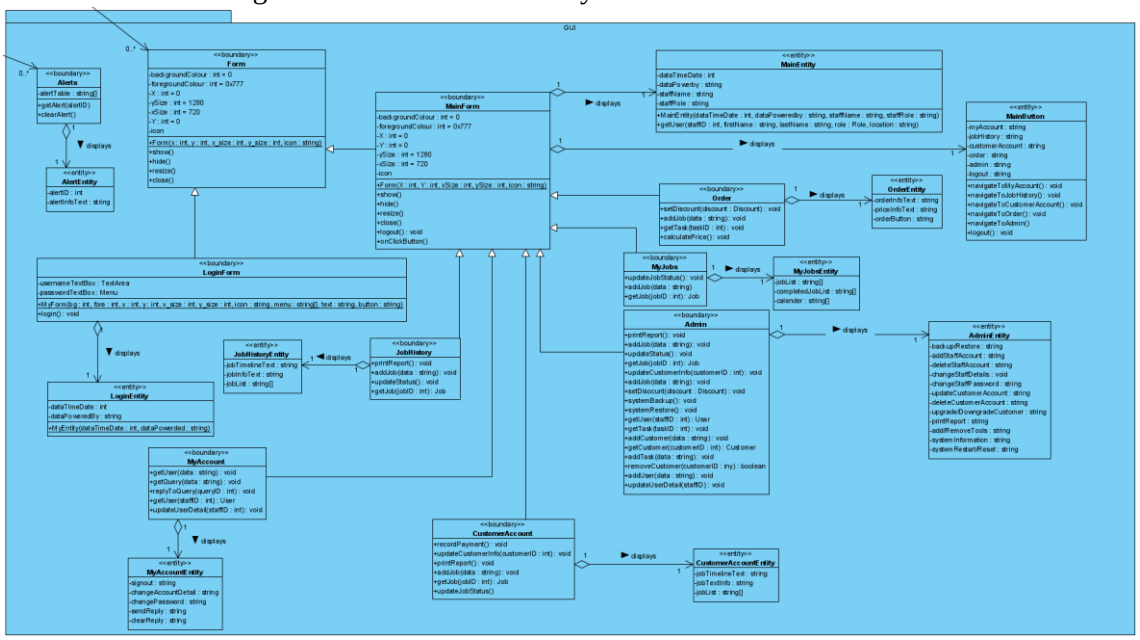
3.1.1 The Job Package: Models how the system will handle orders from customers.



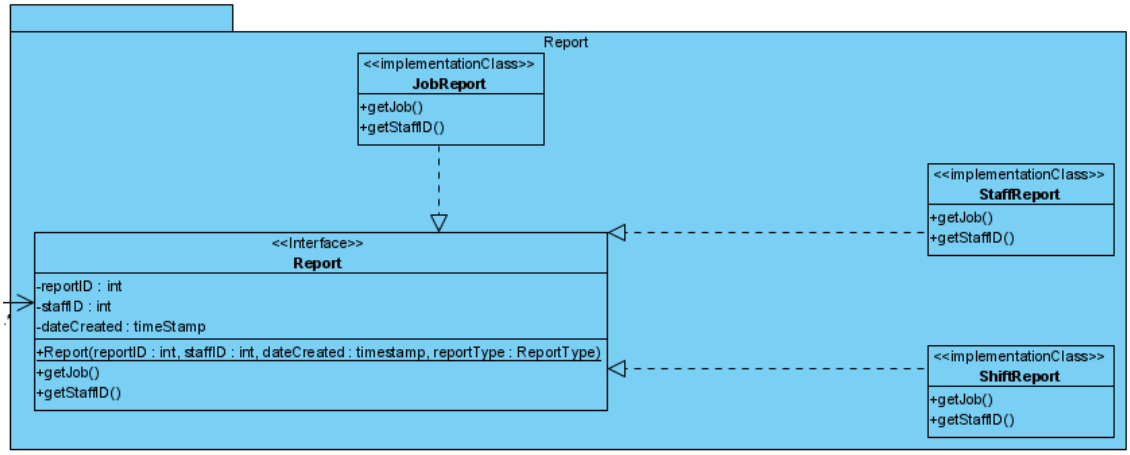
3.1.2 The User Package: Models the staff accounts in the system.



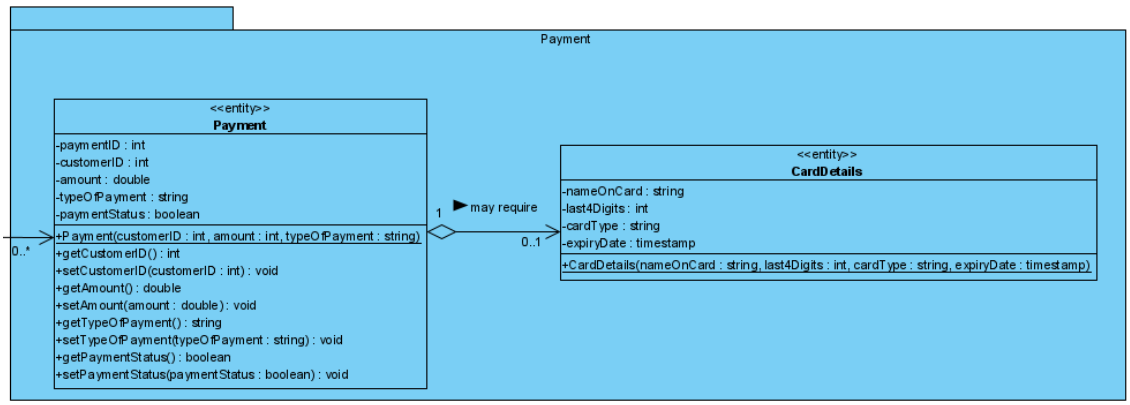
3.1.3 The GUI Package: Models the GUI of the system.



