TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
1	**INTRODUCTION**	**1**
	1.1 Project Overview	**1**
	1.2 Purpose	**1**
2	**LITERATURE SURVEY**	**2**
	2.1 Existing problem	**2**
	2.2 References	**2**
3	**IDEATION & PROPOSED SOLUTION**	**5**
	3.1 Empathy Map Canvas	**5**
	3.2 Ideation & Brainstorming	**6**
	3.3 Proposed Solution	**10**
	3.4 Problem Solution fit	**10**
4	**REQUIREMENTS ANALYSIS**	**12**
	4.1 Functional requirements	**12**

4.2 Non Functional Requirements

5 **PROJECT DESIGN** **13**

5.1 Data Flow Diagrams **13**

5.2 Solution & Technical Architecture **14**

5.3 User Stories **15**

6 **PROJECT PLANNING &** **17**

SCHEDULING

18

6.1 Sprint Planning & Estimation

18

6.2 Sprint Delivery schedule

19

6.3 Reports from JIRA

7 **CODING & SOLUTIONING** **22**

7.1 Feature 1 **22**

7.2 Feature 2 **23**

8 **TESTING** **24**

8.1 Test Cases **24**

8.2 User Acceptance Testing **24**

RESULTS **27**

9 9.1 Performance Metrics **27**

10	**ADVANTAGES & DISADVANTAGES**	**29**
11	**CONCLUSION**	**30**
12	**FUTURE SCOPE**	**31**
13	**APPENDIX**	**32**
	13.1 Source Code	**32**
	13.2 Screenshots	**40**

CHAPTER - 01

1. INTRODUCTION

1.1 Project Overview

Our project Customer Care Registry is a cloud-based web application that is established to provide services for the user's queries. If a user hangs up with a query, then they can raise that specific query in our application. Our application is categorized into three segments. One is the customer side(i.e., the user). The second part is the agent side(i.e., the person who solves the query of the customer). Finally, the third part is the admin side(i.e., the person who handles both the agent and the customer).

On the customer side, the user can be able to have an account on the Customer Care Registry. Using their account, they can log in to their dashboard. There, the user can able to see several sections like FAQ, Knowledge base center, and finally, they can be able to raise a query that is named a Ticket. Using the ticketing system, the user will raise a query to us. Then, here comes the agent. The agent is assigned by the admin. Here, the agent solves the query of the user and the agent communicates with the user through the Omni channels. Three Omni channels are available. They are a Knowledge base, live agent(i.e., chatbot), and Email. The agent communicates with the user with the help of a live agent and an Email. With the help of the live chatbot and Email channels, the agent solves the query of the user. Here, the role of the admin is to assign an agent. The admin can be from any organization. The admin manages all the processes for both the agent and the user.

1.2 Purpose

The brief description of the Omni channels is as follows. In the Knowledge base center, both the user and the agent can able to engage themselves. This Omni channel is like a DIY(Do It Yourself) mechanism. The user and the agent improve their technical skills with the help of the Knowledge base center. The knowledge base contains the customer care registry guide, and solutions to the problems like login issues, sign-up issues, etc.

In the Chatbot Omni channel, the user receives the response in two ways. In the first type, the user will receive the responses in a pre-defined manner. The second type is, the user will receive the responses from the agent in live. Both types are faster and more user-friendly. If the user's query is not solved using the predefined chatbot, then they can use the live chatbot where the user receives the responses based on the query so accurately

CHAPTER - 02

2. LITERATURE SURVEY

2.1 Existing Problem

The existing problems in the Customer Care Application are as follows. There may be many problems that are existing in the application, but we have found three of them. The first existing problem in the Customer Care Application is the Time Delay. Here, the Time Delay refers to the delayed responses from either the Customer side or the Agent side, or even from the Admin side. When the response time is increased, it causes a problem on either side.

Coming to the second problem, the status of the ticket is unavailable. Briefly, in the existing application, the Customer can't able to view the status of the ticket until the end of the service provided. When the Customer quits the chat before the solution is provided, this is considered a problem for the service provider. This problem is defined as a Customer unavailable in the existing application.

2.2 References

1. Knowledge-Based Helpdesk System

Author - Mohamad Safuan Bin Sulaiman, Khairiel Adyani B. Abd. Ghani, Abd. Aziz B. Mhd. Ramli, Mohd. Ashhar B. Hj. Khalid - 2012

A knowledge-Based helpdesk system is a web-based system used to provide technical support to an organization or management. Then, it acts as a Service Provider to that particular organization. The main objective of this Knowledge-based system is to provide technical support to the end users of a particular organization. Using this Knowledge-based Helpdesk system, an organization can improve their end user's performance and make their end users technically well-educated. Once the Knowledge-based helpdesk system is designed, it is tested on the Information Technology (IT) center, Engineering Division (BKJ), etc. To have a better support solution for management, the Knowledge-based system is introduced. Usually, the Knowledge-based system consists of questions that are frequently raised by the end users. All the frequent questions are combined into categories and then, it is provided as a solution. The end users can solve their problems manually by themselves just by reading and implementing the provided solution. Also, the solutions that are provided by the helpdesk team can be used on future problems too. Hence, it is called a continuity and contingency process.

2. Automated Ticket Routing System

Author - Muhammad Zikri bin Zulkifli - 2011

In the existing helpdesk system, the tickets were created and assigned to the end user manually. When the ticket is created, it is assigned to the agent manually before they attend to that specific ticket. This manual process of ticket creation needs more manpower and takes more time. Instead of putting effort and time into this task, the ticket creation and assigning can be done automatically when we create an Automated Ticket Routing system. The automated ticket creation and assignment process reduce the time, and the manpower can be used for other purposes.

Then, by using the manual ticket creation and assignment process, the distribution of good skill sets, and workload balancing will be missed out. Finding a good skill set and automatically assigning the ticket to the specific skilled agent is considered a good job distribution. Also, there might be a chance tickets that are mistakenly routed to the wrong agent. Here, the wrong agent represents the sense that the agent doesn’t know well about that particular problem or issue. If the tickets are mistakenly routed, the resources may get wasted and a lot of time will be spent unnecessarily. Using the location, skill sets, work schedule, and workload balancing, the tickets can be routed automatically to that particular agent perfectly. We can execute the above process perfectly by categorizing the tickets based on the issues.

3. Smart Help Desk Automated Ticketing System

Author - Dhiraj Temkar, Sheetal Singh, Leema Bari, Prof. Snigdha Banga -2021

Automated technical queries help desk is proposed to possess instant real-time quick solutions and ticket categorization. Incorrect routing of tickets to the incorrect resolver group causes delays in resolving the matter. It also causes unnecessary resource utilization, and customer dissatisfaction and affects the business. To beat these problems, the proposed "Smart Automated Ticketing System" supports supervised machine learning techniques that automatically predict the category of the ticket using the natural language ticket description entered by the user through a chat interface. It also helps in the faster resolution of customer issues and sends them an email about the status of the ticket. This process assures customer satisfaction and also keeps the customers within the loop.

4. Corporate IT-Support Help-Desk Process Hybrid-Automation Solution with Machine Learning Approach

Author - Kuruparan Shanmugalingam, Nisal Chandrasekara, Calvin Hindle, Gihan Fernando, Chanaka Gunawaradhana - 2019

In an organization, the Information Technology (IT) support a help desk operation is an important unit that handles the IT services of a business. Many large-scale organizations handle engagement and requests with employees on a 24×7 basis. As with any routine tasks, most processes of the support help desk unit are considered repetitive repetitive tasks such as entering information into an application, resetting passwords, unlocking applications, and credentials errors.

The industry has now realized that many repetitive business processes and tasks can be automated by using Robotic Process Automation (RPA) bots or robotic processes automotive software bots. The idea is to take the repetitive workload and hand it over to the RPA bots so that the employees can focus on more value-adding tasks and decision-making for the organization. The RPA bot would also help to reduce human errors and make processes more efficient, which would finally result in cost savings and productivity increase.

CHAPTER - 03

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

An empathy map is a widely-used visualization tool within the field of UX and HCI practice. Regarding empathetic design, the primary purpose of an empathy map is to bridge the understanding of the end user. Within the context of its application, this tool is used to build a shared understanding of the user's needs and provide context to a user-centered solution.

The traditional empathy map begins with four categories: says, thinks, does, and feels. At the center of the map, a user or persona is displayed to remind practitioners and stakeholders what type of individual this research is centered around. Each category of the empathy map represents a snapshot of the user's thoughts and feelings without any chronological order.

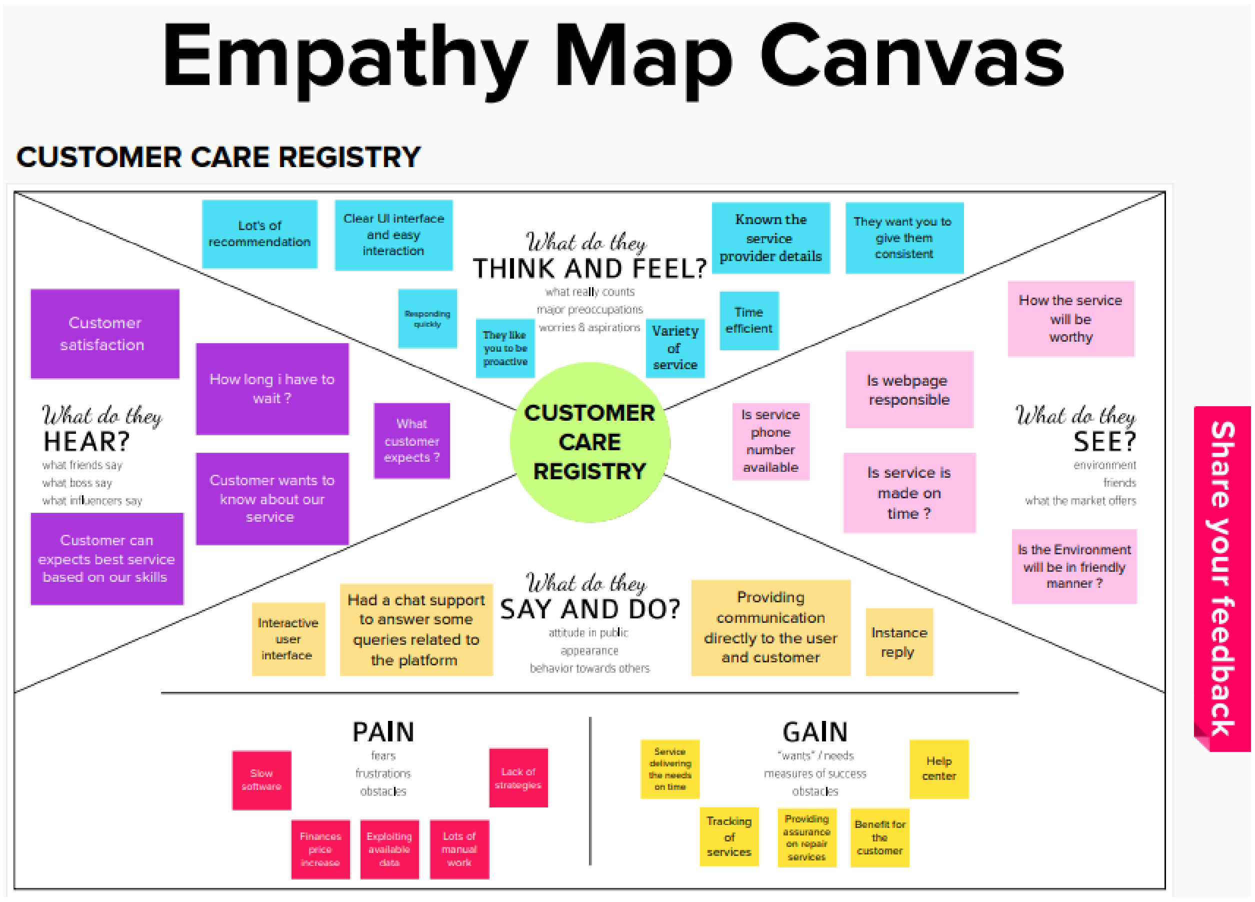
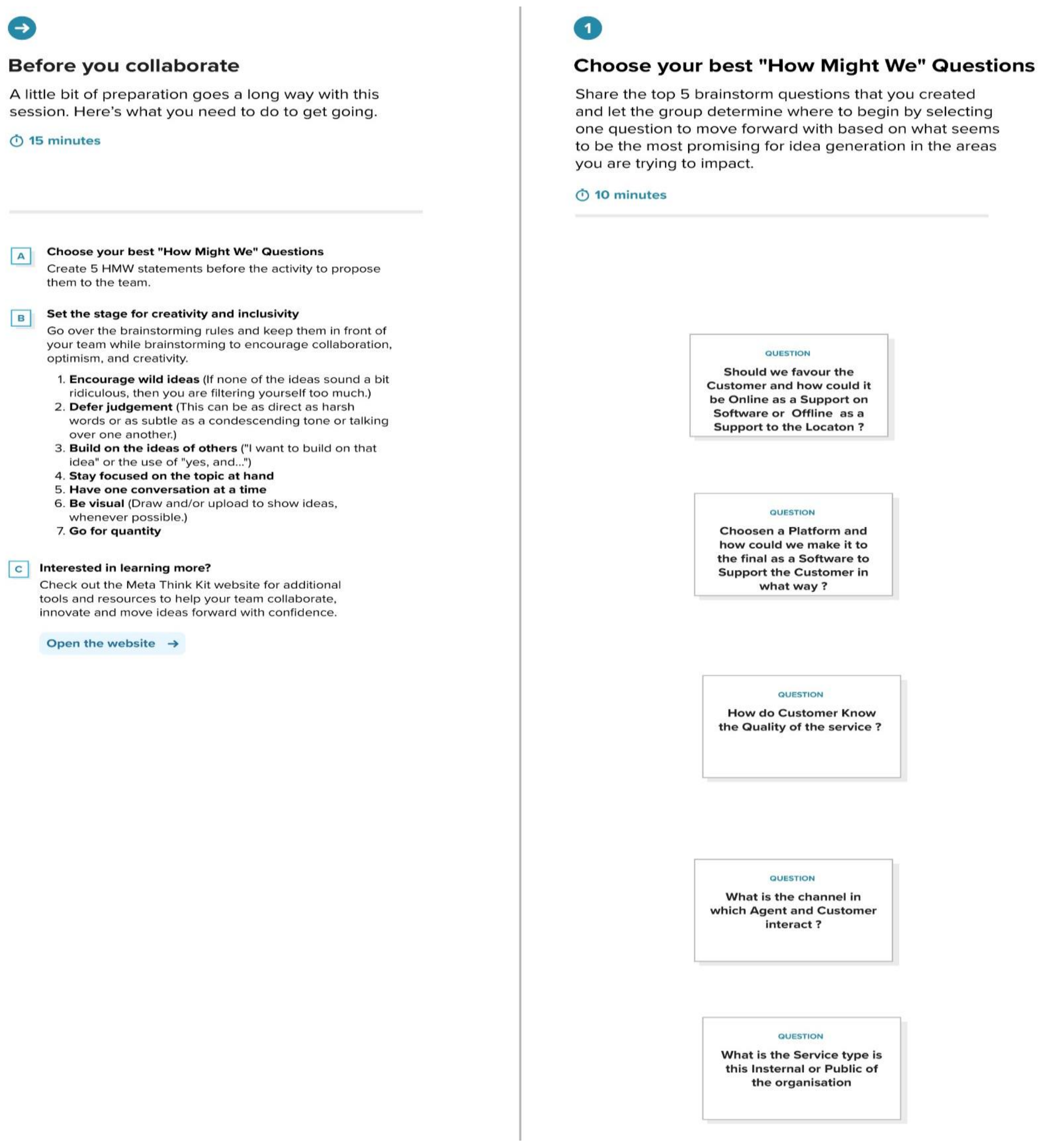


