

CUSTOMER CARE REGISTRY

Project Report

Team ID: PNT2022TMID37544

Team Members

811219205017 SURYA A – Team Leader

811219205001 AKASH V – Team Member 1

811219205011 RANJITH M – Team Member 2

811219205012 RANJITH KUMAR A – Team Member 3

College Name: INDRA GANESAN COLLEGE OF ENGINEERING

CONTENTS

1. INTRODUCTION

1. Project Overview
2. Purpose

2. LITERATURE SURVEY

1. Existing problem
2. References
3. Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

1. Empathy Map Canvas
2. Ideation & Brainstorming
3. Proposed Solution
4. Problem Solution fit

4. REQUIREMENT ANALYSIS

1. Functional requirement
2. Non-Functional requirements

5. PROJECT DESIGN

1. Data Flow Diagrams
2. Solution & Technical Architecture
3. User Stories

6. PROJECT PLANNING & SCHEDULING

1. Sprint Planning & Estimation
2. Sprint Delivery Schedule
3. Reports from JIRA

7. CODING & SOLUTIONING

1. Feature 1
2. Feature 2
3. Database Schema (if Applicable)

8. TESTING

1. Test Cases
2. User Acceptance Testing

9. RESULTS

1. Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Customer care registry application has been developed to help the customer in processing their complaints. The customers can raise the ticket with a detailed description of the issue. An Agent will be assigned to the Customer to solve the problem. Whenever the agent is assigned to a customer, they will be notified with an email alert. Customers can view the status of the ticket till the service is provided.

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 if the information relates to a service as opposed to a Customer.

Customer Service may be provided by a Service Representatives Customer Service is normally an integral part of a company's customer value proposition.

1.2 PURPOSE

The purpose of the project is to:

- Provide a common platform to the customers to clarify their queries
- Having expert agents in the platform for better answering
- Customer's tickets (queries) are answered quickly by the agents
- Customers and Agents can chat with one another for better understanding
- While doing so, the former asks questions
- Later, answers those questions as quickly and as legitimately as possible
- Customers can raise as many tickets as they want.
- 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 if the information relates to a service as opposed to as Customer.
- Customer Service may be provided by a Service Representatives Customer Service is normally an integral part of a company's customer value proposition.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

- ❖ The existing system is a semi-automated at where the information is stored in the form of excel sheets in disk drives. The information sharing to the Volunteers, Group members, etc. is through mailing feature only. The information storage and maintenance is more critical in this system. Tracking the member's activities and progress of the work is a tedious job here. This system cannot provide the information sharing by 24x7 days.
- ❖ Reviews and rating in the e-commerce websites are not reliable
- ❖ Even more so, they are often been given by the manufactures themselves
- ❖ Reviews are not from the authentic individuals
- ❖ After buying the products, I am left with no option to clear my doubts
- ❖ There is no common platform available to us, the customers, to have our doubts cleared
- ❖ If it is existing, we are not getting fast replies. By the time, the reply comes, the issue might have been cleared or of not worth of being cleared to the customers.

2.2 REFERENCES

<https://www.helpdesk.com/>

<https://freshdesk.com/helpdesk-software>

<https://freshdesk.com/resources/case-study/hamleys>

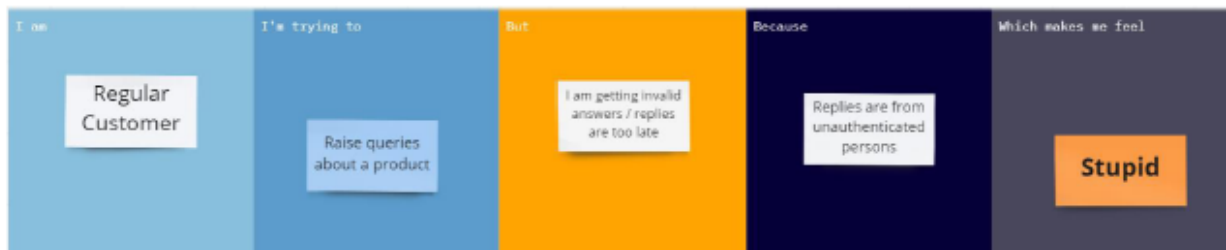
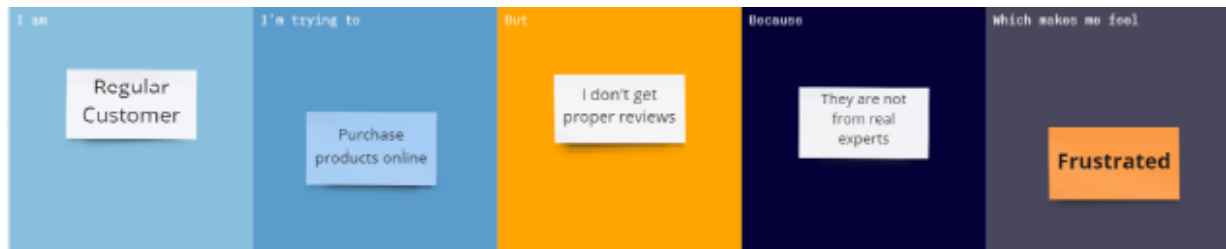
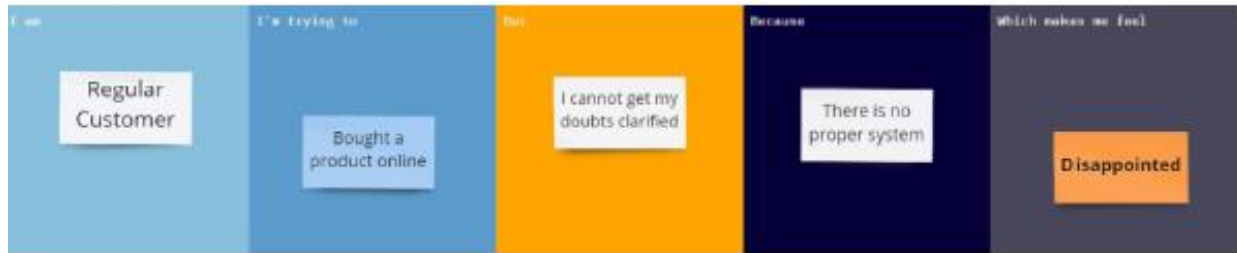
<https://pulsedesk.com/>

<https://www.redpoints.com/blog/amazon-fake-reviews/>

2.3 PROBLEM STATEMENT DEFINITION

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

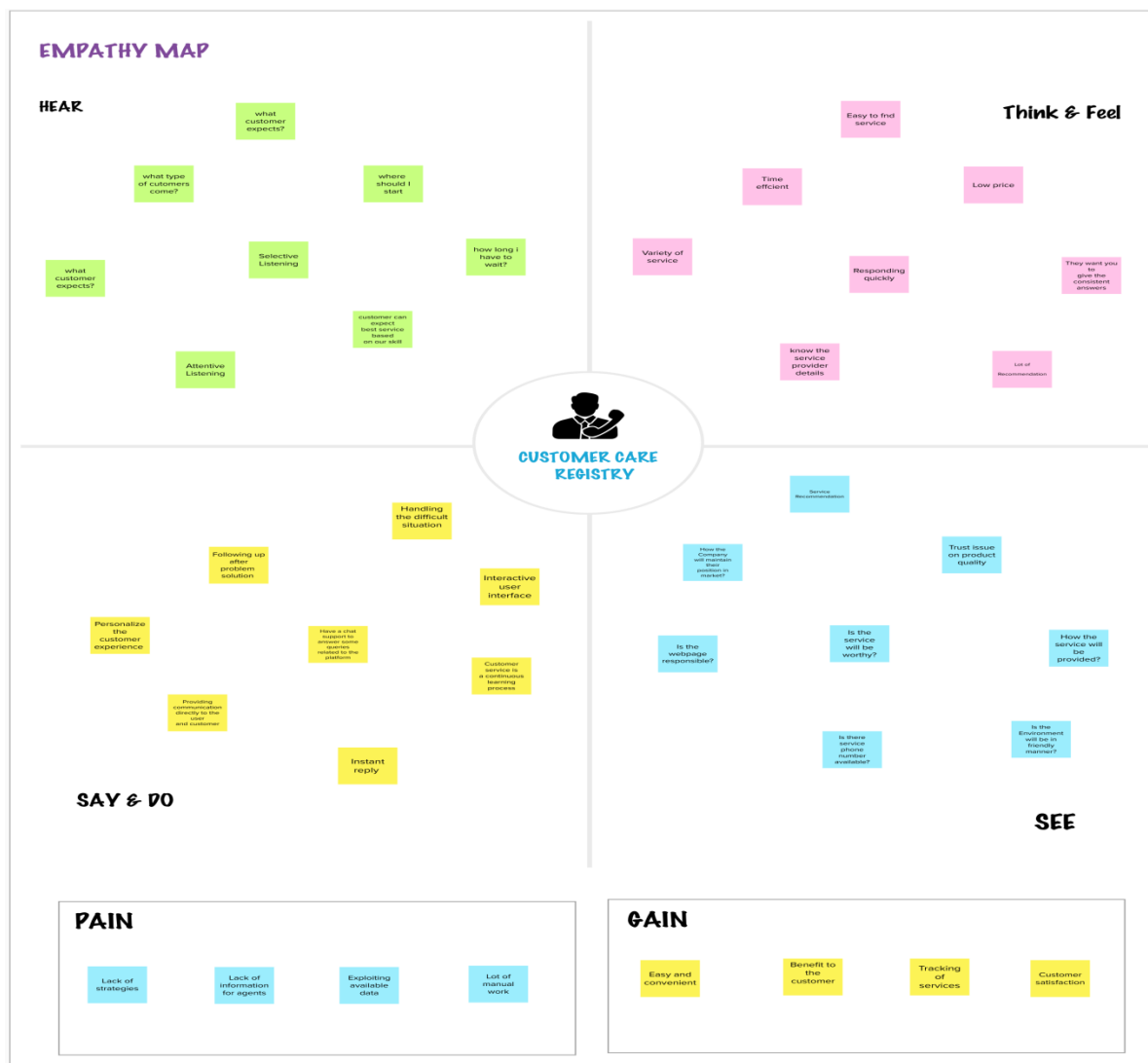


Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Regular Customer	Bought a product	I cannot get my doubts clarified	There is no proper system	Disappointed
PS-2	Regular Customer	Purchase products online	I don't get proper reviews	They are not from real experts.	Frustrated
PS-3	Regular customer	Raise queries about a product	I am getting invalid answers / replies are too late	Replies are from unauthenticated persons	Stupid

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

- ❖ 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 to 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.



3.2 IDEATION & BRAINSTORMING

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.