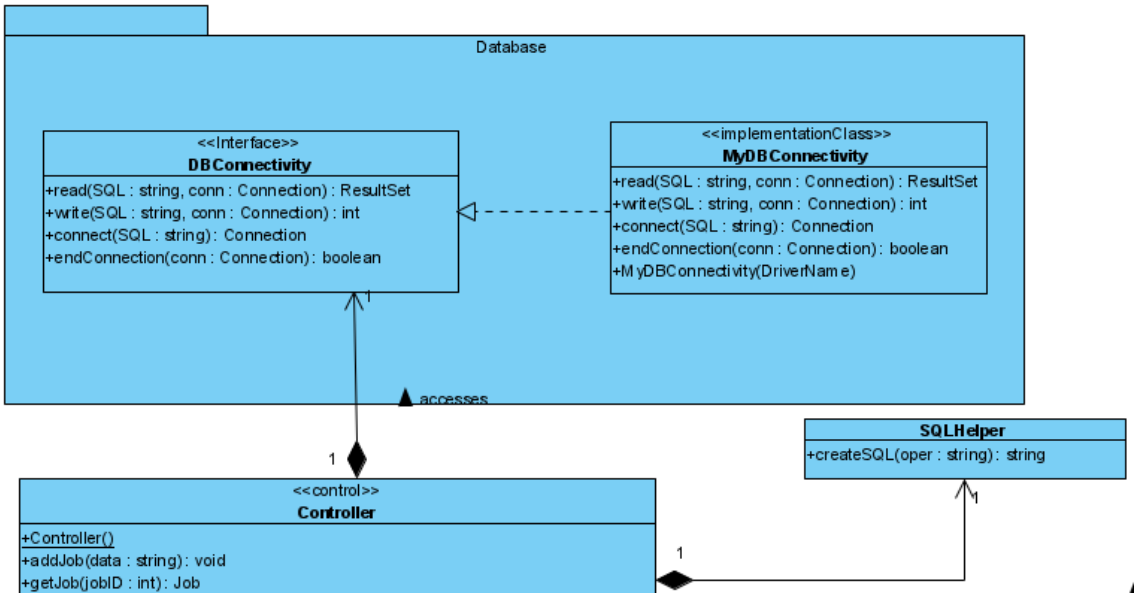
3.1.4 The Report Package: Models the report handelingof the system.



3.1.5 The Payment Package: Models the payment handeling done by staff



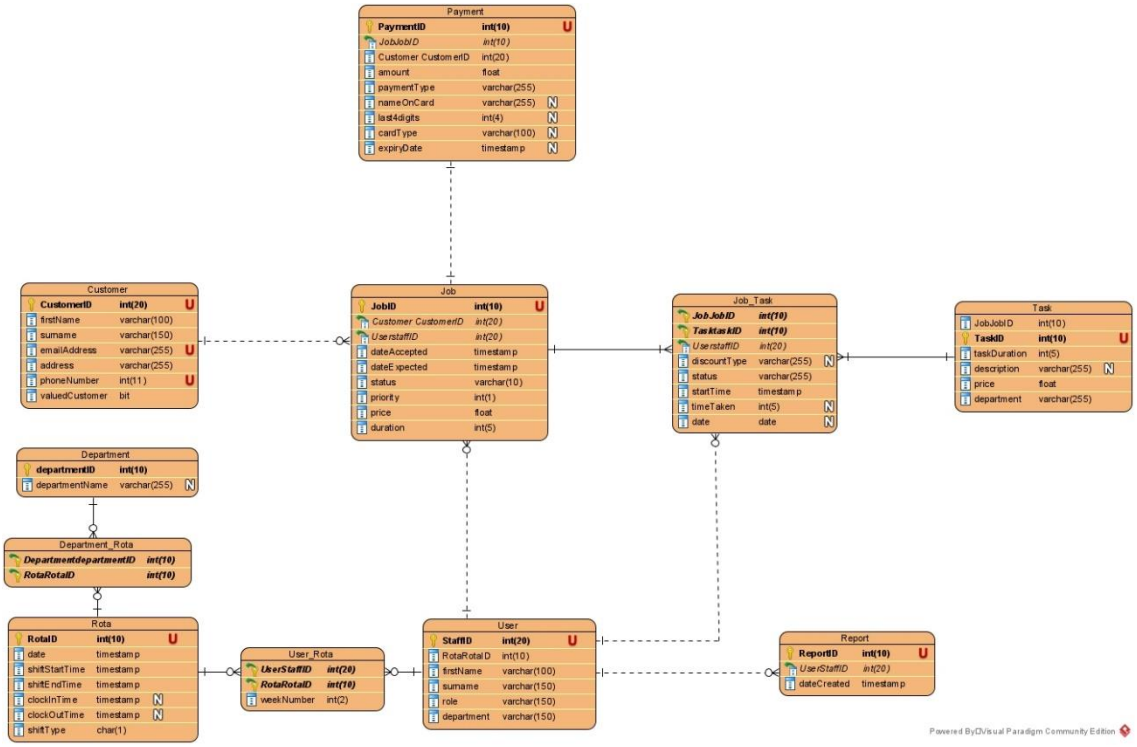
3.1.6 The Database Package: Models how the system will interact with the databse system.



3.1.7 The Controller: Models the conntroller class of the system

4.1 Entity Relationship Diagram:

The following diagram models the entity relationship diagram of the system. It shows how the system will store and access data for such operations as creating reports, timing alerts and storing payment information.



Case Diagram V2.3

4.2.1 DDL Commands:

These DDL commands generate the tables required for the database system.

CREATE TABLE Customer (CustomerID int(20) NOT NULL AUTO_INCREMENT, firstName varchar(100) NOT NULL, surname varchar(150) NOT NULL, emailAddress varchar(255) NOT NULL UNIQUE, address varchar(255) NOT NULL, phoneNumber int(11) NOT NULL UNIQUE, valuedCustomer bit(1) NOT NULL, PRIMARY KEY (CustomerID));

CREATE TABLE Department (departmentID int(10) NOT NULL AUTO_INCREMENT, departmentName varchar(255), PRIMARY KEY (departmentID));

CREATE TABLE Department_Rota (DepartmentdepartmentID int(10) NOT NULL, RotaRotaID int(10) NOT NULL, PRIMARY KEY (DepartmentdepartmentID, RotaRotaID));

CREATE TABLE Job (JobID int(10) NOT NULL AUTO_INCREMENT, `Customer CustomerID` int(20) NOT NULL, UserstaffID int(20) NOT NULL, dateAccepted timestamp NOT NULL, dateExpected timestamp NOT NULL, status varchar(10) NOT NULL, priority int(1) NOT NULL, price float NOT NULL, duration int(5) NOT NULL, PRIMARY KEY (JobID));

CREATE TABLE Job_Task (JobJobID int(10) NOT NULL, TasktaskID int(10) NOT NULL, UserstaffID int(20) NOT NULL, discountType varchar(255), status varchar(255) NOT NULL, startTime timestamp NOT NULL, timeTaken int(5), `date` date, PRIMARY KEY (JobJobID, TasktaskID));

