A logo of a planet with yellow rings around it

Description automatically generated

**GROUP ASSIGNMENT**

**INTRODUCTION TO OBJECT & ORIENTED PROGRAMMING**

APPLICATION DOCUMENTATION

|  |  |
| --- | --- |
| **Group Members** | **TP Number** |
| DEMETRIUS KOH ZIHIN | TP069529 |
| SHRINIVAS HOH | TP071957 |
| CHAW YAN CHENG | TP078508 |

**Lecturer:** DR. KULOTHUNKAN A/L PALASUNDRAM

**Module:** INTRODUCTION TO OBJECT & ORIENTED PROGRAMMING

**Module Code:** CT044-3-1-IOOP

**Intake Code:** APU1F2309CS(AI)

**Hand Out Date:** 18th March 2024

**Hand In Date:** 9th June 2024

Table of Contents

[Table of Figures 1](#_Toc168847022)

[Storyboard 3](#_Toc168847023)

[Login Page 3](#_Toc168847024)

[Admin Pages 4](#_Toc168847025)

[Manager Pages 11](#_Toc168847026)

[Worker Pages 15](#_Toc168847027)

[Customer Pages 18](#_Toc168847028)

[Database Concepts 21](#_Toc168847029)

[Use-case Diagram 22](#_Toc168847030)

[Class Diagrams 24](#_Toc168847031)

[Login Page 24](#_Toc168847032)

[Admin Pages 27](#_Toc168847033)

[Worker Pages 32](#_Toc168847034)

[Manager Pages 34](#_Toc168847035)

[Customer Pages 36](#_Toc168847036)

[Conclusion 39](#_Toc168847037)

[Appendix 40](#_Toc168847038)

[Test Plan & Test Cases 40](#_Toc168847039)

[Workload Matrix 43](#_Toc168847040)

# Table of Figures

[Figure 1: Log in page 4](#_Toc166062433)

[Figure 2: Admin’s main page and update user profile page 5](#_Toc166062434)

[Figure 3: Admin’s forms and message boxes for updating/deleting users 5](#_Toc166062435)

[Figure 4: Admin’s interface for generating reports part I 6](#_Toc166062436)

[Figure 5: Admin's interface for generating reports part II 6](#_Toc166062437)

[Figure 6: Manager's main page and interface for assigning workers 7](#_Toc166062438)

[Figure 7: Updating personal profile interface for all roles 7](#_Toc166062439)

[Figure 8: Worker's main page and interface for updating request statuses 8](#_Toc166062440)

[Figure 9: Worker's message boxes for validations 8](#_Toc166062441)

[Figure 10: Customer's main page and interface for submitting a request 9](#_Toc166062442)

[Figure 11: Forms for viewing customer's requests, formula for calculating prices and message box for request submission 9](#_Toc166062443)

[Figure 12: Database concept for SQL part I 10](#_Toc166062444)

[Figure 13: Database concept for SQL part II 10](#_Toc166062445)

# Storyboard

## Login Page

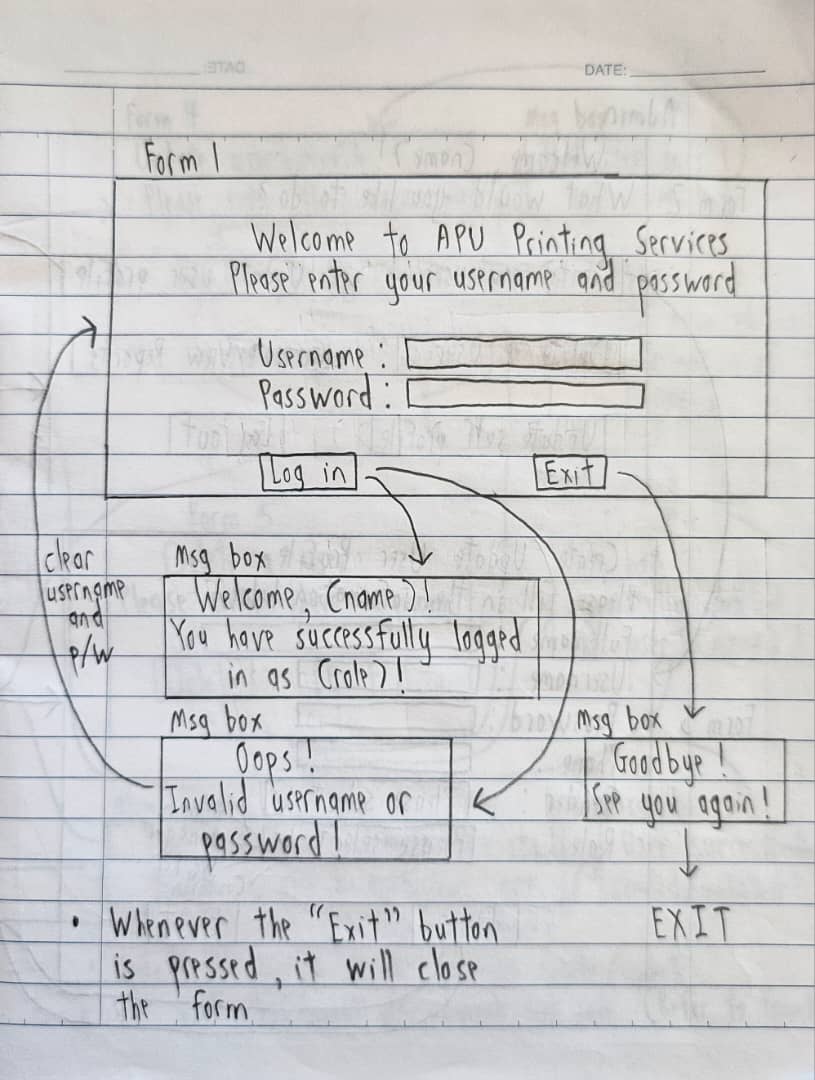


Figure 1: Log in page

**Form 1 (Login Page)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblWelcomeMsg | Displays the welcome message text |
| Label | lblUsername | Displays the username text |
| Label | lblPassword | Displays the password text |
| Textbox | txtboxUsername | This is where the user enters their username |
| Textbox | txtboxPassword | This is where the user enters their password |
| Button | btnLogin | When clicked, it will direct to the respective forms based on the user’s role. If the username or password is wrong, it will display an error message |
| Button | btnExit | When clicked, it will exit the entire application |

