

IBM – NALAIYA THIRAN PROJECT

CUSTOMER CARE REGISTRY

INDUSTRY MENTOR : VASUDEVAHANUSH

FACULTY MENTOR : KOUSIKA N

TEAM ID : PNT2022TMTD50240

TEAM LEAD : MANIKANDAN R

TEAM MEMBER : JAMES JACOBRAJ A

TEAM MEMBER : KUMARAN ANANDHAN K

TEAM MEMBER : ELANGAMANI T

**TABLE OF
CONTENT**

CHAPT ER	CONTEN TS	PA GE NO
1	INTRODUCTION a. PROJECT OVERVIEW b. PURPOSE	04
2	LITERATURE SURVEY a. EXISTING PROBLEM b. REFERENCES c. PROBLEM STATEMENT DEFINITION	05
3	IDEATION &PROPOSED SOLUTION a. EMPATHY MAPCANVAS b. IDEATION &BRAINSTROMING c. PROPOSED SOLUTION d. PROBLEM SOLUTION FIT	08
4	REQUIREMENT ANALYSIS a. FUNCTIONAL REQUIREMENT b. NON-FUNCTIONAL REQUIREMENTS	11
5	PROJECT DESIGN a. DATA FLOW DIAGRAMS b. SOLUTION &TECHNICAL ARCHITECTURE c. USER STORIES	14

6	PROJECT PLANNING & SCHEDULING <ul style="list-style-type: none"> a. SPRINT PLANNING & ESTIMATION b. SPRINT DELIVERY SCHEDULE c. REPORTS FROM JIRA 	18
----------	---	-----------

7	CODING & SOLUTIONING <ul style="list-style-type: none"> a. FEATURE 1 b. FEATURE 2 c. DATABASE SCHEMA 	23
8	TESTING <ul style="list-style-type: none"> a. TEST CASES b. USER ACCEPTANCE TESTING 	26
9	RESULTS <ul style="list-style-type: none"> 9.1 PERFORMANCE METRICS 	28
10	ADVANTAGES & DISADVANTAGES	31
11	CONCLUSION	33
12	FUTURE SCOPE	34

13	APPENDIX SOURCE CODE GITHUB & PROJECTDEMO LINK	34
-----------	---	-----------

1. INTRODUCTION

a. Project Overview

Customer care describes how people are treated when they interact with a brand. This includes all experiences with the company and its employees before, during, and after a purchase. Customer care is an important aspect of customer service because it fosters an emotional connection with the brand's community.

Customer care isn't measured in the same way as customer loyalty or success. That's because things like loyalty and success are a by-product of caring for your customers. It's impossible to build a trustworthy, emotional connection with your customer base if you're too focused on measuring it. Customer care goes a step further by ignoring the metrics and instead fully investing in your customers' goals and needs.

Customer Care vs Customer Services:

Customer care is the process of building an emotional connection with your customers, whereas customer service is simply the advice or assistance your business provides them. Customer care is less quantifiable than customer service and is more concerned with one-to-one customer interactions.

While both functions increase customer satisfaction, customer service does this by

answering questions and providing support. Customer care, on the other hand, focuses on active listening and understanding the customer's emotional needs as much as the physical or business ones.

2. Purpose

An online comprehensive Customer Care Solution is to manage customer interaction and complaints with the Service Providers over phone or through and e-mail. The system should have capability to integrate with any Service Provider from any domain or industry like Banking, Telecom Insurance, etc. Customer Service also known as Client Service is the provision of service to customers its significance varies by product, industry and domain In many cases customer services is more important of the purchase relates to a service as opposed to a product Customer Service may be provided by a Person or Sales & Service Representatives Customer Service is normally an integral part of a company's customer value propositions.

2. LITERATURE SURVEY

In a literature survey, students analyze critically, and concisely earlier research and literature related to a particular research problem and utilize them for their own research purposes. It helps students in concluding the significances of new research and its connections to earlier work.

Purpose of a Literature Survey:

Conducting a literature review establishes your familiarity with and understanding of current research in a particular field before carrying out a new investigation. After doing a

literature review, you should know what research has already been done and be able to identify what is unknown within your topic

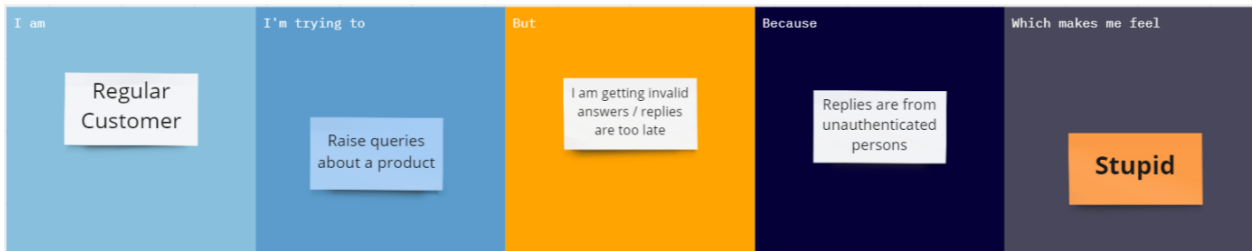
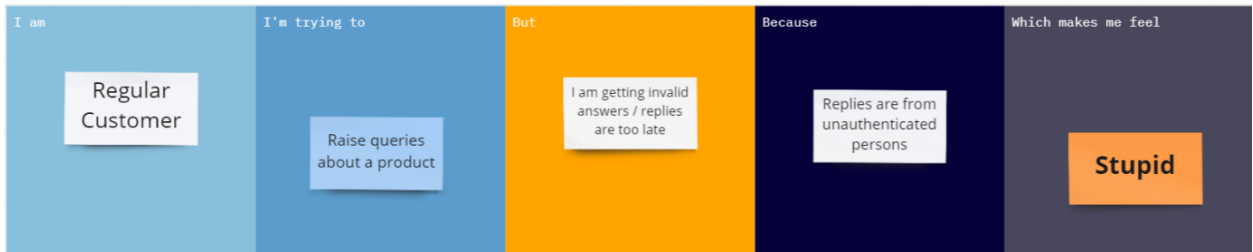
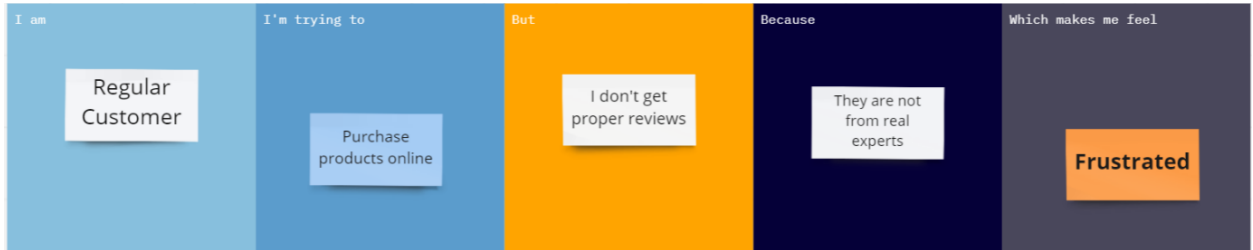
a. Existing problem

I am Surya and I am a regular customer in famous e-commerce websites like Amazon, Flipkart. I order regularly. The problem I have is that in most times, I don't have any reliable sources to clear my doubts in some of the products I buy.

There are reviews and customer ratings in those websites, but somehow, I don't feel they are authentic and real. It would make my world if those replies were from a real expert, and I could clarify all my doubts in a single platform. Of course, I would need instant replies from a real expert who knows about the products I am asking for.