CREATE TABLE Payment (PaymentID int(10) NOT NULL AUTO_INCREMENT, JobJobID int(10) NOT NULL, `Customer CustomerID` int(20) NOT NULL, amount float NOT NULL, paymentType varchar(255) NOT NULL, nameOnCard varchar(255), last4digits int(4), cardType varchar(100), expiryDate timestamp NULL, PRIMARY KEY (PaymentID));

CREATE TABLE Report (ReportID int(10) NOT NULL AUTO_INCREMENT, UserStaffID int(20) NOT NULL, dateCreated timestamp NOT NULL, PRIMARY KEY (ReportID));

CREATE TABLE Rota (RotaID int(10) NOT NULL AUTO_INCREMENT, `date` timestamp NOT NULL, shiftStartTime timestamp NOT NULL, shiftEndTime timestamp NOT NULL, clockInTime timestamp NULL, clockOutTime timestamp NULL, shiftType char(1) NOT NULL, PRIMARY KEY (RotaID));

CREATE TABLE Task (JobJobID int(10) NOT NULL, TaskID int(10) NOT NULL AUTO_INCREMENT, taskDuration int(5) NOT NULL, description varchar(255), price float NOT NULL, department varchar(255) NOT NULL, PRIMARY KEY (TaskID));

CREATE TABLE `User` (StaffID int(20) NOT NULL AUTO_INCREMENT, RotaRotaID int(10) NOT NULL, firstName varchar(100) NOT NULL, surname varchar(150) NOT NULL, role varchar(150) NOT NULL, department varchar(150) NOT NULL, PRIMARY KEY (StaffID));

CREATE TABLE User_Rota (UserStaffID int(20) NOT NULL, RotaRotaID int(10) NOT NULL, weekNumber int(2) NOT NULL, PRIMARY KEY (UserStaffID, RotaRotaID));

ALTER TABLE Job ADD CONSTRAINT FKJob118602 FOREIGN KEY (`Customer CustomerID`) REFERENCES `Customer ` (CustomerID);

```
ALTER TABLE Job_Task ADD CONSTRAINT FKJob_Task457140 FOREIGN KEY
(JobJobID) REFERENCES Job (JobID);

ALTER TABLE Job_Task ADD CONSTRAINT FKJob_Task968081 FOREIGN KEY
(TasktaskID) REFERENCES Task (TaskID);

ALTER TABLE Job ADD CONSTRAINT FKJob577887 FOREIGN KEY (UserstaffID)
REFERENCES `User` (StaffID);

ALTER TABLE Payment ADD CONSTRAINT FKPayment170102 FOREIGN KEY
(JobJobID) REFERENCES Job (JobID);

ALTER TABLE Report ADD CONSTRAINT FKReport501948 FOREIGN KEY (UserStaffID)
REFERENCES `User` (StaffID);

ALTER TABLE Job_Task ADD CONSTRAINT FKJob_Task131706 FOREIGN KEY
(UserstaffID) REFERENCES `User` (StaffID);

ALTER TABLE User_Rota ADD CONSTRAINT FKUser_Rota677344 FOREIGN KEY
(UserStaffID) REFERENCES `User` (StaffID);

ALTER TABLE User_Rota ADD CONSTRAINT FKUser_Rota239915 FOREIGN KEY
(RotaRotaID) REFERENCES Rota (RotaID);

ALTER TABLE Department_Rota ADD CONSTRAINT FKDepartment756872 FOREIGN
KEY (DepartmentdepartmentID) REFERENCES `Department ` (departmentID);

ALTER TABLE Department_Rota ADD CONSTRAINT FKDepartment185518 FOREIGN
KEY (RotaRotaID) REFERENCES Rota (RotaID);
```

4.2.2 DML Statements:

The statements are written in MYSQL/SQLite, they are demonstations of possible commands given to the database system in order to perform the required functions.

Insert

1. INSERT INTO `user`(`StaffID`, `RotaRotaID`, `firstName`, `surname`, `role`, `location`) VALUES
(1,1,"Emmanuel","Mukungwa","Shift Manager,"Back Room")

2. INSERT INTO `customer`(`CustomerID`, `firstName`, `surname`, `emailAddress`, `address`, `phoneNumber`, `valuedCustomer`) VALUES
(1,"Emmanuel","Muk","test@test134.com","10 Downing Street",01234567891,0)

Delete

1. DELETE FROM `customer`
WHERE `CustomerID` = 1

2. DELETE FROM Job
Where status = “Archived”

Select

1. SELECT CustomerID, firstName, surname, valuedCustomer
FROM Customer
Where valuedCustomer = 1
Order By surname ASC

2. SELECT CustomerID, JobID, Status, price
From Customer C, Job J, Payment P
Where C.CustomerID = J.CustomerCustomerID
AND J.JobID = P.PaymentID
HAVING status = “completed”
ORDER BY price DESC

Trivial Report Statements:

3. SELECT firstName, TaskID, Department,
Date, startTime, timeTaken, SUM(timeTaken)
FROM User T1, Job_Task T2
Where T1.StaffID = T2.UserstaffID
GROUP BY firstName
GROUP BY SUM(timeTaken)
HAVING X <= Date AND Date <= Y

4. SELECT departmentName,
date, shiftStartTime, shiftEndTime, shiftType, SUM(total)
From Rota R, Department_Rota DR, Department D
Where D.departmentID = DR. DepartmentdepartmentID
AND R.RotaID = DR. DepartmentdepartmentID
HAVING shiftType = “D”
ORDER BY DATE DESC

Create Table

1. CREATE TABLE Customer (
CustomerID int(20) NOT NULL AUTO_INCREMENT,
firstName varchar(100) NOT NULL,
surname varchar(150) NOT NULL,
emailAddress varchar(255) NOT NULL UNIQUE,
address varchar(255) NOT NULL,
phoneNumber int(11) NOT NULL UNIQUE,
valuedCustomer bit(1) NOT NULL,
PRIMARY KEY (CustomerID));

2. CREATE TABLE Report (
int(10) NOT NULL AUTO_INCREMENT,
UserStaffID int(20) NOT NULL,
dateCreated timestamp NOT NULL,
PRIMARY KEY (ReportID));