## Admin Pages

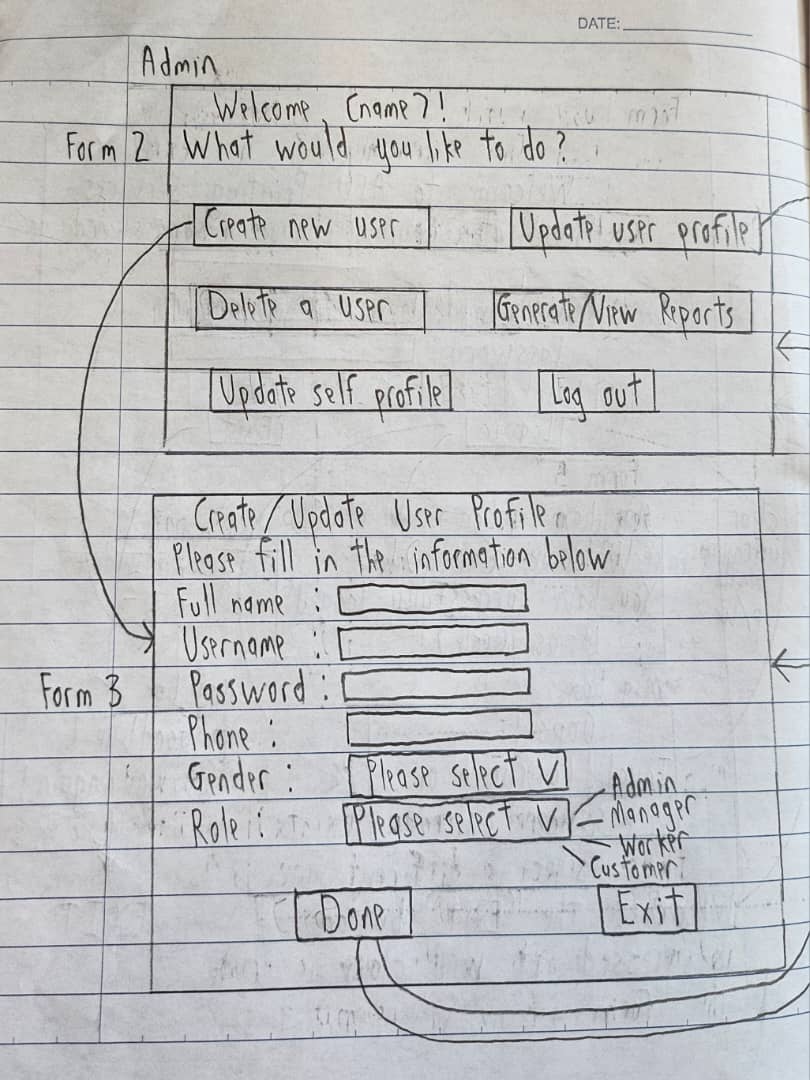


Figure 2: Admin’s main page and update user profile page

**Form 2 (Admin Home Page)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblWelcomeMsg | Displays the welcome message text, the “(name)” will change based on the user’s name that is using the application |
| Button | btnCreateNewUser | When clicked, it will redirect to the create new user form |
| Button | btnUpdateUserProfile | When clicked, it will redirect to the update user profile form |
| Button | btnDeleteAUser | When clicked, it will redirect to the delete user form |
| Button | btnGenerateReports | When clicked, it will redirect to the generate reports form |
| Button | btnUpdatePersonalProfile | When clicked, it will redirect to the update personal profile form |
| Button | btnLogout | When clicked, it will log the user out from the application |

**Form 3 (Create/Update User Profile)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Displays the title of the form and tells the user what to do |
| Label | lblFullname | Displays the full name text |
| Label | lblUsername | Displays the username text |
| Label | lblPassword | Displays the password text |
| Label | lblPhone | Displays the phone text |
| Label | lblGender | Displays the gender text |
| Label | lblRole | Displays the role text |
| Textbox | txtboxFullname | This is where the user enters their full name |
| Textbox | txtboxUsername | This is where the user enters their username |
| Textbox | txtboxPassword | This is where the user enters their password |
| Textbox | txtboxPhone | This is where the user enters their phone number |
| Combo Box | cbGender | Allows the user to decide the gender, whether to be “Male” or “Female” |
| Combo Box | cbRole | Allows the user to decide the role, whether to be “Admin”, “Manager”, “Worker” or “Customer” |
| Button | btnDone | When clicked, it will create or update the user’s profile. If some of the fields are empty or invalid, it will prompt an error message |
| Button | btnExit | When clicked, it will close this form and return to the admin home page |

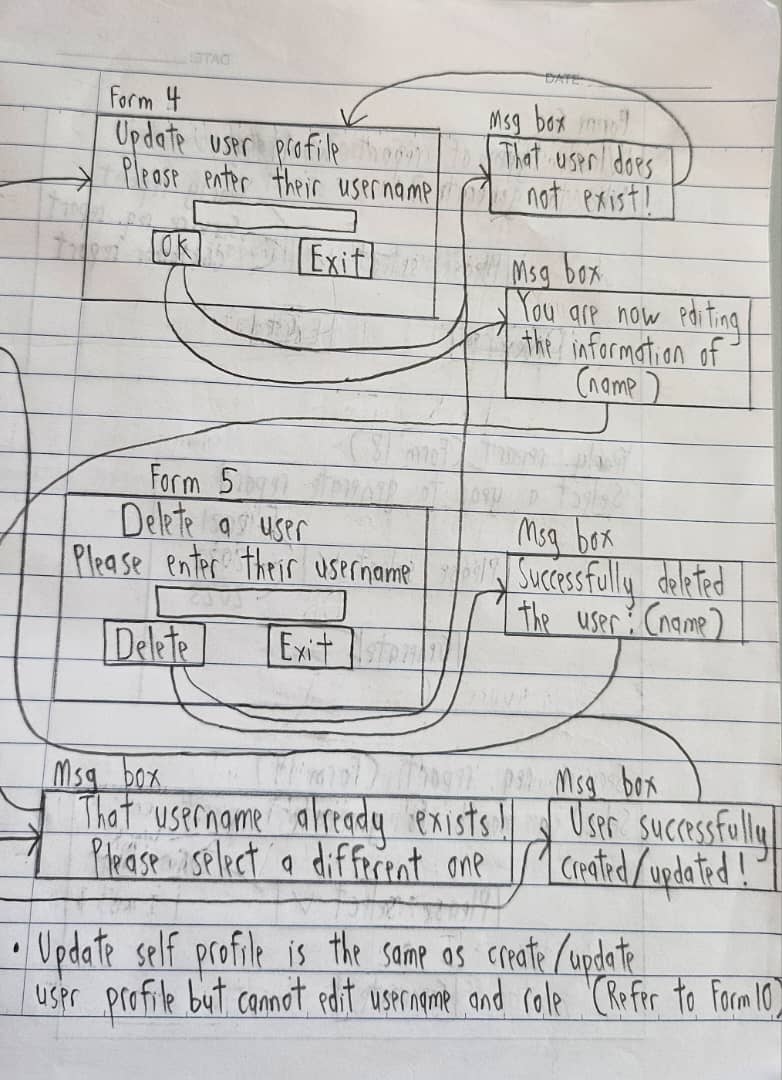


Figure 3: Admin’s forms and message boxes for updating/deleting users

**Form 4 (Username to edit confirmation)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Element Description |
| Label | lblInfo | Shows the title of the form and tells the user on what to enter in the textbox |
| Textbox | txtboxUsername | This is where the user enters the username of the user that they want to edit or update |
| Button | btnOK | When clicked, it will redirect to the update user profile page. If the username textbox is empty or invalid, it will prompt an error message |
| Button | btnExit | When clicked, it will close this form and return to the admin home page |

**Form 5 (Delete a User)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Shows the title of the form and tells the user on what to enter in the textbox |
| Textbox | txtboxUsername | This is where the user enters the username of the user that they want to edit or update |
| Button | btnDelete | When clicked, it will delete the respective user based on the username entered. If the username textbox is empty or invalid, it will prompt an error message |
| Button | btnExit | When clicked, it will close this form and return to the admin home page |