I am	Describe customer with 3-4 key characteristics - <i>who are they?</i>	Describe the customer and their attributes here
I'm trying to	List their outcome or "job" the care about - <i>what are they trying to achieve?</i>	List the thing they are trying to achieve here
but	Describe what problems or barriers stand in the way - <i>what bothers them most?</i>	Describe the problems or barriers that get in the way here
because	Enter the "root cause" of why the problem or barrier exists - <i>what needs to be solved?</i>	Describe the reason the problems or barriers exist
which makes me feel	Describe the emotions from the customer's point of view - <i>how does it impact them emotionally?</i>	Describe the emotions the result from experiencing the problems or barriers

Example:



Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Regular Customer	Purchase products online	I don't get proper reviews	They are not from real experts.	Frustrated
PS-2	Regular Customer	Bought a product	I cannot get my doubts clarified	There is no proper system	Disappointed

PS-3	Regular customer	Raise queries about a product	I am getting invalid answers / replies are too late	Replies are from unauthenticated persons	Stupid
------	------------------	-------------------------------	---	--	--------

b. References

This customer care registry helps to solve the issues and its find customer satisfaction. A Customer had occur a problem when they apply a ticket they need to recovery a solution or result .So the customer will contact a customer care for arise ths issue. After the customer complaint, the company could identify that problem and solved this issue. Now the company wants to avoid these kinds of problems and technical issues So the company needs the customer satisfaction.

c. Problem Statement Definition

A Customer had occur a problem when they apply a ticket they need to recovery a solution or result .So the customer will contact a customer care for arise ths issue. After the customer complaint, the company could identify that problem and solved this issue. Now the company wants to avoid these kinds of problems and technical issues So the company needs the customer satisfaction. This customer care registry helps to solve the issues and its find customer satisfaction

OverData Utilization on connecting to Desktop	
Why this happens?	This happens because the background windows update process is on.
Who does the problem affect?	The user and the users who are connectedto the Desktop.
What is the issue?	This issue is over utilization of mobile dataover connecting to the Desktop
What is the solution?	This issue can be solved by disabling the windows update option in settings

Customer wants to fix a blue screen of death?	
Who does the Problem Affect?	Customer who use the particular thing
What are the boundaries of the problem?	Customer who use the thing for their personal work, office work etc
What is the issue?	Failure of Hardware or driver sometimes it maybe in software too
When does the issue occur?	It frequently occurs after the customer installed new drivers or new piece of software
Where does the issue occur?	It often lies in the Hardware or one of the drivers
Why is it important that we fix the problem?	It is necessary to run the computer or Laptop to do their task or work in order to complete it.
What solution to solve this issue?	A quick reboot is sometimes enough to solve the problem

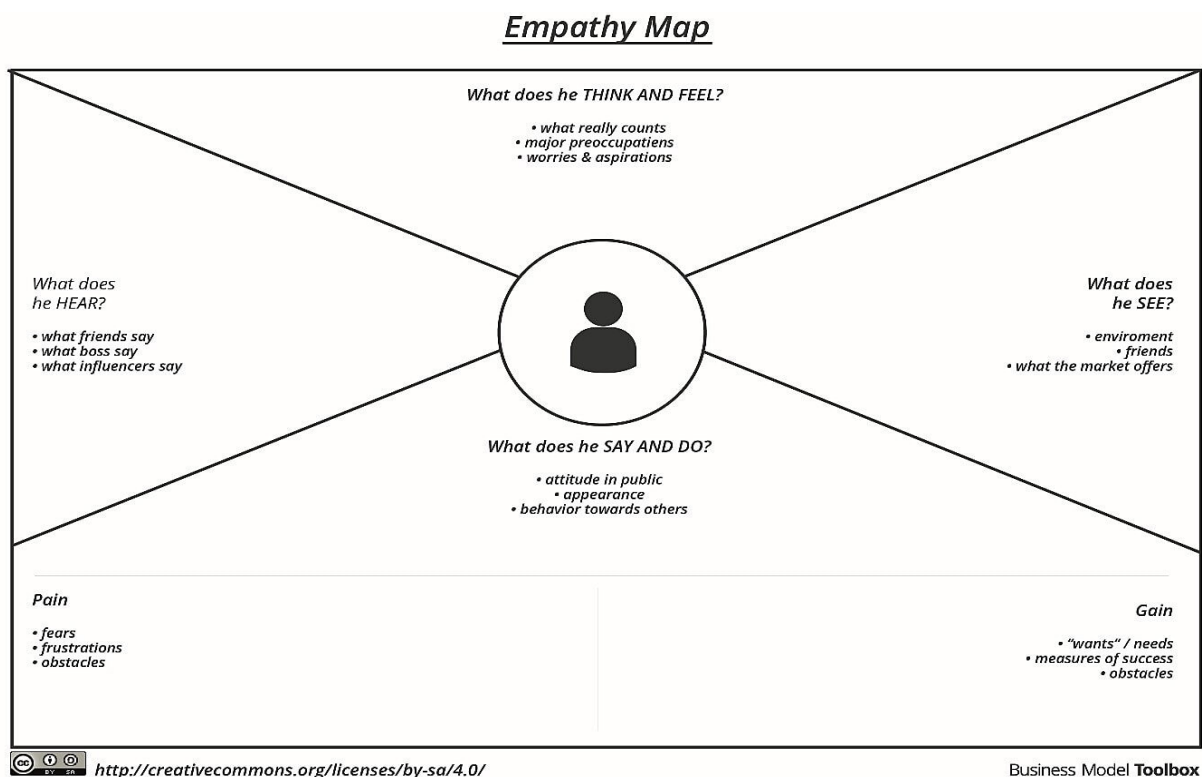
Customer wants to fix the Payment issue?	
Who does the Problem Affect?	Customer who use the particular thing
What is the solution to solve this issue temporarily?	Check payment method is up to date or Try another payment method
How the issue occurs?	Customer who has entered incorrect card information, payment gateway, or the bank institution issue
When does the issue occur?	It occurs when there is insufficient balance in bank account
Why is it important that we fix the problem?	For the welfare of the customer needs