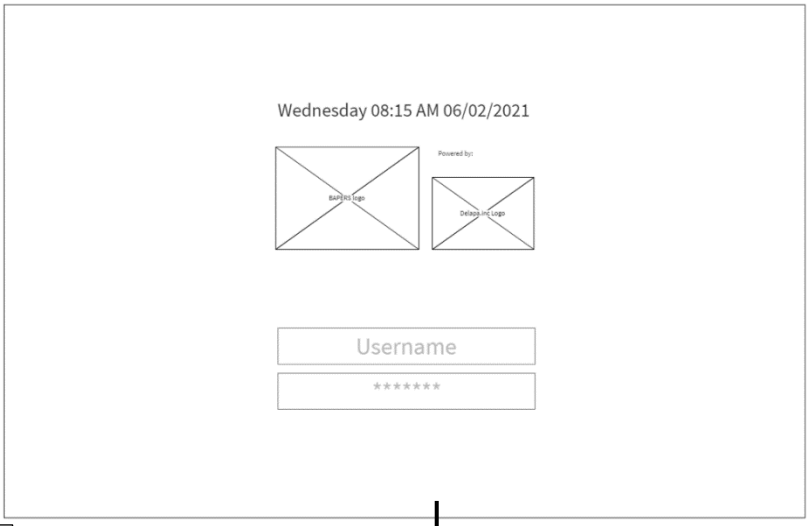
Update

1. Update Job
SET status = “Completed”
Where JobID = 56;

2. Update Customer
SET emailAddress = “mockemail@emailserver.com”
Where CustomerID = 129;

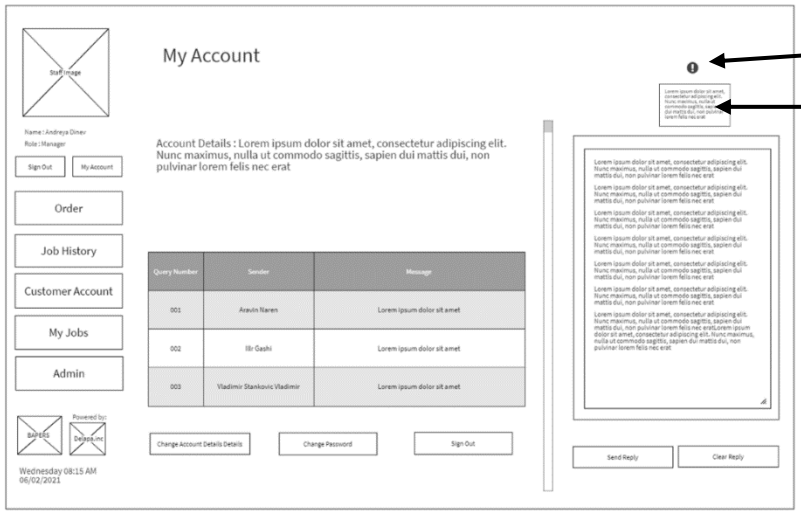
5.1 GUI Design and System Navigation

The GUI design and system navigation below will model how users will interact and navigate through the system.



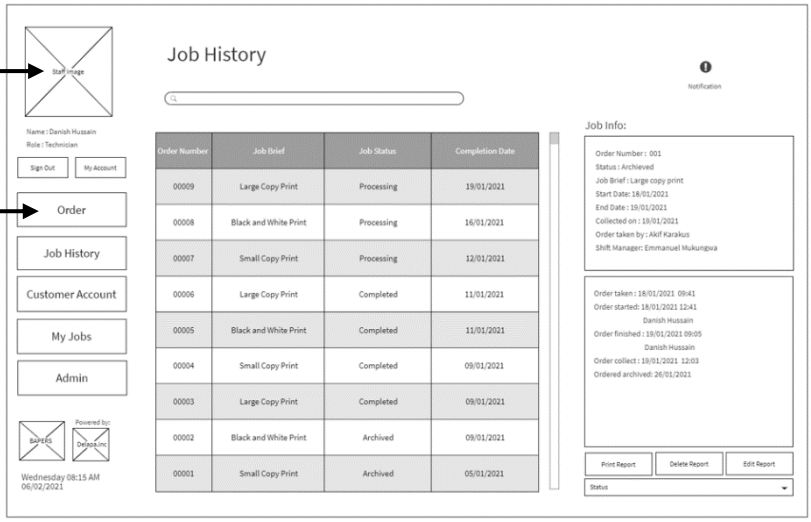
When the user logs in they will be directed to 'My Account' page.

It shows the role of the user so in this case the Manager is the one who is currently on this page.



If you click in the notification icon on the top left a small text box will appear at the bottom showing you the notification.

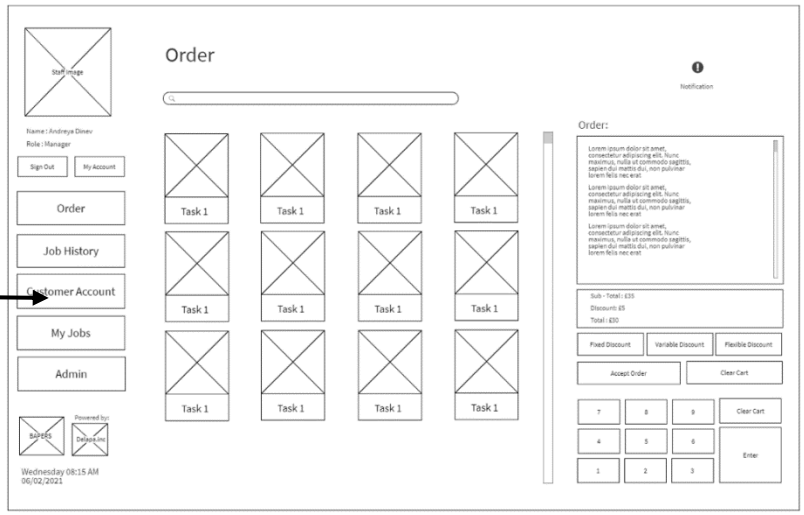
On the left-hand side, you have buttons that have links to the main web pages. If you click any of them it will direct, you to these pages. For example, if you click on the Job History button it will takes you to the Job History web page.



This is the ‘Job History’ page where you can view the recent jobs that have been accepted. You can print, delete, and edit reports on this page. This can be done by clicking on the buttons on the bottom left.