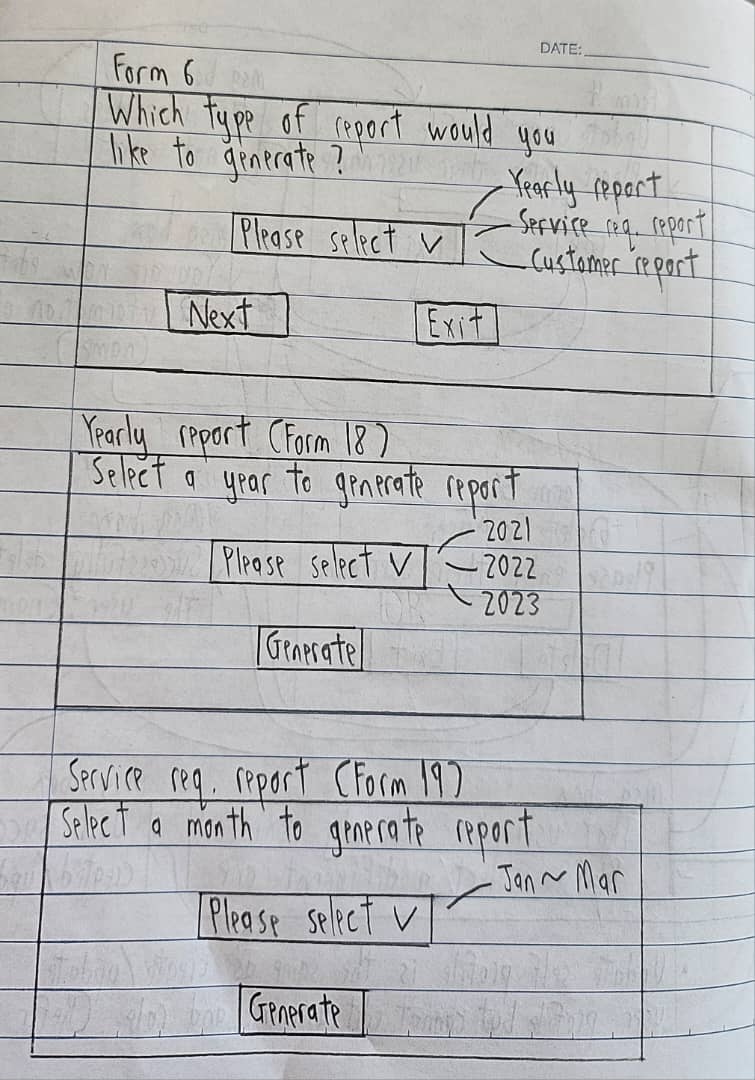


Figure 4: Admin’s interface for generating reports part I

**Form 6 (Generate Reports)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Asks the user on which type of report they want to generate |
| Combo Box | cbReportType | Allows the user to decide the type of report they want to generate, whether to be “Yearly Report”, “Service Request Report”, or “Customer Report” |
| Button | btnNext | When clicked, it will redirect to the year selection form if “Yearly Report” is selected, or the month selection form if “Service Request Report”/”Customer Report” is selected |
| Button | btnExit | When clicked, it will close this form and return to the admin home page |

**Form 18 (Year Selection – Yearly Report)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Tells the user on what to do |
| Combo Box | cbYearSelection | Allows the user to select the year |
| Button | btnGenerate | When clicked, it will generate the report based on the year selected. If no year is selected, it will prompt an error message |

**Form 19 (Month Selection – Service Request Report)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Tells the user on what to do |
| Combo Box | cbMonthSelection | Allows the user to select the month |
| Button | btnGenerate | When clicked, it will generate the report based on the month selected. If no month is selected, it will prompt an error message |

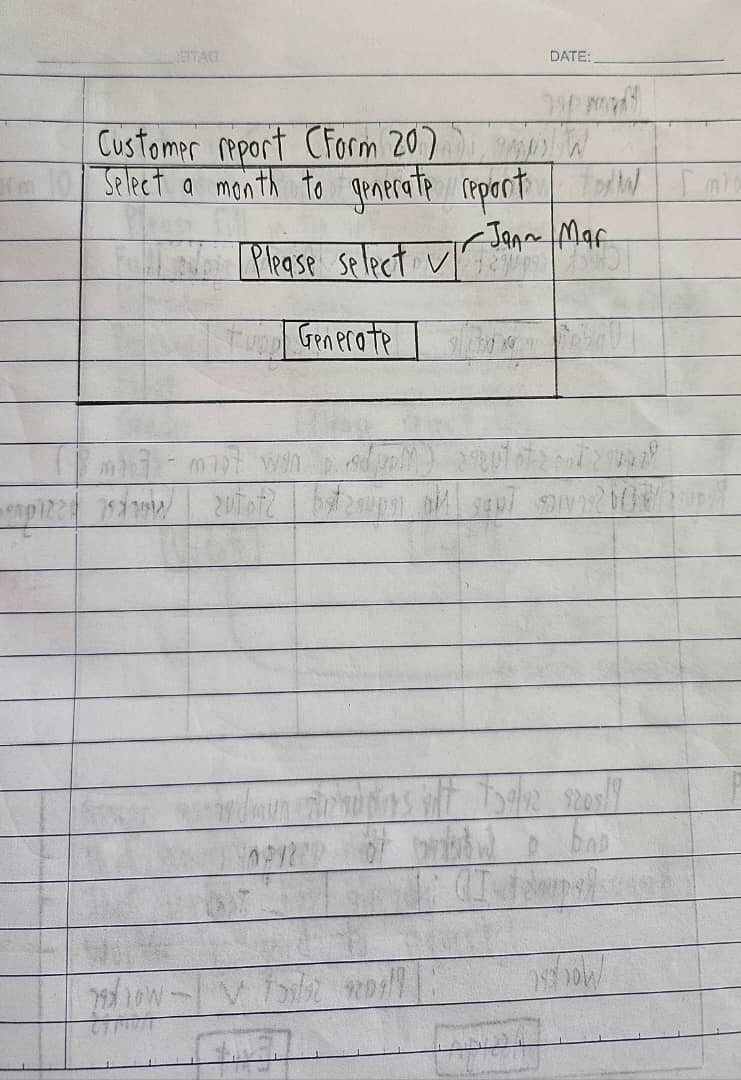


Figure 5: Admin's interface for generating reports part II

**Form 20 (Month Selection – Customer Report)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Tells the user on what to do |
| Combo Box | cbMonthSelection | Allows the user to select the month |
| Button | btnGenerate | When clicked, it will generate the report based on the month selected. If no month is selected, it will prompt an error message |