Figure - 3.1.1

3.2 Ideation & Brainstorming

Ideation is often closely related to the practice of brainstorming, a specific technique that is utilized to generate new ideas. A principal difference between ideation and brainstorming is that ideation is commonly more thought of as being an individual pursuit, while brainstorming is almost always a group activity.



Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

15 minutes

A **Choose your best "How Might We" Questions**
Create 5 HMW statements before the activity to propose them to the team.

B **Set the stage for creativity and inclusivity**
Go over the brainstorming rules and keep them in front of your team while brainstorming to encourage collaboration, optimism, and creativity.

1. **Encourage wild ideas** (If none of the ideas sound a bit ridiculous, then you are filtering yourself too much.)
2. **Defer judgement** (This can be as direct as harsh words or as subtle as a condescending tone or talking over one another.)
3. **Build on the ideas of others** ("I want to build on that idea" or the use of "yes, and...")
4. **Stay focused on the topic at hand**
5. **Have one conversation at a time**
6. **Be visual** (Draw and/or upload to show ideas, whenever possible.)
7. **Go for quantity**

C **Interested in learning more?**
Check out the Meta Think Kit website for additional tools and resources to help your team collaborate, innovate and move ideas forward with confidence.

Open the website →

1

Choose your best "How Might We" Questions

Share the top 5 brainstorm questions that you created and let the group determine where to begin by selecting one question to move forward with based on what seems to be the most promising for idea generation in the areas you are trying to impact.

10 minutes

QUESTION
Should we favour the Customer and how could it be Online as a Support on Software or Offline as a Support to the Locaton ?

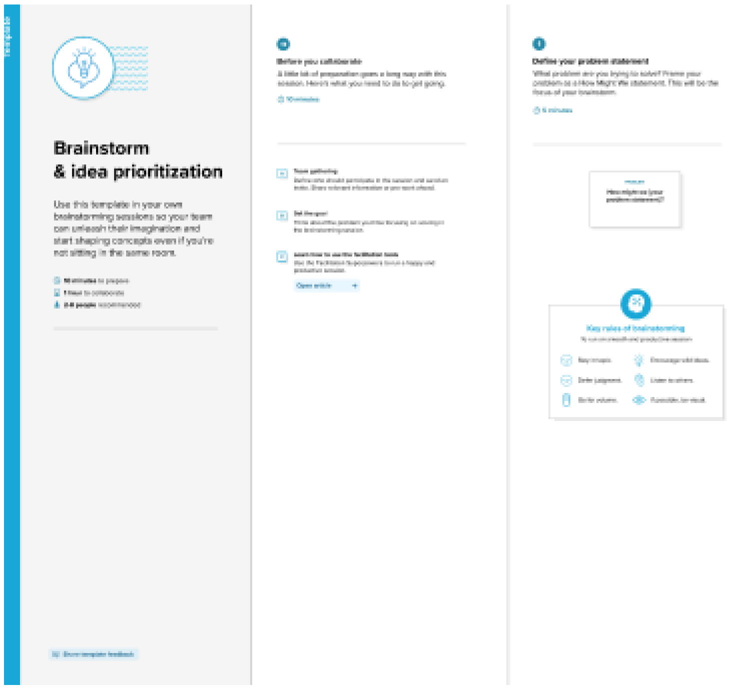
QUESTION
Choosen a Platform and how could we make it to the final as a Software to Support the Customer in what way ?

QUESTION
How do Customer Know the Quality of the service ?

QUESTION
What is the channel in which Agent and Customer interact ?

QUESTION
What is the Service type is this Insternal or Public of the organisation

Figure - 3.2.1



Template

Brainstorm
& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Share template feedback

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
10 minutes

Team gathering
[illegible]

Set the goal
[illegible]

Learn how to use the facilitation tools
[illegible]

Open article

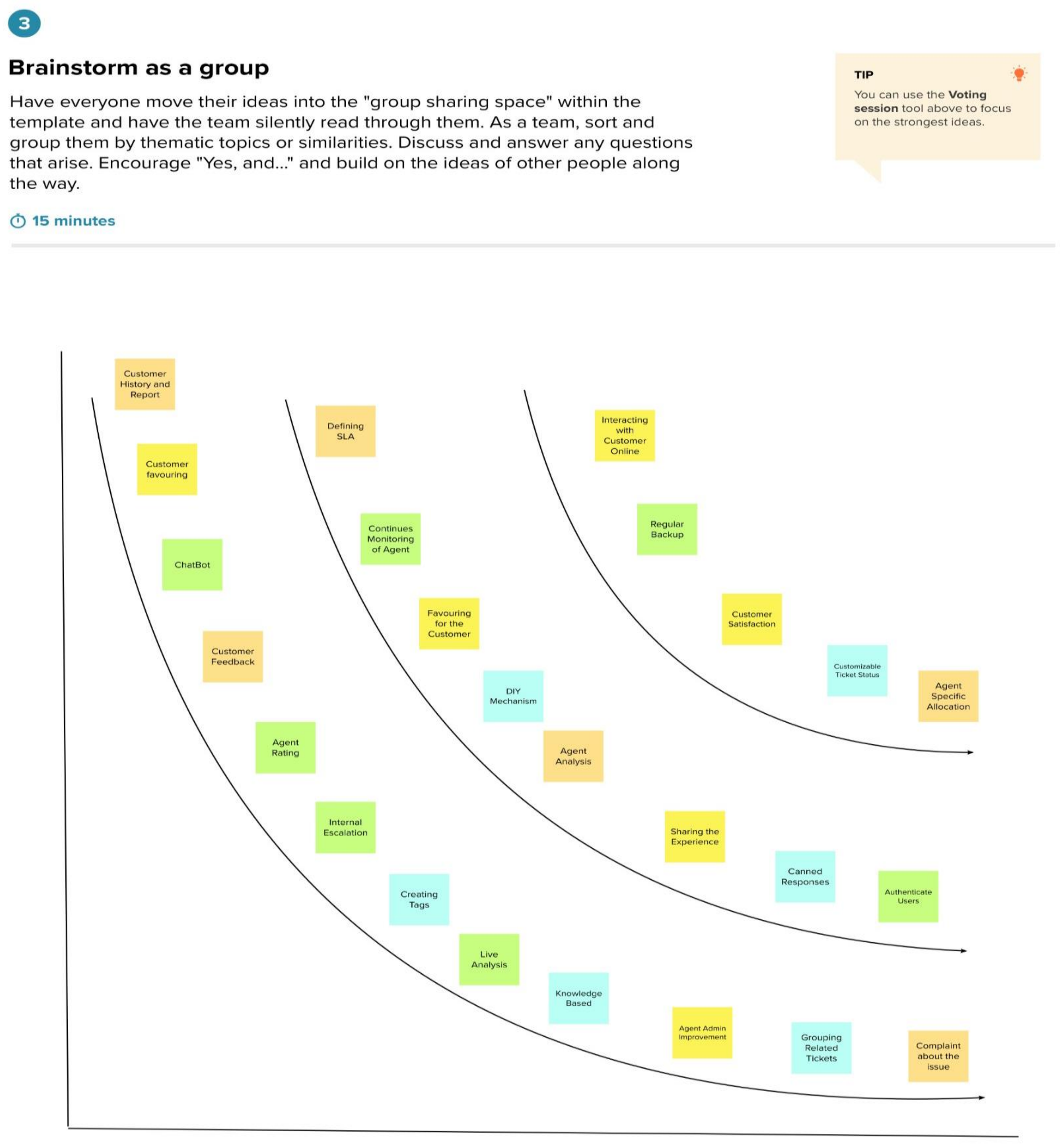
Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes

How might we [your problem statement]?

Key rules of brainstorming
[illegible]

Stay in topic.
Encourage wild ideas.
Defer judgment.
Listen to others.
Go for volume.
If possible, be visual.

Figure - 3.2.2



3

Brainstorm as a group

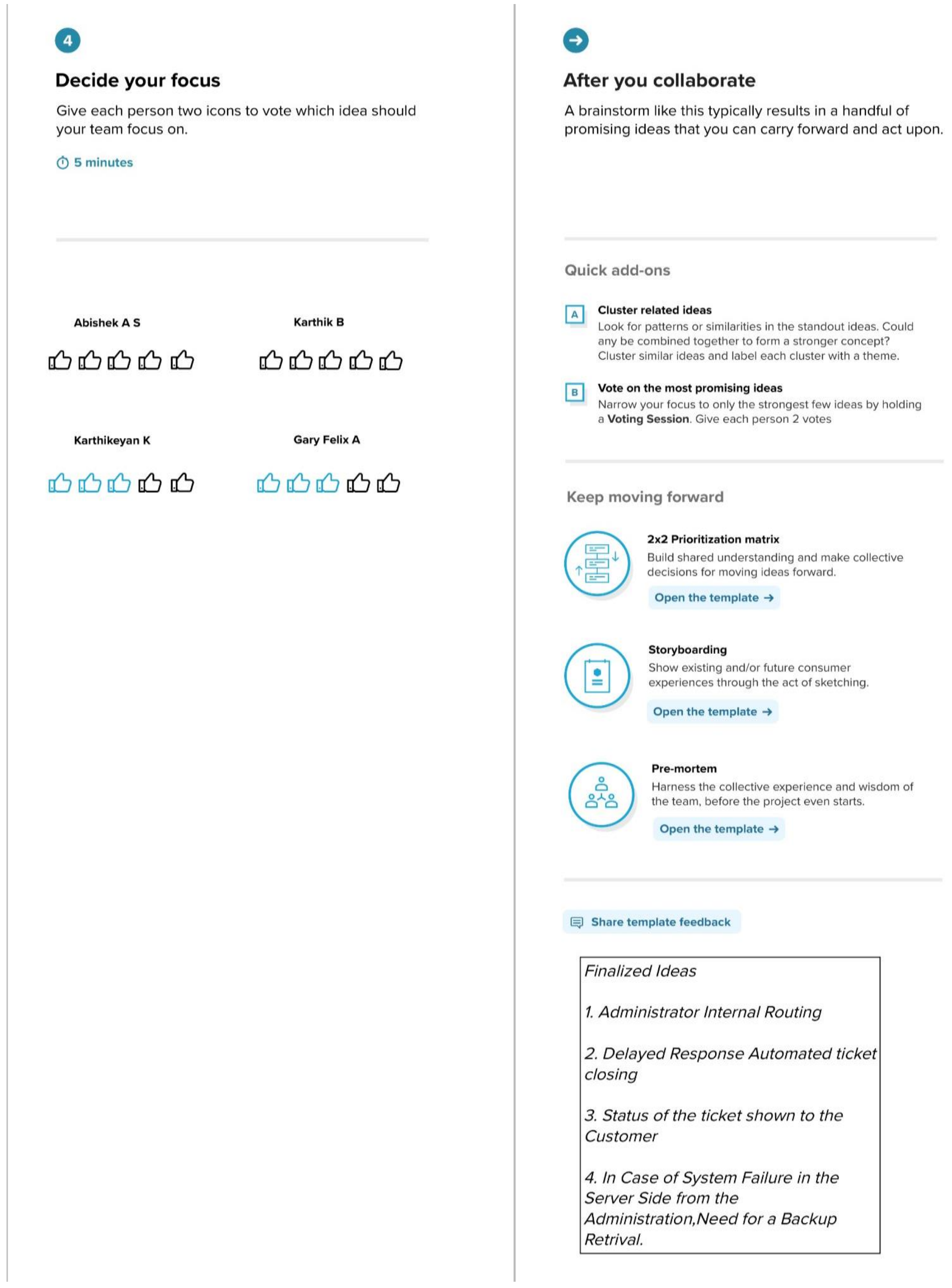
Have everyone move their ideas into the "group sharing space" within the template and have the team silently read through them. As a team, sort and group them by thematic topics or similarities. Discuss and answer any questions that arise. Encourage "Yes, and..." and build on the ideas of other people along the way.