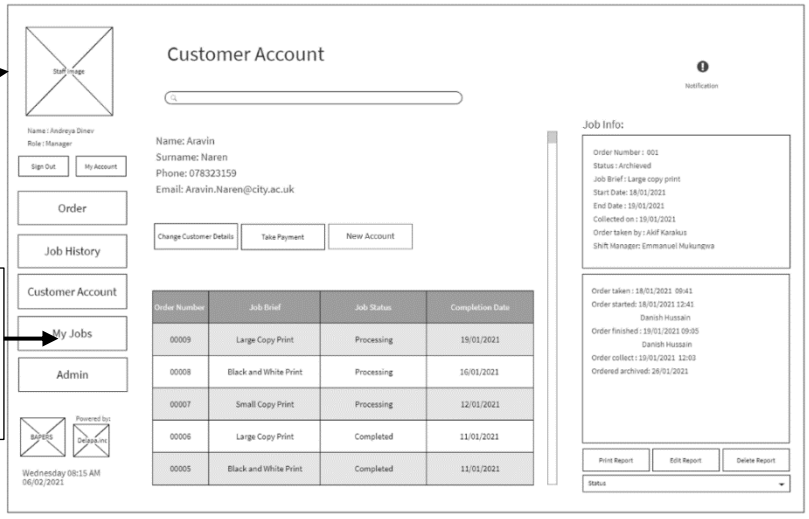
Clicking on the ‘Order’ button will take you to the Order web page.

Continued on the next page



This is the ‘Order’ page where they user can accept jobs.

If you want to go to the ‘Customer Account’ then you just click on the ‘Customer Account’ button.



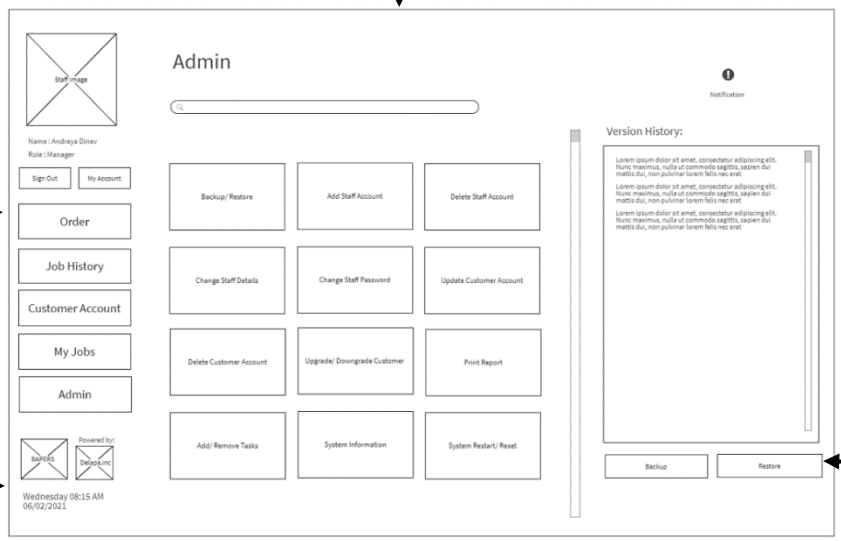
This is the customer account page. Some roles will have features other roles do not. In this case the manager has more features than the other users. The manager is able to ‘Change Customer Details’, ‘Take Payment’ and create a ‘New Account’.

Click on the ‘My Jobs’ button if you want to access the ‘My Jobs’ web page.

This is the ‘My Jobs’ page, in this page the user can see what jobs they need to do. They can also see the urgency level of the job.

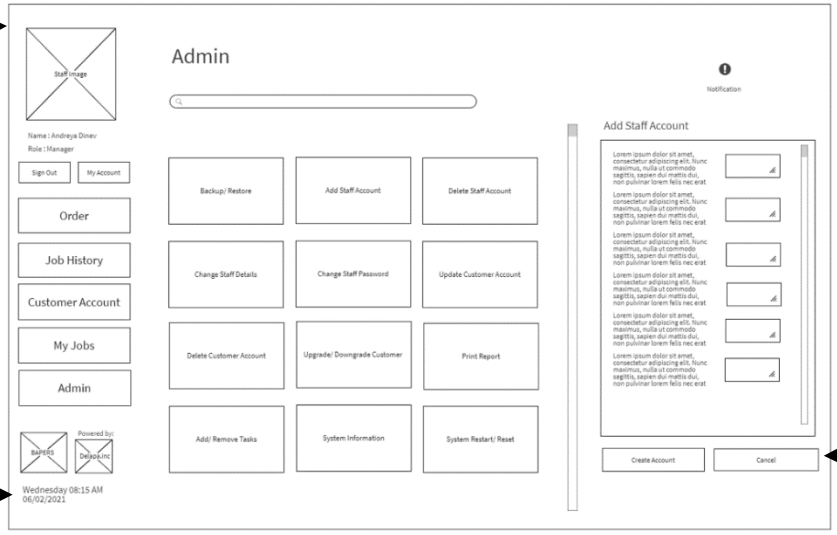
If you want to go to the ‘Admin’ then you just click on the ‘Admin’ button.

This is the admin page; the managers have a lot of features in this page.



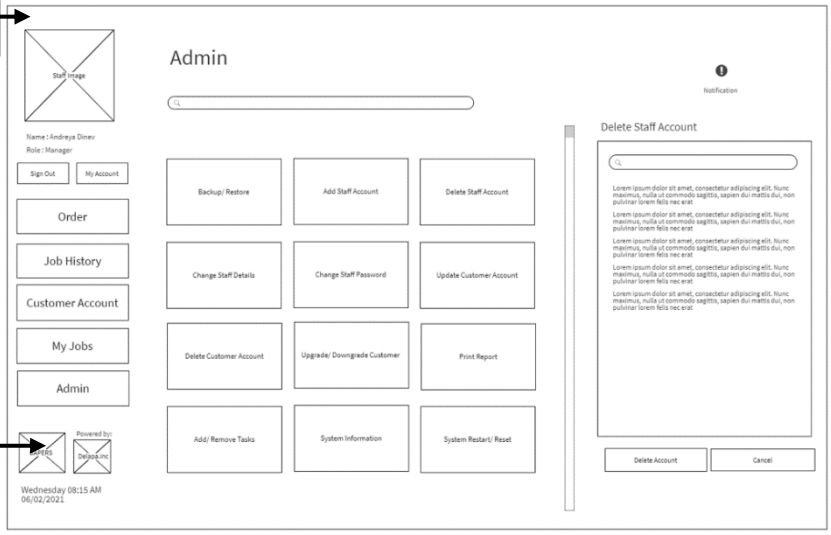
This is where you can backup and restore data, all you must do is to click on the button on the bottom left.

If you want to add staff account, then you just click on the ‘Add Staff Account’ button.