Brainstorm



SURYA A

Checking customer needs	Customer privacy	Listen carefully to the queries
Providing service details	Deals with problem quickly	Solution for customer issues
Providing services on time	Proper solution to any problem	Providing chatbox

AKASH V

User feedback	Notifying custoer	Allocating agent
Service at any time	Email notification	Security
Filtration based on services	Live chatbox	Ask for rating

RANJITH M

Customer privacy	Tracking of services	Quick solution of the problem
Proper information	Managing database	Tracking of services
Customer satisfaction	Proper allocation of staff	Solve the problem in short time

RANJITH KUMAR A

Agent details	Customer details	Kind behaviour among the customers
Customer queries	Appropriate solution	Giving proper clarity
Service at any time	Notifying customer	Future assistance

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

🕒 20 minutes

CUSTOMER



AGENT



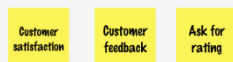
SERVICES



SECURITY



FEEDBACK

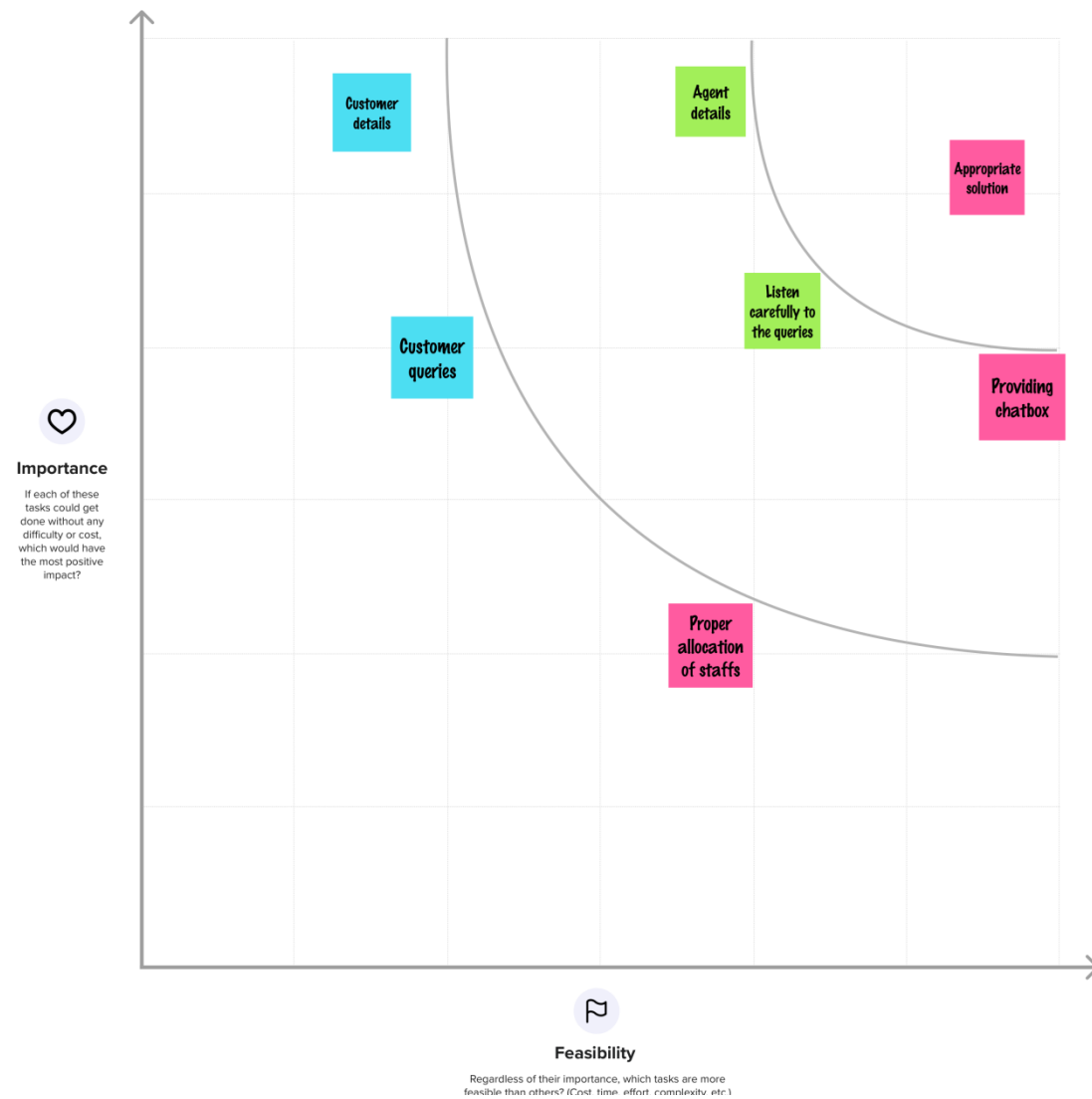


4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

🕒 20 minutes



3.3 PROPOSED SOLUTION

S. No	Parameter	Description
1.	Problem Statement (Problem to be solved)	To solve customer issues using Cloud Application Development.
2.	Idea / Solution description	Assigned Agent routing can be solved by directly routing to the specific agent about the issue using the specific Email. Automated Ticket closure by using daily sync of the daily database. Status Shown to the Customer can display the status of the ticket to the customer. Regular data retrieval in the form of retrieving lost data.
3.	Novelty / Uniqueness	Assigned Agent Routing, Automated Ticket Closure, Status Shown to the Customer, and Backup data in case of failures.
4.	Social Impact / Customer Satisfaction	Customer Satisfaction, Customer can track their status and Easy agent communication.
5.	Business Model (Revenue Model)	<ul style="list-style-type: none"> Key Partners are Third-party applications, agents, and customers. Activities held as Customer Service, System Maintenance. Customer Relationship have 24/7 Email Support, Knowledge-based channel.
6.	Scalability of the Solution	The real goal of scaling customer service is providing an environment that will allow your customer service specialists to be as efficient as possible. An environment where they will be able to spend less time on grunt work and more time on actually resolving critical customer issues.

3.4 PROBLEM SOLUTION FIT

Project Title: Customer Care Registry

Project Design Phase-I Problem Solution Fit

Team ID : PNT2022TMD45009

Define CS, fit into CC	<p>1. CUSTOMER SEGMENT(S) CS</p> <p>Who is your customer? i.e. working parents of 0-5 y.o. kids</p> <p>Our customers are usually above 16 years old. Ranging from college students to working adults to retired professionals. Also, reputed organizations too.</p>	<p>6. CUSTOMER CONSTRAINTS CC</p> <p>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.</p> <ol style="list-style-type: none"> 1. Complicated process to take over 2. Late replies to their queries 3. High chance their queries may not be considered at all 4. Replies irrelevant to their queries 5. Advertisements shown 	<p>5. AVAILABLE SOLUTIONS AS</p> <p>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</p> <p>Customers most probably use helpdesk.</p> <p><u>Pros:</u></p> <ol style="list-style-type: none"> 1. Reasonably priced 2. Highly scalable for team of any size <p><u>Cons:</u></p> <p>They do not understand the severity of all complaints and end up treating them all in the same way</p>	Explore AS, differentiate
	<p>2. JOBS-TO-BE-DONE / PROBLEMS J&P</p> <p>Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.</p> <ol style="list-style-type: none"> 1. Simplifying the user account creation process Giving instant replies to the customers to their queries 2. Providing expert solutions to the queries 3. Assigning individual agents/experts to the customers queries 4. Sending the status of the queries to the customer's mail <p>Overtime, they get disappointed with late and irrelevant replies and triggered to act</p>	<p>9. PROBLEM ROOT CAUSE RC</p> <p>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.</p> <ol style="list-style-type: none"> 1. No proper registry 2. Replies for queries from random persons 3. Lack of experts in a common place 4. High-cost 	<p>7. BEHAVIOUR BE</p> <p>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)</p> <ol style="list-style-type: none"> 1. Asking their friend's opinions 2. Checking solutions in the online forums 3. Using helpdesk 4. Solve the issues themselves based on their own knowledge 5. Seeing reviews posted by the users in the website forums 	Focus on J&P, tap into BE, understand RC
Identify strong TR & EM	<p>4. EMOTIONS:BEFORE/ AFTER EM</p> <ol style="list-style-type: none"> 1. Disappointed - after they do not get instant replies for their queries 2. Dejected - when they get irrelevant replies even after waiting for a long time 	<ol style="list-style-type: none"> 1. Creating a Customer Care Registry 2. Simple User creation process 3. Customers can raise their queries to the experts 4. Individual agents will be assigned to each customer 5. Their queries will be answered earnestly 6. Customers can also check the status of their queries 	<p>8.2 OFFLINE What info do customers take offline? Extract offline channels from #7 and use them for customer development.</p> <p><u>ONLINE:</u></p> <ol style="list-style-type: none"> 1. https://www.helpdesk.com/ 2. https://www.google.com/ 3. https://www.quora.com/ <p><u>OFFLINE:</u></p> <ol style="list-style-type: none"> 1. Asking friends and colleagues 2. Take actions themselves 	Identify strong TR & EM

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

- ❖ 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.
- ❖ It specifies “what should the software system do?”
- ❖ Defined at a component level
- ❖ Usually easy to define
- ❖ 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	Forgot Password	Resetting the password by sending an OTP to user's mail (customer, agent, admin)
FR-3	User Login	Login through Login form (customer, agent, user)
FR-4	Agent creation (admin)	Create an agent profile with username, email and password
FR-5	Dashboard (customer)	Show all the tickets raised by the customer
FR-6	Dashboard (agent)	Show all the tickets assigned to the agent by admin
FR-7	Dashboard (Admin)	Show all the tickets raised in the entire system
FR-8	Ticket creation (customer)	Customer can raise a new ticket with the detailed description of his/her query
FR-9	Assign agent (admin)	Assigning an agent for the created ticket
FR-10	Ticket details (customer)	1. Showing the actual query, status, assigned agent details 2. Status of the ticket
FR-11	Address Column	Agent clarifies the doubts of the customer