3. IDEATION & PROPOSED SOLUTION

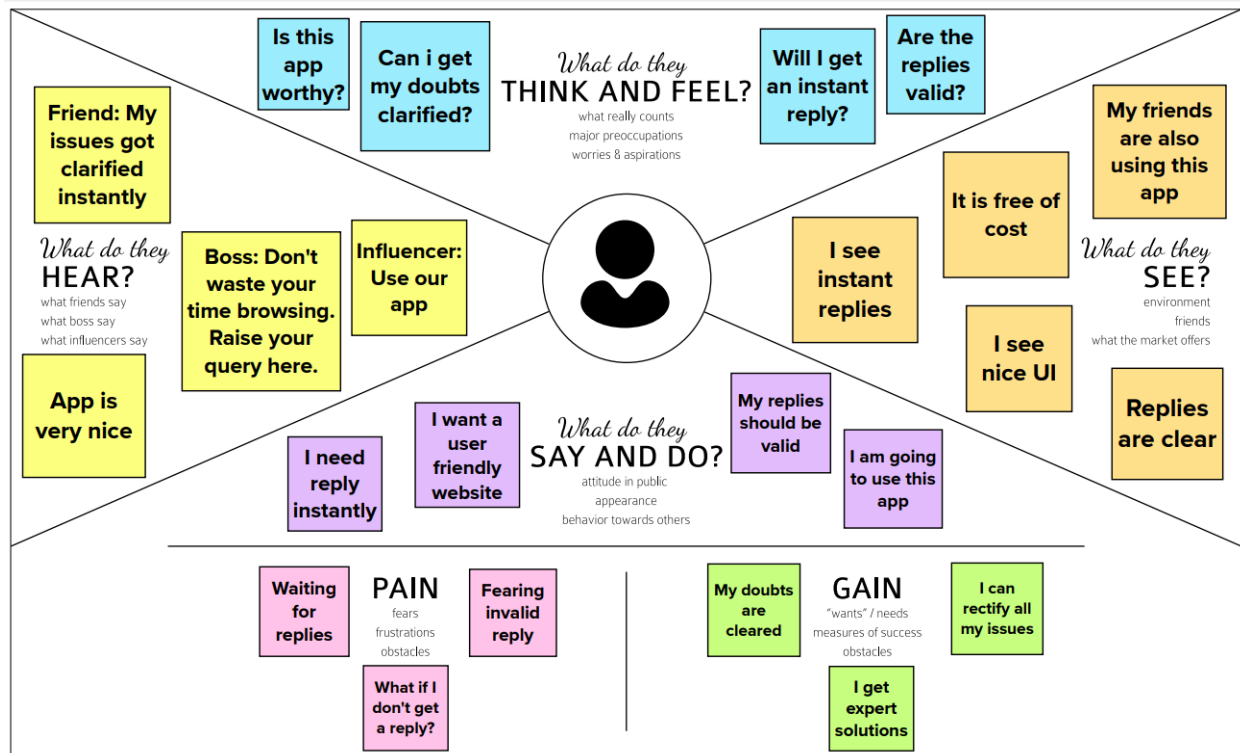
a. Empathy Map Canvas

- An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.
- It is a useful tool to help teams better understand their users.
- Creating an effective solution requires understanding the true problem and the person who is experiencing it.
- The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Example:



Empathy Map for Customer Care Registry:



d. Ideation & Brainstorming

Brainstorm & Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement

Team Gathering:

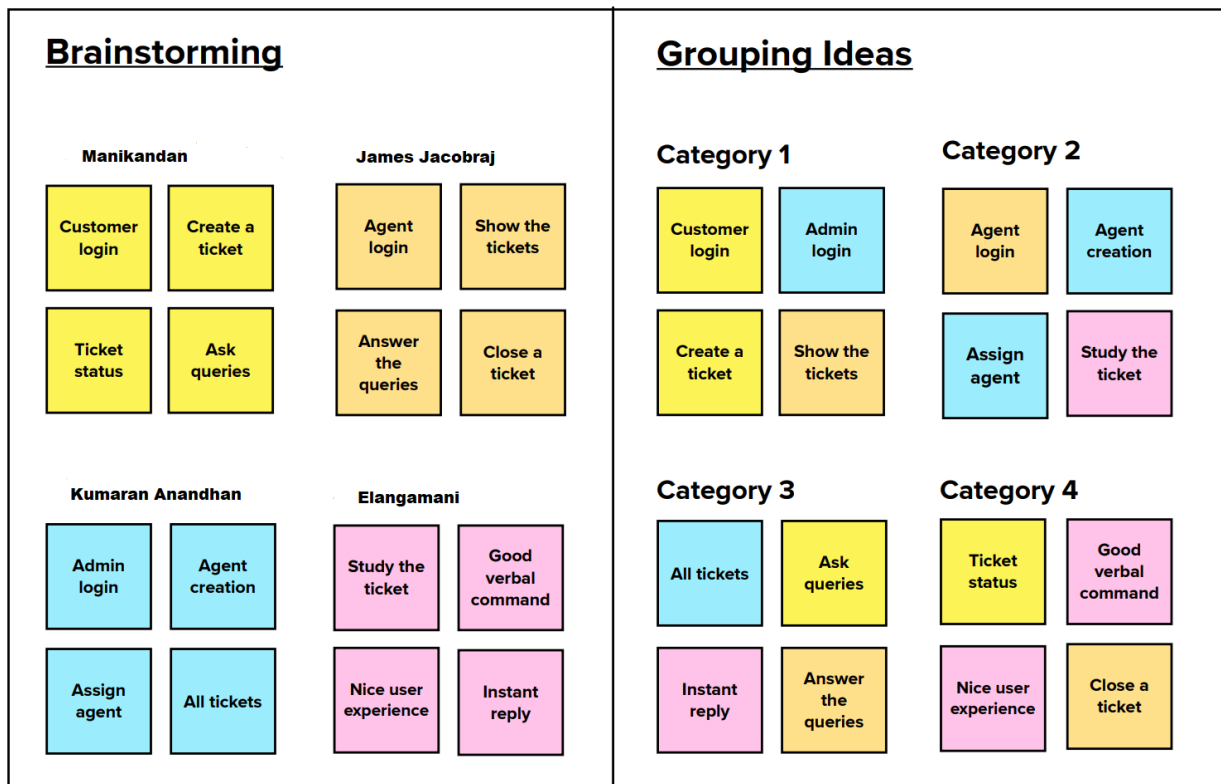
Team Members	
Team Leader	Manikandan
Team Members	James Jacobraj
	Kumaran Anandhan
	Elangamani

Problem Statement:

I am Ajay and I am a regular customer in famous e-commerce websites like Amazon, Flipkart. I order regularly. The problem I have is that in most times, I don't have any reliable sources to clear my doubts in some of the products I buy.

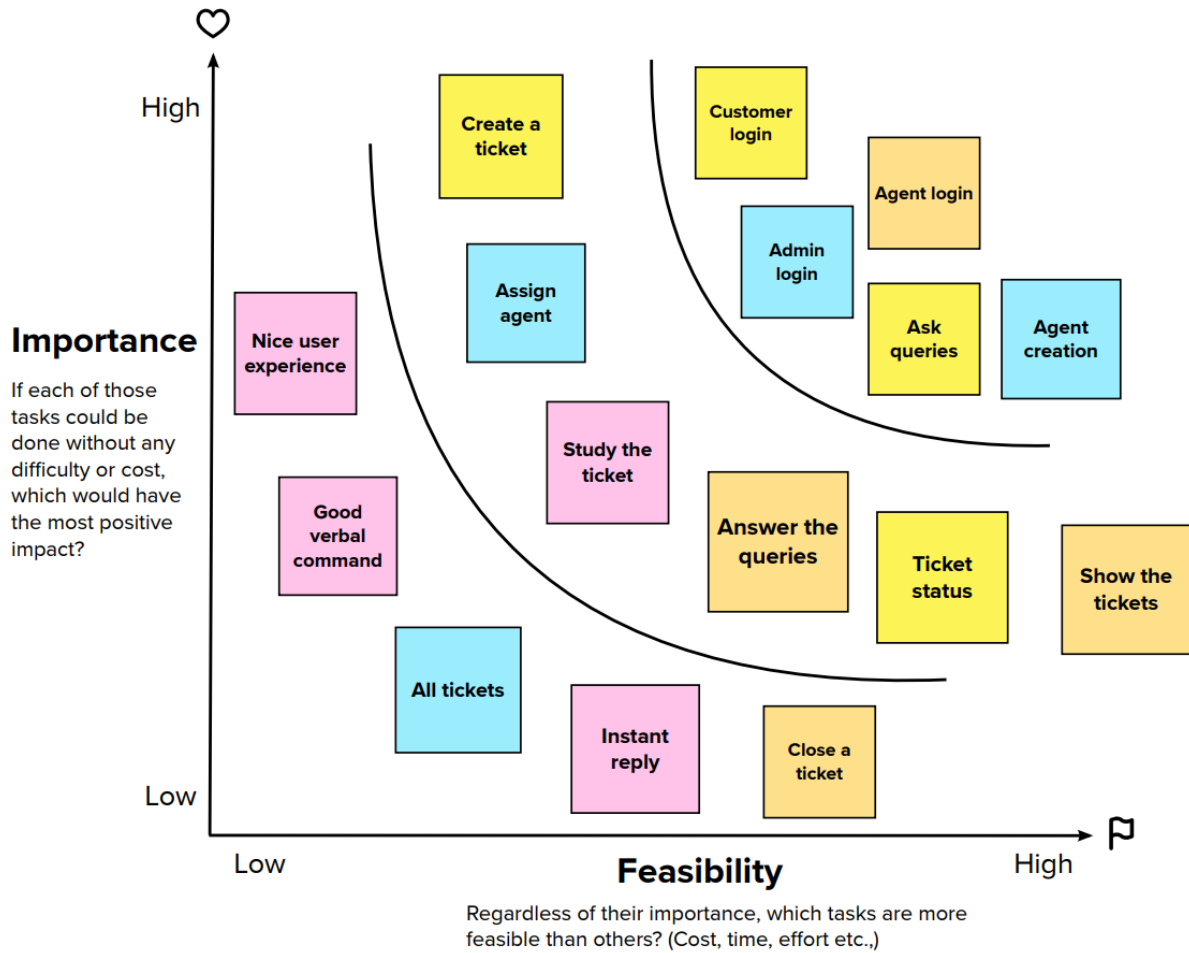
There are reviews and customer ratings in those websites, but somehow, I don't feel they are authentic and real. It would make my world if those replies were from a real expert, and I could clarify all my doubts in a single platform. Of course, I would need instant replies from a real expert who knows about the products I am asking for.

Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization

Prioritization



e. Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To solve the customer issues using web based cloud application.
2.	Idea / Solution description	Creating a Customer Care Registry, where the customers can raise their queries in form of tickets. An agent will be assigned to them for replying/clarifying their issues.
3.	Novelty / Uniqueness	The agents are experts in the product domain and they will communicate well with the customers
4.	Social Impact / Customer Satisfaction	Customers will be satisfied with the instant and valid replies. Also, it creates a doubtless society, that boosts sales.
5.	Business Model (Revenue Model)	Customers can be charged a minimal amount based on the number of queries (tickets) they can rise in a said period of time.
6.	Scalability of the Solution	This idea is so much use to the customers that the latter may refer this registry to their friends and colleagues at work. Naturally, the user base grows so does the number of queries answered. May be in the future, may be a cross-platform mobile application may be developed, making this customer care registry much more accessible to the users.

e. Problem Solution fit

Define CS, fit into CC	<div>1. CUSTOMER SEGMENT(S)<div>CS</div></div> <div>Who is your customer? i.e. working parents of 0-5 y.o. kids</div> <div>Our customers are usually above 16 years old. Ranging from college students to working adults to retired professionals. Also, reputed organizations too.</div>	<div>6. CUSTOMER CONSTRAINTS<div>CC</div></div> <div>What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.</div> <div><div>1. Late replies for their queries</div><div>2. Complicated process to take over</div><div>3. High chance their queries may not be considered at all</div><div>4. Replies irrelevant to their queries</div><div>5. Advertisements shown</div></div>	<div>5. AVAILABLE SOLUTIONS<div>AS</div></div> <div>Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking</div> <div>Customers most probably use helpdesk.</div> <div><div>Pros:</div><div><div>1. Reasonably priced</div><div>2. Highly scalable for team of any size</div></div></div> <div><div>Cong:</div><div>They do not understand the severity of all complaints and end up treating them all in the same way</div></div>	Explore AS, differentiate
	<div>2. JOBS-TO-BE-DONE / PROBLEMS<div>J&P</div></div> <div>Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.</div> <div><div>✓ Simplifying the user account creation process</div><div>✓ Giving instant replies to the customers to their queries</div><div>✓ Providing expert solutions to the queries</div><div>✓ Assigning individual agents/experts to the customers queries</div><div>✓ Sending the status of the queries to the customer's mail</div></div>	<div>9. PROBLEM ROOT CAUSE<div>RC</div></div> <div>What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations.</div> <div><div>1. No proper registry</div><div>2. Lack of experts in a common place</div><div>3. Replies for queries from random persons</div><div>4. Communication lag</div><div>5. High-cost</div></div>	<div>7. BEHAVIOUR<div>BE</div></div> <div>What does your customer do to address the problem and get the job done? (i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace))</div> <div><div>1. Asking their friend's opinions</div><div>2. Checking solutions in the online forums</div><div>3. Using helpdesk</div><div>4. Solve the issues themselves based on their own knowledge</div><div>5. Seeing reviews posted by the users in the website forums</div></div>	Focus on J&P, tap into BE, understand RC
Identify strong TR & EM	<div>3. TRIGGERS<div>TR</div></div> <div>What triggers customers to act? i.e. seeing their neighbor installing solar panels, reading about a more efficient solution in the news.</div> <div>Overtime, they get disappointed with late and irrelevant replies and triggered to act</div>	<div>10. YOUR SOLUTION<div>SL</div></div> <div>If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behavior.</div> <div><div>• Creating a Customer Care Registry</div><div>• Simple User creation process</div><div>• Customers can raise their queries to the experts</div><div>• Individual agents will be assigned to each customer</div><div>• Their queries will be answered earnestly</div><div>• Customers can also check the status of their queries</div></div>	<div>8. CHANNELS of BEHAVIOUR<div>CH</div></div> <div><div>8.1 ONLINE</div><div>What kind of actions do customers take online? Extract online channels from #7</div><div><div>8.2 OFFLINE</div><div>What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.</div></div></div> <div><div>ONLINE:</div><div><div>1. https://www.helpdesk.com/</div><div>2. https://www.google.com/</div><div>3. https://www.quora.com/</div></div></div> <div><div>OFFLINE:</div><div><div>1. Asking friends and colleagues</div><div>2. Take actions themselves</div></div></div>	Identify strong TR & EM
	<div>4. EMOTIONS: BEFORE / AFTER<div>EM</div></div> <div>How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure > confident, in control - use it in your communication strategy & design.</div> <div><div>✗ Disappointed - after they do not get instant replies for their queries</div><div>✗ Dejected - when they get irrelevant replies even after waiting for a long time</div></div>			

3. REQUIREMENT ANALYSIS

a. Functional requirement

1. A functional requirement defines a function of a system or its component, where a function is described as a specification of behaviour between inputs and outputs.
2. It specifies “what should the software system do?”
3. It is mandatory
4. Defined at a component level
5. Usually easy to define

6. Helps you verify the functionality of the software

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Signup form (customer)
FR-2	User Login	Login through Login form (customer, agent, user)
FR-3	Agent creation (admin)	Create an agent profile with username, email and password
FR-4	Dashboard (customer)	Show all the tickets raised by the customer
FR-5	Dashboard (agent)	Show all the tickets assigned to the agent by admin
FR-6	Dashboard (Admin)	Show all the tickets raised in the entire system
FR-7	Ticket creation (customer)	Customer can raise a new ticket with the detailed description of his/her query
FR-8	Assign agent (admin)	Assigning an agent for the created ticket
FR-9	Ticket details (customer)	1. Showing the actual query, status, assigned agent details 2. Status of the ticket - OPEN, AGENT ASSIGNED, IN PROCESS, COMPLETE, CLOSED
FR-10	Address Column	Agent clarifies the doubts of the customer

b. Non-Functional requirements

1. A non-functional requirement defines the quality attribute of a software system
2. It places constraint on "How should the software system fulfil the functional requirements?"