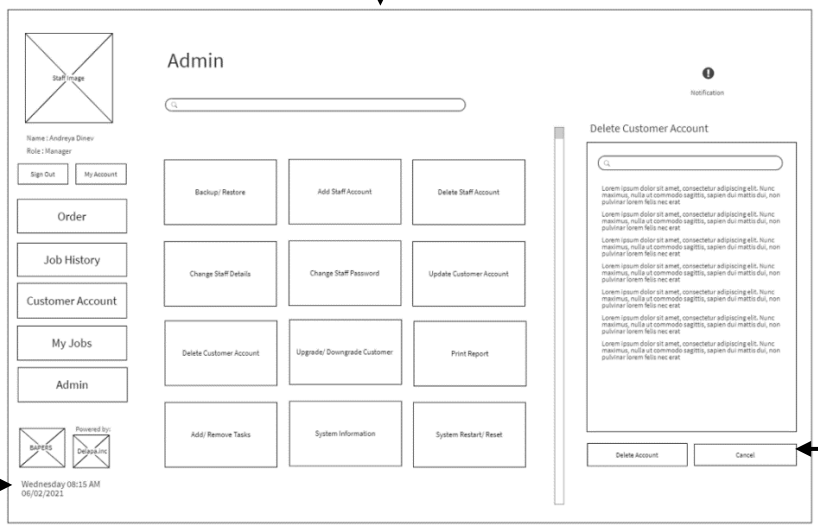
This is where you can add staff account, click on the ‘Create Account’ button on the bottom left.

If you want to delete staff account, then you just click on the ‘Delete Staff Account’ button



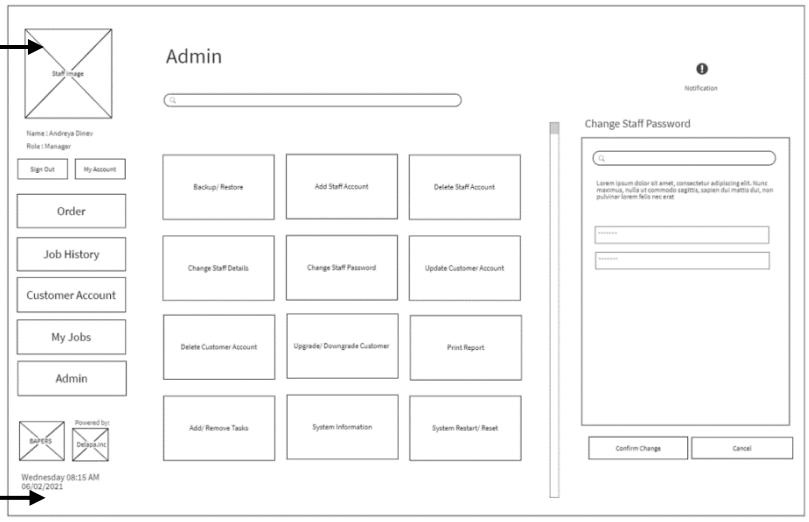
This is where you can delete staff account, click on the ‘Delete Account’ button on the bottom left.

If you want to delete customer account, then you just click on the ‘Delete Customer Account’ button



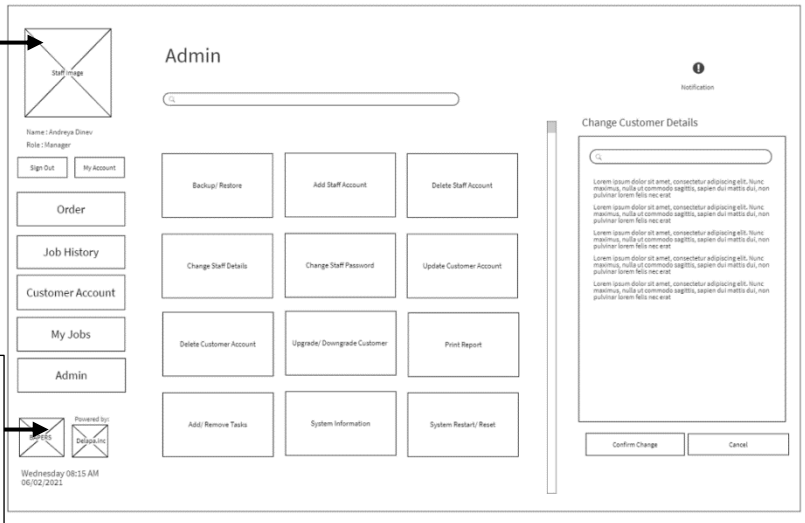
This is where you can delete customer account, click on the 'Delete Account' button on the bottom left.

If you want to change staff password, then you just click on the 'Change Staff Password' button



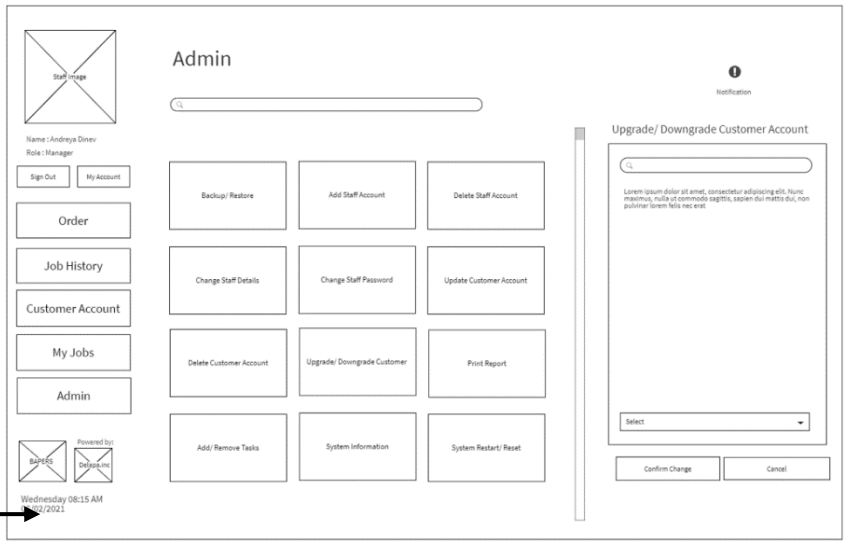
Click on the 'Confirm Change' button on the bottom left to change the password.