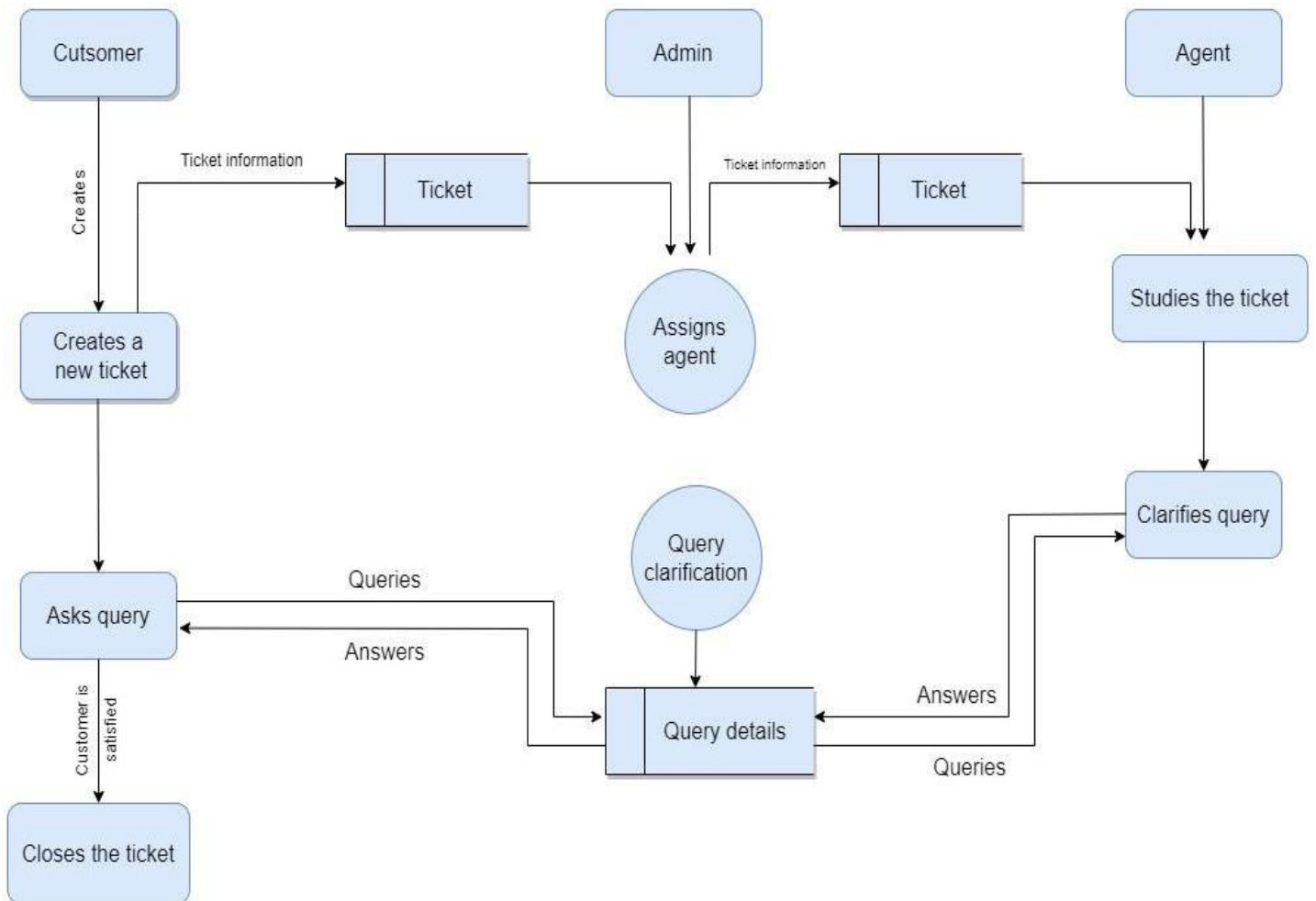
4.2 NON FUNCTIONAL REQUIREMENTS

- ❖ A non-functional requirement defines the quality attribute of a software system
- ❖ It places constraint on “How should the software system fulfill the functional requirements?”
- ❖ It is not mandatory
- ❖ Applied to system as a whole
- ❖ Usually more difficult to define
- ❖ 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 detailedUI, 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 optimized.
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.

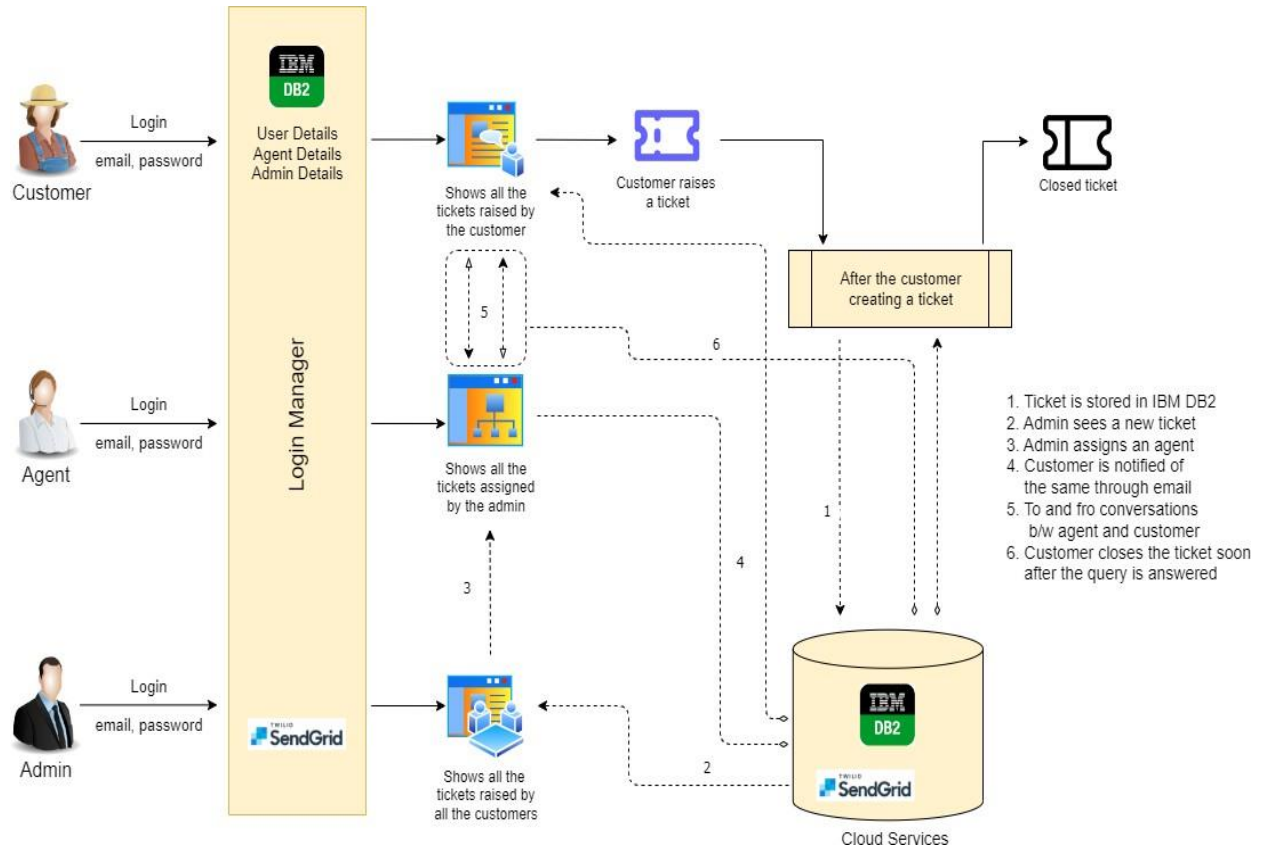
5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

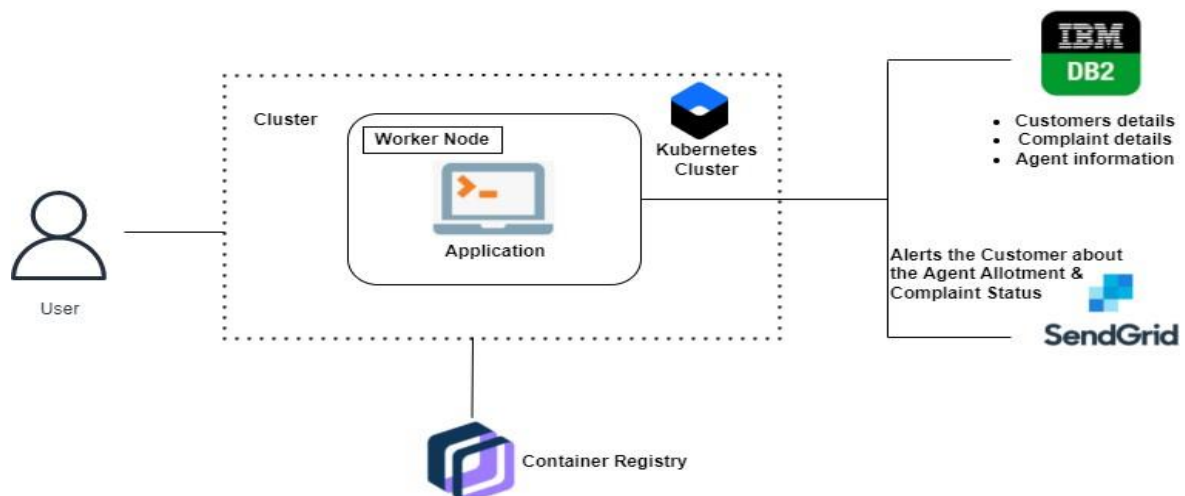


5.2 SOLUTION & TECHNICAL ARCHITECTURE

SOLUTION ARCHITECTURE



TECHNICAL ARCHITECTURE



5.3 USER STORIES

User Type	Functional Requirements (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Registration	USN-1	As a customer, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Login	USN-2	As a customer, I can login to the application by entering correct email and password	I can access my account / dashboard	High	Sprint-1
	Dashboard	USN-3	As a customer, I can see all the tickets raised by me and lot more	I get all the info needed in my dashboard	High	Sprint-1
	Ticket creation	USN-4	As a customer, I can create a new ticket with the detailed description of my query	I can ask my query	High	Sprint-2
	Address Column	USN-5	As a customer, I can have conversations with the assigned agent and get my queries clarified	My queries are clarified	High	Sprint-3
	Forgot password	USN-6	As a customer, I can reset my password by this option in case I forgot my old password	I get access to my account again	Medium	Sprint-4
	Ticket details	USN-7	As a customer, I can see the current status of my tickets	I get better understanding	Medium	Sprint-4
Agent (Web user)	Login	USN-1	As an agent, I can login to the application by entering correct email and password	I can access my account / dashboard	High	Sprint-3
	Dashboard	USN-2	As an agent, I can see all the tickets assigned to me by the admin	I can see the tickets to which I could answer	High	Sprint-3

	Address Column	USN-3	As an agent, I get to have conversations with the customer and clear his/her queries	I can clarify the issues	High	Sprint-3
	Forgot password	USN-4	As an agent, I can reset my password by this option in case I forgot my old password	I get access to my account again	Medium	Sprint-4
Admin (Web user)	Login	USN-1	As an admin, I can login to the application by entering correct email and password	access my account / dashboard	High	Sprint-1
	Dashboard	USN-2	As an admin, I can see all the tickets raised in the entire system and lot more	I can assign agents by seeing those tickets	High	Sprint-1
	Agent creation	USN-3	As an admin, I can create an agent for clarifying the customer's queries	I can create agents	High	Sprint-2
	Assigning agent	USN-4	As an admin, I can assign an agent for each ticket created by the customer	Enables agent to clarify the queries	High	Sprint-2
	Forgot password	USN-4	As an admin, I can reset my password by this option in case I forgot my old password	I get access to my account again	Medium	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

SPRINT	User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Customer (Web User)	Registration	USN-1	As a customer, I can register for the application by entering my email, password, and confirming my Password.	2	High	Surya, Ranjith
Sprint-1		Login	USN-2	As a customer, I can login to the application by entering correct email and password	1	High	Akash,Ranjith Kumar
Sprint-1		Dashboard	USN-3	As customer, I can see all the tickets raised by me and lot more	3	High	Surya
Sprint-2		Ticket Creation	USN-4	As a customer, I can create a new ticket with the detailed description of my query	2	High	Surya
Sprint-3		Address column	USN-5	As a customer, I can have conversations with the assigned agent and get my queries clarified	3	High	Ranjith Kumar, Surya
Sprint-4		Forgot password	USN-6	As a customer, I can reset my password by this option in case I forgot my old password	2	Medium	Ranjith Kumar, Ranjith
Sprint-4		Ticket Details	USN-7	As a customer, I can see the current status of my tickets	2	Medium	Surya, Ranjith
Sprint-3	Agent (Web user)	Login	USN-1	As an agent, I can login to the application by entering correct email and password	2	High	Akash