## Manager Pages

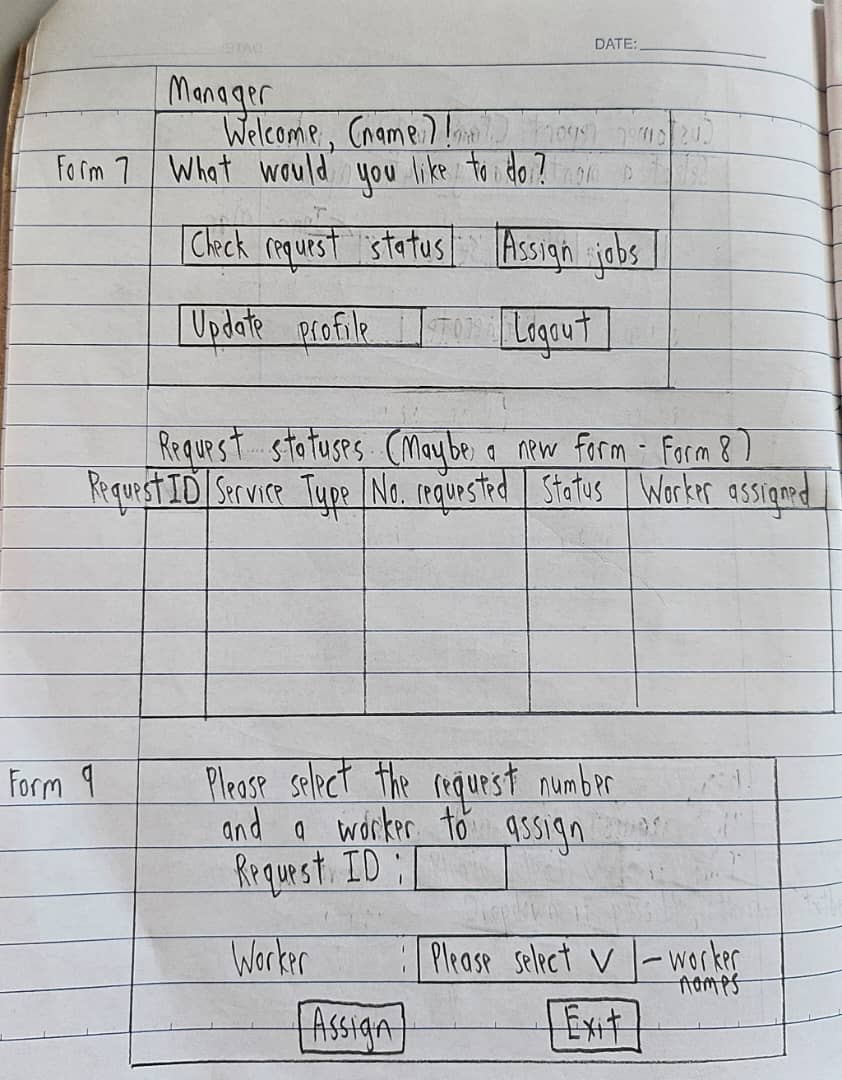


Figure 6: Manager's main page and interface for assigning workers

**Form 7 (Manager Home Page)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblWelcomeMsg | Displays the welcome message text, the “(name)” will change based on the user’s name that is using the application |
| Button | btnCheckRequestStatus | When clicked, it will redirect to the check request status form |
| Button | btnAssignJobs | When clicked, it will redirect to the assign jobs form |
| Button | btnUpdatePersonalProfile | When clicked, it will redirect to the update personal profile form |
| Button | btnLogout | When clicked, it will log the user out from the application |

**Form 8 (Request Status)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Data Grid View | datagridviewRequestStatus | Displays the list of requests, consisting of:   1. Its ID 2. Service type 3. Number of requests 4. Its urgency 5. Its current status 6. Whether it has a worker assigned |

**Form 9 (Assign Jobs)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Displays the title of the form and tells the user what to do |
| Label | lblRequestID | Displays the request ID text |
| Label | lblWorker | Displays the worker text |
| Textbox | txtboxRequestID | Allows the user to enter the request ID |
| Combo Box | cbWorkerName | Allows the user to choose the worker from the list of worker names |
| Button | btnAssign | When clicked, it will assign the chosen worker to the chosen request. If either of the fields are empty or invalid, it will prompt an error message |
| Button | btnExit | When clicked, it will close this form and return to the manager home page |

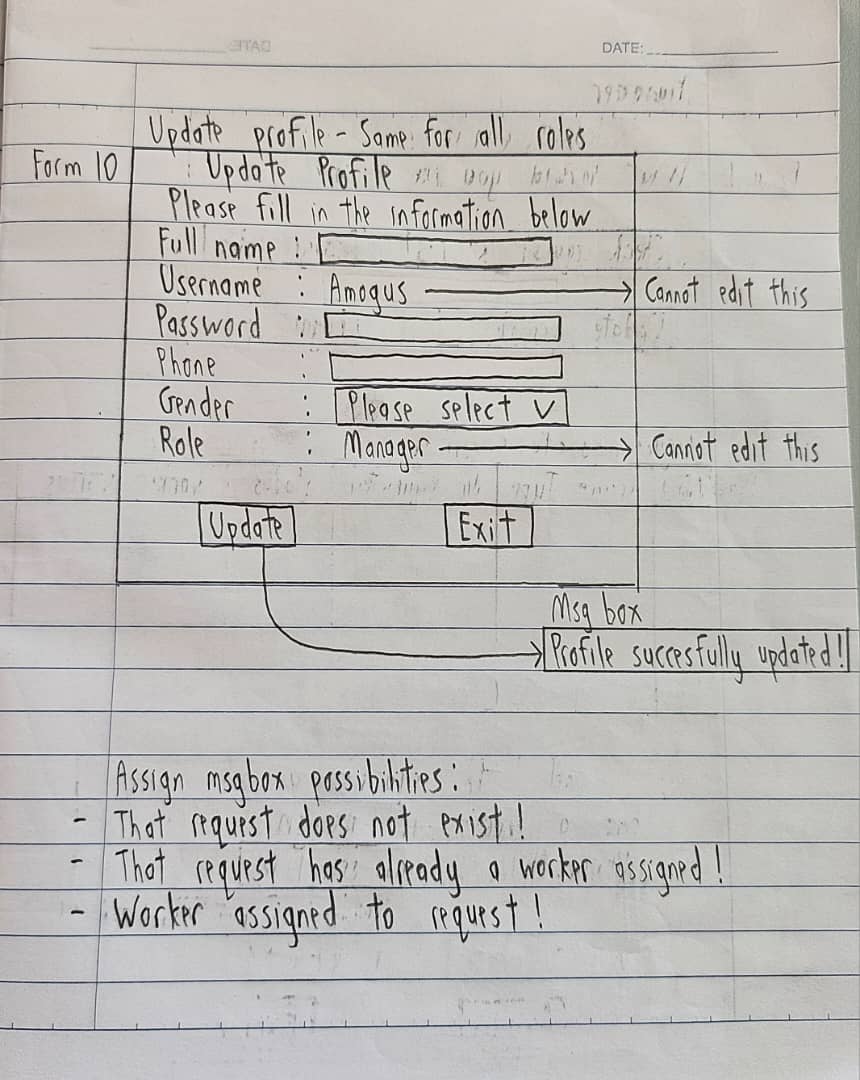


Figure 7: Updating personal profile interface for all roles

**Form 10 (Update Personal Profile)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Element Description |
| Label | lblInfo | Displays the title of the form and tells the user what to do |
| Label | lblFullname | Displays the full name text |
| Label | lblUsername | Displays the username text |
| Label | lblFixedUsername | Displays the current username of the user, this cannot be changed |
| Label | lblPassword | Displays the password text |
| Label | lblPhone | Displays the phone text |
| Label | lblGender | Displays the gender text |
| Label | lblRole | Displays the role text |
| Label | lblFixedRole | Displays the current role of the user, this cannot be changed |
| Textbox | txtboxFullname | This is where the user enters their full name |
| Textbox | txtboxPassword | This is where the user enters their password |
| Textbox | txtboxPhone | This is where the user enters their phone number |
| Combo Box | cbGender | Allows the user to decide the gender, whether to be “Male” or “Female” |
| Button | btnUpdate | When clicked, it will update the user’s profile. If some of the fields are empty or invalid, it will prompt an error message |
| Button | btnExit | When clicked, it will close this form and return to the home page depending on the user’s role |

