CRM APPLICATION

Software Requirements Specification

Revision History

Date	Revision	Description	Author
6/15/2021	1.0	Initial Version	GROUP 3
6/22/2021	2.0	Cleanup, formatting, case id's	GROUP 3
6/30/2021	3.0	Additional info in functional requirements, additional diagrams	GROUP 3

Table of Contents

1. Purpose **4**

1.1. Scope 4
1.2. Definitions, Acronyms, Abbreviations 4
1.3. References 4
1.4. Overview 4

2. Overall Description **5**

2.1. Product Perspective 5
2.2. Product Architecture 5
2.3. Product Functionality/Features 5
2.4. Constraints 5
2.5. Assumptions and Dependencies 5

3. Specific Requirements **6**

3.1. Functional Requirements 6
3.2. External Interface Requirements 6
3.3. Internal Interface Requirements 7

4. Non-Functional Requirements **8**

4.1. Security and Privacy Requirements 8
4.2. Environmental Requirements 8
4.3. Performance Requirements 8

1. Purpose

This document outlines the requirements for the Customer Relationship Management (CRM) system

1.1. Scope

This document will catalog the user, system, and hardware requirements for the CRM system. It will not, however, document how these requirements will be implemented.

1.2. Definitions, Acronyms, Abbreviations

CRM - Customer Relationship Management.

Admin - Administrator with total control over the CRM system.

Manager - A higher level user with some higher level privileges.

User - A typical user with no admin level privileges.

Group - A collection of clients belonging to a user.

Client - A customer.

Record - A record or log of data.

1.3. References

Use Case ID	1.0
Use Case Name	System startup
Relevant Requirements	
Primary Actor	IT Admin
Pre-Conditions	System is not running
Post-Conditions	System is up and running
Basic Flow or Main Scenario	1. Check the username and password 2. Check the user's authority 3. Check the system status (runnable) 4. Load the database and open port 5. send a message to tell the user the system is already started
Extensions or Alternate Flows	
Exceptions	1. Invalid account 2. The system is already running 3. Cannot open the port 4. Cannot connect the database
Related Use Cases	

Use Case ID	2.1
Use Case Name	User login
Relevant Requirements	
Primary Actor	User
Pre-Conditions	System is running
Post-Conditions	User logged in
Basic Flow or Main Scenario	1. Ask the user enter the username and password 2. Send the username and passwords to system server 3. System check the username and passwords if they are valid 4. Tell the user he/she is already logged in.

Extensions or Alternate Flows	
Exceptions	1. username and password is not valid
Related Use Cases	

Use Case ID	3.1
Use Case Name	IT Admin edit the user
Relevant Requirements	
Primary Actor	IT Admin
Pre-Conditions	The user is already exist & account is already login
Post-Conditions	The user is locked or unlocked or change the password of the user
Basic Flow or Main Scenario	1. Check the authority of the current user 2. Enter the user name you want to lock/unlock & check the username if valid 3. Successful lock/unlock the user
Extensions or Alternate Flows	
Exceptions	1. User not found 2. IT admin cannot lock/unlock another IT Admin
Related Use Cases	

Use Case ID	4.1
Use Case Name	User creates record
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	New record created by user
Basic Flow or Main Scenario	1. Enter new record

	2. System validates record entry 3. System saves record
Extensions or Alternate Flows	
Exceptions	● Record already exist ● Invalid record parameters
Related Use Cases	

Use Case ID	
Use Case Name	User reads record
Relevant Requirements	4.2
Primary Actor	System
Pre-Conditions	None
Post-Conditions	Record read by user
Basic Flow or Main Scenario	1. User request record 2. System searches for record 3. System returns record contents to user
Extensions or Alternate Flows	
Exceptions	● Record not found
Related Use Cases	

Use Case ID	4.3
Use Case Name	User deletes record
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	Record deleted by user
Basic Flow or Main Scenario	
Extensions or Alternate Flows	
Exceptions	● Record not found

Related Use Cases	

'

Use Case ID	4.4
Use Case Name	User updates record
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	Record is updated
Basic Flow or Main Scenario	1. User requests record to be updated 2. User passes updated record parameters 3. System searches for record 4. System validates new record parameters 5. System updates record old parameters with new parameters 6. System confirms success
Extensions or Alternate Flows	● Record not found ● Record parameters invalid
Exceptions	
Related Use Cases	

Use Case ID	5.1
Use Case Name	User creates group
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	Group is created
Basic Flow or Main Scenario	1. User request to create a new group with group name 2. System checks that group does not already exist 3. System creates new group 4. System confirms success

Extensions or Alternate Flows	
Exceptions	● Users not found ● Group already exist
Related Use Cases	

Use Case ID	5.2
Use Case Name	User adds client to group
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	Client added to group
Basic Flow or Main Scenario	1. User requests client be added to a group 2. System checks that client and group exists 3. System adds client to group 4. System confirms success
Extensions or Alternate Flows	1. System can ask User if System should create group should it not exist already
Exceptions	● Client not found ● Group does not exist
Related Use Cases	

Use Case ID	5.3
Use Case Name	User removes client from group
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	Client removed from group
Basic Flow or Main Scenario	1. User requests client be removed from a group 2. System checks that client and group exists