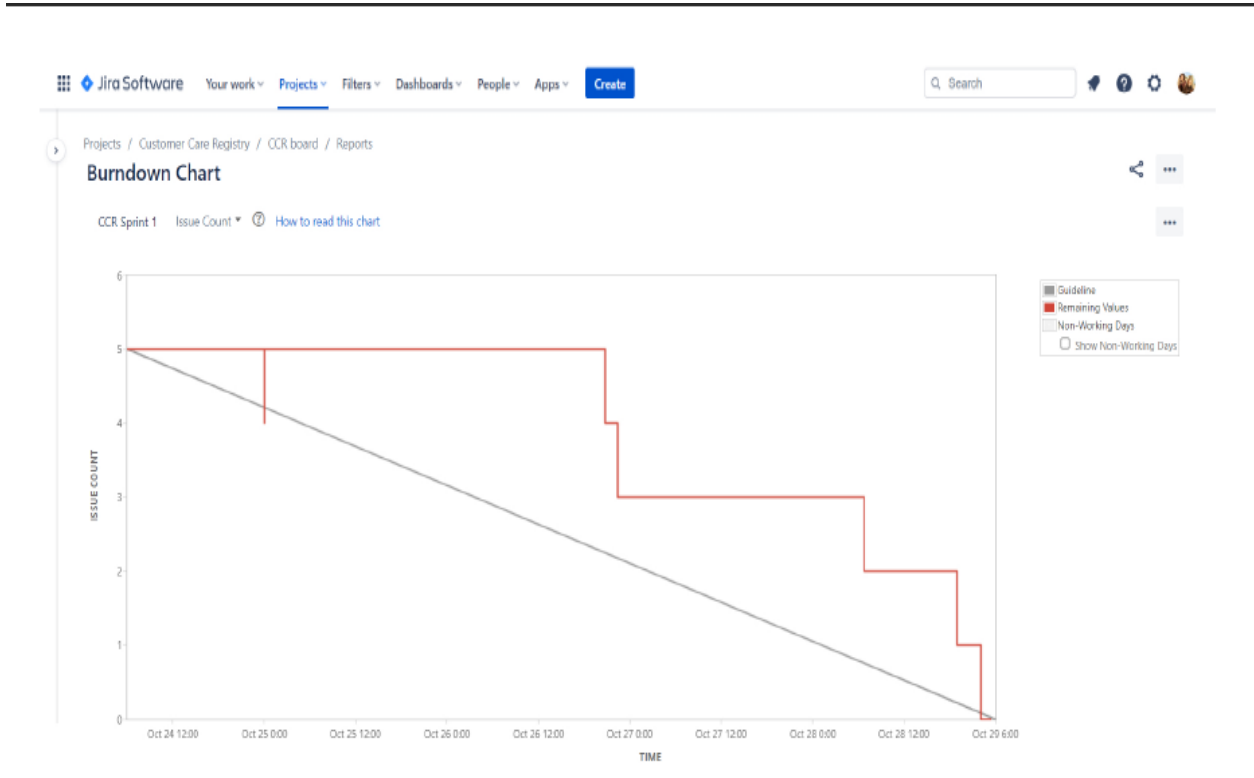
Sprint-3		Dashboard	USN-2	As an agent, I can see all the tickets assigned to me by the admin	3	High	Ranjith
Sprint-3		Address column	USN-3	As an agent, I get to have conversations with the customer and clear his/her queries	3	High	Surya, Ranjith Kumar
Sprint-4		Forgot password	USN-4	As an agent, I can reset my password by this option in case I forgot my old password	2	Medium	Akash, Surya
Sprint-1	Admin (Web user)	Login	USN-1	As an admin, I can login to the application by entering correct email and password	1	High	Ranjith, Akash
Sprint-1		Dashboard	USN-2	As an admin, I can see all the tickets raised in the entire system and lot more	3	High	Ranjith Kumar
Sprint-2		Agent creation	USN-3	As an admin, I can create an agent for clarifying the customer's queries	2	High	Ranjith Kumar
Sprint-2		Assigning agent	USN-4	As an admin, I can assign an agent for each ticket created by the customer	3	High	Akash, Ranjith
Sprint-4		Forgot Password	USN-4	As an admin, I can reset my password by this option incase I forgot my old password	2	Medium	Ranjith Kumar, Akash

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Total story points	Duration	Sprint Start Date	Sprint End Date	Story Points Completed	Sprint Release Date (Actual)
Sprint-1	10	6 days	24 Oct 2022	29 Oct 2022	10	29 Oct 2022
Sprint-2	7	6 days	31 Oct 2022	01 Nov 2022	7	05 Nov 2022
Sprint-3	11	4 days	06 Nov 2022	11 Nov 2022	11	09 Nov 2022
Sprint-4	8	4 days	10 Nov 2022	15 Nov 2022	8	13 Nov 20022

6.3 REPORTS FROM JIRA

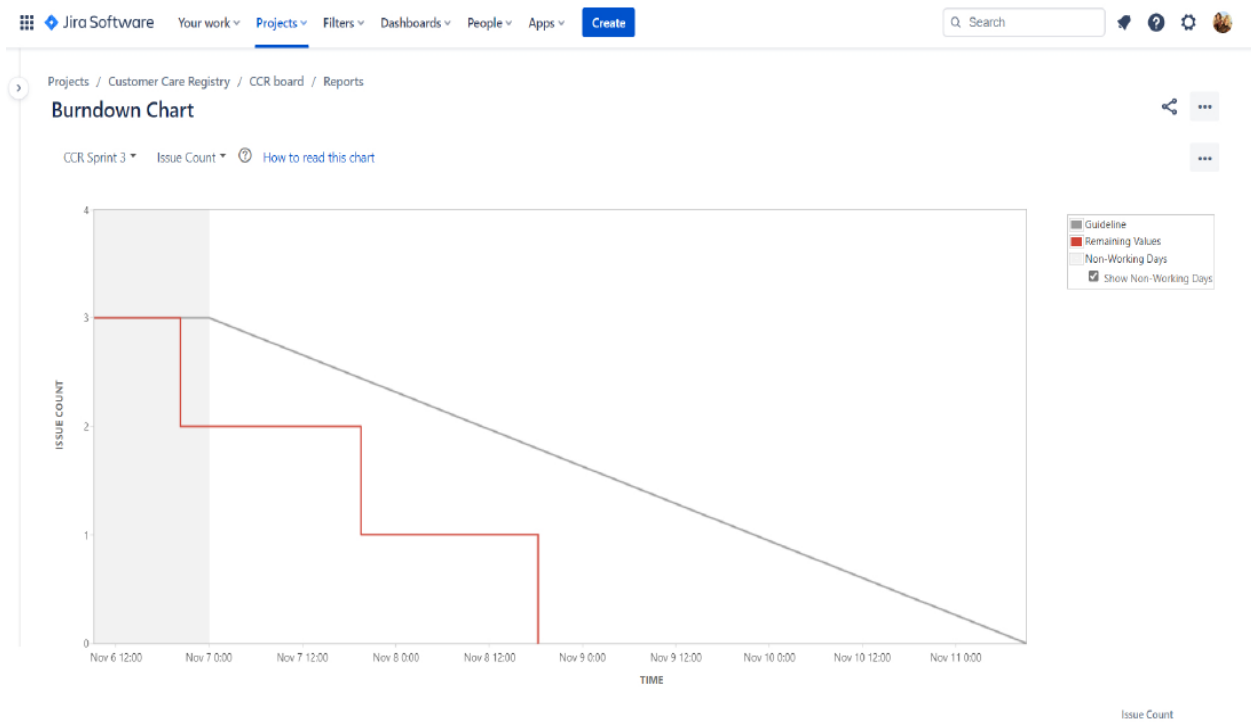
Sprint-1 Burndown Chart



Sprint-2 Burndown Chart



Sprint-3 Burndown Chart



Sprint-4 Burndown Chart



7. CODING & SOLUTIONING

7.1 FEATURE 1

Admin assigning an agent to a ticket

```
@admin.route('/admin/update/<agent_id>/<ticket_id>*
```

```
@login required
```

```
assign(agent_id, ticket_id)
```

Assigning an agent to the ticket

views admin

```
hasattr(admin, 'email')
```

```
# query to update the ASSIGNED_TO of a ticket
```

```
assign_agent_query = *
```

```
UPDATE tickets SET assigned_to = ? WHERE ticket_id = ?
```

```
stmt = ibm_db.prepare(conn, assign_agent_query)
```

```
ibm_db.bind_param(stmt, 1, agent_id)
```

```
ibm_db.bind_param(stmt, 2, ticket_id)
```

```
ibm_db.execute(stmt,
```

```
“None”
```

```
# Logging out
```

```
redirect (url_for('blue_print.logout))
```

Explanation

- ❖ User creates a ticket by describing the query
- ❖ Admin views the newly created ticket in the dashboard
- ❖ In the dropdown given, admin selects an agent
- ❖ Once selected, using fetch() the request is sent to the server
- ❖ The request URL contains both the Ticket ID and the selected Agent ID

- ❖ Using the shown SQL query, the assigned_to column of the tickets table is set to agent_id where the ticket_id column = ticket_id
- ❖ Then, the dashboard of the admin gets refreshed.

7.2 FEATURE

Customer closing the ticket

```
@cust.route("/customer/close/<ticket_id>/
```

```
@login_required
```

```
close(ticket_id
```

Customer can close the ticket

param ticket_id ID of the ticket that should be closed

views customer

```
hasattr(customer, "uuid")):
```

```
# query to close the ticket
```

```
close_ticket = ""
```

```
UPDATE tickets SET query_status = ? WHERE ticket_id = ?
```

```
stmt = ibm_db.prepare(conn, close_ticket'
```

```
ibm_db.bind_param(stat, 1, "CLOSED
```

```
Lbm_db.bind_param(stat, 2, ticket_id
```

```
ibm_db.execute(stat:
```

```
redirect(url_for
```

```
customer.tickets").
```

```
# logging out
```

```
' redirect(url_for('blue_print.logout" )
```

Explanation

- ❖ User creates a ticket by describing the query
- ❖ Admin assigns an agent to this ticket
- ❖ The customer and the agent, chat with each other, in the view of clearing the customer's doubts
- ❖ Once the customer is satisfied, the customer decides to close the ticket