## Worker Pages

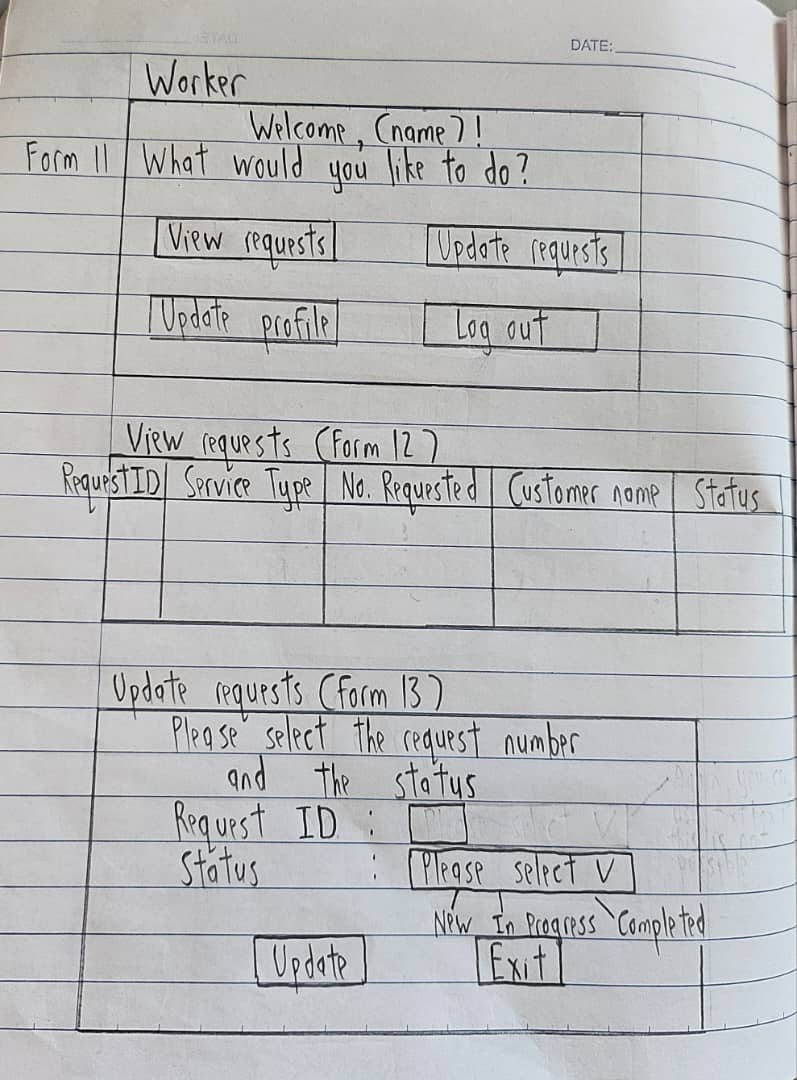


Figure 8: Worker's main page and interface for updating request statuses

**Form 11 (Worker Home Page)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblWelcomeMsg | Displays the welcome message text, the “(name)” will change based on the user’s name that is using the application |
| Button | btnViewRequests | When clicked, it will redirect the user to the view requests form |
| Button | btnUpdateRequests | When clicked, it will redirect the user to the update requests form |
| Button | btnUpdatePersonalProfile | When clicked, it will redirect the user to the update personal profile form |
| Button | btnLogout | When clicked, it will log the user out from the application |

**Form 12 (View Requests)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Data Grid View | datagridviewRequests | Displays the list of requests assigned to the user, consisting of   1. Its ID 2. Service type 3. Number of requests 4. The customer who made the request 5. Its current status |

**Form 13 (Update Request Status)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Displays the title of the form and tells the user what to do |
| Label | lblRequestID | Displays the request ID text |
| Label | lblStatus | Displays the status text |
| Textbox | txtboxRequestID | Allows the user to enter the request ID |
| Combo Box | cbStatus | Allows the user to choose the status whether to be “In Progress” or “Completed” |
| Button | btnUpdate | When clicked, it will update the selected request with the chosen status. If either of the fields are empty or invalid, it will prompt an error message |
| Button | btnExit | When clicked, it will close this form and return to the worker home page |

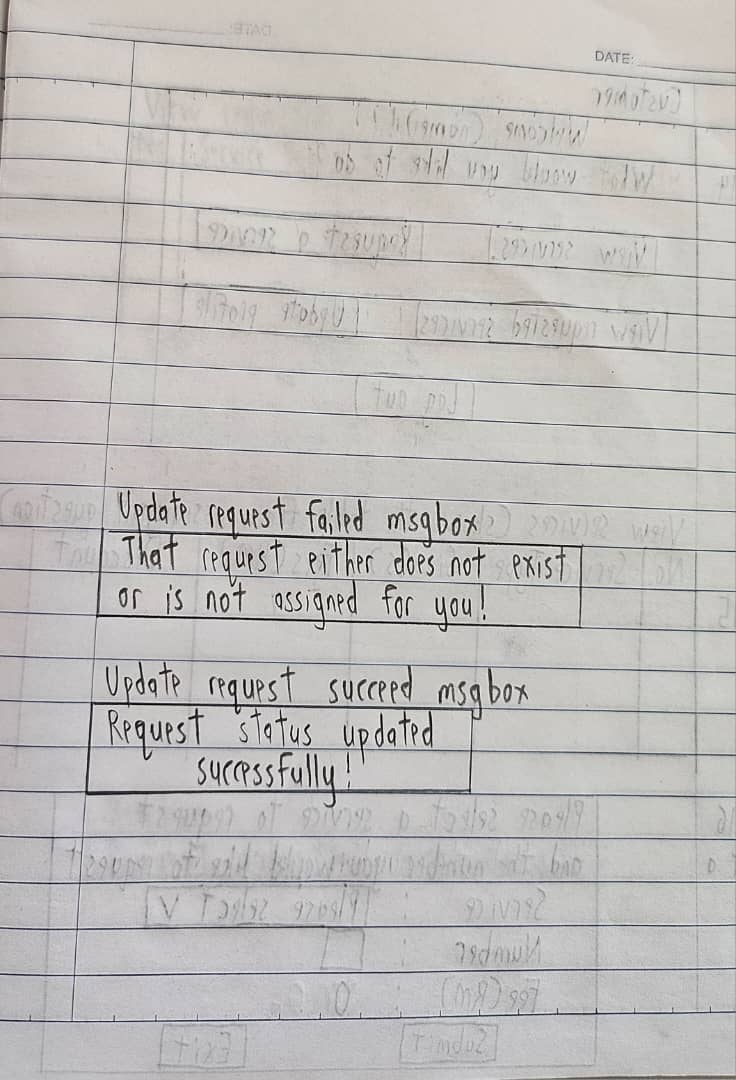


Figure 9: Worker's message boxes for validations

## Customer Pages

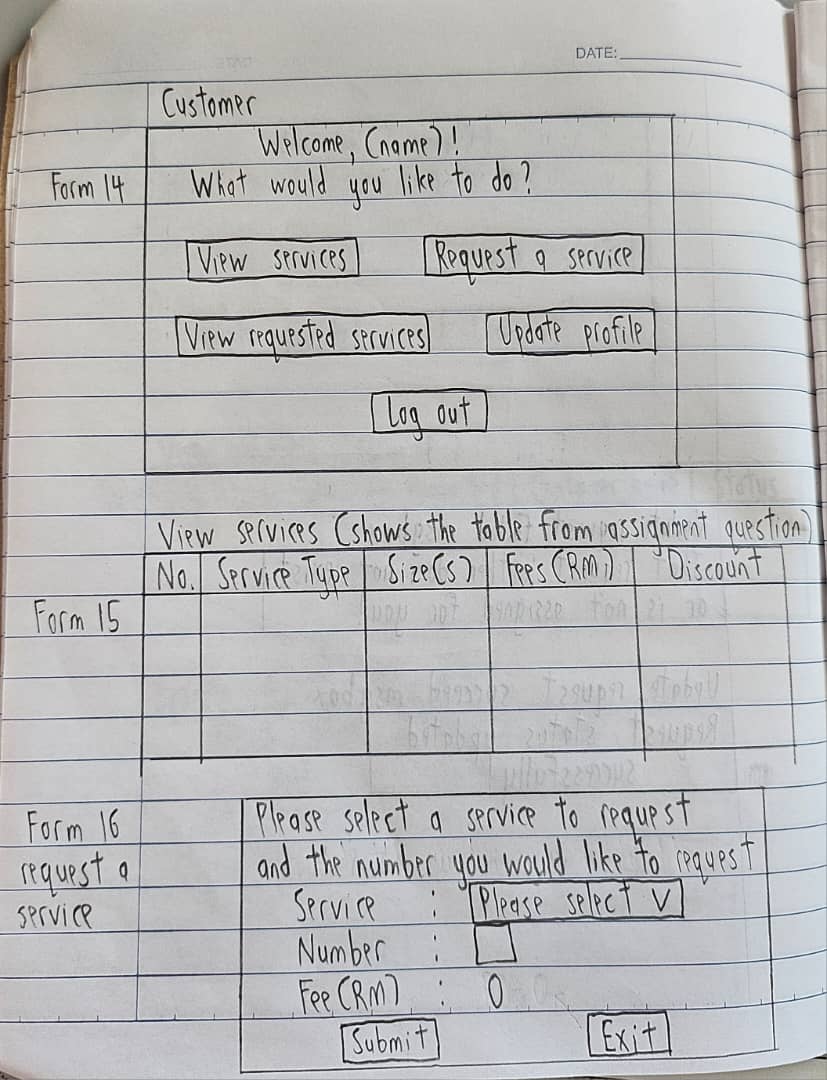


Figure 10: Customer's main page and interface for submitting a request

**Form 14 (Customer Home Page)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblWelcomeMsg | Displays the welcome message text, the “(name)” will change based on the user’s name that is using the application |
| Button | btnViewServices | When clicked, it will redirect the user to the view services form |
| Button | btnRequestService | When clicked, it will redirect the user to the request service form |
| Button | btnViewRequestedServices | When clicked, it will redirect the user to the view requested services form |
| Button | btnUpdatePersonalProfile | When clicked, it will redirect the user to the update personal profile form |
| Button | btnLogout | When clicked, it will log the user out from the application |