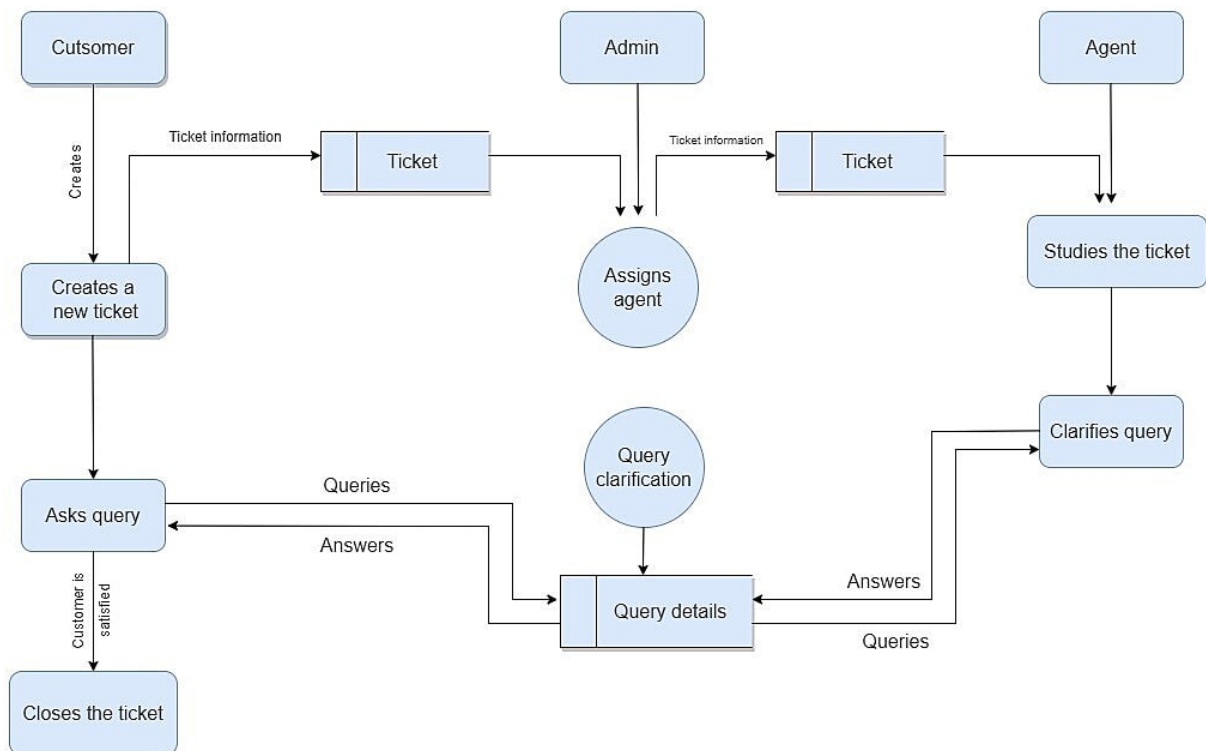
3. It is not mandatory
4. Applied to system as a whole
5. Usually more difficult to define
6. Helps you verify the performance of the software

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Customers can use the application in almost all the web browsers. Application is with good looking and detailed UI, which makes it more friendly to use.
NFR-2	Security	Customers are asked to create an account for themselves using their email which is protected with an 8 character-long password, making it more secure.
NFR-3	Reliability	Customers can raise their queries and will be replied with a valid reply, as soon as possible, making the application even more reliable and trust-worthy.
NFR-4	Performance	Customers will have a smooth experience while using the application, as it is simple and is well optimised.
NFR-5	Availability	Application is available 24/7 as it is hosted on IBM Cloud
NFR-6	Scalability	In future, may be cross-platform mobile applications can be developed as the user base grows.

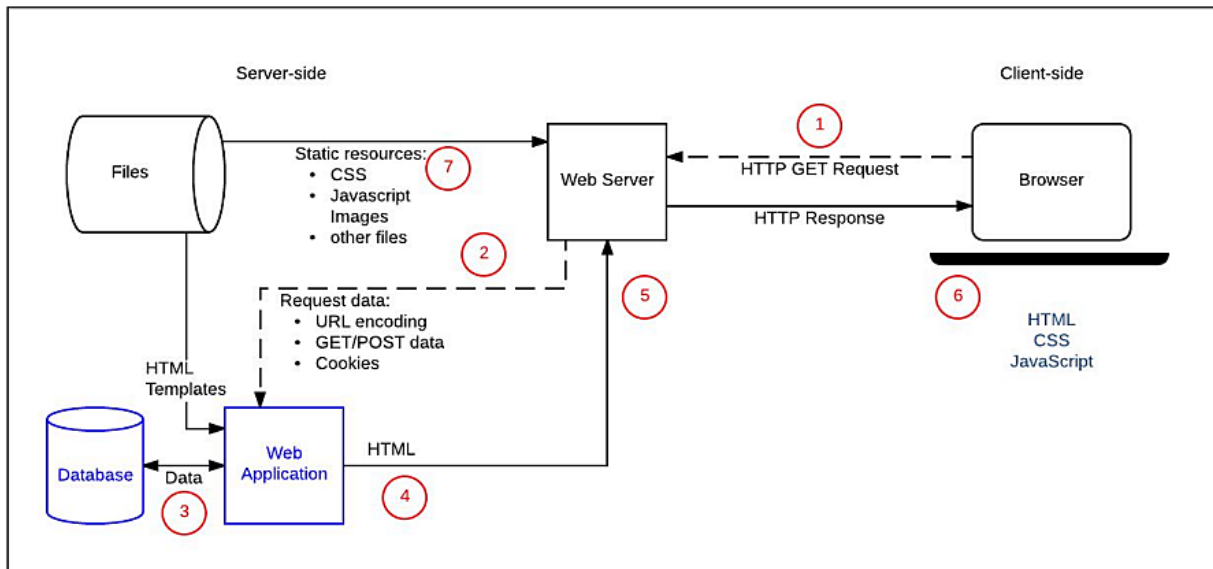
4. PROJECT DESIGN

a. Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



b. Solution & Technical Architecture



c. User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the	I can receive confirmation on email & click	High	Sprint-1

			application	confirm		
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-2
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard	USN-6	As a user , I can register the complaint in the register complaint page	I can register complaint(s)	High	Sprint-1
		USN-7	As a user , I can view the status of the complaint.	I can view status of complaint	Medium	Sprint-1
		USN-8	As a user, I can logout of the application	I can logout from the application	Low	Sprint-2
Customer Care Executive	Dashboard	USN-8	As a customer care Executive, I can resolve a complaint registered by	I can provide solution to a problem.	High	Sprint - 1

			user.			
Administrator	Registration	USN-9	As an admin, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-10	As an admin, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-11	As an admin I can log into the application(admin panel) by entering email & password		High	Sprint-1
	Dashboard	USN-12	As an admin, I can update the status of the complaint to the user with the help of customer care executive.	I can satisfy the customer on his/her query.	Medium	Sprint-2
		USN-13	As an admin , I can logout from the application	I can logout from the application	Low	Sprint - 2

5. PROJECT PLANNING & SCHEDULING

a. Sprint Planning & Estimation

Sprint 1:

1. We created a Flask Project.
2. Added all the routes needed for our project.
3. Created Tables in IBM Cloud.

Sprint 2:

1. We added all the html templates needed for our project.
2. We styled those pages using CSS and Bootstrap.
3. We wrote Queries to connect IBM Cloud Database.
4. Finished all the Fetching and Posting Stuff of IBM Cloud Database Integration.