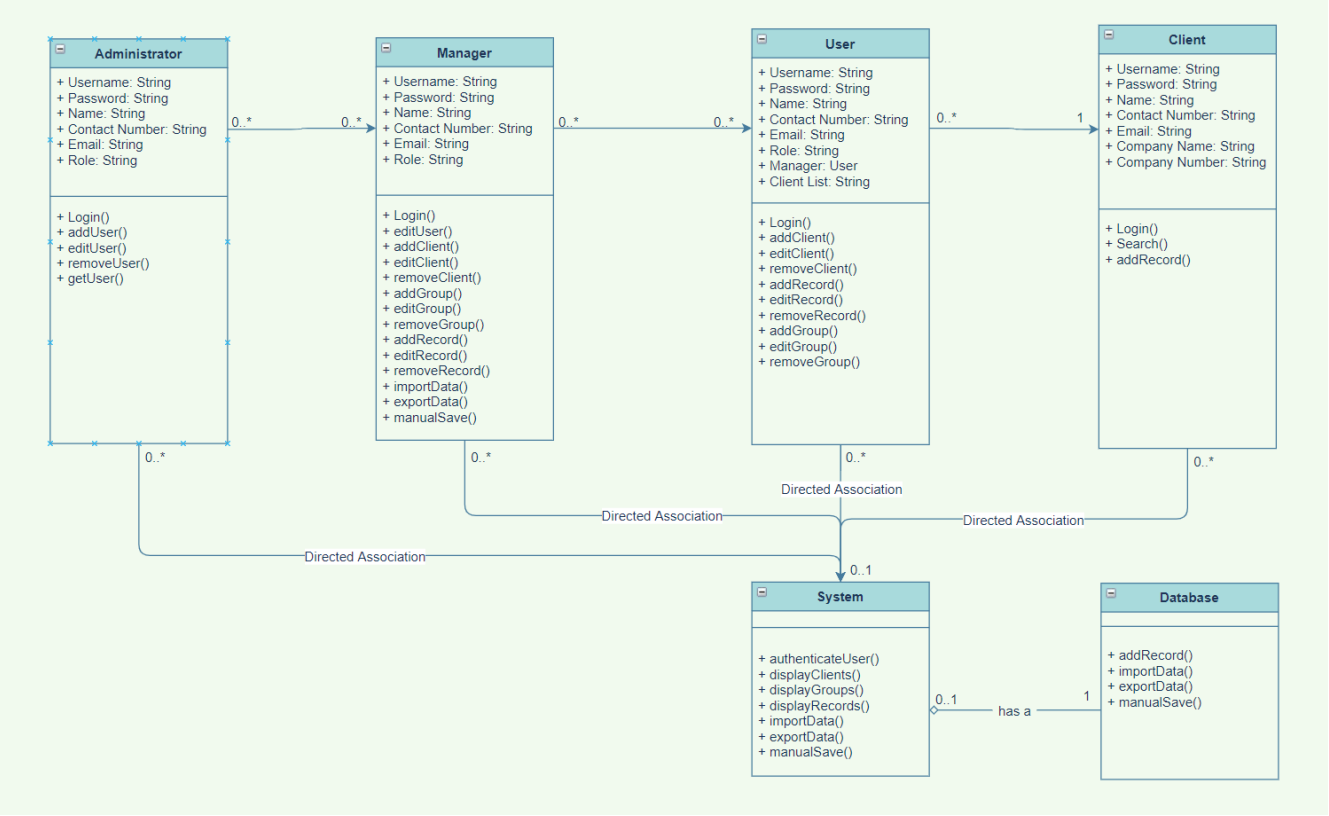
	3. System removes client from group 4. System confirms success
Extensions or Alternate Flows	
Exceptions	● Client not found ● Group not found
Related Use Cases	

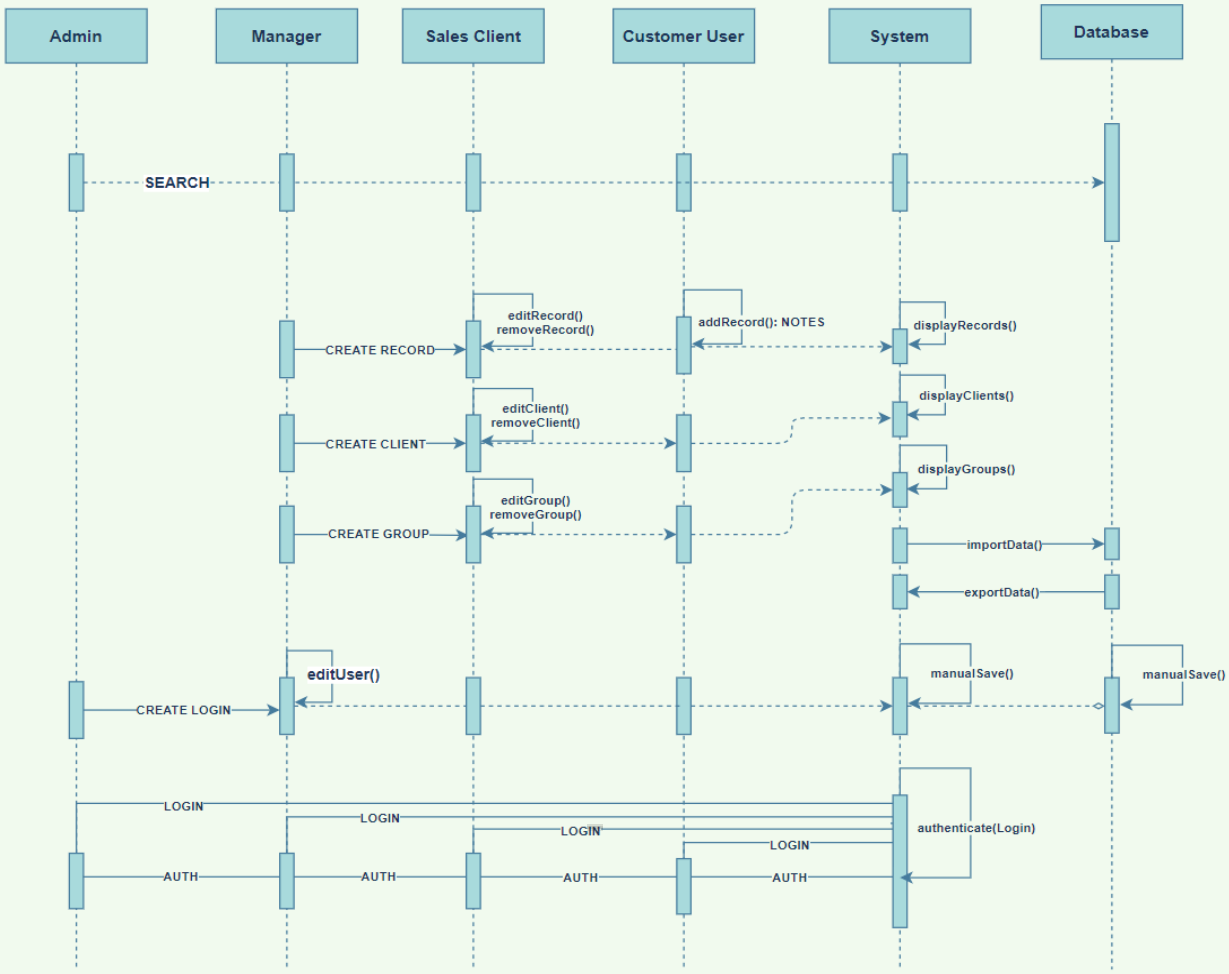
Use Case ID	6.0
Use Case Name	User prints report
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	User record data printed
Basic Flow or Main Scenario	1. User requests report based on parameters 2. System checks for records that match criteria 3. System collates all matching records 4. System returns results to User
Extensions or Alternate Flows	
Exceptions	● Search could yield no results
Related Use Cases	