**Form 15 (View Services)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Data Grid View | datagridviewServices | Displays the list of services provided, consisting of:   1. Service number 2. Service name 3. Available size(s) 4. Fees 5. Discount (If any) |

**Form 16 (Request a Service)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblInfo | Displays the title of the form and tells the user what to do |
| Label | lblService | Displays the service text |
| Label | lblNumber | Displays the number text |
| Label | lblFee | Displays the fee text |
| Label | lblCalculatedFee | Displays the calculated fees, based on the type of service selected and the number of that service |
| Textbox | txtboxNumofService | Allows the user to insert their number of services they would like to request |
| Combo Box | cbService | Allows the user to choose the type of service to request |
| Button | btnSubmit | When clicked, it will submit the request based on the service selected and number of the service to the database. If either fields above are empty or invalid, it will prompt an error message |
| Button | btnExit | When clicked, it will close this form and return to the worker home page |

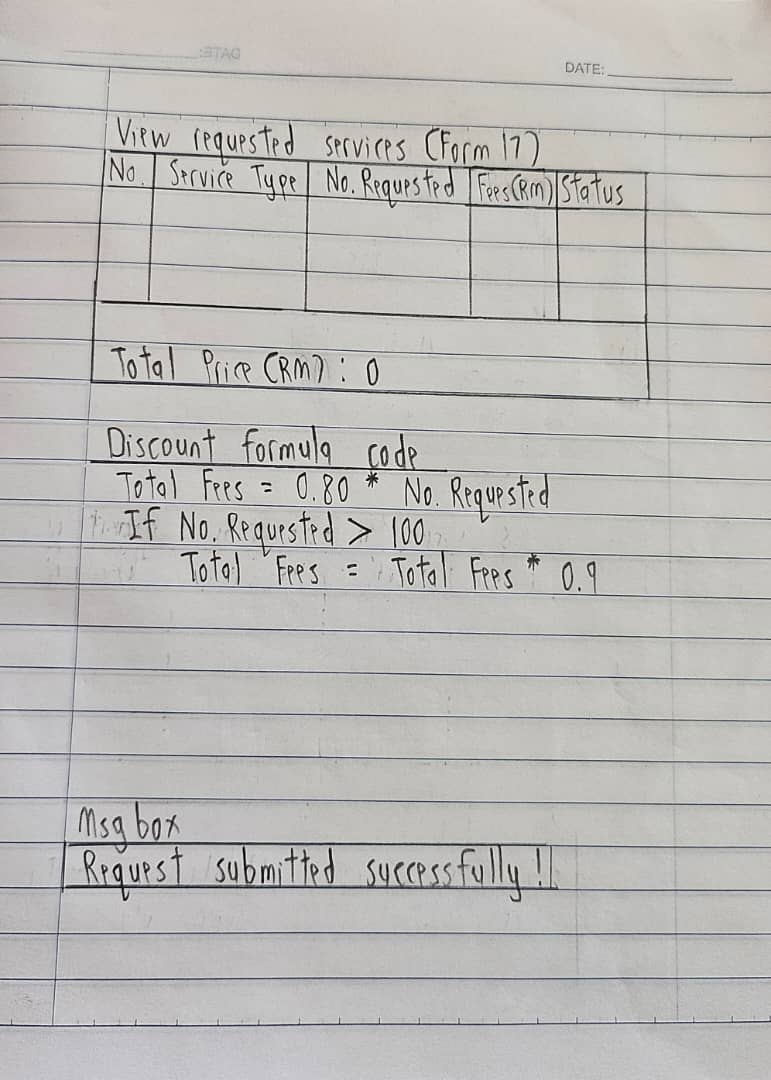


Figure 11: Forms for viewing customer's requests, formula for calculating prices and message box for request submission

**Form 17 (View Requested Services)**

|  |  |  |
| --- | --- | --- |
| Element Type | Element Name | Description |
| Label | lblTotalPrice | Displays the total price text |
| Label | lblCalculatedPrice | Displays the calculated total price based on all the requests submitted |
| Data Grid View | datagridviewRequestedServices | Displays the list of requested services by the user, consisting of:   1. Request ID 2. Service type 3. Number requested 4. Fees 5. Its status |