15 minutes

TIP

You can use the **Voting session** tool above to focus on the strongest ideas.

Figure - 3.2.3



4

Decide your focus

Give each person two icons to vote which idea should your team focus on.

5 minutes

Abishek A S

Karthik B

Karthikeyan K

Gary Felix A

After you collaborate

A brainstorm like this typically results in a handful of promising ideas that you can carry forward and act upon.

Quick add-ons

A **Cluster related ideas**
Look for patterns or similarities in the standout ideas. Could any be combined together to form a stronger concept? Cluster similar ideas and label each cluster with a theme.

B **Vote on the most promising ideas**
Narrow your focus to only the strongest few ideas by holding a **Voting Session**. Give each person 2 votes

Keep moving forward

2x2 Prioritization matrix
Build shared understanding and make collective decisions for moving ideas forward.
Open the template →

Storyboarding
Show existing and/or future consumer experiences through the act of sketching.
Open the template →

Pre-mortem
Harness the collective experience and wisdom of the team, before the project even starts.
Open the template →

Share template feedback

Finalized Ideas

1. Administrator Internal Routing

2. Delayed Response Automated ticket closing

3. Status of the ticket shown to the Customer

4. In Case of System Failure in the Server Side from the Administration,Need for a Backup Retrival.

Figure - 3.2.4

3.3 Proposed Solution

- **Assigned Agent Internal Routing**

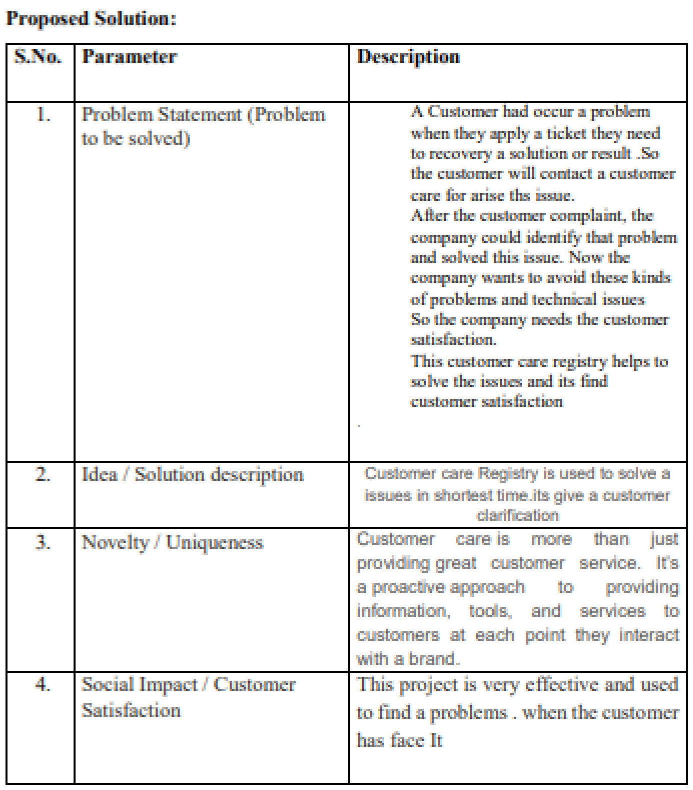
  Assigned Agent routing can be solved by directly routing to the specific agent about the issue using the specific Email. Here, the customer chooses the specific issue type, and based on the issue type, the routing occurs. Clearly, the admin assigns every agent a role, and based on the role, the issue type is defined for the Customer.

- **Automated Ticket Closure**

  Automated Ticket closure by using daily sync of the daily database. Instead of closing the tickets manually, we can automate the process to close the ticket. The process will be, if the user doesn't make any response to the agent for a particular time period, then the tickets must be closed automatically. This automation process will save much time and we can use that manpower for any other purposes.

- **Status Unavailable to the Customer**

  Status Shown to the Customer can display the status of the ticket to the customer. Here, in this solution, the user can be able to view the status of the ticket. Also, the customer can able to resume the ticket again just by entering the query which is displayed in their profile(Customer Profile).

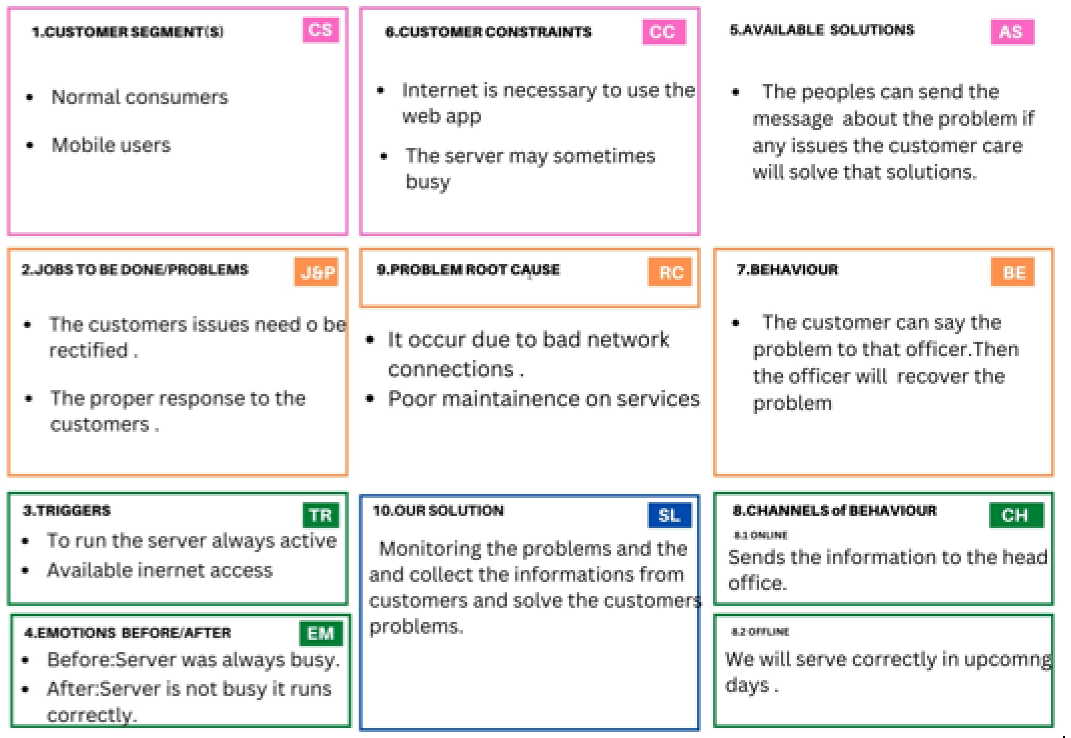


Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	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 .
2.	Idea / Solution description	Customer care Registry is used to solve a issues in shortest time.its give a customer clarification
3.	Novelty / Uniqueness	Customer care is more than just providing great customer service. It's a proactive approach to providing information, tools, and services to customers at each point they interact with a brand.
4.	Social Impact / Customer Satisfaction	This project is very effective and used to find a problems . when the customer has face It

3.4 Problem Solution Fit

Proposed Solution Template: Problem Fit Control



1.CUSTOMER SEGMENT(S) CS

- Normal consumers
- Mobile users

6.CUSTOMER CONSTRAINTS CC

- Internet is necessary to use the web app
- The server may sometimes busy

5.AVAILABLE SOLUTIONS AS

- The peoples can send the message about the problem if any issues the customer care will solve that solutions.

2.JOBS TO BE DONE/PROBLEMS J&P

- The customers issues need o be rectified .
- The proper response to the customers .

9.PROBLEM ROOT CAUSE RC

- It occur due to bad network connections .
- Poor maintainence on services

7.BEHAVIOUR BE

- The customer can say the problem to that officer.Then the officer will recover the problem

3.TRIGGERS TR

- To run the server always active
- Available inernet access

10.OUR SOLUTION SL

Monitoring the problems and the and collect the informations from customers and solve the customers problems.

8.CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Sends the information to the head office.

4.EMOTIONS BEFORE/AFTER EM

- Before:Server was always busy.
- After:Server is not busy it runs correctly.

8.2 OFFLINE

We will serve correctly in upcomng days .

CHAPTER - 04

4. REQUIREMENT ANALYSIS

4.1 Functional Requirement

S. No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
1.	Admin/Agent Registration	Registration through Gmail
2.	Admin/Agent Confirmation	Confirmation via Email.
3.	Customer Query	Access through Email, Chatbot from targeted websites
4.	Customer Confirmation	Confirmation through Ticket ID in Email

5.	Database	Storing the object model.

4.2 Non-Functional Requirement

S. No.	Non-Functional Requirement (Epic)	Description
1.	Usability	User Friendly, Easily Accessible.
2.	Security	IBM Digital Security Certificate(SSL) for Database.
3.	Reliability	Providing Quality Content.
4.	Performance	Quick Access, Flexible, and Responsive
5.	Availability	24/7 Support
6.	Scalability	Good performance for large Customers and workload

CHAPTER - 05

5. PROJECT DESIGN

5.1 Data Flow Diagram

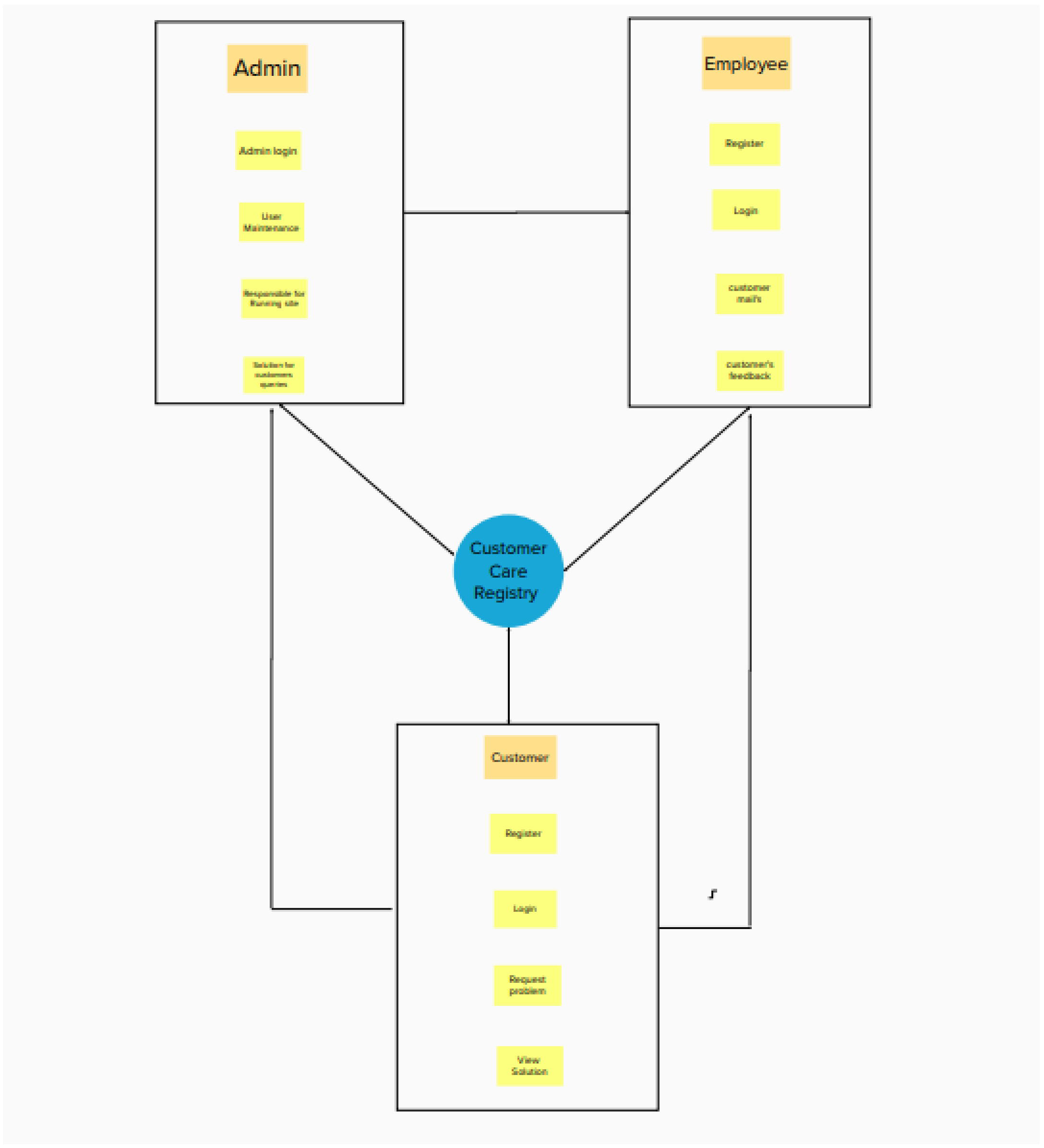


Figure 5.1.1

5.2 Solution and Technical Architecture

Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Example - Solution Architecture Diagram:

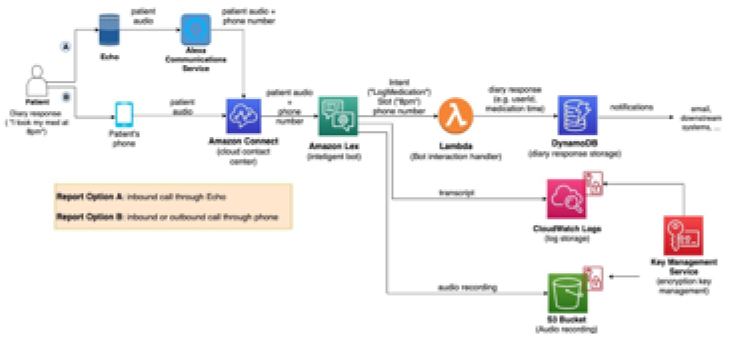

patient audio
patient audio + phone number
Echo
Alexa Communications Service
A
B
Patient
Diary response ("I took my med at 8pm")
Patient's phone
patient audio
Amazon Connect (cloud contact center)
patient audio + phone number
Amazon Lex (intelligent bot)
Intent ("LogMedication") Slot ("8pm") phone number
Lambda (Bot interaction handler)
diary response (e.g. userId, medication time)
DynamoDB (diary response storage)
notifications
email, downstream systems, ...
Report Option A: inbound call through Echo
Report Option B: inbound or outbound call through phone
transcript
CloudWatch Logs (log storage)
Key Management Service (encryption key management)
audio recording
S3 Bucket (Audio recording)

Figure 5.2.1

5.3 User Stories

User Type	Functional Requirement	User Story Number	User Story/Task	Acceptance Criteria	Priority	Release
Customer (Client User)	Registration	USN-1	As a User, I will register for the application by entering my email, password, and confirming my password	I will be redirected to Email Verification	High	Sprint-1
Customer(Automate d User)	Registration	USN-2	As a User, I will Validate the Customer Credentials once after the Email Verification.	I will receive a confirmation Message from Administrator	High	Sprint-2

Customer(Automated User)	Registration	USN-3	As a User, I will issue the Customer with Login Id and Password through Object Creation from the Customer Credentials.	I can register & access the dashboard with Facebook Login	Medium	Sprint-1
Customer(Client User)	Login	USN-4	As a User, I will log in to the Portal using the Login Credentials Provided.	I will be redirected to the Portal Dashboard Page	Medium	Sprint-2
Customer (Mobile User)	Dashboard	USN-5	As a User, I will book a ticket from available sections along the Application and Submit the Ticket to the Portal	I will be Issued with a Ticket Applied Message from the Portal.	High	Sprint-3
Customer (Admin User)	Validation	USN-6	As a User, I will issue a Suitable Agent to the Customer and provide a Bot Connectivity with the Agent.	I will send an email about the agent issuing the application.	High	Sprint-2
Customer(Automated User)	Bot Connected	USN-7	As a User, I will connect the Bot to the Customer and provide repeated Status of the Query to the Customer	I will Receive a Message from the Responsive Server Bot.	Low	Sprint-4

Customer(Agent User)	Agent	USN-8	As a User, I will satisfy all the queries to the Customer for all the repetitive responses from the Customers.	I will communicate with a Query from the Server Bot.	Medium	Sprint-3
Customer(Server User)	Feedback	USN-9	As a User, I will fill up the Feedback form provided to improve or service provided by the Application	I will accept the Feedback and issue a message for queries	High	Sprint-4
Customer(Applicatio n User)	Log out	USN-10	As a User, I will Log out of the Application when my Queries are over or else will begin again from the beginning.	I will Estimate the User Response and React to end the Process	Low	Sprint-1

CHAPTER - 06

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	**Functional Requireme nt (Epic)**	**User Story Numbe r**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**

Sprint-1	Registration (Admin and Customer)	USN-1	As a user, I can register for the application by entering My Registration Credentials like name,username,password ,confirm password…	3	Medium	Gary Felix A,Karthikeyan K
Sprint-1	Login and Dashboard(Admin and Customer)	USN-2	As a user, I will Login into the Dashboard Page by using the Login Credentials once after the Mail Activation link Authentication.	5	Medium	Gary Felix A,Karthikeyan K
Sprint-1	Authentication and IBM DB2	USN-3	As a user, I will be authenticated from the Administration and store the Credentials back to the Database using IBM DB2.	5	High	Gary Felix A,Karthikeyan K
Sprint-2	Email Integration and SendGrid API	USN-4	As a user, I will send the email to the Customer automatically using SendGrid API with the Subject to ticket id ,agent name…	5	Medium	Karthik B,Abishek A S
Sprint-2	DB Schema for Queries	USN-5	As a user, I will create and map the credentials of the Customer from the Application through Tables and Schema.	3	High	Gary Felix A,Karthikeyan K

Sprint-3	Watson Assistant	USN-6	As a user, I will walk through the customer to resolve the queries and also connect the live agent to the Application.	3	High	Karthik B
Sprint-3	Knowledge Base Assistant	USN-7	As a user, I will provide predefined Queries like FAQs so that the customer can be solved by DIY Mechanism.	5	Medium	Abishek A S
Sprint-4	Deployment with Docker	USN-8	As a User, I will deploy the entire Application using Docker.	1	Medium	Karthik B, Abishek A S
Sprint-4	Orchest with Kubernetes	USN-9	As a User, I will allocate the server nodes and balance the workloads in the server.	2	Medium	Gary Felix A, Karthikeyan K, Abishek A S and Karthik B

6.2 Sprint Delivery Schedule

Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	13	6 Days	24 Oct 2022	29 Oct 2022		
Sprint-2	8	6 Days	31 Oct 2022	05 Nov 2022		
Sprint-3	8	6 Days	07 Nov 2022	12 Nov 2022		
Sprint-4	3	6 Days	14 Nov 2022	19 Nov 2022		

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day).

Average Velocity of Sprint-1 = 13/6 =2.17

Average Velocity of Sprint-2 = 8/6 =1.25

Average Velocity of Sprint-3 = 8/6 =1.25

Average Velocity of Sprint-4 = 3/6 =0.5

6.3 Reports from JIRA

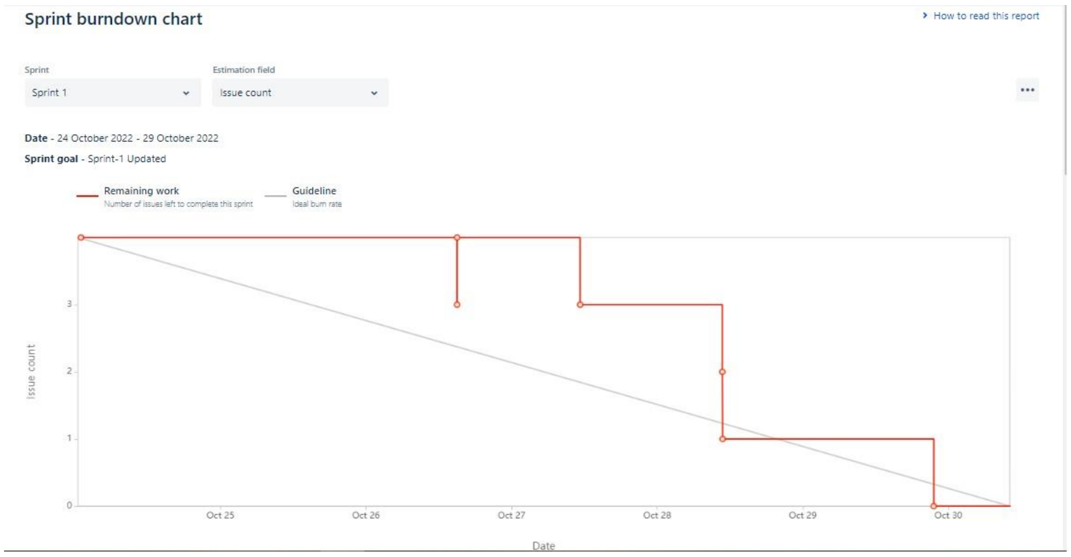


Figure - 6.3.1

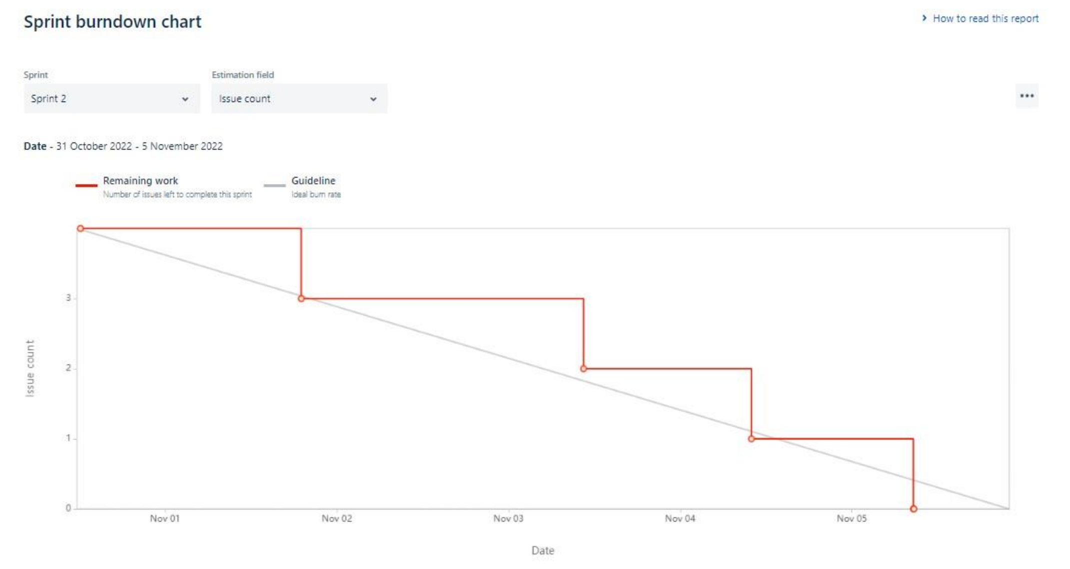


Figure - 6.3.2

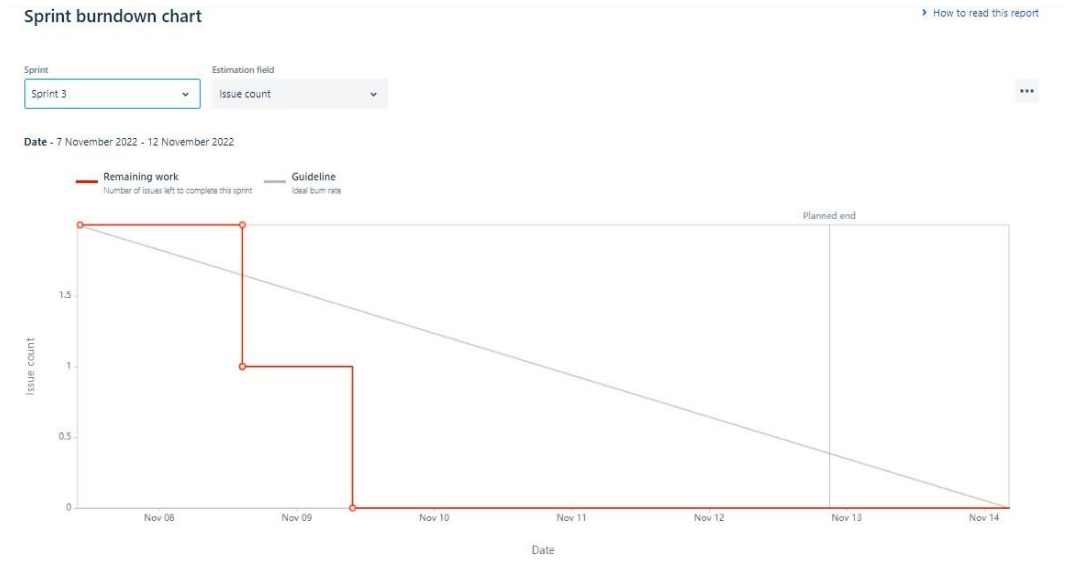


Figure - 6.3.3

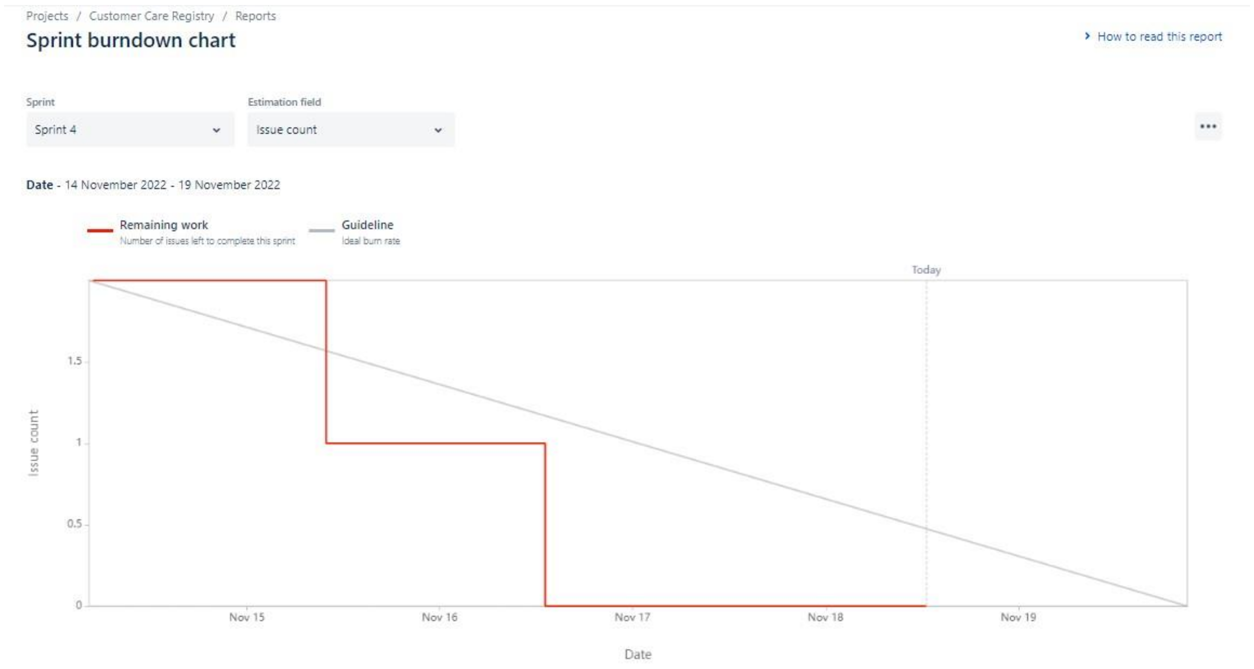
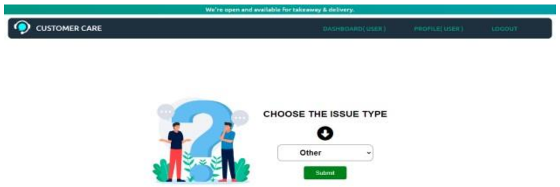