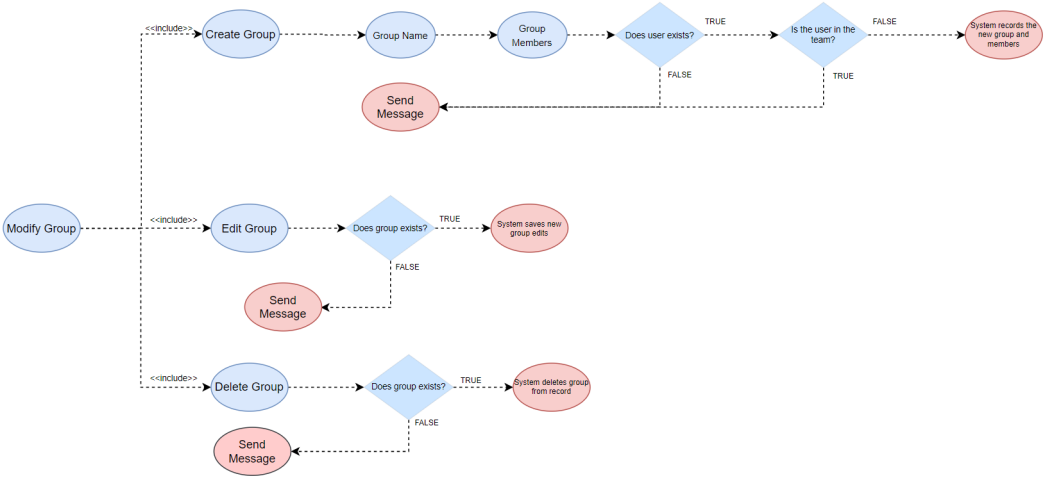
Use Case ID	7.1
Use Case Name	System saves data to database
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	Data saved to database
Basic Flow or Main Scenario	1. System collects all data of all

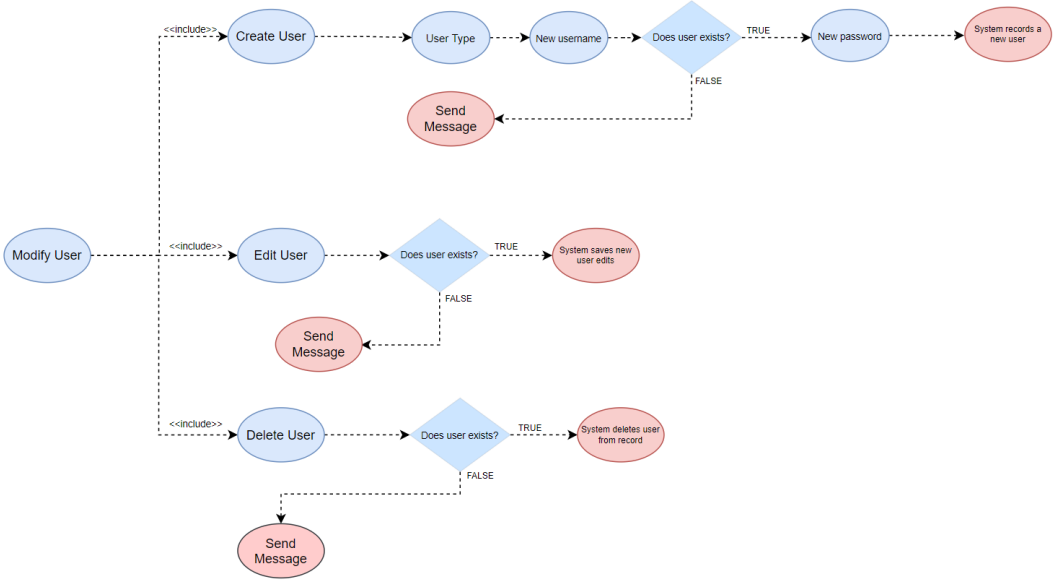
	clients 2. System sends data to database for storage
Extensions or Alternate Flows	
Exceptions	● Data corruption
Related Use Cases	

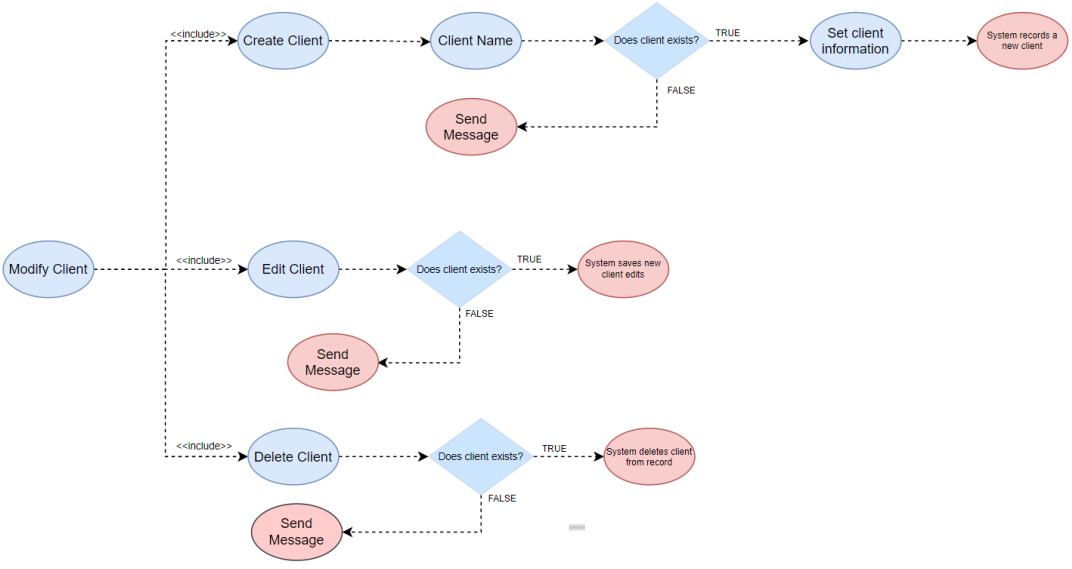
Use Case ID	7.2
Use Case Name	System loads data from database
Relevant Requirements	
Primary Actor	System
Pre-Conditions	None
Post-Conditions	Data saved to database
Basic Flow or Main Scenario	1. Upon system initialization, system will fetch data from storage 2. Data will be read from storage and loaded by system
Extensions or Alternate Flows	
Exceptions	● Data corruption ● Empty file
Related Use Cases	