If you want to change the customer details, then you just click on the 'Update Customer Account button



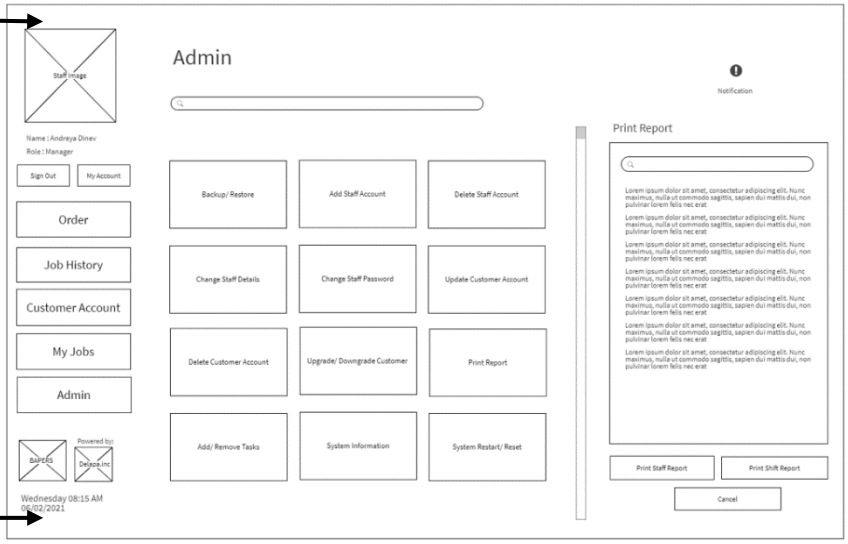
Click on the 'Confirm Change' button on the bottom left to change the customers details.

Click on the 'Upgrade/Downgrade Customer' button if you want to upgrade or down grade customer account.



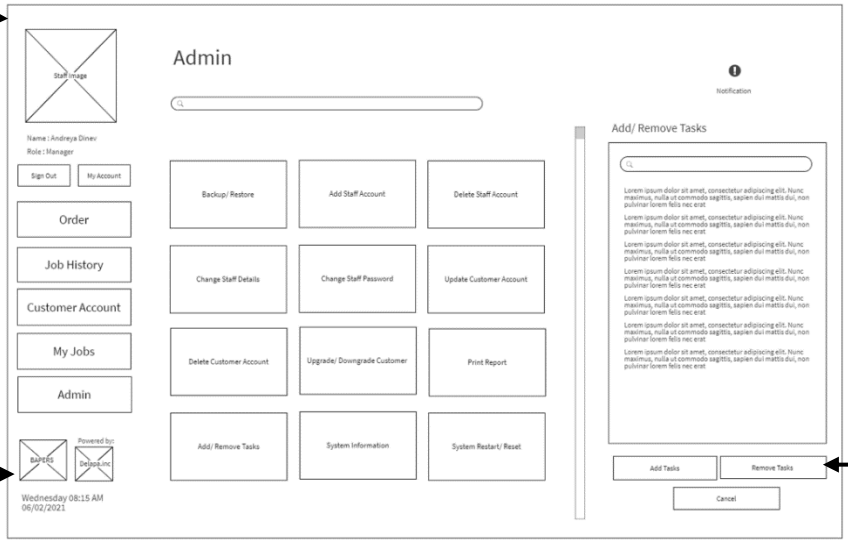
Click on the 'Confirm Change' button on the bottom left to upgrade or downgrade a customer account.

Click on the 'Print Report' button if you want to print a report.



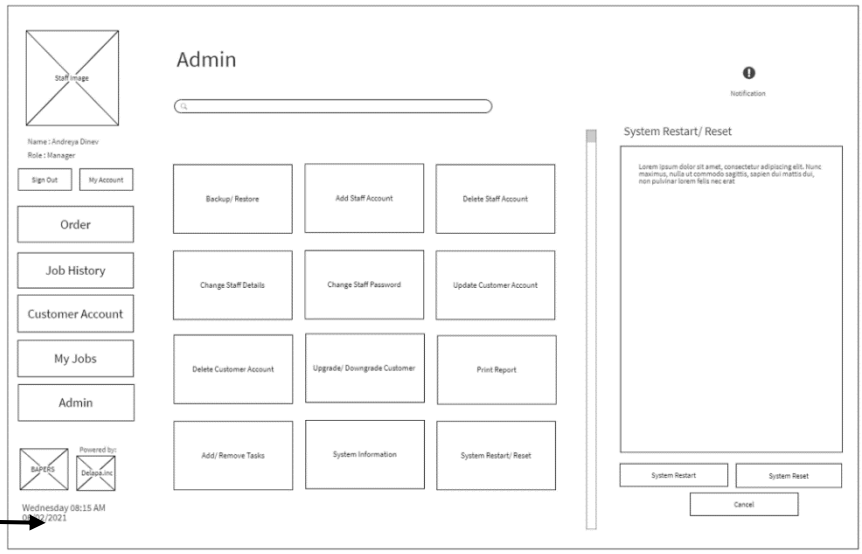
If you want to print staff report then click on the 'Print Staff Report' and if you want to print shift report then click on the 'Print Shift Report' button. Both buttons can be found on the bottom left.

Click on the 'Add/Remove Tasks' button if you want to add or remove tasks.



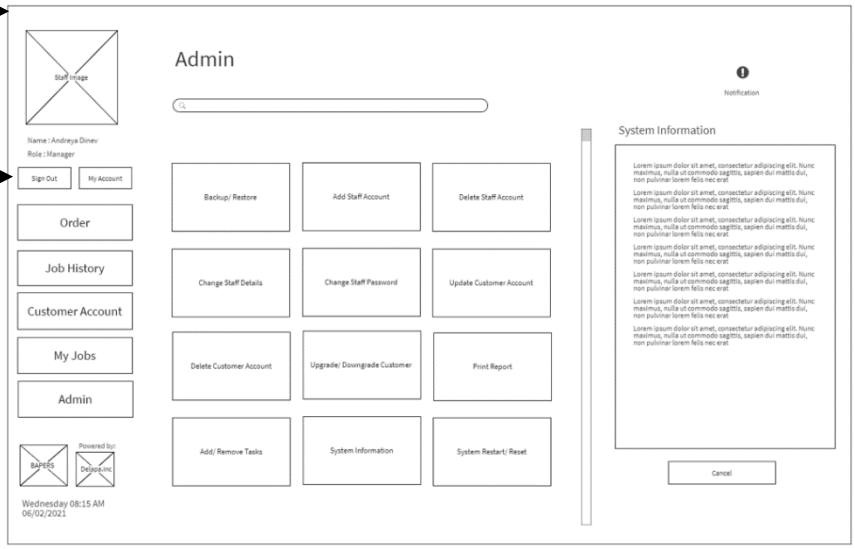
If you want to add tasks, then click on the 'Add Tasks' and if you want to remove tasks then click on the 'Remove Tasks' button. Both buttons can be found on the bottom left.