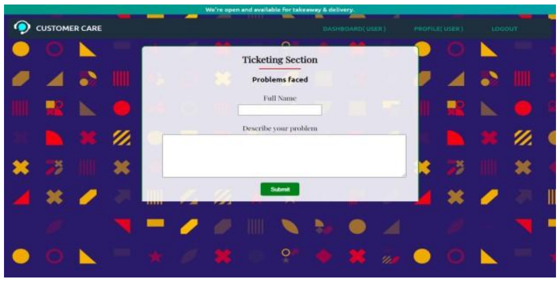
Figure - 6.3.4

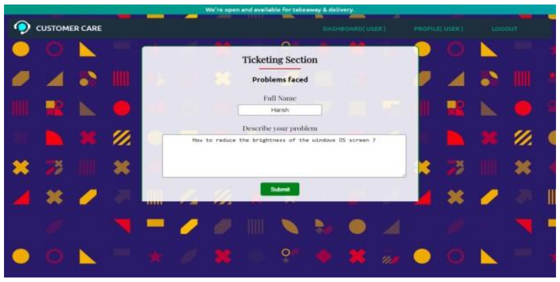
CHAPTER - 07

7.1 Assigned Agent Internal Routing

Assigned Agent routing can be solved by directly routing to the specific agent about the issue using the specific Email. Here, the customer chooses the specific issue type, and based on the issue type, the routing occurs. Clearly, the admin assigns every agent a role, and based on the role, the issue type is defined for the Customer.

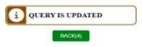


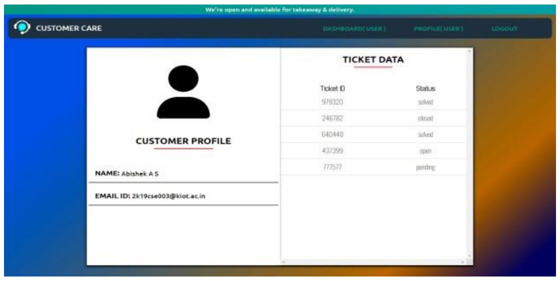




Full Name: Harish

Describe your problem: How to reduce the brightness of the windows OS screen ?

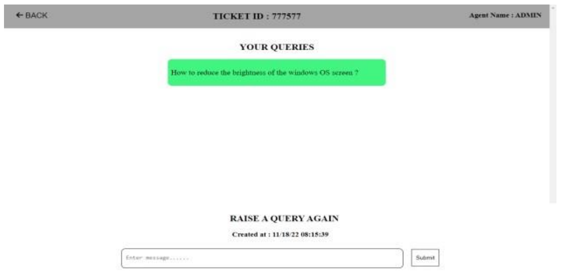




NAME: Abishek A S

EMAIL ID: 2k19cse003@kiot.ac.in

Ticket ID	Status
979320	solved
246782	closed
640440	solved
437399	open
777577	pending



TICKET ID : 777577

Agent Name : ADMIN

YOUR QUERIES

How to reduce the brightness of the windows OS screen ?

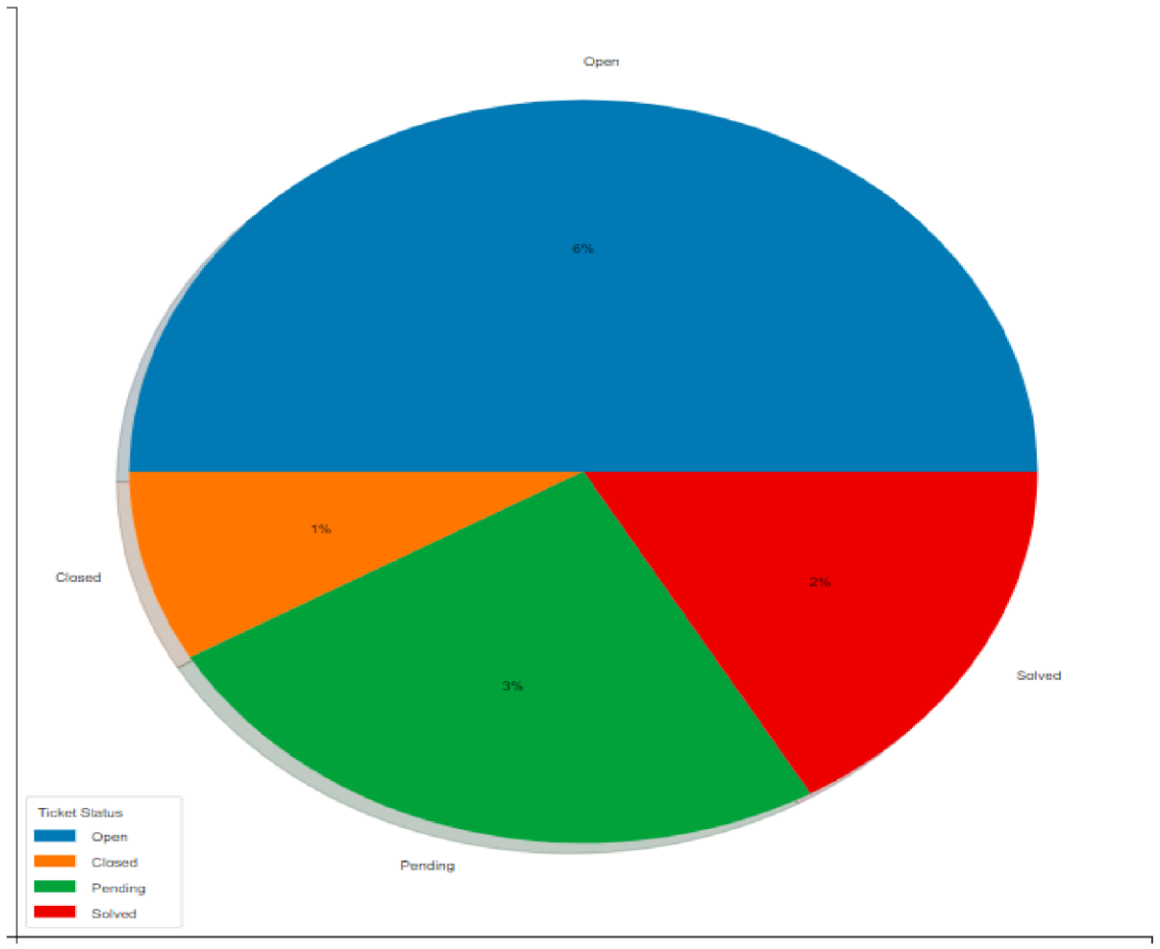
RAISE A QUERY AGAIN

Created at : 11/18/22 08:15:39

Enter message.......

7.2 Report of the Status

Dynamic report generation using Numpy and Matplotlib to create a Piechart for the available tickets namely open, pending, closed, and solved. For every ticket type, there is a creation of pie section and this chart provides a complete status measure for the application on ticket management.



CHAPTER - 08

8. TESTING

8.1 Test Cases

Test Case	Description	Test Step	Expected Result	Status

Sign Up	Sign Up as Admin /User	1. Enter a valid Email and Password as an Admin / User 2. Verify your OTP from email	The Link will be taken to log in.	**Pass or Fail**
Login	Login as Admin/Agent/User	1. Enter a Valid Email and Password to Validate	The Link will be redirected to the Dashboard Page	**Pass or Fail**
Forgot Password	Forget Passwords for All.	1. Enter a Valid Email 2. Enter OTP and New Password to verify	The Password will be Updated and Allowed to Login	**Pass or Fail**
Raise a Ticket	Raise a Ticket for the Issues for All Users.	1. Select Issue Type and Enter your name as well as Query 2. Submit the Query	Query will be Updated	**Pass or Fail**

8.2 User Acceptance Testing

Test Case ID	**Feature Type**	**Compone nt**	**Test Scenario**	**Prerequi site**	**Steps To Execute**
SignUpPag e_TC_001	Functional	Home Page and Sign up Page	Verify users are able to see the Signup pop-up when the user clicked on Sign up.		1. Enter URL and click go 2. Click on Sign up from Navbar3. Verify Signup pop up displayed or not.

SignUpPage_TC_002	UI	Home Page and Sign-up Page	Verify the UI elements in the Signup popup.		1. Enter URL and click go 2. Click on the Signup button 3. Verify the Signup popup with thebelow UI elements: a. Username text box b. Email text box c. Password text box d. Re-enter password text box e. Signup button f. Existing customer? Login accountlink
LoginPage_TC_001	Functional	Home Page and Login Page	Verify the user is able to see the Login when the user is clicked on Login.		1. Enter URL and click go 2. Click on Login from Navbar 3. Verify whether the Login popup isdisplayed or not
LoginPage_TC_002	UI	Home Page and Login Page	Verify the UI elements in the Login popup.		1. Enter URL and click go 2. Click on the Login button 3. Verify the Login popup with thebelow UI elements: a.username/email text box b.password text box c. Login button d.New customer? Signup Account link e.Forgot Password Link
AdminInvitePage_TC_001	Functional/UI	Admin Invite Page	Verify if the agent is not invited unless the admin invites him.	User logged as Admin	1. Verify Agent details and invite Agent a.email text box b.password text box

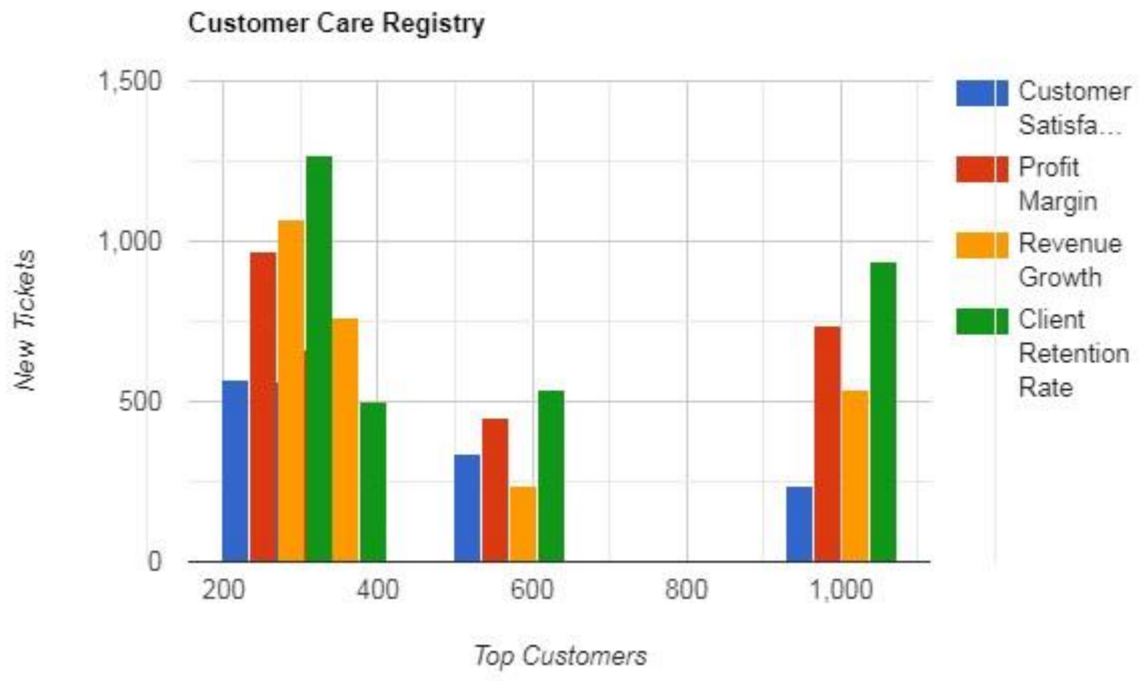
Test Case ID	**Test Data**	**Expected Result**	**Actual Result**	**Status**	**Comments**	**TC for Automation (Y/N)**	**Bug ID**	**Executed By**
SignUpPage_TC_001		Signup popup should display	Working as expected	Pass		Y		