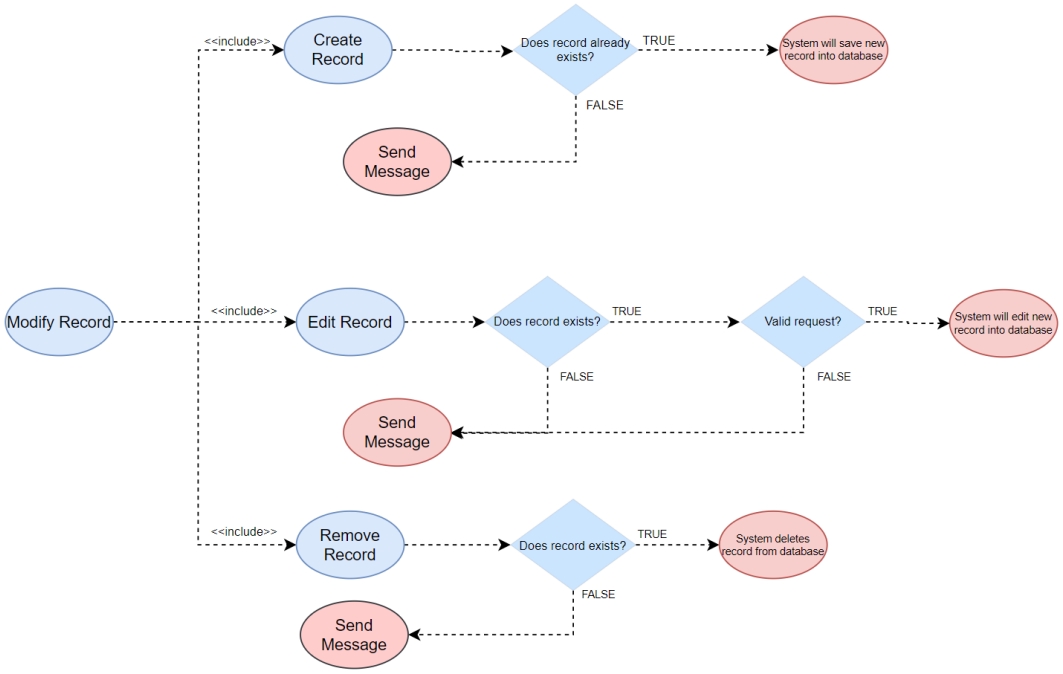

Administrator
+ Username: String
+ Password: String
+ Name: String
+ Contact Number: String
+ Email: String
+ Role: String
+ Login()
+ addUser()
+ editUser()
+ removeUser()
+ getUser()
Manager
+ Username: String
+ Password: String
+ Name: String
+ Contact Number: String
+ Email: String
+ Role: String
+ Login()
+ editUser()
+ addClient()
+ editClient()
+ removeClient()
+ addGroup()
+ editGroup()
+ removeGroup()
+ addRecord()
+ editRecord()
+ removeRecord()
+ importData()
+ exportData()
+ manualSave()
User
+ Username: String
+ Password: String
+ Name: String
+ Contact Number: String
+ Email: String
+ Role: String
+ Manager: User
+ Client List: String
+ Login()
+ addClient()
+ editClient()
+ removeClient()
+ addRecord()
+ editRecord()
+ removeRecord()
+ addGroup()
+ editGroup()
+ removeGroup()
Client
+ Username: String
+ Password: String
+ Name: String
+ Contact Number: String
+ Email: String
+ Company Name: String
+ Company Number: String
+ Login()
+ Search()
+ addRecord()
0..*
1
Directed Association
System
+ authenticateUser()
+ displayClients()
+ displayGroups()
+ displayRecords()
+ importData()
+ exportData()
+ manualSave()
Database
+ addRecord()
+ importData()
+ exportData()
+ manualSave()
0..1
has a