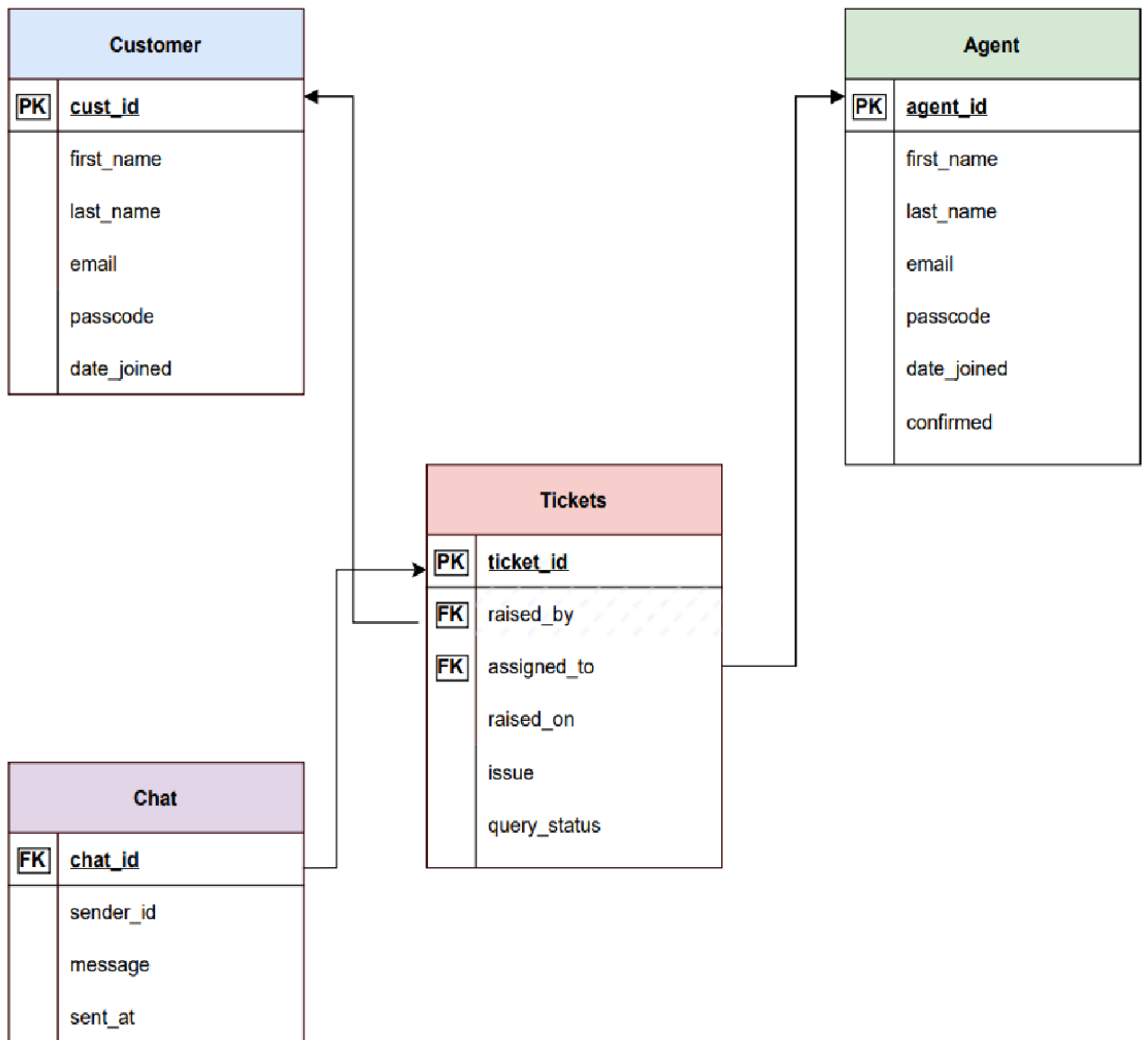
- ❖ Using `fetch()` the request is sent to the server. The requested URL contains the Ticket ID
- ❖ Using the shown SQL query, the status of the ticket is set to “CLOSED”
- ❖ Thus the ticket is closed
- ❖ Then the customer gets redirected to the all-tickets page

7.3 DATABASE SCHEMA

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful.

A database schema defines how data is organized within a relational database; this is inclusive of logical constraints such as, table names, fields, datatypes, and the relationships between these entities. Schemas commonly use visual representations to communicate the architecture of the database, becoming the foundation for an organization's data management discipline. This process of database schema design is also known as data modeling.



8. TESTING

8.1 TEST CASES

The test case is defined as a group of conditions under which a tester determines whether software application is working as per the customer's requirements or not. Test case designing includes preconditions, case name, input conditions, and expected result. A test case is a first level action and derived from test scenarios.

Test case gives detailed information about testing strategy, testing process, preconditions, and expected output. These are executed during the testing process to check whether the software application is performing the task for that it was developed or not.

Test case helps the tester in defect reporting by linking defect with test case ID. Detailed test case documentation works as a full proof guard for the testing team because if developer missed something, then it can be caught during execution of these full-proof test cases.

To write the test case, we must have the requirements to derive the inputs, and the test scenarios must be written so that we do not miss out on any features for testing. Then we should have the test case template to maintain the uniformity, or every test engineer follows the same approach to prepare the test document.

SPRINT-1 TEST CASES

Test Case ID	Test case description	Test Steps	Test Data	Expected Results	Actual Result	Pass/Fail
1.	Customer registration with invalid data	1. Go to application 2. Enter first name, last name, select the role, password and confirm password 3. Click Register	First Name = Surya Last Name = Ranjith Role = Customer Email = itsurya2002@gmail.com Password = 12345678 Confirm Password = 123456789	Customer should get an alert saying "Passwords do not match"	As expected	Pass
2.	Customer registration with invalid data	1. Go to application 2. Enter first name, last name, select the role, password and confirm password 3. Click Register	First Name = Surya Last Name = Ranjith Role = Customer Email = itsurya2002@gmail.com Password = 12345678 Confirm Password = 12345678	Customer should get an alert saying "Invalid email"	As expected	Pass

3.	Customer registration with invalid data	1. Go to application 2. Enter first name, last name, select the role, password and confirm password 3. Click Register	First Name = Surya Last Name = Ranjith Role = Customer Email = itsurya2002@gmail.com Password = 12345678 Confirm Password = 12345678	Customer should get an alert saying "First name should be at least 3 characters long!"	As expected	Pass
4.	Customer registration with invalid data	1. Go to application 2. Enter first name, last name, select the role, password and confirm password 3. Click Register	First Name = Surya Last Name = Ranjith Role = Customer Email = itsurya2002@gmail.com Password = 1234 Confirm Password = 1234	Customer should get an alert saying "Passwords must be at least 8 characters long!"	As expected	Pass
5.	Customer registration with valid data	1. Go to application 2. Enter first name, last name, select the role, password and confirm password 3. Click Register	First Name = Surya Last Name = Ranjith Role = Customer Email = itsurya2002@gmail.com Password = 12345678 Confirm Password = 12345678	Customer should be able to register to the application	As expected	Pass
6.	Customer login using the invalid data	1. Go to application 2. Enter email, password 3. Click Login	Email = itsurya2002@gmail.com Password = 12345678	Customer should get an alert saying "Invalid email"	As expected	Pass
7.	Customer login using the invalid data	1. Go to application 2. Enter email, password 3. Click Login	Email = itsurya2002@gmail.com Password = 12345678	Customer should get an alert saying "User does not exist"	As expected	Pass
8.	Customer login using invalid data	1. Go to the application 2. Enter email, password 3. Click Login	Email = itsurya2002@gmail.com Password = 12345678999	Customer should get an alert saying "Wrong Password!"	As expected	Pass

9.	Customer Login using Valid data	1. Go to application 2. Enter email, password 3. Click Login	Email = itsurya2002@gmail.com Password = 12345678	Customer should login to the application	As expected	Pass
10.	Admin login using invalid data	1. Go to application 2. Enter email, password 3. Click Login	Email = admin.ccr@gmail.com Password = admin.ccr	Admin should get an alert saying 'Invalid Password!'	As expected	Pass
11.	Admin login using invalid data	1. Go to application 2. Enter email, password 3. Click Login	Email = admin@gmail.com Password = admin.ccr@2022	Admin should get an alert saying 'Invalid Email!'	As expected	Pass
12.	Admin login using valid data	1. Go to application 2. Enter email, password 3. Click Login	Email = admin.ccr@gmail.com Password = admin.ccr@2022	Admin should login to the application	As expected	Pass
13.	Customer logging out	1. Go to the Nav Bar 2. Click on the right-side circular image 3. Click Logout	-	Customer should be able to log out from the application	As expected	Pass
14.	Customer logging out	1. Go to the Nav Bar 2. Click on the right-side circular image 3. Click Logout	-	Admin should be able to log out from the application	As expected	Pass

SPRINT-2 TEST CASES

Test Case ID	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Pass / Fail
15.	Customer creating a new ticket with empty query	<ol style="list-style-type: none"> Go to site Customer login using email and password Click "New Ticket" option in the Dashboard Clicking the "New Ticket" button without typing any query in the given text area 	Query = NULL	Customer should get an alert saying "Query cannot be empty!"	As expected	Pass
16.	Customer creating a new ticket with a valid query	<ol style="list-style-type: none"> Go to site Customer login using email and password Click "New Ticket" option in the Dashboard Typing the query in the given text area Clicking the "New Ticket" button 	Query = "Hi. My I Phone 14 pro max is not turning on. It is a new unit I bought it just 2 days back. I don't know what happened. Can you help me please?"	The ticket gets inserted in the database. After that customer gets an alert saying 'Ticket created'	As expected	Pass

17.	Customer seeing all the tickets raised by him/her	<ol style="list-style-type: none"> Go to site Customer login using email and password Click "Tickets" option in the Dashboard 	Tickets created by the customer which are already being inserted in the database	Customer should see the list of all the tickets raised by him/her	As expected	Pass
18.	Customer seeing all the tickets raised by him/her	<ol style="list-style-type: none"> Go to site Customer login using email and password Click "Tickets" option in the Dashboard 	-	Customer should see a message "You are yet to raise a ticket"	As expected	Pass
19.	Customer seeing the query of a ticket	<ol style="list-style-type: none"> Go to site Customer login using email and password Click "Tickets" option in the Dashboard Click "View" option in a ticket from the list of tickets 	Tickets created by the customer which are already being inserted in the database	An alert should be shown having the actual query posted by the customer	As expected	Pass
20.	Customer seeing the assigned agent for a ticket	<ol style="list-style-type: none"> Go to site Customer login using email and password Click "Tickets" option in the Dashboard 	<ul style="list-style-type: none"> Tickets created by the customer which are already being inserted in the database Admin assigned the agent for the ticket 	Customer should be able to see the first name of the agent assigned	As expected	Pass
21.	Customer seeing the assigned agent for a ticket	<ol style="list-style-type: none"> Go to site Customer login using email and password Click "Tickets" option in the Dashboard 	<ul style="list-style-type: none"> Tickets created by the customer which are already being inserted in the database Admin is yet to assign the agent 	Customer should be able to see the "N/A" message displayed	As expected	Pass

22.	Admin seeing all the unassigned tickets	<ol style="list-style-type: none"> Go to site Admin login using email and password Click "Tickets" option in the Dashboard 	<ul style="list-style-type: none"> Tickets created by the customers which are already being inserted in the database Admin did not assign agent for the tickets 	Showing the tickets that are yet to be assigned an agent by the admin	As expected	Pass
23.	Admin seeing all the unassigned tickets	<ol style="list-style-type: none"> Go to site Admin login using email and password Click "Tickets" option in the Dashboard 	<ul style="list-style-type: none"> Tickets created by the customers which are already being inserted in the database Admin assigned agents for all the tickets 	Admin should just see the message "There is nothing left to assign"	As expected	Pass
24.	Admin assigning an agent for a ticket	<ol style="list-style-type: none"> Go to site Admin login using email and password Click "Tickets" option in the Dashboard Select an agent from the dropdown given 	<ul style="list-style-type: none"> Tickets created by the customers which are already being inserted in the database Admin did not assign the agent yet 	Admin should get an alert saying "Do you really want to assign the agent for this ticket?". If admin clicks OK, then the agent is assigned for the ticket. The list gets updated	As expected	Pass
25.	Admin seeing the requests section	<ol style="list-style-type: none"> Go to site Admin login using email and password Click "Requests" option in the Dashboard 	<ul style="list-style-type: none"> Agent details in the database Admin is yet to accept the agent 	Admin should be able to see the list of all the requests made by the agents to the admin	As expected	Pass