SignUpPage_TC_002		Application should show below UI elements: a. Username textbox b. Email text box c. Password textbox d. Re-enterpassword text box e. Signup button f. Existing customer? Login account link	Working as expected	Pass		Y		
LoginPage_TC_001		Login popup should display	Working as expected	Pass		Y		
LoginPage_TC_002		Application should show below UI elements: a.email text box b.password text box c. Signup button d.Existing customer? Login account links	Working as expected	Pass		Y		
AdminInvitePage_TC_001		Admin User popup should display and Agent should be able to be allocated.	Working as expected	Pass		Y		

CHAPTER - 09

9. RESULTS

9.1 Performance Metrics



S. No	Project Name	Scope/ feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/ Volume Changes	Risk Score	Justification
1.	Customer Care Registry	New	Moderate	No Change	Low	-	>30% to 50%	Orange	Changes have been observed
2.	Customer Care Registry	New	Low	No Change	Low	-	>40% to 60%	Orange	Changes have been observed
3.	Customer Care Registry	New	Low	No Change	Low	-	>20% to 50%	Green	Changes have been observed

S.No	Project Overview	NFT Test Approach	Assumptions/Dependencies/Risks	Approvals/SignOff
1.	Customer Care Registry	Usability	Low	Kavirajan K
2.	Customer Care Registry	Scalability	Low	Narayanan K

S. No	Project Overview	NFT Test Approach	NFT - MET	Test Outcome	GO/ NO-GO Decision	Recommendation	Identified Defects(Detected/Closed/Open)	Approvals / SignOff
1.	Customer Care Registry	Scalability	Yes	Good		Increase number of pods	Closed	Gary Felix A

CHAPTER - 10

10. ADVANTAGES & DISADVANTAGES

10.1 ADVANTAGES

- The Main Advancement in our Application is Covering the Assigned Agent Internal Routing which covers first selecting the Type of Issue, then routing directly to the Particular Available Agent.

- The second advantage of our application is Automated Ticket Closure. This happens when they don't make their response to the agent or admin.

- Status Shown to the Customer can display the status of the ticket to the customer. Here, in this solution, the user can be able to view the status of the ticket. Also, the customer can able to resume the ticket again just by entering the query which is displayed in their profile(Customer Profile).

10.2 DISADVANTAGE

- Our application doesn’t have any kind of disadvantages, but in the future, we will improve our application to the next level by adding amazing features. For example, in the future, customers will be able to raise a query through videos or images.

CHAPTER - 11

11. CONCLUSION

This is our Customer Care Registry Application. For your better understanding, here is the brief conclusion of our project. Customer Care Registry is developed in Cloud technology and our project is called a Support Channel. Here, the users can able to raise technical-related queries to Agent. The agent is assigned by the Admin.

The agent solves the query that is raised by the user. Also, the user can solve their problem by themselves with the help of the knowledge base center. This Knowledge base center is called a DIY Mechanism. Apart from that, we have three channels. They are a Knowledge base center, Email Channel and Chatbot.

With the help of these Omni channels, user can resolve their queries. Our application has several advantages Assigned Agent Internal Routing, Automated Ticket Closure, and Status to the user. In the future, we’ll provide some updates like the user can raise a query with the help of images or videos.

CHAPTER - 12

12. FUTURE SCOPE

- In the future we will improve our application to the next level by adding amazing features. For example, in the future, customers will be able to raise a query through videos or images.

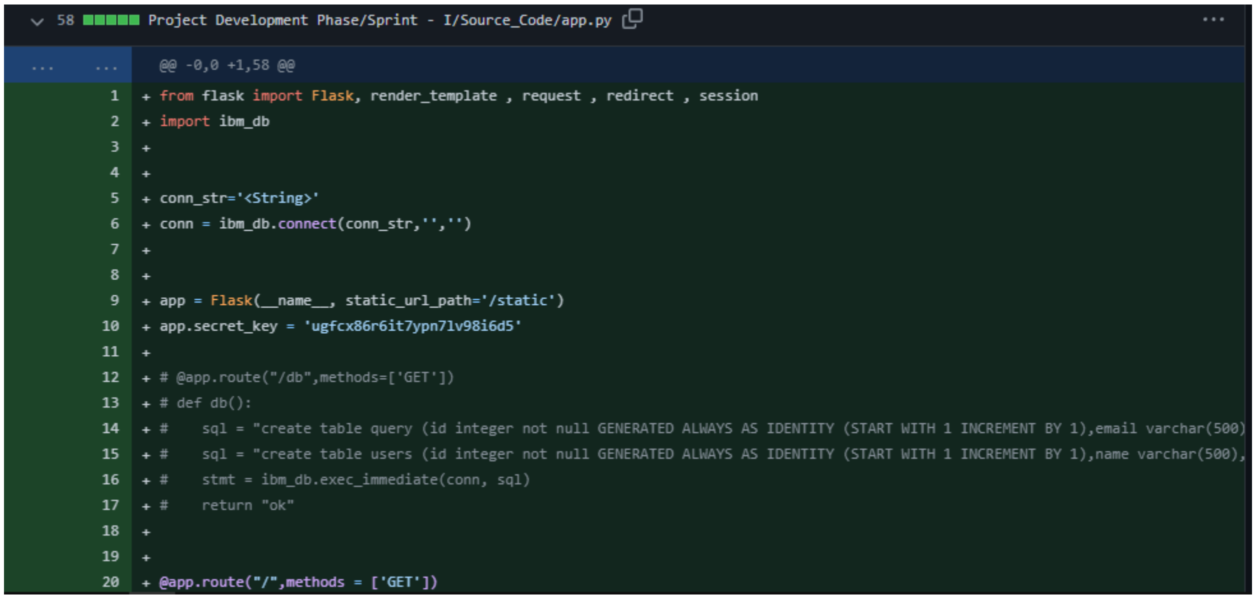
- Also, in the future, we will bring up Service Level Agreement(SLA), and based on that the user will be able to set the priority for their raised ticket.

- Additionally, in the future, the Knowledge base center will be upgraded to the next level dynamically.

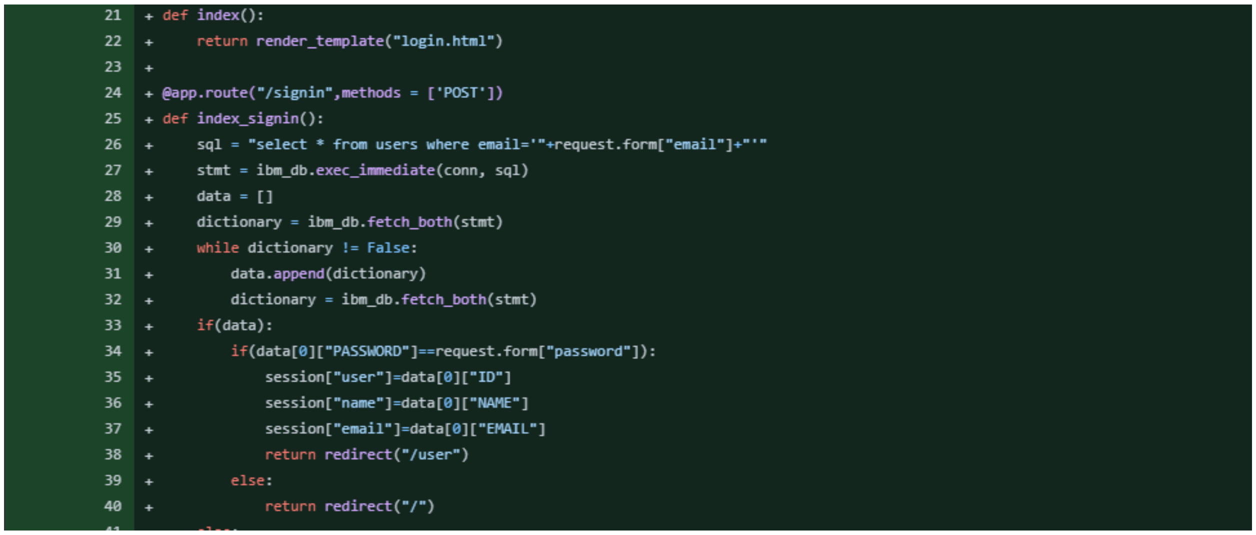
CHAPTER - 13

13. APPENDIX

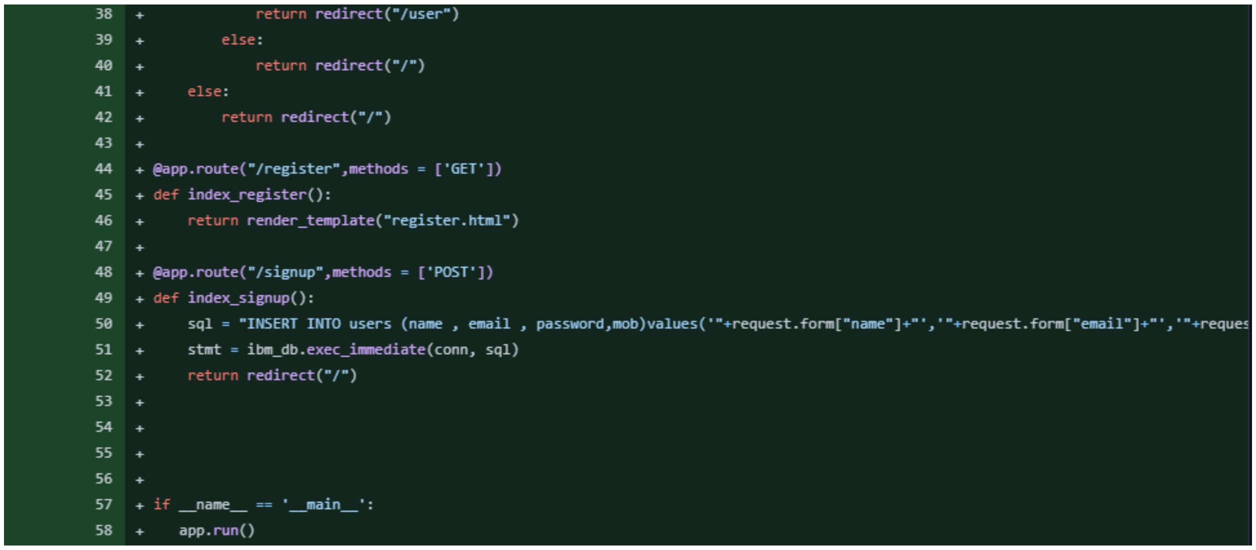
13.1 Source Code



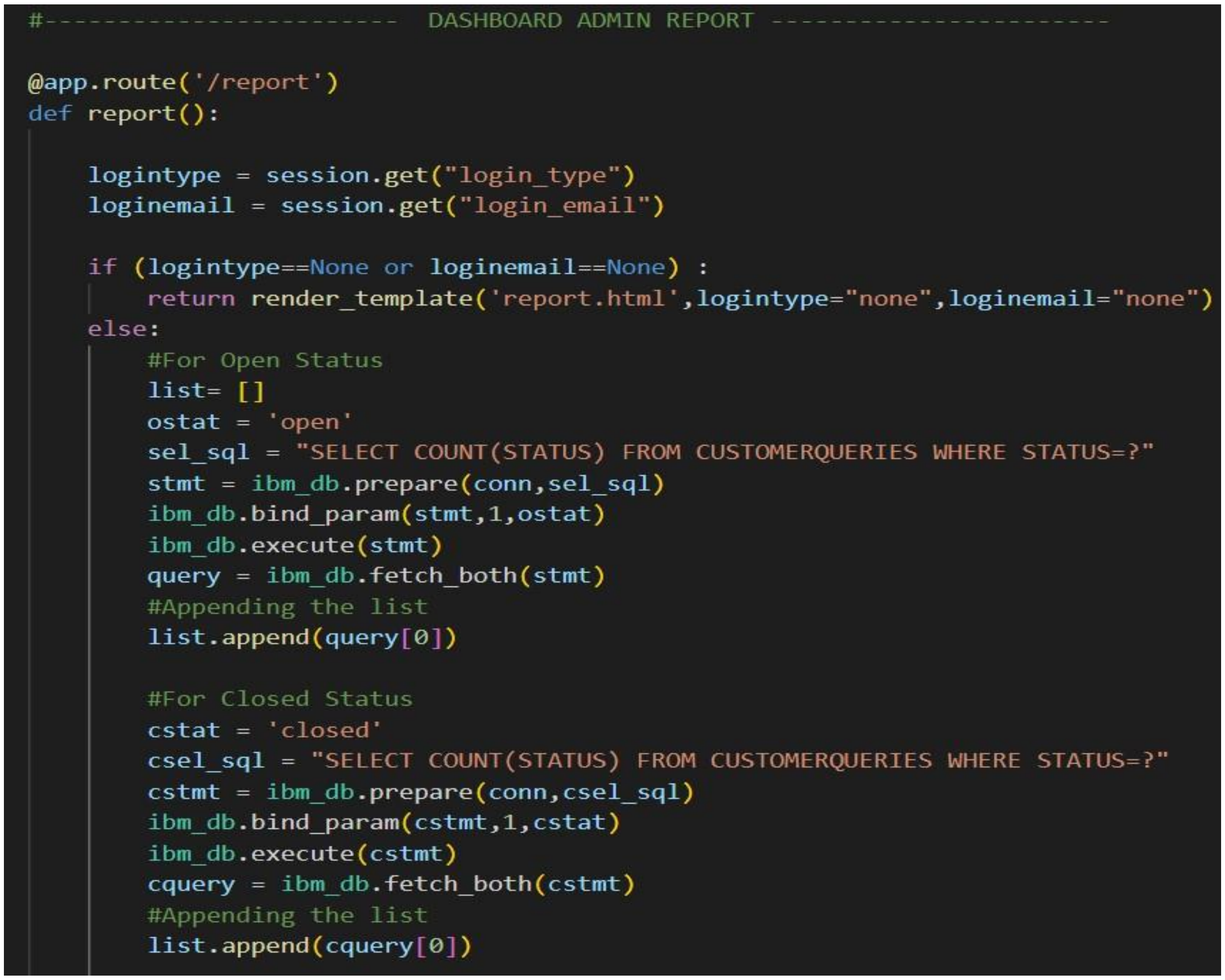
```
58 Project Development Phase/Sprint - I/Source_Code/app.py
@@ -0,0 +1,58 @@
+ from flask import Flask, render_template , request , redirect , session
+ import ibm_db
+
+
+ conn_str='<String>'
+ conn = ibm_db.connect(conn_str,'','')
+
+
+ app = Flask(__name__, static_url_path='/static')
+ app.secret_key = 'ugfcx86r6it7ypn7lv98i6d5'
+
+ # @app.route("/db",methods=['GET'])
+ # def db():
+ #     sql = "create table query (id integer not null GENERATED ALWAYS AS IDENTITY (START WITH 1 INCREMENT BY 1),email varchar(500)
+ #     sql = "create table users (id integer not null GENERATED ALWAYS AS IDENTITY (START WITH 1 INCREMENT BY 1),name varchar(500),
+ #     stmt = ibm_db.exec_immediate(conn, sql)
+ #     return "ok"
+
+
+ @app.route("/",methods = ['GET'])
```