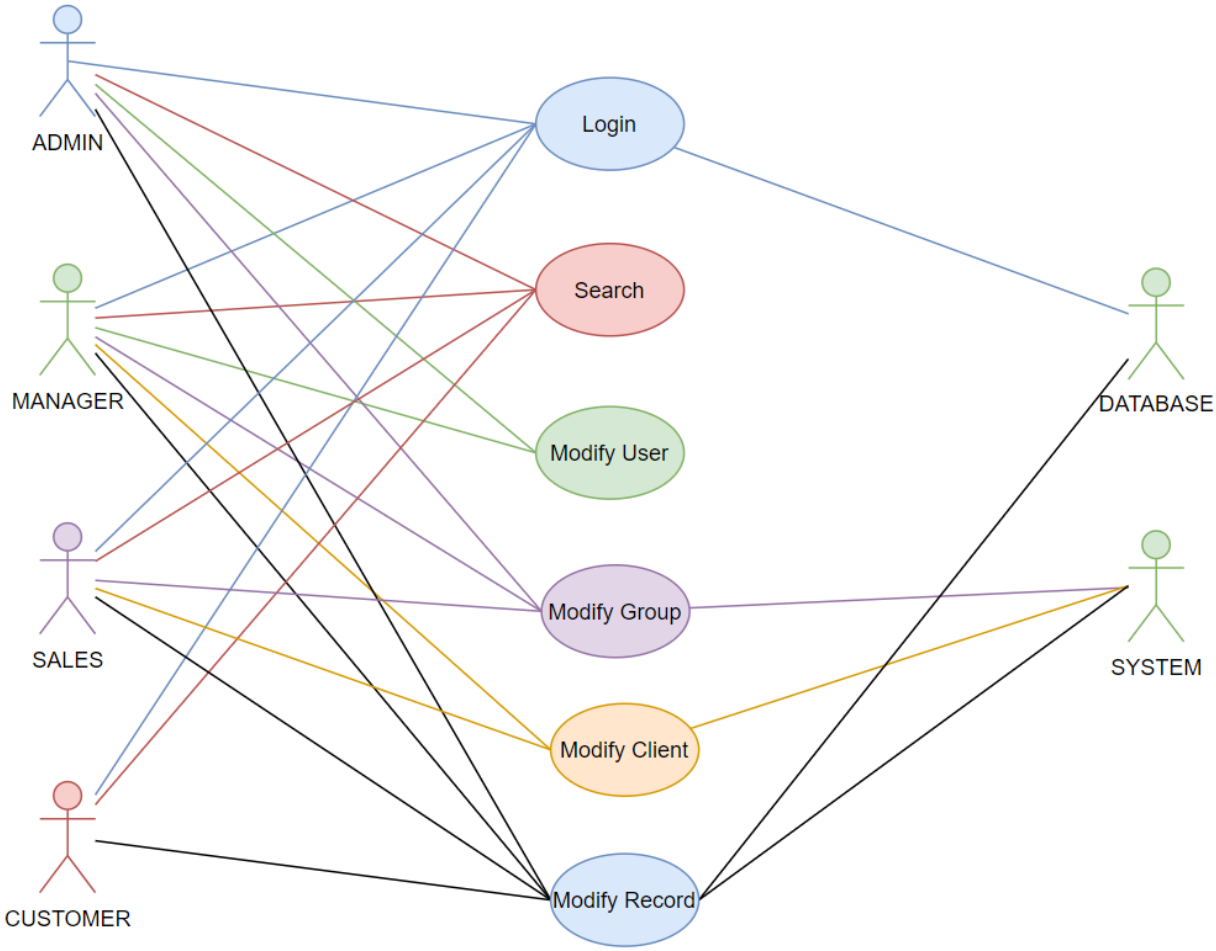

Admin
Manager
Sales Client
Customer User
System
Database
SEARCH
CREATE RECORD
editRecord()
removeRecord()
addRecord(): NOTES
displayRecords()
CREATE CLIENT
editClient()
removeClient()
displayClients()
CREATE GROUP
editGroup()
removeGroup()
displayGroups()
importData()
exportData()
CREATE LOGIN
editUser()
manualSave()
manualSave()
LOGIN
LOGIN
LOGIN
LOGIN
authenticate(Login)
AUTH
AUTH
AUTH
AUTH

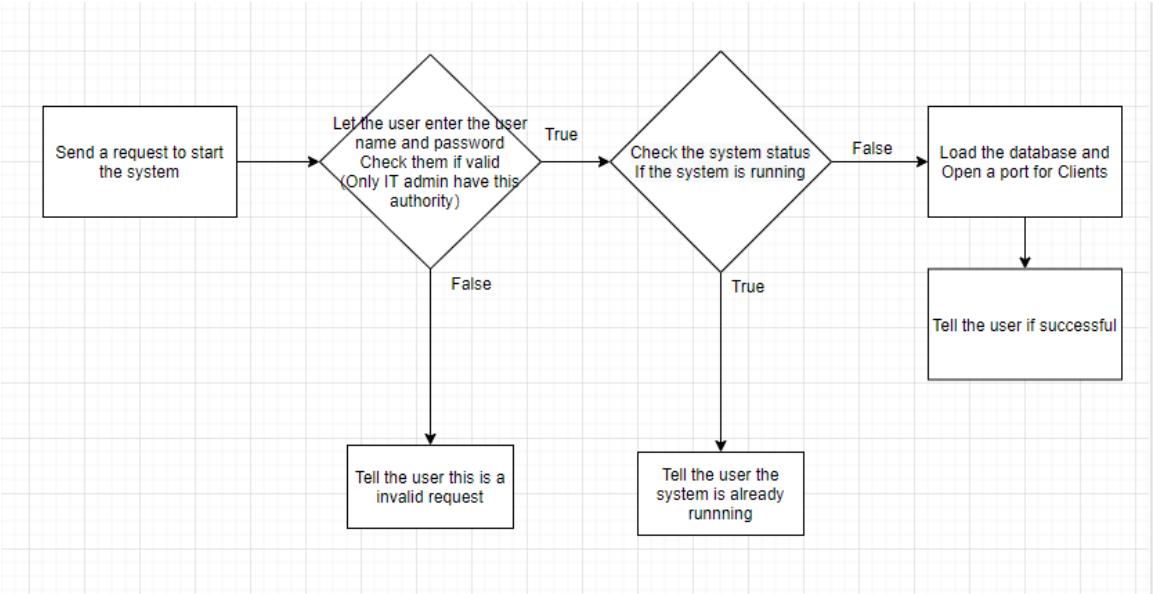

Modify Group
<<include>>
Create Group
Group Name
Group Members
Does user exists?
TRUE
Is the user in the team?
FALSE
System records the new group and members
TRUE
FALSE
Send Message
<<include>>
Edit Group
Does group exists?
TRUE
System saves new group edits
FALSE
Send Message
<<include>>
Delete Group
Does group exists?
TRUE
System deletes group from record
FALSE
Send Message


Modify User
<<include>>
Create User
User Type
New username
Does user exists?
TRUE
New password
System records a new user
FALSE
Send Message
<<include>>
Edit User
Does user exists?
TRUE
System saves new user edits
FALSE
Send Message
<<include>>
Delete User
Does user exists?
TRUE
System deletes user from record
FALSE
Send Message

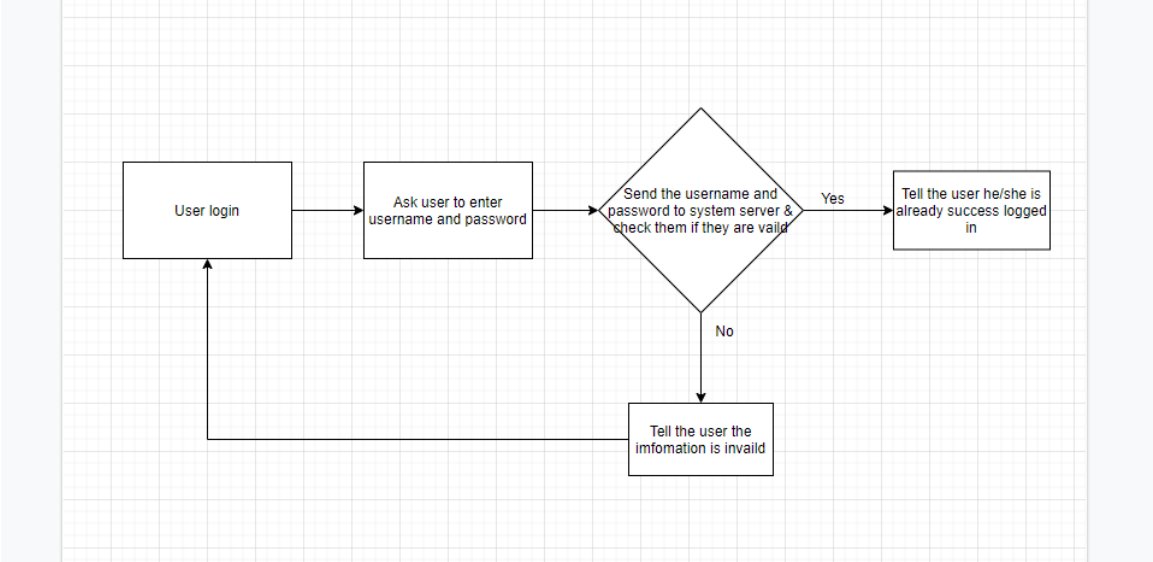

Modify Client
<<include>>
Create Client
Client Name
Does client exists?
TRUE
Set client information
System records a new client
FALSE
Send Message
<<include>>
Edit Client
Does client exists?
TRUE
System saves new client edits
FALSE
Send Message
<<include>>
Delete Client
Does client exists?
TRUE
System deletes client from record
FALSE
Send Message