26.	Admin seeing the requests section	1. Go to site 2. Admin login using email and password 3. Click "Requests" option in the Dashboard	<ul style="list-style-type: none"> Agent details in the database Admin accepted all the agents 	Admin should just see the message "There are no pending requests"	As expected	Pass
27.	Admin accepting an agent from the request section	1. Go to site 2. Admin login using email and password 3. Click "Requests" option in the Dashboard 4. Click "Tick" mark that is against the agent details	<ul style="list-style-type: none"> Agent details in the database Admin is yet to accept the agent 	The agent gets accepted and the same is updated in the database. The list gets updated	As expected	Pass
28.	Agent registration using invalid data	1. Go to site 2. Click on "Don't have an account yet? Register" option 3. Fill the form	First Name = Agent 1 Last Name = NULL Email = agent1@gmail.com Password = 12345678 Confirm password = 12345678	Agent should get an alert saying "Last Name must be at least 1 character long!"	As expected	Pass
29.	Agent registration using invalid data	1. Go to site 2. Click on "Don't have an account yet? Register" option 3. Fill the form	First Name = Agent 1 Last Name = Agent Email = agent1@gmail.com Password = 12345678 Confirm password = 12345678	Agent should get an alert saying "invalid Email"	As expected	Pass
30.	Agent registration using invalid data	1. Go to site 2. Click on "Don't have an account yet? Register" option 3. Fill the form	First Name = Agent 1 Last Name = Agent Email = agent1@gmail.com Password = 123456789 Confirm password = 12345678	Agent should get an alert saying "Passwords do not match!"	As expected	Pass

SPRINT-3 TEST CASES

Test Case ID	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Pass / Fail
37.	Customer changing the existing password using invalid data	1. Go to site 2. Login as a customer using valid credentials 3. Click "Change Password" in the dashboard	Password = 123456 New Password = 123456789 Confirm Password = 123456789	Customer should get an alert saying "Passwords must be at least 8 characters long!"	As expected	Pass
38.	Customer changing the existing password using invalid data	1. Go to site 2. Login as a customer using valid credentials 3. Click "Change Password" in the dashboard	Password = 12345678 New Password = 123456789 Confirm Password = 123456780	Customer should get an alert saying "Passwords do not match!"	As expected	Pass

39.	Customer changing the existing password using invalid data	1. Go to site 2. Login as a customer using valid credentials 3. Click "Change Password" in the dashboard	Password = 12345678 New Password = 12345678 Confirm Password = 12345678	Customer should get an alert saying "Old and New password cannot be the same!"	As expected	Pass
40.	Customer changing the existing password	1. Go to site 2. Login as a customer using valid credentials 3. Click "Change Password" in the dashboard	Password = 12345678 New Password = 123456789 Confirm Password = 123456789	Customer should get an alert saying "Password changed! Please Login". The customer is then redirected to the login page for logging in	As expected	Pass
41.	Customer opening the address column	1. Go to site 2. Login as a customer using valid credentials 3. Click "Tickets" in the dashboard 4. Click "Chat/Visit" in the address column of a ticket	Tickets in the database	Customer should be able to get into the address column, where the latter can chat with the agent	As expected	Pass
42.	Customer opening the address column	1. Go to site 2. Login as a customer using valid credentials 3. Click "Tickets" in the dashboard 4. Click "Chat" in the address column of a ticket	<ul style="list-style-type: none"> Ticket in the database Ticket is still OPEN Still, no messages with the agent Agent first name = 'Agent 1' 	Customer should see an alert saying "Start the conversation with the Agent 1"	As expected	Pass

SPRINT-4 TEST CASES

Test Case ID	Test Case Description	Test Steps	Test Data	Expected Results	Actual Results	Pass / Fail
61.	Customer forgot the password and trying to update the password with invalid email	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button	Email = itsurya2002@gmail.com Role = "Customer"	Customer should get an alert saying "Invalid email!"	As expected	Pass
62.	Customer forgot the password and trying to update the password with invalid email	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button	Email = itsurya2002@gmail.com Role = "Customer"	Customer should get an alert saying "Customer does not exist"	As expected	Pass

67.	Customer entering the new passwords in the change password page	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button 5. Enter the valid code received in the email 6. Click "Submit" button 7. Enter the passwords	Email = itsurya2002@gmail.com Role = "Customer" Code = "87436601" Password = 12345678 Confirm password = 12345678	Customer's password gets updated. Then the customer is redirected to the login page to login	As expected	Pass
68.	Agent forgot the password and trying to update the password with invalid email	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button	Email = agent1@gmail.com Role = "Agent"	Agent should get an alert saying "Invalid email!"	As expected	Pass
69.	Agent forgot the password and trying to update the password with invalid email	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button	Email = agent44@gmail.com Role = "Agent"	Agent should get an alert saying "Agent does not exist"	As expected	Pass
70.	Agent forgot the password and trying to update the password with valid email	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button	Email = agent1@gmail.com Role = "Agent"	Agent should receive an 8-digit code in the email and redirected to the code entering page	As expected	Pass

71.	Agent entering invalid code to change the password	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button 5. Enter invalid code 6. Click "Submit" button	Email = agent1@gmail.com Role = "Agent" Code = "bhuudbsgygdy2"	Agent should get an alert saying "Invalid code!"	As expected	Pass
72.	Agent entering valid code to change the password	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button 5. Enter the valid code received in the email 6. Click "Submit" button	Email = agent1@gmail.com Role = "Agent" Code = "87436601"	Agent should be redirected to the passwords entering page	As expected	Pass
73.	Agent entering the invalid passwords in the change password page	1. Go to the site 2. Click "Forgot Password?" option in the Login form 3. Enter the email 4. Click "Get Code" button 5. Enter the valid code received in the email 6. Click "Submit" button 7. Enter the passwords	Email = agent1@gmail.com Role = "Agent" Code = "87436601" Password = 12345678 Confirm password = 87654321	Agent should get an alert saying "Passwords do not match!"	As expected	Pass

8.2 USER ACCEPTANCE TESTING

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the **Customer Care Registry** project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	5	0	0	2	7
External	0	2	0	0	2
Fixed	12	11	35	45	103
Not Reproduced	0	5	0	0	5
Skipped	0	0	0	0	0
Totals	17	18	35	47	117

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

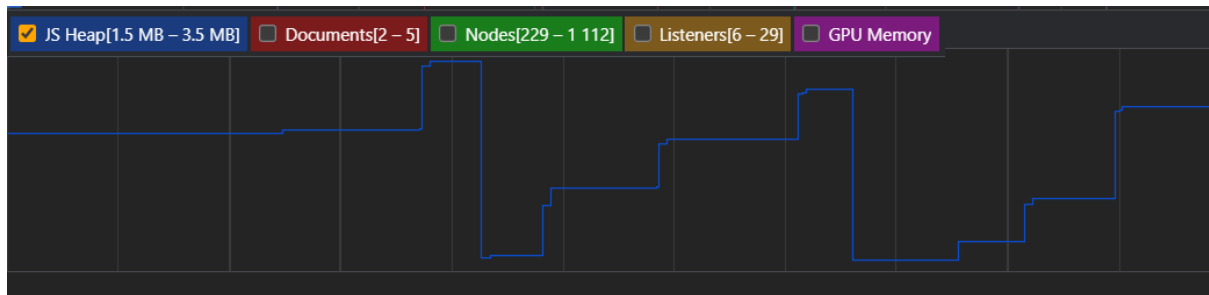
Section	Total Cases	Not Tested	Fail	Pass
Client Application	72	0	0	72
Security	7	0	0	7
Exception Reporting	5	0	0	5
Final Report Output	4	0	0	4

9. RESULTS

9.1 PERFORMANCE METRICS

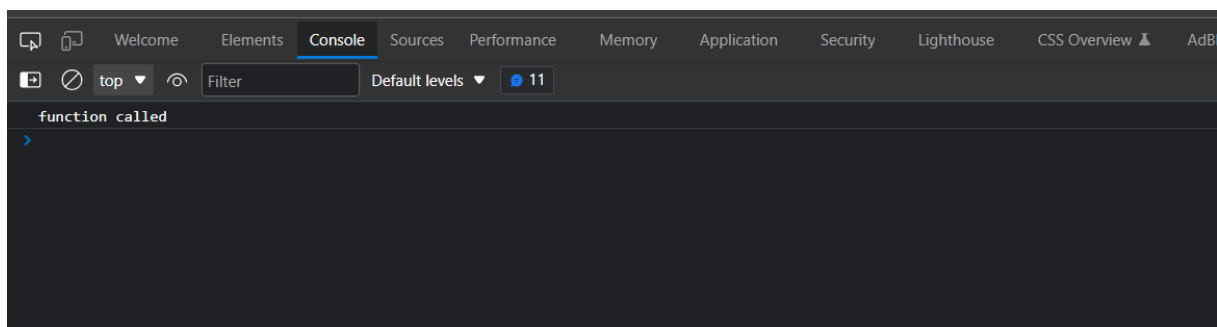
CPU usage:

- ❖ Since all the operations run using Flask is in server-side, the client (browser) need not worry about the CPU usage. Just rendering the page, static contents take place in the client-side.
- ❖ Memory for client-side functions (JavaScript) is allocated using heap. It can be either increased based upon the requirement or removed from the heap.



Errors:

- ❖ Since all the backend functions are done using flask, any exceptions / errors rising are well-handled. Though they appear, user's interaction with the site is not affected in any way



Latency and Response time:

It takes less than a second to load a page in the client. From this it is evident that there is low latency

11 requests 238 kB transferred 285 kB resources Finish: 892 ms DOMContentLoaded: 810 ms Load: 905 ms

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- ❖ Customers are provided with a unique account, to which the latter can login at any time
- ❖ Customers can clarify their doubts just by creating a new ticket
- ❖ Application is very simple to use, with well-known UI elements
- ❖ Customers are given clear notifications through email, of all the processes related login, ticket creation etc.,
- ❖ Free of cost
- ❖ Customer's feedbacks are always listened
- ❖ Not only the replies are faster but also the replies are more authentic and practical
- ❖ Very minimal account creation process
- ❖ Customers can raise as many tickets as they want
- ❖ Very minimal account creation process
- ❖ Customers can clarify their doubts just by creating a new ticket
- ❖ Customers can clarify their doubts just by creating a new ticket
- ❖ Customer gets replies as soon as possible

DISADVANTAGES

- ❖ No login alerts
- ❖ Account cannot be deleted, once created
- ❖ Cannot update the mobile number
- ❖ Customers cannot give feedback to the agent for clarifying the queries
- ❖ No SMS alerts
- ❖ Supports only text messages while chatting with the Agent
- ❖ No tap to reply feature
- ❖ No automated replies
- ❖ Only web application is available right now (as of writing)
- ❖ UI is not so attractive, it's just simple looking

11. CONCLUSION

It is a web-enabled project. With this project the details about the product will be given to the customers in detail within a short span of time. Queries regarding the product or the services will also be clarified.