Sprint 3:

1. Integration of Send grid into our application

Sprint 4:

1. Deploying the application using Docker and Kubernetes

b. Reports from JIRA

IT organizations have the challenge of ensuring system uptime, supporting users, and managing inventory of both hardware and software. IT teams gain significant efficiencies when one tool can support multiple business operations. According to Gartner, mastering the discipline of effective asset management is a huge cost savings for companies.

6. CODING & SOLUTIONING (Explain the features added in the project along with code)

a. Feature 1

Flask Framework is added.

b. Feature 2

Send Mail using SendGrid

We recommend using SendGrid Python, our client library, available on G..

We recommend using SendGrid Python, our client library, available on GitHub, with full documentation...

<https://docs.sendgrid.com/for-developers/sending-email/v3-python-code-example>

c. Database Schema (if Applicable)

DB2 is used as database.

There are various ways of accessing databases such as JDBC, JavaScript..

There are various ways of accessing databases such as JDBC, JavaScript, JSP, Python and many others. Here, we will be specifically talking.....

<https://medium.com/mozilla-firefox-club/accessing-ibm-db2-database-using-python-c356a4a76bf3>

7. TESTING

a. Test Cases

No	Feature Type	Component	Test Scenario	Test Data	Expected Result	Actual Result	Status
1	Functional	Registration Page	Customer is trying to register with the invalid data	First Name = Bala Last Name = Abinеш Role = Customer Email = suryathaya10@gmail.com Password = 12345678	Customer should get an alert saying "Passwords do not match"	Working as expected	Pass

				Confirm Password = 123456789			
2	Functional	Registration Page	Customer is trying to register with the invalid data	First Name = Bala Last Name = Abinеш Role = Customer Email = suryathaya10gmail.com Password = 12345678 Confirm Password = 12345678	Customer should get an alert saying "Invalid email"	Working as expected	Pass
3	Functional	Registration Page	Customer is trying to register with the invalid data	First Name = Ba Last Name = Abinеш Role = Customer Email = suryathaya10@gmail.com Password = 12345678 Confirm Password = 12345678	Customer should get an alert saying "Firstname should be atleast 6 characters long!"	Working as expected	Pass
4	Functional	Login page	Customer is trying to register with the invalid data	First Name = Bala Last Name = Abinеш Role = Customer Email = suryathaya10@gmail.com Password = 1234 Confirm Password = 1234	Customer should get an alert saying "Passwords must be at least 8 characters long!"	Working as expected	Pass

5	Functional	Registration Page	Customer is trying to register with the valid data	First Name = Bala Last Name = Abinеш Role = Customer Email = suryathaya@10gmail.com Password = 12345678 Confirm Password = 12345678	Customer's profile is added in the database and the customer is registered. Then, the customer is re-directed to the Login page to login	Working as expected	Pass
---	------------	-------------------	--	--	--	---------------------	------

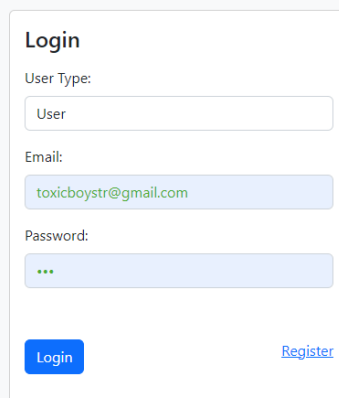
b. User Acceptance Testing

Test Scenarios

		User Type
1	Verifying customer is able to login to the application	Customer
2	Verifying customer is able to logout of the application	Customer
3	Verifying customer is able to change the password	Customer
4	Verifying customer is able to create a new ticket	Customer
5	Verifying customer is able to see all the tickets created	Customer
6	Verifying customer is able to have a chat with the Agent	Customer
7	Verifying customer is able to close the ticket	Customer
8	Verifying customer is able to see the past chats with the agents	Customer
9	Verifying customer is able to change the password using the Forgot password option	Customer
10	Verifying customer is able to receive all the necessary mails	Customer
1	Verifying agent is able to login to the application	Agent
2	Verifying agent is able to logout of the application	Agent

3	Verifying agent is able to change the password	Agent
4	Verifying agent is able to create a new ticket	Agent
5	Verifying agent is able to see all the tickets created	Agent
6	Verifying agent is able to have a chat with the Agent	Agent
7	Verifying agent is able to close the ticket	Agent
8	Verifying agent is able to see the past chats with the agents	Agent
9	Verifying agent is able to change the password using the Forgot password option	Agent
0	Verifying agent is able to receive all the necessary mails	Agent
1	Verifying admin is able to login to the application	Admin
2	Verifying admin is able to logout of the application	Admin
3	Verifying admin is able to see all the requests by the agents	Admin
4	Verifying admin is able to see all the unassigned tickets	Admin
5	Verifying admin is able to assign an agent for a ticket	Admin
6	Verifying admin is able to see all the feedbacks submitted	Admin

8. RESULTS



The screenshot shows a 'Login' form with the following fields and elements:

- Title:** Login
- User Type:** A dropdown menu with 'User' selected.
- Email:** A text input field containing 'toxicboyst@gmail.com'.
- Password:** A text input field with masked characters '***'.
- Buttons:** A blue 'Login' button and a blue 'Register' link.

Register

User Type:

Name:

Mobile:

Email:

Password:

Customer Care Registry

A future revolution system.

[Dashboard](#) [Assign Agent to Ticket](#) [Closed Ticket\(s\)](#) [Logout](#)

Welcome **manikandan!** [admin](#)

Welcome to Customer Care Registry

Customer Care Registry

A future revolution system.

Dashboard Active Ticket(s) Ready to Close Tickets Closed Ticket(s) Logout

Welcome **James Jacobraj!** agent

Ready to Close Tickets

Add New Ticket

Title	Description	Priority	User	Email	Mobile	Agent	Status	Note	
No Data Found.									

Customer Care Registry

A future revolution system.

Dashboard Assign Agent to Ticket Closed Ticket(s) Logout

Welcome **manikandan!** admin

Active Tickets

Add New Ticket

Title	Description	Priority	User	Email	Mobile	Agent	Status	Note	
123	123	high	karthick	karthickmce@gmail.com	04443858955	Selva	OPEN		Edit Close
test rajesh	chat is not working	high	karthick	karthickmce@gmail.com	04443858955	Selva	OPEN		Edit Close
website	not loaded	high	muthu	muthulakshmiraj26@gmail.com	9942203285	priya	OPEN		Edit Close
SMS not Working	SMS is not working properly	medium	muthu	muthulakshmiraj26@gmail.com	9942203285	Selva	OPEN		Edit Close
Hardware problem	software not working properly	high	kumar	toxicboyst@gmail.com	9361311374	James Jacobraj	OPEN		Edit Close

159.122.186.219:5000/tickets/active

9. ADVANTAGES & DISADVANTAGES

Advantages

- To solve the customer problem immediately using web portal
- To send the email alert to the customer and the agent.
- To use the user authentication as admin, agent and customer

Disadvantages

- Able to use the small level company
- Unable to send the SMS

9. CONCLUSION

Thus, there are many customer service applications available on the internet. Noting down the structural components of those applications and building a customer care registry. It will be web application build with Flask (Python micro-web framework), HTML, JavaScript. It will be a ticket-based customer service registry.