Modify Record
<<include>>
Create Record
Does record already exists?
TRUE
System will save new record into database
FALSE
Send Message
<<include>>
Edit Record
Does record exists?
TRUE
Valid request?
TRUE
System will edit new record into database
FALSE
FALSE
Send Message
<<include>>
Remove Record
Does record exists?
TRUE
System deletes record from database
FALSE
Send Message

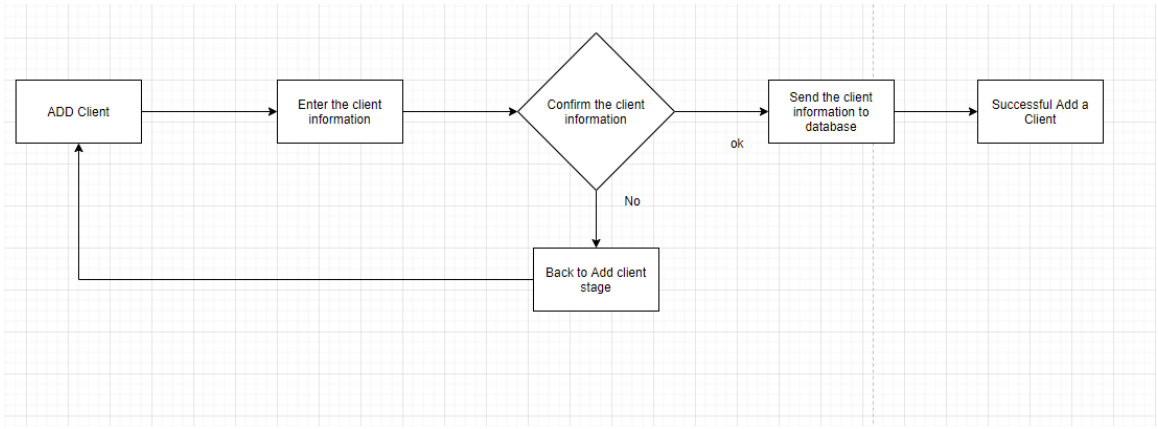

ADMIN
MANAGER
SALES
CUSTOMER
Login
Search
Modify User
Modify Group
Modify Client
Modify Record
DATABASE
SYSTEM

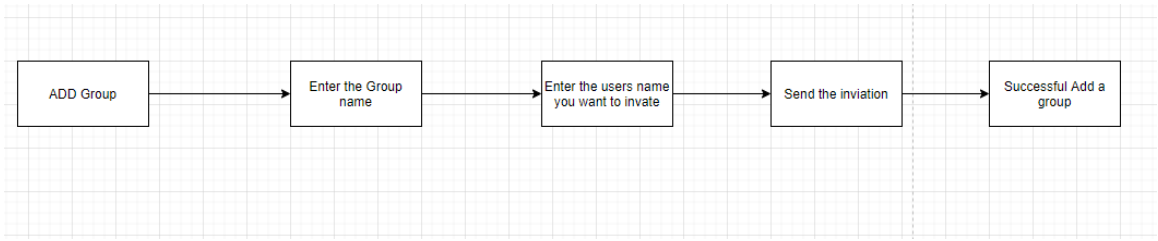


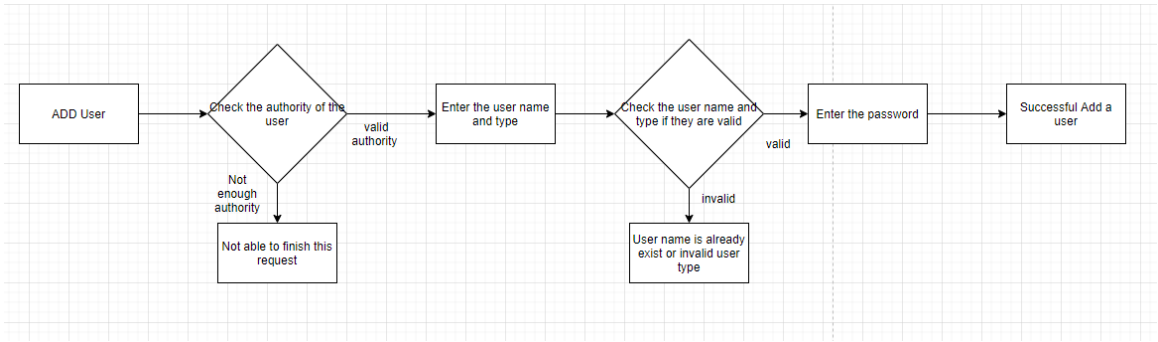
User login:

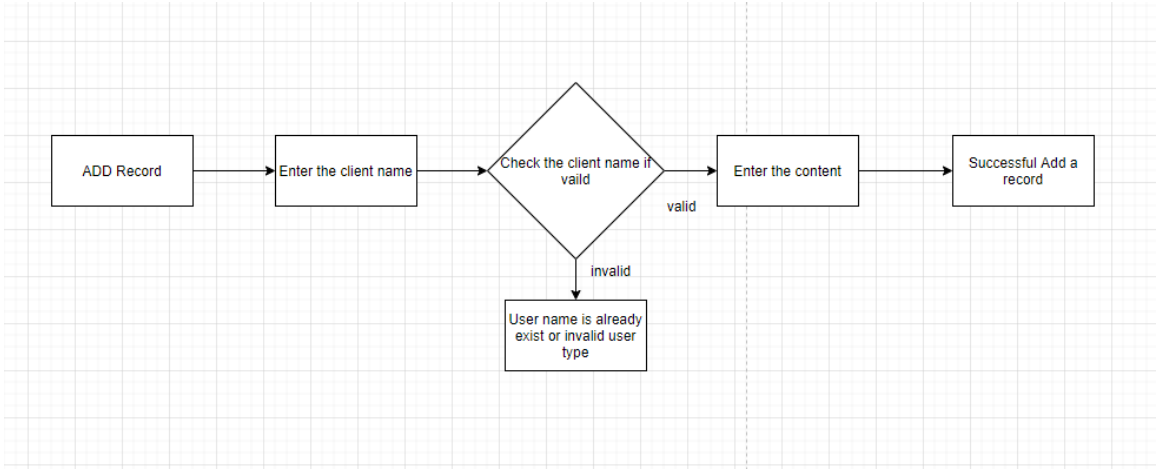


ADD:

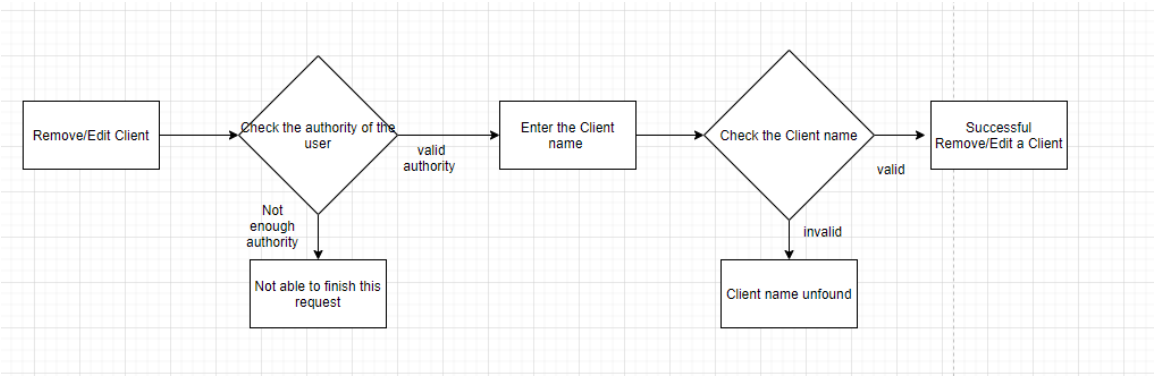


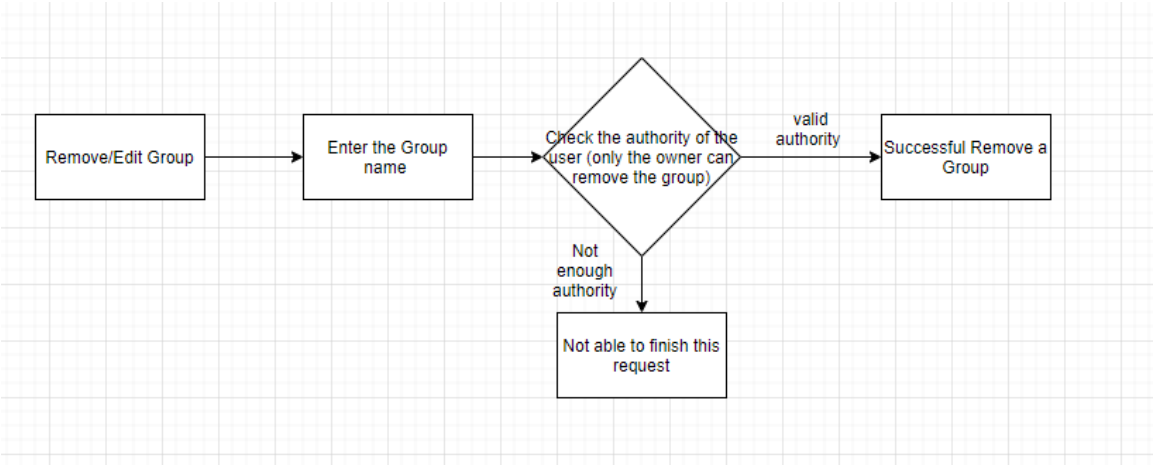


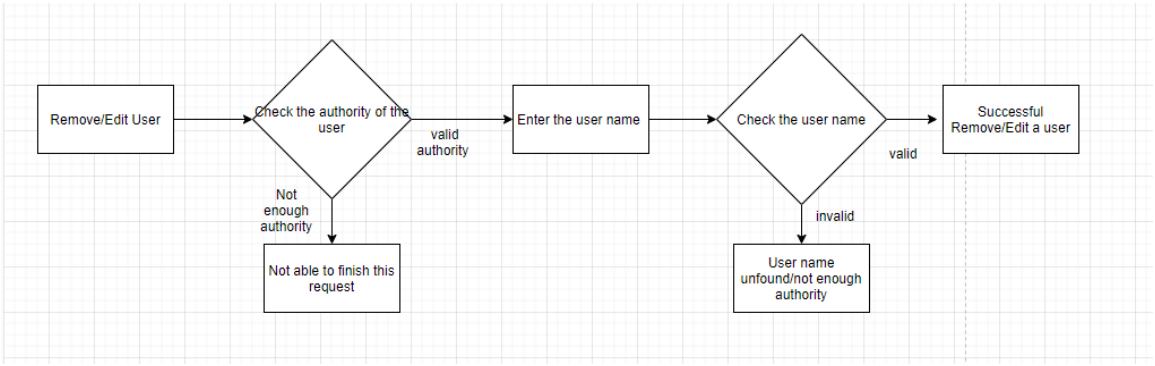


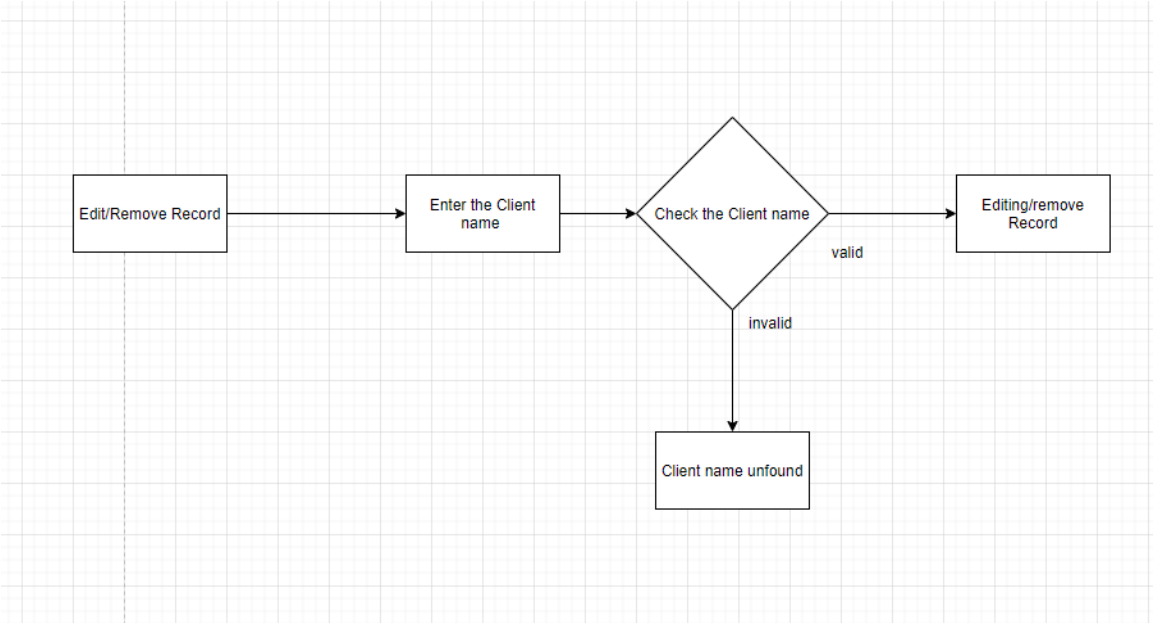


Edit and Remove:

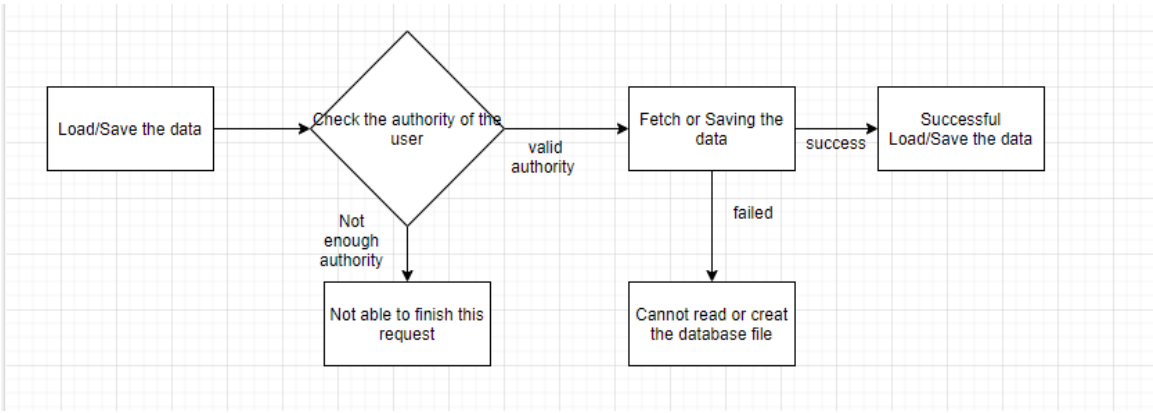

Remove/Edit Client
Check the authority of the user
valid authority
Enter the Client name
Check the Client name
valid
Successful Remove/Edit a Client
Not enough authority
Not able to finish this request
invalid
Client name unfound


Remove/Edit Group
Enter the Group name
Check the authority of the user (only the owner can remove the group)
valid authority
Successful Remove a Group
Not enough authority
Not able to finish this request


Remove/Edit User
Check the authority of the user
valid authority
Enter the user name
Check the user name
valid
Successful Remove/Edit a user
Not enough authority
Not able to finish this request
invalid
User name unfound/not enough authority



Load/Save the data:



1.4. Overview

The CRM is a Customer Relationship Management application that is used to maintain records of client relationships.

2. Overall Description

2.1. Product Perspective

2.2. Product Architecture

The system will be organized into 3 major modules: the controller module, storage module and the client module.

2.3. Product Functionality/Features

The high-level features of the system are as follows (see section 3 of this document for more detailed requirements that address these features):

2.4. Constraints

2.4.1. The system does not use any off-the-shelf software for data storage

2.4.2. Administrators are responsible for creating subsequent users of the system

2.4.3. Users cannot manipulate or edit the data of other users

2.5. Assumptions and Dependencies

2.5.1. It is assumed that the system will have 3 levels of users: Administrator, Manager, User.

2.5.2. It is assumed that Administrator-level users will create all users.

2.5.3. It is assumed that data will persist for all connected users such that upon restarting or reaccessing the application, all their data will be present from the last session.

2.5.4. It is assumed that User-level users can create/read/update/delete data pertaining to their customers.