Thus, there are many customer service applications available on the internet. Noting down the structural components of those applications and we built a customer care registry application.

It will be a web application build with Flask (Python micro-web framework), HTML, and JavaScript. It will be a ticket-based customer service registry.

Customers can register into the application using their email, password, first name and last name. Then, they can login to the system, and raise as tickets 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.

12. FUTURE SCOPE

- ❖ Incorporating automatic replies in the chat columns
- ❖ Releasing cross-platform mobile applications
- ❖ Attracting and much more responsive UI throughout the application
- ❖ Call support
- ❖ Instant SMS alerts
- ❖ Deleting the account whenever customer wishes to
- ❖ Supporting multi-media in the chat columns

13. APPENDIX

SOURCE CODE

base.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>{% block title %}{% endblock %}</title>

    <link rel="icon" type="image" href="{% url_for('static', filename='images/cart logo white-
modified.png') %}">

    <!-- Linking css, js, Google fonts -->

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link rel="stylesheet" href="{% url_for('static', filename='css/style.css') %}" />

    <link
href="https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100;0,300;0,400;0,500;
0,700;0,900;1,100;1,300;1,400
;1,500;1,700;1,900&display=swap" rel="stylesheet">

    <script src="{% url_for('static', filename='js/pass.js') %}"></script>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
awesome.min.css">

    <!-- Linking Watson Assistant -->

    {% block watson %}

    {% endblock %}

</head>
```

```

<body>

    {% block alert %}

        {% if to_show %}

<script>

            alert('{{ message }}')

</script>

{% endif %}

    {% endblock %}

    {% block main %}

    {% endblock %}

</body>

</html>

```

Login.html

```

{% extends 'base.html' %}

{%block  title
%}Login

{% endblock %}

{% block main %}

<div class="bg-main-div">

<section class="login-
section">

    <div class="login-div">

<div    class="login-
header">

    <h2>Sign in</h2>

```

```
<p>Use your Registry Account</p>

</div>

<div class="login-remind">

    <form action="{{ url_for('blue_print.login') }}" method="POST" class="login-form">

<label>Email</label>

        <input type="email" required value="{{ email }}" name="email" placeholder="Enter your
        email"/>

        <label>Password</label>

        <input type="password" required value="{{ password }}"
name="password" id="password-input" placeholder="Enter your password"/>

        <div class="show-pass-div">

            <input type="checkbox" onclick="showPassword()" style="height: 20px;"/>

            <p>Show Password</p>

        </div>

        <div class="role-div">

            <p>Role : </p>

            <div>

                <div>

                    <input type="radio" style="height: 20px;" value="Customer" checked name="role-
                    check"/>

                    <p>Customer</p>

                </div>

                <div>

                    <input type="radio" style="height: 20px;" value="Agent" name="role-check"/>

                    <p>Agent</p>

                </div>

            </div>

        </div>

    </div>
```


</div>

<button class="submit-btn" type="submit">Login</button>

<div>

<!-- {{ url_for("blue_print.forgot") }} -->

Forgot Password?

<div>

Don't have an account yet?
Register

</div>

</div>

</form>

</div>

</div>

</section>

</div>

{% endblock %}

address.html

```
{% extends 'base.html' %}

{% block title %} Address Column

{% endblock %}

{% block main %}

    <div class="dashboard-div">

        <nav>

<div class="dash-nav">

<div>

    <div class="dash-img-text">

        {% if user == "AGENT" %}

            <a href="{{ url_for('agent.assigned') }}">

                <i class="fa fa-arrow-left" aria-hidden="true"></i>

            </a>

        {% else %}

            <a href="{{ url_for('customer.tickets') }}">

                <i class="fa fa-arrow-left" aria-hidden="true"></i>

            </a>

        {% endif %}

        <h3>{{ name }}</h3>

    </div>

</div>

</div>

<div>
```

```

<div style="align-items: center;">

    {% if value == "True" %}

        {% if user == "CUSTOMER" %}

<a href="/customer/close/{{ id }}"><button class="logout-btn">CLOSE TICKET</button></a>

        {% endif %}

    {% endif %}

</div>

</div>

</div>

</nav>

<div class="chat-body">

    <div class="chat-contents" id="content">

        {% if msgs_to_show %}

            {% for chat in chats %}

                {% if chat['SENDER_ID'] == sender_id %}

                    <div class="message-sent">{{ chat['MESSAGE'] }}</div>

                {% else %}

                    <div class="message-sent received">{{ chat['MESSAGE'] }}</div>

                {% endif %}

            {% endfor %}

        {% endif %}

    </div>

    <div class="chat-input-div">

        {% if value == "True" %}

            <form method="POST" action="{{ post_url }}">

<input name="message-box" class="chat-input" type="text" placeholder="Type something" required/>

```

```

        <button type="submit" class="chat-send">

        <i class="fa fa-paper-plane-o" aria-hidden="true"></i>

    </button>

</form>

{% else %}

<div>

    {% if user == "CUSTOMER" %}

        <h4>You closed this ticket. Chats are disabled</h4>

    {% else %}

        <h4>{{ name }} closed this ticket. Chats are disabled</h4>

    {% endif %}

</div>

{% endif %}

</div>

{% endblock %}

```

chat.py

```

from flask import render_template, Blueprint, request, session,
redirect, url_for import ibm_db

from datetime import
date timeimport time

chat = Blueprint("chat_bp", __name__)

@chat.route('/chat/<ticket_id>/<receiver_name>', methods =
['GET', 'POST'])def address(ticket_id, receiver_name):

'''

```

Address Column - Agent and Customer chats with one another

: param ticket_id ID of the ticket for which the chat is being opened

: param receiver_name Name of the one who receives the texts, may be Agent/ Customer

```
""

sessionuser = ""

sender_id = ""

value = ""

can_trust = False

post_url = f'/chat/{ticket_id}/{receiver_name}/'

if session['LOGGED_IN_AS'] is not None:

    if session['LOGGED_IN_AS'] == "CUSTOMER":

        from .views import customer

        if(hasattr(customer, 'uuid')):

            user = "CUSTOMER"

            sender_id = customer.uuid

            can_trust = True

        else:

            redirect(url_for('blue_print.logout'))

    elif session['LOGGED_IN_AS'] == "AGENT":

        if (hasattr(agent, 'uuid')):

            user = "AGENT"

            sender_id = agent.uuid

            can_trust = True

        else:

            redirect(url_for('blue_print.logout'))

to_show = False

message = ""
```

```

if can_trust:

    from .views import conn

    if request.method == 'POST':

        myMessage = request.form.get('message-box')

        if len(myMessage)
            == 0: to_show =
                True

        message = "Type something!"

    else:

        message_insert_query = ""

        INSERT INTO chat

            (chat_id, sender_id, message, sent_at) VALUES

            (?, ?, ?, ?)

        ""

    try:

        stmt=ibm_db.prepare(conn,
            message_insert_query)
        ibm_db.bind_param(stmt, 1, ticket_id)

        ibm_db.bind_param(stmt, 2, sender_id)

        ibm_db.bind_param(stmt, 3, myMessage)

        ibm_db.bind_param(stmt, 4, datetime.now())

        ibm_db.execute(stmt)

    except:

        to_show = True

        message = "Please send again!"

    return redirect(post_url)

else:

```

```
msgs_to_show = False
```

```
get_messages_query = ""
```

```
SELECT * FROM chat
WHERE chat_id = ?
```

```
ORDER BY sent_at ASC
```

```
""
```

```
query_status_check = ""
```

```
SELECT query_status FROM tickets WHERE ticket_id = ?
```

```
""
```

```
try:
```

```
check = ibm_db.prepare(conn, query_status_check)
```

```
ibm_db.bind_param(check,1,ticket_id)
```

```
ibm_db.execute(check)
```

```
value = "True" if ibm_db.fetch_assoc(check)
```

```
['QUERY_STATUS'] == "OPEN" else "False"
```

```
ibm_db.prepare(conn,get_messages_query)
```

```
ibm_db.bind_param(stmt, 1, ticket_id) ibm_db.execute(stmt)
```

```
messages = ibm_db.fetch_assoc(stmt)
```

```
messages_list = []
```

```
while messages != False:
```

```
    messages_list.append(messages)
```

```
    print(messages)
```

```
    messages = ibm_db.fetch_assoc(stmt)
```

```
len(messages_list) > 0:
```

```
    msgs_to_show = True
```

```
elif len(messages_list) == 0 and value == "True":
```

```
    msgs_to_show = False
```

```
to_show = True
```

```
message = f'Start the conversation with the {"Customer" if user == "AGENT" else "Agent"}'
```

```

except:

    to_show = True

    message = "Something happened! Try Again"

return render_template(

'address.html', to_show = to_show, message = message, id = ticket_id,

    chats=messages_list,
    msgs_to_show=msgs_to_show,
    sender_id = sender_id,

    name= receiver_name,

    user = user,

    post_url=post_url,

    value = value

)

else:

    return redirect(url_for('blue_print.logout'), user = user)

```

init.py

```

from flask import Flask, session

from flask_login import LoginManager

def create_app():

    app = Flask(__name__)

    app.config['SECRET_KEY'] = "PHqtYfAN2v@CCR2022"

    # registering the blue prints with the app from .routes.views
    import views app.register_blueprint(views, appendix='/')

    from .routes.cust import cust app.register_blueprint(cust, appendix='/customer/')

    from .routes.admin import admin app.register_blueprint(admin, appendix='/admin/')

    from .routes.agent import agent app.register_blueprint(agent, appendix='/agent/')

```



```
from .routes.chat import chat app.register_blueprint(chat, appendix='/chat/')
```

```
# setting up the login manager login_manager =  
LoginManager() login_manager.login_view =  
"blue_print.login" login_manager.init_app(app)
```