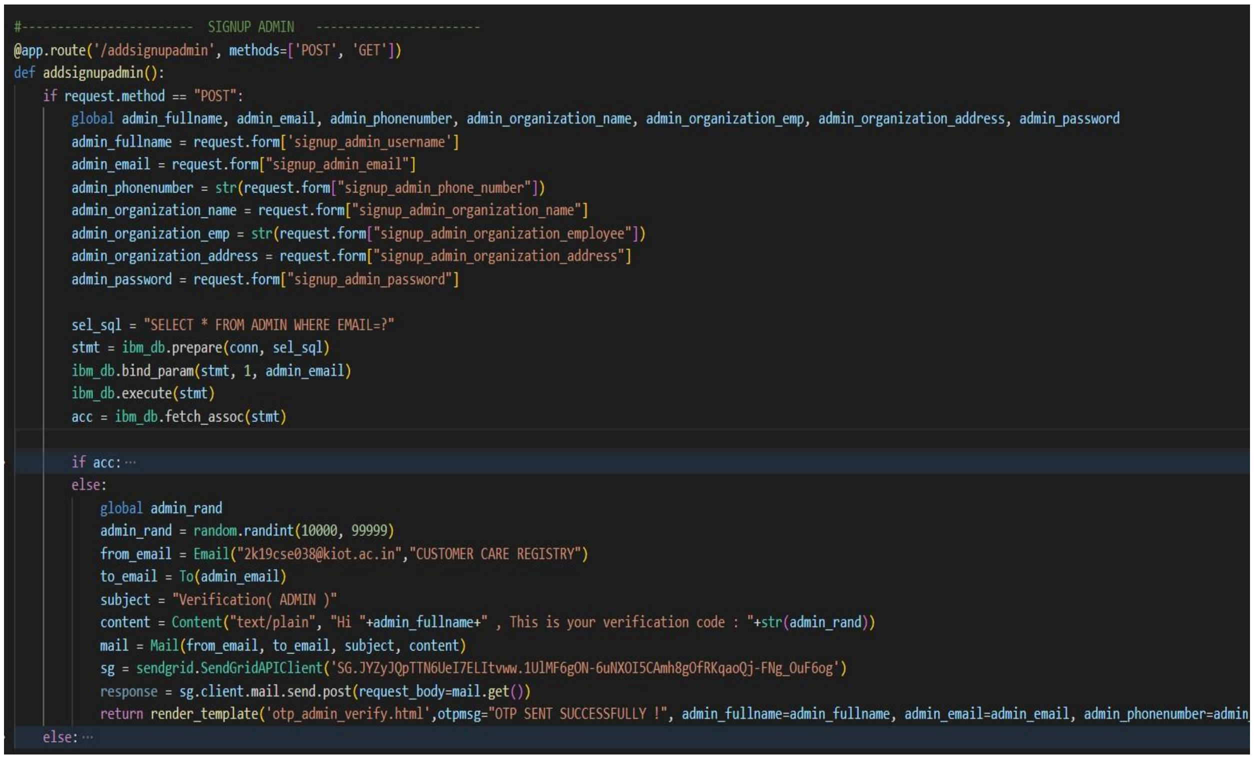
```
+ def index():
+     return render_template("login.html")
+
+ @app.route("/signin",methods = ['POST'])
+ def index_signin():
+     sql = "select * from users where email='"+request.form["email"]+"'"
+     stmt = ibm_db.exec_immediate(conn, sql)
+     data = []
+     dictionary = ibm_db.fetch_both(stmt)
+     while dictionary != False:
+         data.append(dictionary)
+         dictionary = ibm_db.fetch_both(stmt)
+     if(data):
+         if(data[0]["PASSWORD"]==request.form["password"]):
+             session["user"]=data[0]["ID"]
+             session["name"]=data[0]["NAME"]
+             session["email"]=data[0]["EMAIL"]
+             return redirect("/user")
+         else:
+             return redirect("/")
```



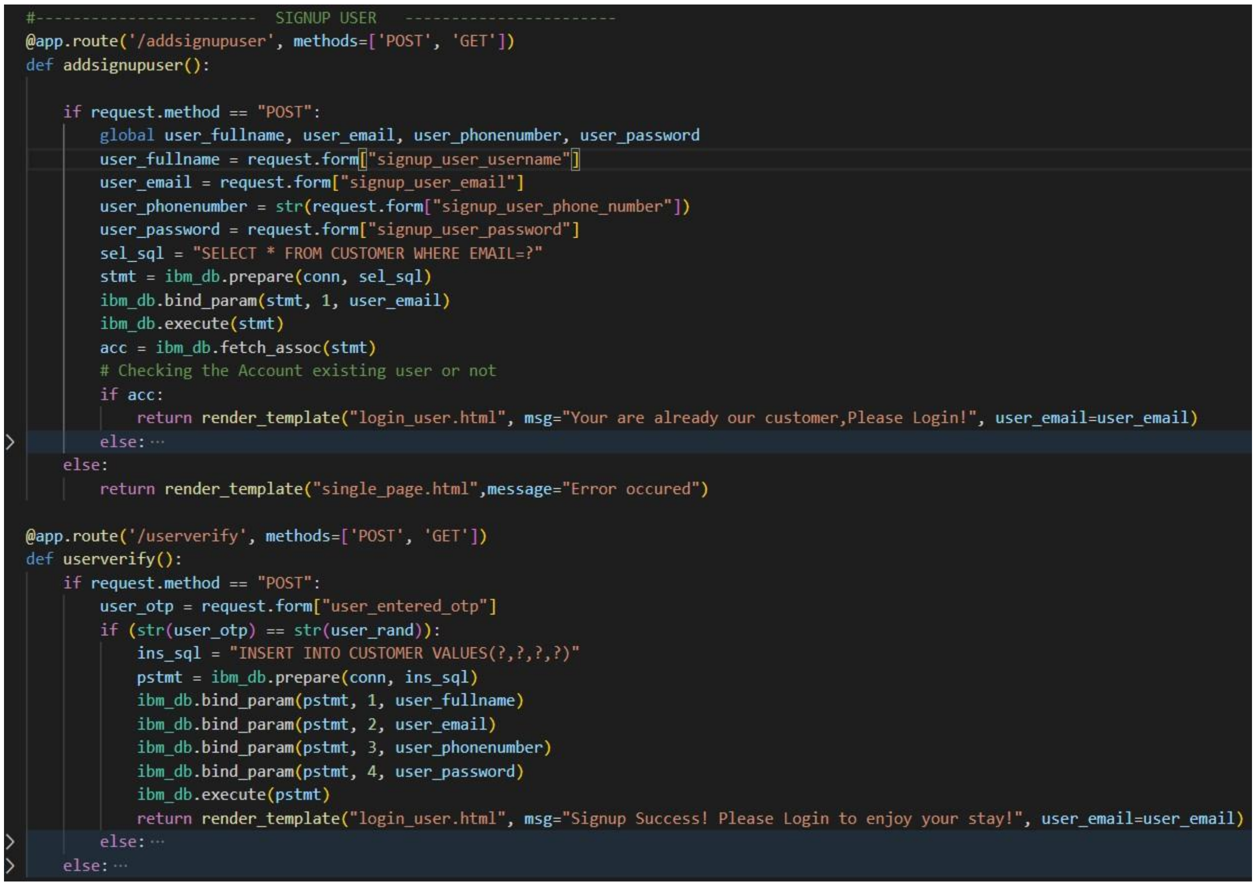
```
+             return redirect("/user")
+         else:
+             return redirect("/")
+     else:
+         return redirect("/")
+
+ @app.route("/register",methods = ['GET'])
+ def index_register():
+     return render_template("register.html")
+
+ @app.route("/signup",methods = ['POST'])
+ def index_signup():
+     sql = "INSERT INTO users (name , email , password,mob)values('"+request.form["name"]+"','"+request.form["email"]+"','"+reques
+     stmt = ibm_db.exec_immediate(conn, sql)
+     return redirect("/")
+
+
+
+
+ if __name__ == '__main__':
+    app.run()
```



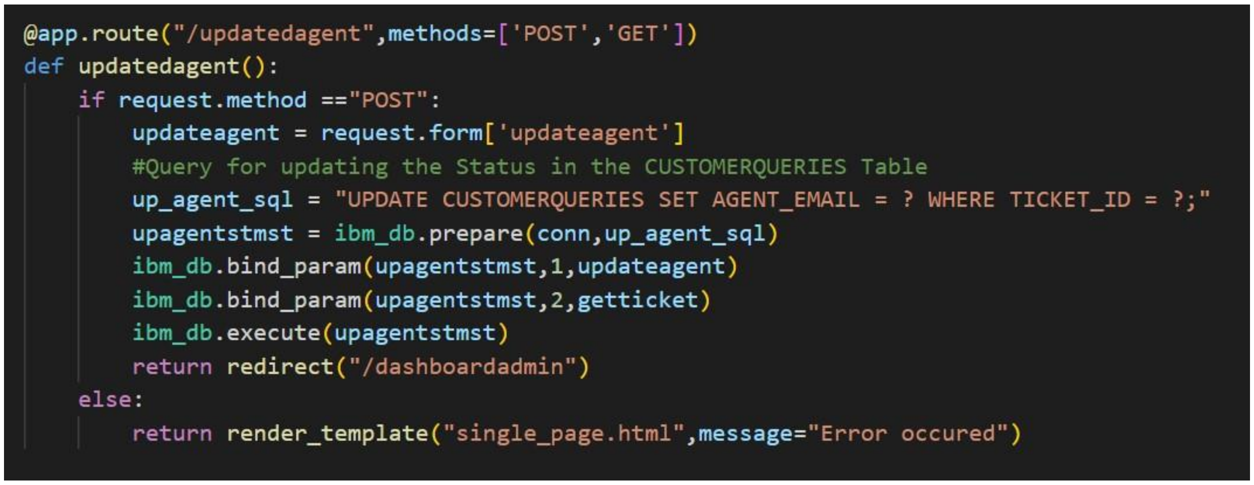
```
#------------------------  DASHBOARD ADMIN REPORT -----------------------

@app.route('/report')
def report():

    logintype = session.get("login_type")
    loginemail = session.get("login_email")

    if (logintype==None or loginemail==None) :
        return render_template('report.html',logintype="none",loginemail="none")
    else:
        #For Open Status
        list= []
        ostat = 'open'
        sel_sql = "SELECT COUNT(STATUS) FROM CUSTOMERQUERIES WHERE STATUS=?"
        stmt = ibm_db.prepare(conn,sel_sql)
        ibm_db.bind_param(stmt,1,ostat)
        ibm_db.execute(stmt)
        query = ibm_db.fetch_both(stmt)
        #Appending the list
        list.append(query[0])

        #For Closed Status
        cstat = 'closed'
        csel_sql = "SELECT COUNT(STATUS) FROM CUSTOMERQUERIES WHERE STATUS=?"
        cstmt = ibm_db.prepare(conn,csel_sql)
        ibm_db.bind_param(cstmt,1,cstat)
        ibm_db.execute(cstmt)
        cquery = ibm_db.fetch_both(cstmt)
        #Appending the list
        list.append(cquery[0])
```



```
#-----------------------  SIGNUP ADMIN   ----------------------
@app.route('/addsignupadmin', methods=['POST', 'GET'])
def addsignupadmin():
    if request.method == "POST":
        global admin_fullname, admin_email, admin_phonenumber, admin_organization_name, admin_organization_emp, admin_organization_address, admin_password
        admin_fullname = request.form['signup_admin_username']
        admin_email = request.form["signup_admin_email"]
        admin_phonenumber = str(request.form["signup_admin_phone_number"])
        admin_organization_name = request.form["signup_admin_organization_name"]
        admin_organization_emp = str(request.form["signup_admin_organization_employee"])
        admin_organization_address = request.form["signup_admin_organization_address"]
        admin_password = request.form["signup_admin_password"]

        sel_sql = "SELECT * FROM ADMIN WHERE EMAIL=?"
        stmt = ibm_db.prepare(conn, sel_sql)
        ibm_db.bind_param(stmt, 1, admin_email)
        ibm_db.execute(stmt)
        acc = ibm_db.fetch_assoc(stmt)

        if acc:···
        else:
            global admin_rand
            admin_rand = random.randint(10000, 99999)
            from_email = Email("2k19cse038@kiot.ac.in","CUSTOMER CARE REGISTRY")
            to_email = To(admin_email)
            subject = "Verification( ADMIN )"
            content = Content("text/plain", "Hi "+admin_fullname+" , This is your verification code : "+str(admin_rand))
            mail = Mail(from_email, to_email, subject, content)
            sg = sendgrid.SendGridAPIClient('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')
            response = sg.client.mail.send.post(request_body=mail.get())
            return render_template('otp_admin_verify.html',otpmsg="OTP SENT SUCCESSFULLY !", admin_fullname=admin_fullname, admin_email=admin_email, admin_phonenumber=admin
    else:···
```