## Database Concepts

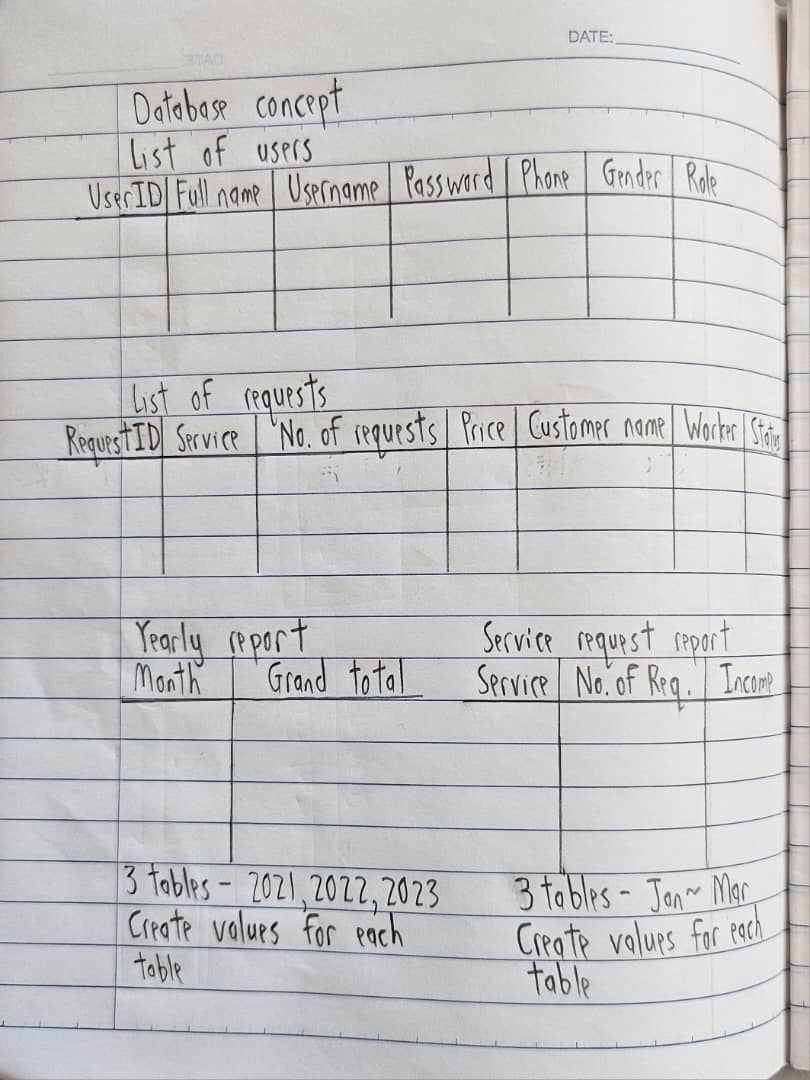


Figure 12: Database concept for SQL part I

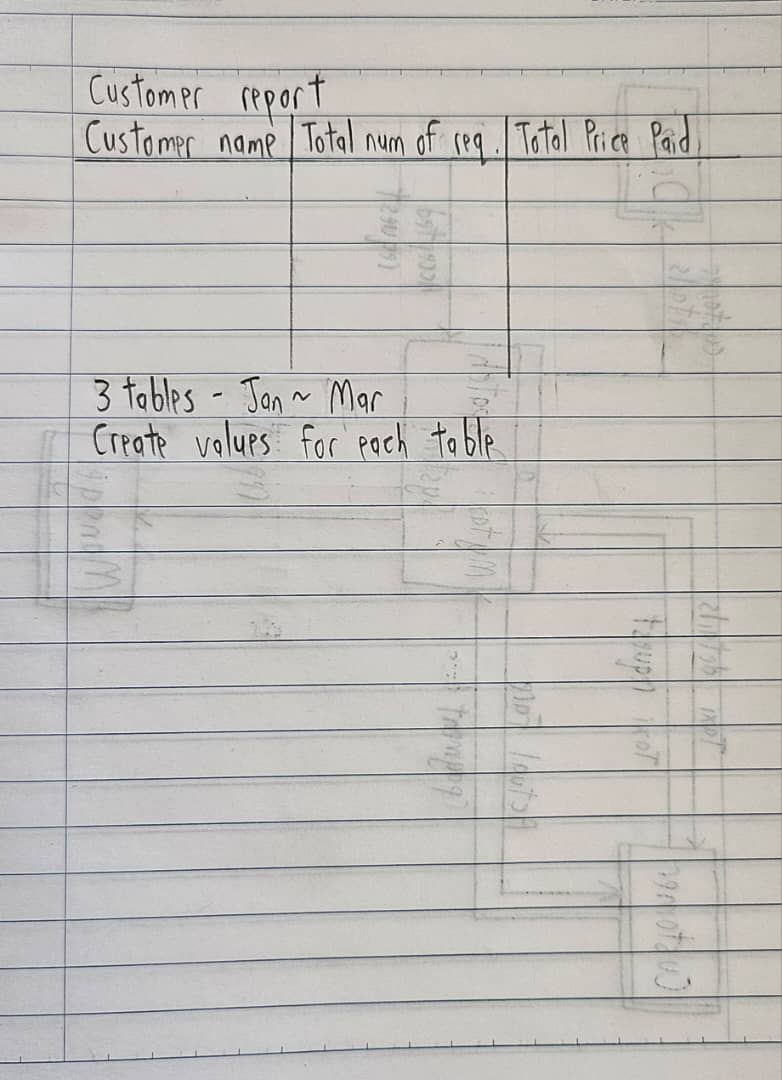


Figure 13: Database concept for SQL part II

# Use-case Diagram

A diagram of a network

Description automatically generated

A diagram of a flowchart

Description automatically generated

# Class Diagrams

## Login Page

A screenshot of a computer

Description automatically generated

1. The fields are created for the users (Manager, Admin, Worker, customer) to fill in their password and usernames in the following textboxes
2. Properties are created for the login form to check if the username is valid or not
3. Methods perform a specific task. The login button helps to redirect to the next required page depending on who is the user. The quit button helps the user to exit the application overall.

A screenshot of a computer

Description automatically generated

**Fields**

\_Fullname: Stores the full name of the user

\_Gender: Stores the gender of the user (Female or male)

\_Password: Stores a unique password which has numbers and words

\_Phone: Stores users phone details

\_Role: Stores the role of the user (Manager, Admin, worker, customer)

\_UserID: Stores the user ID of the user

\_Username: Stores the username of the user

**Methods**

The user is a constructor have two overloaded

* One of it are initialize all the field (userid, fullname, username, password, phone, gender, role) of user class.
* Another are initialize essential field (fullname, username, password) and set other as default

## Admin Pages

A screenshot of a computer

Description automatically generated

1. The field exists for the admin to store a value or a reference to an object inside a class
2. A light/dark mode is featured for the admin to customize the form’s display settings
3. Methods perform a specific task for the admin to add a new user, delete a user, exit the admin form, generate reports according to the types of requests, month and year the admin requires. Besides that, the admin has the authority to update their own profile and the other users’ profiles.

A screenshot of a computer

Description automatically generated

1. The field exists for the add user profile form to create a user profile, exit the form, select a role, select the gender, fill in their full names, password, phone details, and username.
2. Methods perform a specific task for the add user profile where it creates a new added user profile and sends it to the sql to be stored
3. “ValidateUsername” method is important to ensure that the user fills in the correct username or else it would display an error message to fill in again.

A screenshot of a computer

Description automatically generated

1. This edits user profile class form applies to not only admins, but to other users including manager, worker and customer.
2. Any information can be edited depending on the users’ preferences. They can edit a single or multiple part of their info as they prefer.

A screenshot of a computer program

Description automatically generated

1. The delete user profile class form helps admins to delete other users
2. However, they are unable to delete themselves as an error handling message will appear if they somehow execute this.
3. A “datagridviewUserList” is shown in the same form so that the Admin can view which user they want to delete specifically

A screenshot of a computer program

Description automatically generated

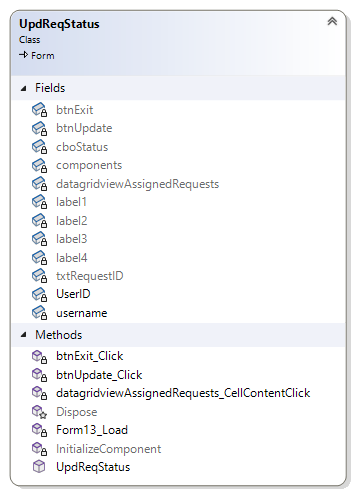
1. The admin can view reports when selecting the specific report, month and year
2. Once the admin selects all the given criteria, the “datagridviewReport” will display the report that is generated from the SQL database.

## Worker Pages

A screenshot of a computer program

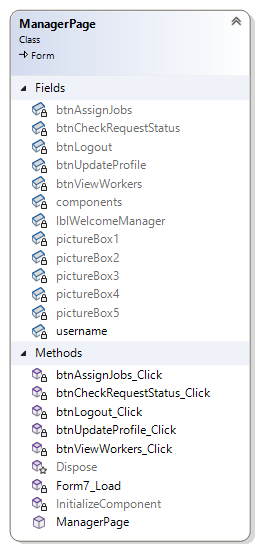
Description automatically generated

1. Once the worker types the username and password through the login form, the worker will be redirected to the worker page class form where they can logout which will redirect them to the login page back, update their profile, update and view requests that comes from the customers’ demand
2. The worker can update their own profile according to their preferences.



1. The “datagridviewAssignedRequests” helps the worker to view the Request ID of the customer so that they can select the specific customer to update.
2. The Worker can update the customer’s request service from the default “Assigned” to either “In Progress” or “Completed” so that the customer can view their updated/ current request status.