2.5.5. It is assumed that User-level users cannot interact with or manipulate data of other users

2.5.6. of the system.

2.5.7. It is assumed that each user has their own account and password.

2.5.8. Groups depend on the the user that created the group

3. Specific Requirements

3.1. Functional Requirements

3.1.1. Common Requirements:

3.1.1.1. System requires the user to enter a username and a password

3.1.1.2. A GUI interface which allows to execute different types of commands graphically.

3.1.1.3. Administrator-level users to maintain the server and also add, create and remove users.

3.1.1.4. User-level users can create/read/update/delete records

3.1.1.5. User-level users can create groups and add clients to these groups

3.1.1.6. All user data persists in external data storage

3.1.2. Storage Module Requirements:

3.1.2.1. Storage will be custom implementation and does not use any sort of off-the-shelf software (Cannot be a MYSQL db for instance)

3.1.2.2. Data for all users should be stored and loaded when the application is started

3.1.2.3. Runs as a separate process to manage queue of data to be saved.

3.1.3. Server Module Requirements:

3.1.3.1. Controls the initial startup of the application.

3.1.3.2. Manages all connections.

3.1.3.3. Binds port for clients to connect to.

3.1.3.4. Saves and loads all user data.

3.1.3.5. Processes user requests.

3.1.3.6. Processes request to save data.

3.1.4. Client Module Requirements:

3.1.4.1. Client connects to the server via a network address and a port.

3.1.4.2. Users of the CRM server interact with the Server from the Client side and through the GUI.

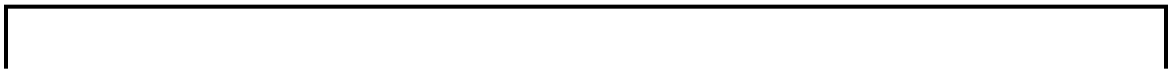
3.1.4.3. The client will pass user requests to the server.

3.2. External Interface Requirements:

3.2.1. None

3.3. Internal Interface Requirements:

3.3.1. None



4. Non-Functional Requirements

4.1. Security and Privacy Requirements

4.1.1. The system requires a username and password for each user of the system.
4.1.2. Authentication is done by the server, clients are not allowed to self-authenticate.

4.2. Environmental Requirements

4.2.1. System is required to run as a Java based application

4.3. Performance Requirements

4.3.1. None