Customers can register into the application using their email, password, and a username. Then, they can login to the system, and raise as queries as they want in the form of their tickets.

These tickets will be sent to the admin, for which an agent is assigned. Then, the assigned agent will have a one-to-one chat with the customer and the latter's queries will be clarified. It is also the responsibility of the admin, to create an agent.

10.FUTURE SCOPE

This project can be extended in future like

- SMS Gateway

- Whatsapp Communication

11. APPENDIX

Source Code

Templates

login :

```
<html>
<head>
  <title>Customer Care System</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body class="bg-light">
<div class="container d-flex justify-content-center pt-5">

  <div class="card col-md-4 mb-4 mt-5">

    {% with messages = get_flashed_messages(with_categories=true) %}
    {% if messages %}
```

```

{% for category, message in messages %}
    <div class="flashes alert alert-{{category}}">
        <strong>{{ message }}</strong>
    </div>
{% endfor %}

{% endif %}
{% endwith %}

<div class="card-body">
    <h4 class="card-title">Login</h4>
<form method="post" action="/login">
    <div class="mb-3 mt-3">
        <label for="user_type" class="form-label">User Type:</label>
        <div class="dropdown">
            <select name="user_type" id="user_type" class="form-control">
                <option value="user">User</option>
                <option value="admin">Admin</option>
                <option value="agent">Agent</option>
            </select>
        </div>
    </div>
<div class="mb-3 mt-3">
    <label for="email" class="form-label">Email:</label>
    <input type="email" class="form-control" id="email" placeholder="Enter email"
name="username">
</div>
<div class="mb-3">
    <label for="pwd" class="form-label">Password:</label>

```



```

        <input type="password" class="form-control" id="pwd" placeholder="Enter password"
name="password">
    </div>
    <div class="form-check mb-3">
        <label class="form-check-label">
            <input class="form-check-input" type="checkbox" name="remember"> Remember me
        </label>
    </div>
    <button type="submit" class="btn btn-primary">Submit</button> <a href="/user/signup"
class="float-end">Register</a>
</form>
</div>
</div>
</div>

</body>
</html>

```

Register:

```

<html>
<head>
    <title>Signup</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet">
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>

```

```

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body class="bg-light">
<div class="container d-flex justify-content-center pt-5">
  <div class="card col-md-4 mb-4 mt-5">
    <div class="card-body">
      <h4 class="card-title">Register</h4>

      <form method="post" action="/user/signup">

        <div class="mb-3 mt-3">
          <label for="user_type" class="form-label">User Type:</label>
          <div class="dropdown">
            <select name="user_type" id="user_type" class="form-control" required>
              <option value="user">User</option>
              <option value="admin">Admin</option>
              <option value="agent">Agent</option>
            </select>
          </div>
        </div>

        <div class="mb-3 mt-3">
          <label for="name" class="form-label">Name:</label>
          <input type="name" class="form-control" id="name" placeholder="Enter name"
name="name" required>
        </div>

        <div class="mb-3 mt-3">

```

```

        <label for="mobile" class="form-label">Mobile:</label>
        <input type="text" class="form-control" id="mobile" placeholder="Enter
mobile" name="mobile" required>
    </div>
    <div class="mb-3 mt-3">
        <label for="email" class="form-label">Email:</label>
        <input type="email" class="form-control" id="email" placeholder="Enter email"
name="email" required>
    </div>
    <div class="mb-3 mt-3">
        <label for="password" class="form-label">Password:</label>
        <input type="password" class="form-control" id="password"
placeholder="Enter password" name="password" required>
    </div>

    <button type="submit" class="btn btn-danger">Cancel</button>
    <button type="submit" class="btn btn-success float-end">Save</button>
    </form>
</div>

</div>

</body>
</html></select>

```

Tickets:

```
{% extends "common_template.html" %}
{% block title %} {{title}} {% endblock %}
{% block content %}
<section class="vh-100">
  <div class="container">
    <div class="row">
      <div class="col-md-6">
        <h4 class="pull-left">{{title}}</h4>
      </div>
      <div class="col-md-6 d-flex flex-row-reverse">
        <a href="/ticket/create"><button class="btn btn-sm btn-success float-right">Add
New Ticket</button></a>
      </div>
    </div>
  </div>

<table class="table table-bordered">
  <tr>
    <th>Title</th>
    <th>Description</th>
    <th>Priority</th>
    <th>User</th>
    <th>Agent</th>
    <th>Note</th>
    <th></th>
  </tr>
  {%if tickets | length == 0%}
  <tr>
```

```

        <td colspan="7"><em>No Data Found.</em></td>
    </tr>
    {%endif%}
    {%for inv in tickets%}
    <tr>
        <td>{{inv['TITLE']}}</td>
        <td>{{inv['DESCRIPTION']}}</td>
        <td>{{inv['PRIORITY']}}</td>
        <td>{{inv['USER_NAME']}}</td>
        <td>{{inv['AGENT_NAME']}}</td>
        <td>{{inv['NOTE']}}</td>
        <td><a href="/ticket/edit/{{inv['ID']}}">Edit</a>
        {%if session.user_type == 'admin' and inv['STATUS'] != 1 %}
        <a href="javascript:void(0)" onclick="if(confirm('Are you sure to close this ticket?'))
window.location = '/ticket/close/{{inv['ID']}}'; ">Close</a>
        {% endif %}
    </td>
    </tr>
    {%endfor%}
</table>
</div>
</section>
{% endblock %}

```

Dashboard:

```

{% extends "common_template.html" %}
{% block title %}Dashboard{% endblock %}
{% block content %}

```

```

<section class="vh-100">
  <div class="container">
    <div class="row d-flex justify-content-center align-items-center h-100">
      <div class="col-xl-9">
        <div class="card" style="border-radius: 15px;">
          <div class="card-body">
            <div class="row align-items-center pt-4 pb-3">
              <p>Welcome to Customer Care System (CCS)</p>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</section>
{% endblock %}

```

Email_ticket_closed:

```

<div class="container d-flex justify-content-center pt-5">
  <div class="card col-md-4 mb-4 mt-5">
    Hi {{user.NAME}}, <br><br>
    Your ticket with following details has been closed. <br><br>
    <table border="1">
      <tr>
        <th>Title</th>
        <td>{{ticket.TITLE}}</td>
      </tr>
      <tr>
        <th>Description</th>
        <td>{{ticket.DESCRPTION}}</td>
      </tr>
    </table>
  </div>
</div>

```

```

<tr>
  <th>Priority</th>
  <td>{{ticket.PRIORITY}}</td>
</tr>
<tr>
  <th>Status</th>
  <td>{{ticket.STATUS == 1 and 'Closed' or 'Opened'}}</td>
</tr>
</table>
</div>
</div>

```

Email_agent_assigned:

```

<div class="container d-flex justify-content-center pt-5">
  <div class="card col-md-4 mb-4 mt-5">
    Hi {{agent.NAME}}, <br><br>
    Your have assigned to the ticket with following details. Please help the customer to solve
    this problem. <br><br>
    <table border="1">
      <tr>
        <th>Customer Name</th>
        <td>{{user.NAME}}</td>
      </tr>
      <tr>
        <th>Customer Mobile</th>
        <td>{{user.MOBILE}}</td>
      </tr>
      <tr>