```
@login_manager.user_loader
```

```
def load_user(id):
```

```
    if session.get('LOGGED_IN_AS') is not None:
```

```
        if session['LOGGED_IN_AS'] == "CUSTOMER":
```

```
            from .routes.views import customer
```

```
            if hasattr(customer, 'first_name'):
```

```
                return customer
```

```
        elif session['LOGGED_IN_AS'] == "AGENT":
```

```
            from .routes.views import agent
```

```
            if hasattr(agent, 'first_name'):
```

```
                return agent
```

```
        elif session['LOGGED_IN_AS'] == "ADMIN":
```

```
            from .routes.views import admin
```

```
            if hasattr(admin, 'email'):
```


```
                return admin
```


```
    else:
```

```
        return None
```

```
    return app
```

OUTPUT

Customer Care Registry



Profile

Tickets Assigned

Change Password


About


Feedback

Welcome to CCR!

Profile	
First Name	Agent 1
Last Name	Agent
Role	Agent
Email	agent1@gmail.com
Date joined	2022-11-04

Agent Dashboard

Customer Care Registry



Profile

Tickets Assigned

Change Password

About

Feedback

Change Password

Feeling your old password is not good enough?

Password

Current Password

New Password

New Password

Confirm Password

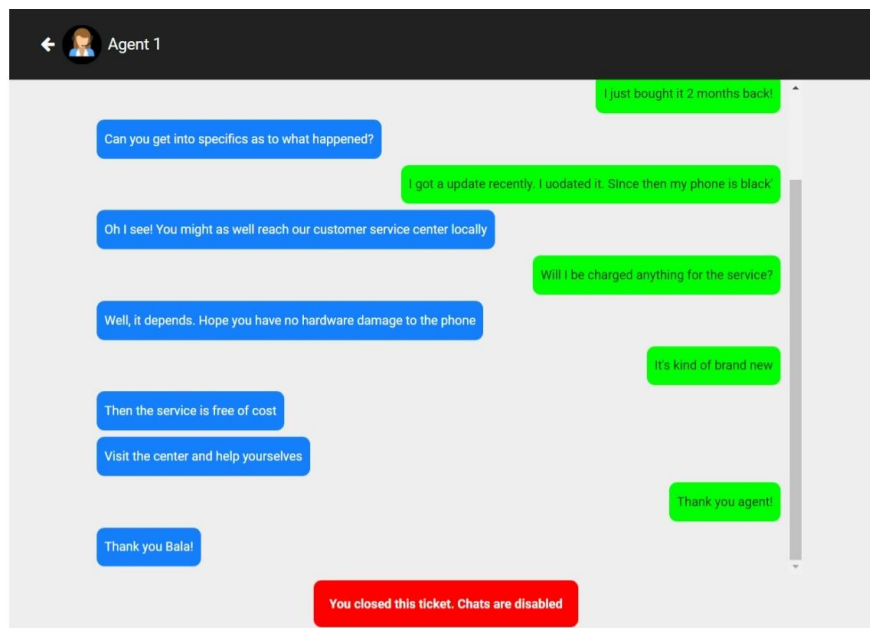
New Password

Use 8 or more characters with a mix of just letters and numbers

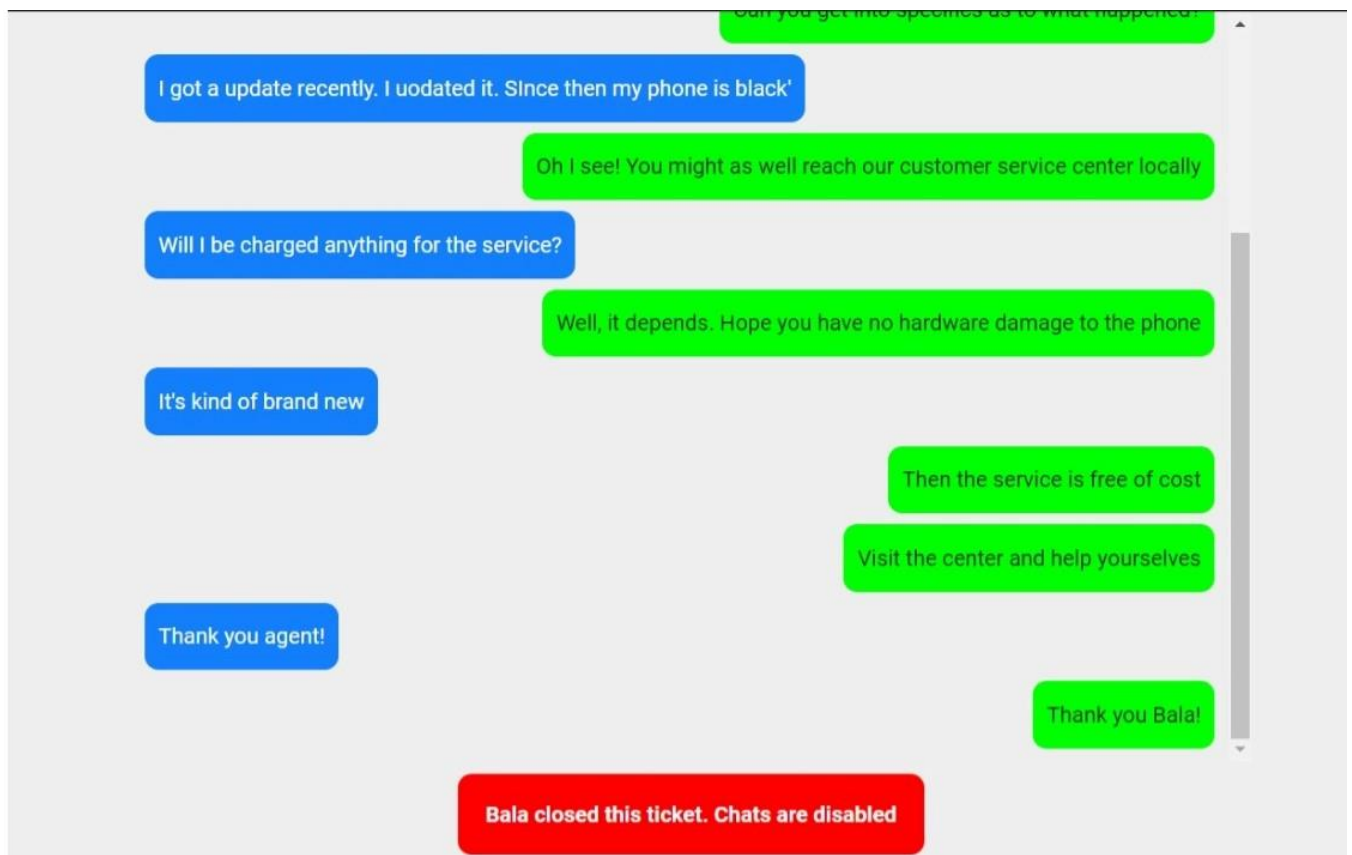
☐ Show Password

Submit

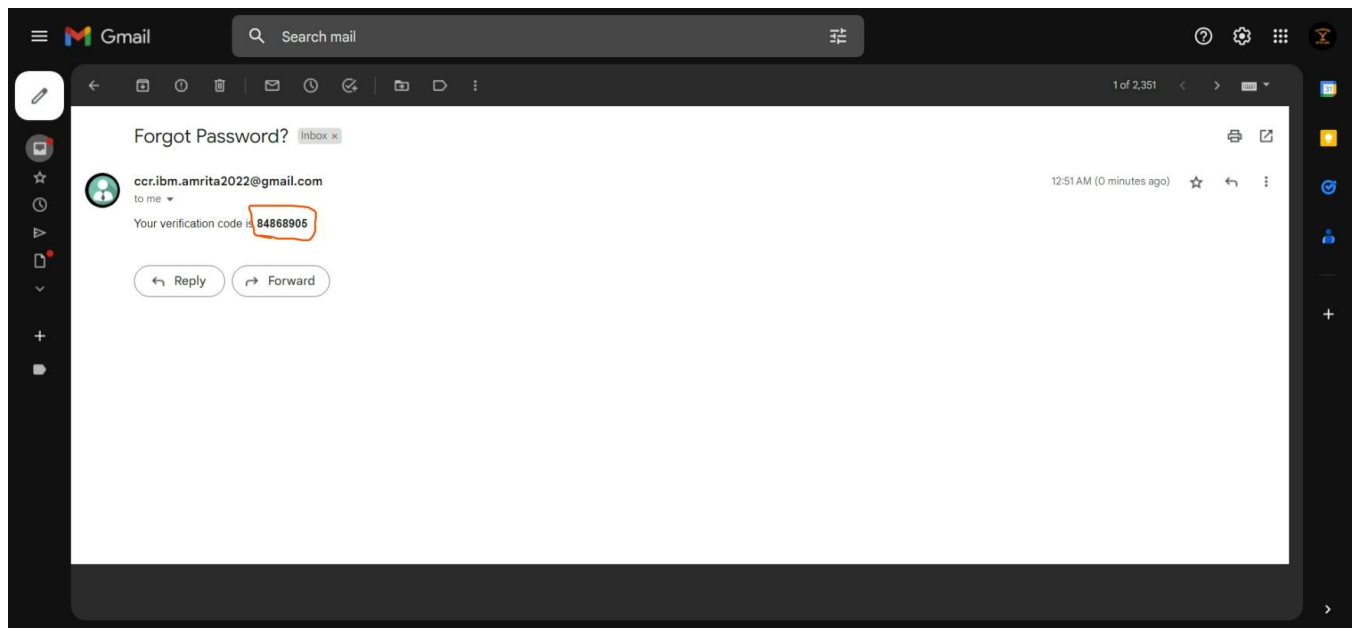
Change Password





Chatting with Agent





Chatting with Customer





Code Sent


 Customer Care Registry 

 Tickets

 Agents

 Requests

 About


 Feedback

All Agents

List of confirmed agents

AGENT ID	JOINED DATE	FIRST NAME	LAST NAME	EMAIL
3a7de	2022-11-04	Agent 1	Agent	agent1@gmail.com

Confirmed Agents



Forgot Password?

Use your Registry Account

Email


Role : ☒ Customer ☐ Agent

[Signin instead](#)

[Don't have an account yet? Register](#)

[Get Code](#)

Forgot Password




Welcome agent, Agent 1


It seems your account is not yet verified by the admin! So would you please login after a while.


LOGOUT


Not Confirmed





Customer Care Registry




 Tickets

 Agents

 Requests

 About


 Feedback

Pending Requests


These are the pending requests


AGENT ID	EMAIL	FIRST NAME	JOINED DATE	ACCEPT?
6fc82	agent2@gmail.com	Agent 2	2022-11-04	<input checked="" type="checkbox"/> <input type="checkbox"/>


Pending Requests





Customer Care Registry





 Profile

 New ticket

 Tickets

 Change Password

 About

 Feedback

Raise a new ticket

In case you have any issues, you can raise a new ticket with the detailed description of the issue. An agent will be assigned for your ticket. And you will receive expert answers

Your Query...

* Please make sure the query is legible

New Ticket

Raise a New Ticket

Create your Registry Account

First Name

Last Name

Email

Your email address

This will be your Registry account email

Role : ☒ Customer ☐ Agent

Password

Confirm Password


Password

Confirm password

Use 8 or more characters with a mix of just letters and numbers



☐ Show Password






[Sign-in instead](#)



Politeness goes far, yet costs nothing

Registration Page


Customer Care Registry




 Profile
  Tickets Assigned
  Change Password
  About
  Feedback







Tickets assigned to you!

Admin assigned these tickets to you

TICKET ID	DATE	RAISED BY	STATUS	QUERY	ADDRESS COLUMN
077e6	2022-11-09		CLOSED	View	Visit
c097e	2022-11-09		OPEN	View	Chat

Tickets Assigned


Customer Care Registry


 Profile
  New ticket
  Tickets
  Change Password
  About
  Feedback

Tickets raised by you!

These are your tickets

TICKET ID	DATE	STATUS	QUERY	AGENT	ADDRESS COLUMN
c097e	2022-11-09	OPEN	View	Agent 1	Chat
077e6	2022-11-09	CLOSED	View	Agent 1	Visit

Tickets Raised



Tickets

Agents

Requests

About

Feedback

Unassigned Tickets

These are the unassigned tickets

TICKET ID	DATE	CUSTOMER	QUERY	ASSIGN
03945	2022-11-04	Bala	View	<input type="text" value="Choose"/>
89d49	2022-11-04	Bala	View	<input type="text" value="Choose"/>
b7474	2022-11-04	Bala	View	<input type="text" value="Choose"/>
f0ded	2022-11-04	Bala	View	<input type="text" value="Choose"/>

Unassigned Tasks

Git Hub & Project Demo Link

<https://github.com/IBM-EPBL/IBM-Project-46533-1660749331>