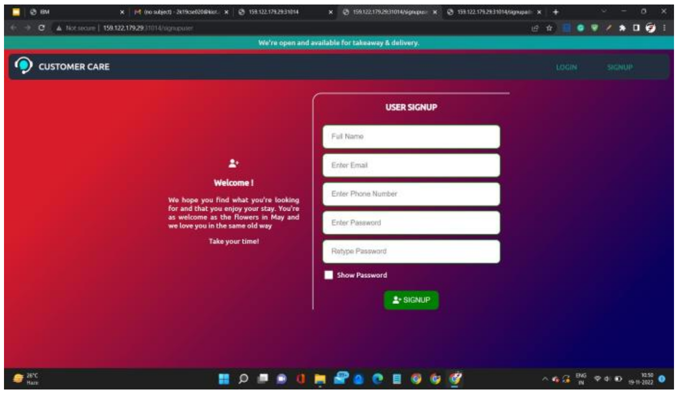


```
#-----------------------  SIGNUP USER   ----------------------
@app.route('/addsignupuser', methods=['POST', 'GET'])
def addsignupuser():

    if request.method == "POST":
        global user_fullname, user_email, user_phonenumber, user_password
        user_fullname = request.form["signup_user_username"]
        user_email = request.form["signup_user_email"]
        user_phonenumber = str(request.form["signup_user_phone_number"])
        user_password = request.form["signup_user_password"]
        sel_sql = "SELECT * FROM CUSTOMER WHERE EMAIL=?"
        stmt = ibm_db.prepare(conn, sel_sql)
        ibm_db.bind_param(stmt, 1, user_email)
        ibm_db.execute(stmt)
        acc = ibm_db.fetch_assoc(stmt)
        # Checking the Account existing user or not
        if acc:
            return render_template("login_user.html", msg="Your are already our customer,Please Login!", user_email=user_email)
        else:···
    else:
        return render_template("single_page.html",message="Error occured")

@app.route('/userverify', methods=['POST', 'GET'])
def userverify():
    if request.method == "POST":
        user_otp = request.form["user_entered_otp"]
        if (str(user_otp) == str(user_rand)):
            ins_sql = "INSERT INTO CUSTOMER VALUES(?,?,?,?)"
            pstmt = ibm_db.prepare(conn, ins_sql)
            ibm_db.bind_param(pstmt, 1, user_fullname)
            ibm_db.bind_param(pstmt, 2, user_email)
            ibm_db.bind_param(pstmt, 3, user_phonenumber)
            ibm_db.bind_param(pstmt, 4, user_password)
            ibm_db.execute(pstmt)
            return render_template("login_user.html", msg="Signup Success! Please Login to enjoy your stay!", user_email=user_email)
        else:···
    else:···
```



```python
@app.route("/updatedagent",methods=['POST','GET'])
def updatedagent():
    if request.method =="POST":
        updateagent = request.form['updateagent']
        #Query for updating the Status in the CUSTOMERQUERIES Table
        up_agent_sql = "UPDATE CUSTOMERQUERIES SET AGENT_EMAIL = ? WHERE TICKET_ID = ?;"
        upagentstmst = ibm_db.prepare(conn,up_agent_sql)
        ibm_db.bind_param(upagentstmst,1,updateagent)
        ibm_db.bind_param(upagentstmst,2,getticket)
        ibm_db.execute(upagentstmst)
        return redirect("/dashboardadmin")
    else:
        return render_template("single_page.html",message="Error occured")
```

13.2 Screenshots

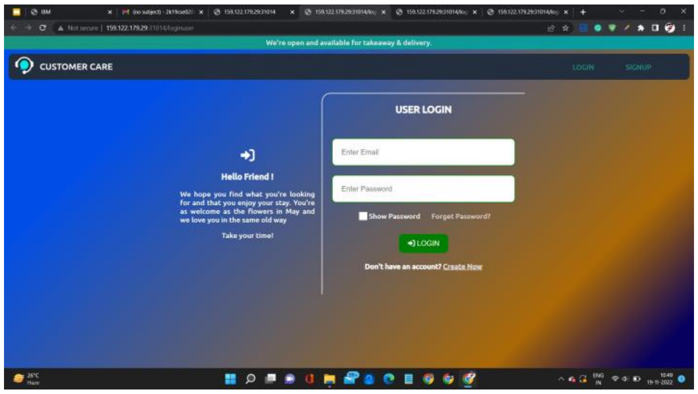


We're open and available for takeaway & delivery.

Welcome !

We hope you find what you're looking for and that you enjoy your stay. You're as welcome as the flowers in May and we love you in the same old way

Take your time!



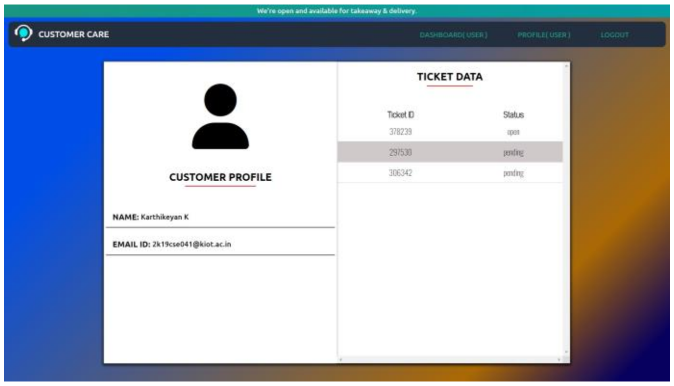
We're open and available for takeaway & delivery.

Hello Friend !

We hope you find what you're looking for and that you enjoy your stay. You're as welcome as the flowers in May and we love you in the same old way

Take your time!

Don't have an account? Create Now



We're open and available for takeaway & delivery.

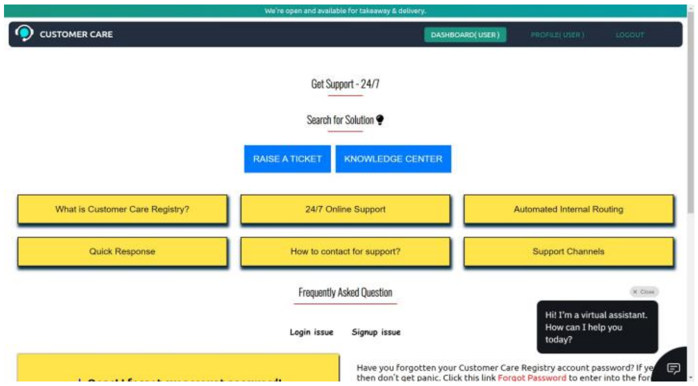
CUSTOMER PROFILE

NAME: Karthikeyan K

EMAIL ID: 2k19cse041@kiot.ac.in

TICKET DATA

Ticket ID	Status
378239	open
297530	pending
306342	pending



We're open and available for takeaway & delivery.

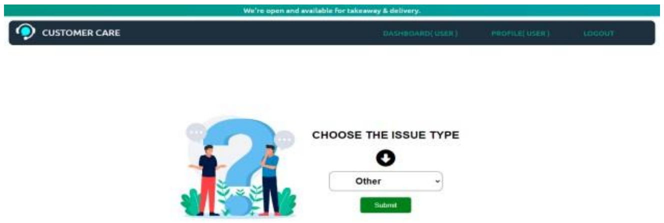
Get Support - 24/7

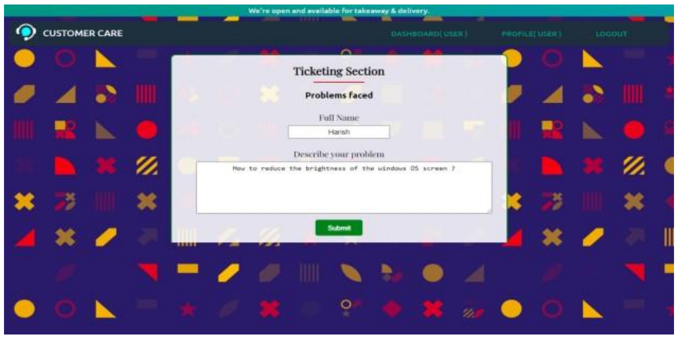
Search for Solution

Frequently Asked Question

Hi! I'm a virtual assistant. How can I help you today?

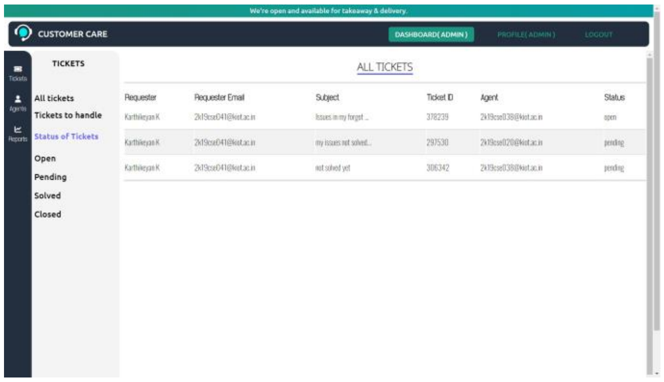
Have you forgotten your Customer Care Registry account password? If ye
then don't get panic. Click this link Forgot Password to enter into the for





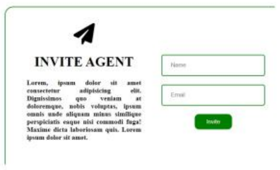
Harish

How to reduce the brightness of the windows OS screen ?

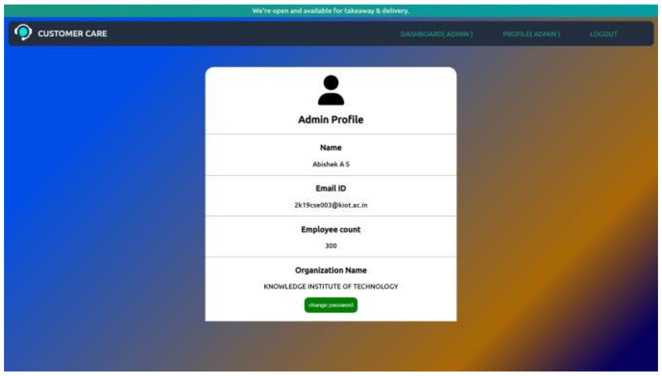


ALL TICKETS

Requester	Requester Email	Subject	Ticket ID	Agent	Status
Karthikeyan K	2k19cse041@kiot.ac.in	Issues in my forgot ...	378239	2k19cse038@kiot.ac.in	open
Karthikeyan K	2k19cse041@kiot.ac.in	my issues not solved...	297530	2k19cse020@kiot.ac.in	pending
Karthikeyan K	2k19cse041@kiot.ac.in	not solved yet	306342	2k19cse038@kiot.ac.in	pending



Lorem, ipsum dolor sit amet consectetur adipisicing elit. Dignissimos quo veniam at doloremque, nobis voluptas, ipsum omnis unde aliquam minus similique perspiciatis eaque nisi commodi fuga! Maxime dicta laboriosam quis. Lorem ipsum dolor sit amet.

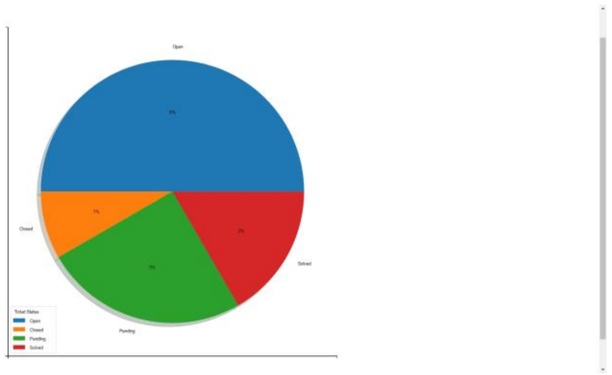


Name
Abishek A S

Email ID
2k19cse003@kiot.ac.in

Employee count
300

Organization Name
KNOWLEDGE INSTITUTE OF TECHNOLOGY


Open
Closed
Pending
Solved
Ticket Status
Open
Closed
Pending
Solved