## Manager Pages



1. Once the worker types the username and password through the login form, the worker will be redirected to the worker page class form where they can logout which will redirect them to the login page back, update their profile, update and view requests that comes from the customers’ demand
2. The manager can update their own profile according to their preferences

A screenshot of a computer

Description automatically generated

1. The manager is able to view customer requests so that the manager can assign it to the available workers
2. The manager can also view completed requests by ticking the “ShowCompletedRequests” checkbox

A screenshot of a computer

Description automatically generated

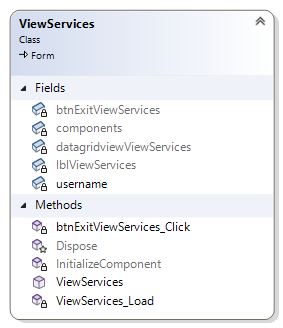
1. The manager can assign customers’ requests to workers by clicking on the assign job button
2. It will show all requests that customers have made and all the workers registered
3. The manager can assign a request to a worker by inserting the requestID and workerID

## Customer Pages

A screenshot of a computer program

Description automatically generated

1. The customer can choose to view available services, request a service, view their requested services and update their own profile

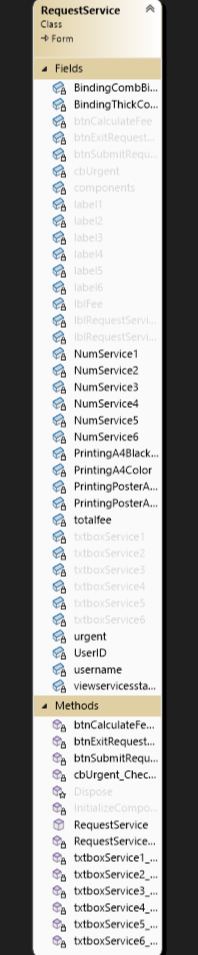


1. The customer is able to view all the services provided

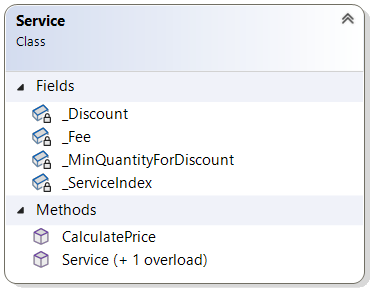
A screenshot of a computer

Description automatically generated

1. The customer can view all their requested services, including the current status of each service
2. The customer can choose to cancel their requests if the request status is still “new”
3. The customer can also choose to filter the results to only show the completed requests



1. The customer is able to insert the number of requests for each service in the provided text box
2. The customer then clicks the on “btnCalculateFee” to calculate their fees
3. The customer then clicks on “btnSubmitRequest” to submit their request(s), which will be updated in the database
4. The customer can also tick the “cbUrgent” checkbox to state that their request(s) is urgent, this will impose an additional 10% to their fee



**Fields**

\_Discount: Store the discount as a factor, like if 20% discount it store as 0.8

\_Fee: Store the base fee of the service

\_MinQuantityForDiscount: Hold the minimum quantity for the discount to apply

\_ServiceIndex: Unique identifier for the service

**Methods**

Service is a constructor which are two overloaded

* One of the service included \_MinQuantityForDiscount and \_Discount
* One of it initialize \_MinQuantityForDiscount and Discount to 0 which is not included

CalculatePrice is a method used to calculate TotalFee

* Price that applied discount multiply quantity

# Conclusion

The development of APU Printing Service Management System allows us to demonstrate our ability to apply basic object-oriented programming techniques to create a useful application. The extensive features of the system which are suited to the various user role like Admin, Customer, Manager and Worker, and make it possible for each user to interact with the system in a way that best fit their requirements. And also, we design and implement applications that are practical by including functions like generate report, update user profile and request service into a unified system.

However, there is still room for improvement. The system can implement more robust error handling to prevent system crashes and improve security procedures to safeguard user data. In future time, our system has potential to implement more advanced reporting features, like ability to predict service demands using predictive analytics. Adding mobile compatibility to improve the system usability and accessibility. And adding automatic service update notifications, that may enhance the system communication and result in improved service management.

In conclusion, even if the APU Printing Service Management System successfully satisfies the project original criteria, it still need ongoing development and maintenance to stay effective in a changing environment.

# Appendix

## Test Plan & Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | Description | Expected Result | Actual Result | Remarks |
| 1 | To test the buttons of the customer’s user interface | Once a button is clicked, a new form will open while closing the previous form | When the button is clicked, a new form opens but does not close the previous form | Add a code that will close the previous form once the button is clicked |
| 2 | To calculate the fees of a service after a choosing a service and the number of requests | Once the “Calculate” button is clicked, it will calculate the total fee based on the chosen service and the number of request(s) | The fee is calculated but its not according to the chosen service | Edit the index of the chosen service in the code |
| 3 | To handle errors when choosing a service and the number of requests | When the “Calculate” button is clicked while the service not being chosen or the number of requests inputted is invalid, it will prompt an error to the user | An error is prompted when the user attempts to click on the “Calculate” button while not choosing any service or inputting an invalid number of requests. However, it still accepts negative values for the number of requests | Add a handler where it rejects negative numbers inputted |
| 4 | To test the buttons of the admin’s user interface | Once a button is clicked, a new form will open while closing the previous form | When the button is clicked, a new form opens and the previous form closes | - |
| 5 | To test the buttons of the manager’s user interface | Once a button is clicked, a new form will open while closing the previous form | When the button is clicked, a new form opens and the previous form closes | - |
| 6 | To test the buttons of the worker’s user interface | Once a button is clicked, a new form will open while closing the previous form | When the button is clicked, a new form opens and the previous form closes | - |
| 7 | To test the login function | Once the “Login” button is clicked, the program will detect whether the input username matches with their password. It will also open the correct user interface based on the user’s role | When the “Login” button is clicked, the program fails to detect the input username and password. It is also unable to detect the role | * Ensure that the program is able to access values from the database correctly * Assign the input values to different variables |
| 8 | To connect to the database to generate and view each report | When chosen year, month and report type, it will read data from database and generate a report to allow admin to view the certain report | When chosen year, month and report type, its successful form a correct report | Add a sql code in visual studio to connect to the database |
| 9 | View completed service | When ticked the checkbox, it will show completed service and hide the new and In-progress service | The completed services can be view once the checkbox is ticked. However, when unticked, it will still only show the completed services | Add an event handler to the checkbox to show incomplete requests when the checkbox is unticked |
| 10 | Log Out | When clicked “Log Out” button, it will go back to the login page | When users click the “Log Out” button, its successful return to login page | - |
| 11 | Admin update customer, manager and worker profile | Admin able to edit everything in their profile including username, full name, password, phone, gender and role | The updated details won’t be updated in the database | Declare variables to store the updated information and use them in sql code to update the database |
| 12 | Customer, Worker and Manager update their own profile | They can edit everything except role | The edited part successfully updated at the database | - |
| 13 | Manager assign job to worker | Manager assign job to worker by entering request ID and worker ID | Worker able to receive the job assign by manager. However, the manager is able to input request IDs that does not exist in the database | Add a handler that checks for all the requests IDs in the database and ensure that the request ID inputted by the manager matches one of them |
| 14 | Worker update request status | Worker can update the request from new to In-progress to complete | Worker update successful and manager can view the updated request status. However, the worker is able to update the status of requests that is not assigned to them | Add a handler where is ensures the worker is updating the request that is assigned to them, user ID can be used |
| 15 | View User | Admin can view manager, admin and customer which list down at view user | When click button “View user” it list down all the account that created | - |

## Workload Matrix

|  |  |  |  |
| --- | --- | --- | --- |
|  | Shrinivas Hoh | Chaw Yan Cheng | Demetrius Koh |
| User Interface and Programming | | | |
| Admin |  | 100% |  |
| Customer |  |  | 100% |
| Manager | 100% |  |  |
| Worker | 100% |  |  |
| Database | | | |
| User Database |  | 50% | 50% |
| Request Database | 100% |  |  |
| Admin Reports |  | 50% | 50% |
| Documentation | | | |
| Use-Case diagram | 34% | 33% | 33% |
| Class Diagram | 33% | 34% | 33% |
| Test plan and Test Case | 33% | 34% | 33% |
| Conclusion | 34% | 33% | 33% |
| References | 33% | 33% | 34% |
|  | | | |
| Overall Contribution | 33% | 33% | 34% |
| Signature | Shri | Chaw | Deme |