Click on the 'System Restart/Reset' button if you want to reset or restart the system.

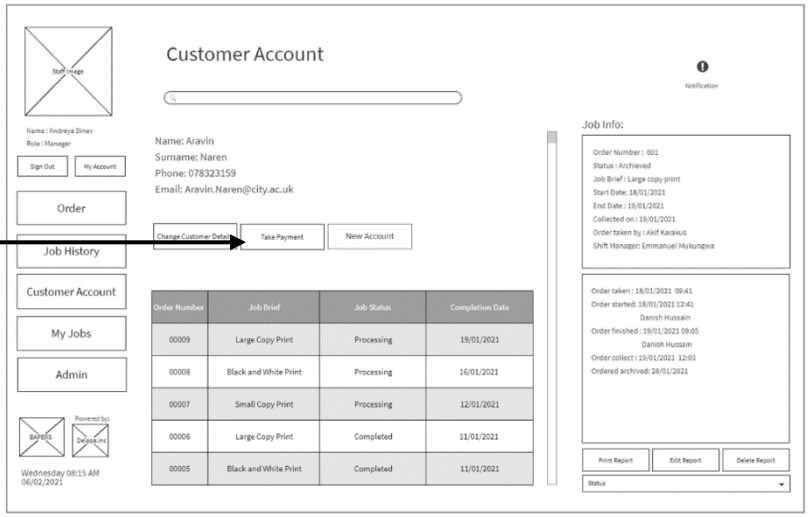


If you want to restart the system, then click on the 'System Restart' and if you want to reset the system then click on the 'System Reset' button. Both buttons can be found on the

Click on the 'System Information' button if you want to view the system information.



If you want to view your account, then all you must click on the 'My Account' button and it will take you to the My Account page. If you want to sign out, then click on the 'Sign Out butting' and it will take you back to the login page.

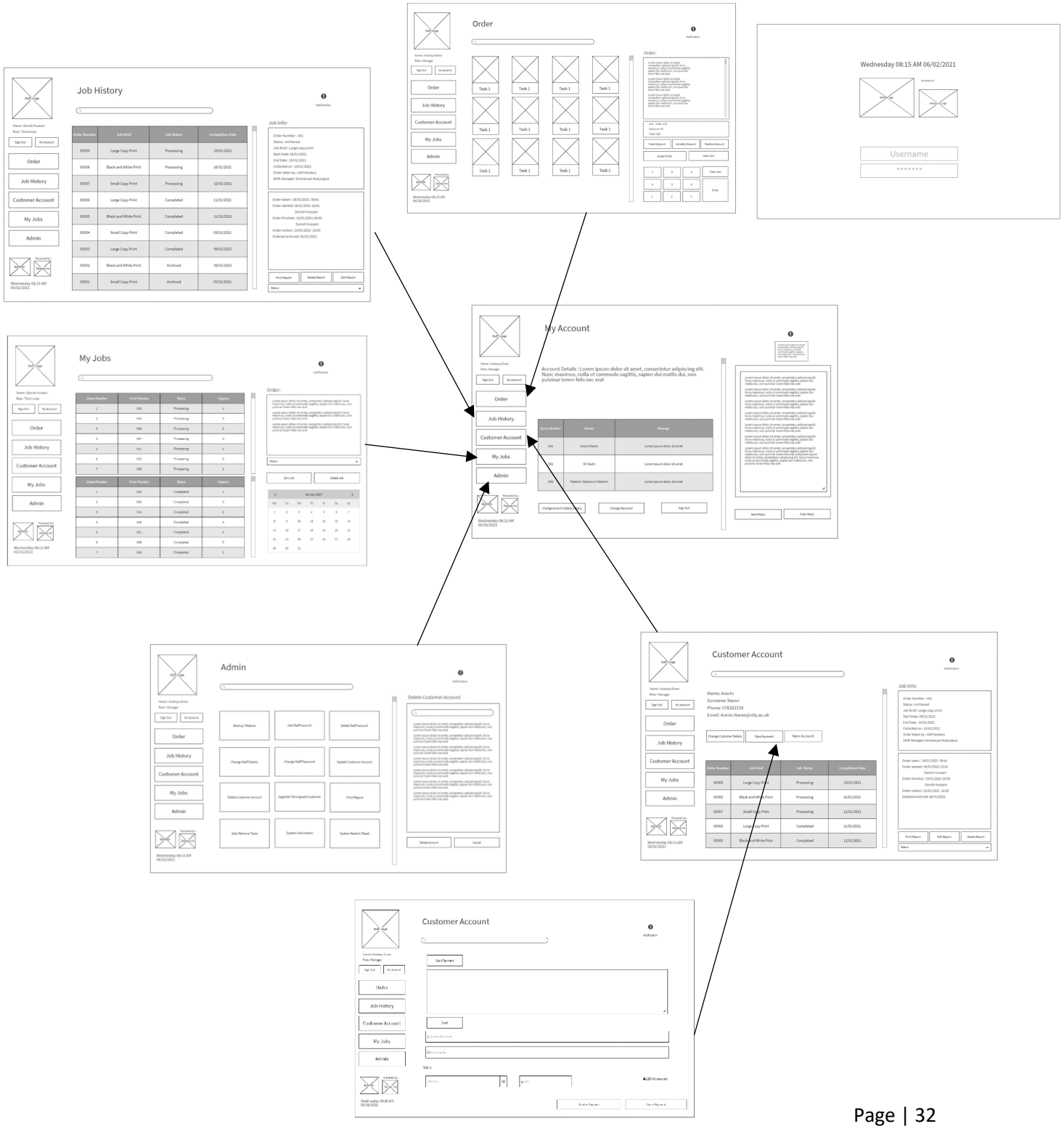


In order to take payments, the staff member needs to look up the customers account and select "Make Payment". This will display the amount due and navigate to the payment page

Customers have the option to pay by card or cash. The payment is taken and stored, after which the GUI navigates back to the Customer Account page

5.2 Overview of the GUI

This is the overview of the wireframe of the sytem. It shows the bigger picture of where pages are in reltionship to each other.



6.1 Conclusion

This is the design and analysis of the system, it has shown both the overview and the specifics of the systems and how we intend to implement them. In the next part we will create a working demonstartion of the system. For any further enquieres please contact our team.