```

```

        <th>Title</th>
        <td>{{ticket.TITLE}}</td>
    </tr>

    <tr>
        <th>Description</th>
        <td>{{ticket.DESCRPTION}}</td>
    </tr>

    <tr>
        <th>Priority</th>
        <td>{{ticket.PRIORITY}}</td>
    </tr>

    <tr>
        <th>Status</th>
        <td>{{ticket.STATUS == 1 and 'Closed' or 'Opened'}}</td>
    </tr>
</table>
</div>
</div>

```

Main python

```

main.py
# This is a sample Python script.
# Press Shift+F10 to execute it or replace it with your code.
# Press Double Shift to search everywhere for classes, files, tool windows, actions, and settings.
import ibm_db
from flask import Flask, render_template, request, redirect, url_for, flash, session
from ticket.User import User
from ticket.Ticket import Ticket
import sendgrid
import os
from sendgrid.helpers.mail import *

```



```

app = Flask(__name__)
app.secret_key = b'_4#z2G"F5Q9z\n\xec]/'

@app.route("/")
def show_login():
    return redirect(url_for('login'))

@app.route("/login", methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        print("hi")
        if request.form['username'] != "" and request.form['password'] != "":
            user = User()
            user.User_Type = request.form['user_type']
            user.Email = request.form['username']
            user.Password = request.form['password']
            result = user.login()

            print("login result", result)
            if len(result) > 0:
                session['name'] = result[0]['NAME']
                session['user_id'] = result[0]['ID']
                session['user_type'] = result[0]['USER_TYPE']

            return redirect(url_for('dashboard'))

        else:
            flash(u'username or password is incorrect.', 'danger')
            return redirect(url_for('login'))

    else:
        return render_template('login.html')

```

```

@app.route("/user/signup", methods=['GET', 'POST'])
def vendor_signup():
    if request.method == 'POST':
        user = User()
        user.Id = ""
        user.Name = request.form['name']
        user.User_Type = request.form['user_type']
        user.Mobile = request.form['mobile']
        user.Email = request.form['email']
        user.Password = request.form['password']
        user.save()

        flash(u'User Sign up done, you login now with your username and password.', 'success')

        return redirect(url_for('login'))
    else:
        return render_template('register.html')

@app.route("/dashboard", methods=['GET'])
def dashboard():
    if session['name'] is None:
        return redirect(url_for('login'))

    # inventory = Inventory()
    # inventory = inventory.display()
    return render_template('dashboard.html')

@app.route("/ticket/create", methods=['GET', 'POST'])
def create_ticket():
    if session['name'] is None:
        return redirect(url_for('login'))

    if request.method == 'POST':
        ticket = Ticket()
        ticket.Title = request.form['title']

```

```

ticket.Description = request.form['description']
ticket.Priority = request.form['priority']
id = request.form.get('id')

old_ticket = Ticket()

if id is not None:
    ticket.Id = id
    tickets = old_ticket.get(id)
    old_ticket = tickets[0]
agent_id = request.form.get('agent_id')
if agent_id is not None:
    ticket.AgentId = agent_id
status = request.form.get('status')
if status is not None:
    ticket.Status = status
ticket.Status = 0
ticket.save()

if ticket.AgentId != 0 and ticket.AgentId != old_ticket["AGENTID"]:
    return redirect(url_for('ticketagentassigned', ticket_id=id))

flash(u'Ticket has been saved successfully.', 'success')

return redirect(url_for('active_tickets'))
else:
    ticket = Ticket()
    agents = []
    return render_template('createcomplaint.html', ticket=ticket, agents=agents)

@app.route("/ticket/edit/<id1>", methods=['GET'])
def edit_ticket(id1):
    if session['name'] is None:
        return redirect(url_for('login'))

    ticket = Ticket()

```

```
tickets = ticket.get(id1)
ticket = tickets[0]

user = User()
agents = user.agents()
return render_template('createcomplaint.html', ticket=ticket, agents=agents)
```

```
@app.route("/tickets/active", methods=['GET'])
def active_tickets():
    if session['name'] is None:
        return redirect(url_for('login'))

    ticket = Ticket()
    ticket.Status = 0
    tickets = ticket.display()

    print(tickets)
    return render_template('tickets.html', title='Active Tickets', tickets=tickets)
```

```
@app.route("/tickets/closed", methods=['GET'])
def closed_tickets():
    if session['name'] is None:
        return redirect(url_for('login'))

    ticket = Ticket()
    ticket.Status = 1
    tickets = ticket.display()
    return render_template('tickets.html', title='Closed Tickets', tickets=tickets)
```

```
@app.route('/logout')
def logout():
    session.clear()
    return redirect(url_for('login'))
```

```

@app.route('/ticket/agent-assigned/<ticket_id>', methods=['GET'])
def ticketagentassigned(ticket_id):
    if session['name'] is None:
        return redirect(url_for('login'))

    id1 = ticket_id

    ticket = Ticket()
    ticket.close(id1)

    ticket = Ticket()
    tickets = ticket.get(id1)
    ticket = tickets[0]

    user = User()
    user.Id = ticket["USERID"]
    users = user.get()
    user = users[0]

    agent = User()
    agent.Id = ticket["AGENTID"]
    users = agent.get()
    agent = users[0]

    sg =
sendgrid.SendGridAPIClient(api_key="SG.PEMDvdpVSeqVl9BCQP5xjw.KSZztqZz5nx291w0
SmyXvug_nrTm5HpelEMCSkFj4Cs")
    from_email = Email("rajesh@malaris.com")
    to_email = To(user.Email)
    subject = "Customer Care Agent Assigned Notification"
    html_content = str(render_template('email_agent_assigned.html', ticket=ticket, user=user,
agent=agent))
    content = Content("text/html", html_content)
    print(html_content)
    mail = Mail(from_email, to_email, subject, content)
    response = sg.client.mail.send.post(request_body=mail.get())

```

```

print(response.status_code)
print(response.body)
print(response.headers)
return redirect(url_for('active_tickets'))

@app.route('/ticket/close/<ticket_id>', methods=['GET'])
def ticketclose(ticket_id):
    if session['name'] is None:
        return redirect(url_for('login'))

    id1 = ticket_id

    ticket = Ticket()
    ticket.close(id1)

    ticket = Ticket()
    tickets = ticket.get(id1)
    ticket = tickets[0]

    user = User()
    user.Id = ticket["USERID"]
    users = user.get()
    user = users[0]
    sg =
sendgrid.SendGridAPIClient(api_key="SG.PEMDvdpVSeqVl9BCQP5xjw.KSZztqZz5nx291w0
SmyXvug_nrTm5HpelEMCSkFj4Cs")
    from_email = Email("rajesh@malaris.com")
    to_email = To(user["EMAIL"])
    subject = "Customer Care Ticket Closed Notification"
    content = Content("text/html", render_template('email_ticket_closed.html', ticket=ticket,
user=user))
    mail = Mail(from_email, to_email, subject, content)
    response = sg.client.mail.send.post(request_body=mail.get())
    print(response.status_code)
    print(response.body)
    print(response.headers)

```

```
return redirect(url_for('active_tickets'))
```

```
if __name__ == "__main__":  
    port = int(os.environ.get('PORT', 5000))  
    app.run(debug=True, host='0.0.0.0', port=port)  
# See PyCharm help at https://www.jetbrains.com/help/pycharm/
```

GitHub & Project Demo Link

GITHUB LINK:<https://github.com/IBM-EPBL/IBM-Project-43602-1660718377>

PROJECT DEMO LINK:

<https://drive.google.com/file/d/1bv2Sf9bgO0R XF9R2oXrTMerCpiCa8p9z/view?usp=drivesdk>