CUSTOMER CARE REGISTRY

TEAM ID: PNT2022TMID47962

TEAM MEMBERS:

Pandi Selvi R (TEAM LEAD)	**912219104023**
Chithra Banu M	**912219104005**
Fathima Kani M	**912219104011**
Rajarajeshwari S	**912219104025**
Sneha S	**912219104033**

- PROJECT OVERVIEW
-

• INTRODUCTION

Customer care and customer service together help create a positive customer experience, or the overall impression a person has when interacting with your company. Both are vital, but there are subtle differences in how they are implemented. High-quality customer care is proactive. The needs of customers throughout the buyer's journey are anticipated, making customers feel supported. That, in turn, helps create an emotional connection between the customer and the

company. Customer service is reactive. Here, the focus is on helping customers solve problems or answer questions before purchase, either in a self-serve fashion or via the customer support team. 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. If a company neglects customer care, it can negatively impact the customer service experience. For example, when a website chatbot can't provide key information about a product, customers are more likely to get frustrated and reach out to a customer service agent for help. Consumer expectations are extremely high, putting increased pressure on companies to improve their customer relationships. This can lead to lost information when the same person reaches out via multiple channels. When a customer service agent doesn't know the whole story and the customer has to repeatedly share the problem, it leaves both people frustrated. They can register for an account. After the login, they can create a complaint with a description of the problem they are facing. Each user will be assigned an agent. They can view the status of their complaint.

- Customers get the insights they need to make an informed purchase.
- Customer satisfaction can increase and customer loyalty can improve.
- Customer service agents spend less time on routine tasks and answering commonly askedquestions, enabling agents to do more meaningful task.

• PURPOSE

There are two sides to customer service objectives. First, there are the goals and KPIs customer service teams attempt to achieve. Then, there's customer service resume objectives. It's important to understand the connection between the two: Writing a strong customer serviceresume objective starts with understanding the objectives of the field and its depth and possibilities. To provide insight into both levels of customer service objectives. The prime objective of customer service is to answer customer questions quickly and effectively, resolve issues with empathy and care, document pain points to share with internal teams, nurture relationships, and improve brand credibility. Great customer service can make people loyal to your brand, products, and services for years to come.

A strong customer service resume objective underscores your skills and experiences in contributing to customer service's overall goals and objectives.

Meeting key customer service KPIs doesn’t just involve answering phones and emails. It's a whole world of solutions development, intuition, empathy, brand management, time management-and the soft skills thathelp connect people and create trust. I guide my team toward giving the best service possible. Sometimes, we’re not delivering good news. But the objective is to do that with compassion and empathy and in a way that we give the customer constructive next steps to move forward. We also know that as a newer, younger brand, customers may be wary of our credibility. It usually takes a few consistently excellent customer experiences to feel connected and loyal to the brand. That awesome experience starts from the very first touchpoint, whether it be web, email, brick and mortar, or Instagram, and carries through to when they're wearing our product

- EXISTING PROBLEM
-

• LITERATURE SURVEY

A strong customer problem statement should provide a detailed description of your customer’s current situation. Consider how they feel, the financial and emotional impact oftheir current situation, and any other important details about their thoughts or feelings.

Customer Satisfaction is an attitude that is decided based on the experience obtained.Satisfaction is an assessment of the characteristics or privileges of a product or service, or theproduct itself, that provides a level of consumer pleasure with regard to meeting consumer consumption needs.

Customer Satisfaction is the customer's response to the evaluation of perception of differences in initial expectations prior to purchase (or other performance standards) and theactual performance of the product as perceived after wearing or consuming the product in question.

The level of complaint is how high the complaint or delivery of dissatisfaction, discomfort, irritation, and anger over the service of the service or product. The dimension or indicator of complaint level is the high level of complaint.

Product Quality affects Customer Satisfaction, where the dimensions or indicatorsof Product Quality are quality products, in accordance with the price

offered, and ease of use affects the dimensions or indicators of Customer Satisfaction in relation to subscription decisions.

- **REFERENCES**

1] Uddi Executive Overview: Enabling Service Oriented Architecture, 2004 Oct.

- "Web Services Architecture", http://www.w3.org/TR/ws-arch/. Date Accessed:11/02/2004.
- UDDI V3 Specification. http://uddi.org/pubs/uddi-v3.00-published-20020719.html.19/07/2002.
- Christensen E, Curbera F, Meredith G, Weerawarana S, Web Services DescriptionLanguage (WSDL) 1.1, W3C Note, 2001
- Ali A S, Rana O F, Ali R A, Walker D W, UDDIe: an extended registry for Webservices, SAINT-w'03 Proceedings of the 2003 Symposium on Applications and the Internet Workshops, 2003 Jan, pp .85-89.
- Tsai W T, Paul R, Cao Z, Yu L, Saimi A, Xiao B, Verification of Web Services Usingan Enhanced UDDI Server ,Proceedings of The Eighth IEEE International Workshop on Object-Oriented Real-Time Dependable Systems ,2003 Jan, pp 131-38.
- Liu J, Gu N, Zong Y Ding Z, Zhang S, Zhang Q Service Registration and Discovery ina Domain-Oriented UDDI Registry ,

Proceedings of the 2005 The Fifth International Conference on Computer and InformationTechnology (CIT'05),2005, pp .276-83.

- Jian W, Zhaohui W, Similarity-based Web Service Matchmaking Proceedings of the2005 IEEE International Conference on Services Computing (SCC'05), 2005.
- Tretola G, Zimeo E , Structure Matching for Enhancing UDDI Queries Results , IEEEInternational Conference on Service-Oriented Computing and Applications(SOCA'07),2007.
- Ayorak E, Bener a, Super Peer Web Service Discovery Architecture, IEEE Proceedings - International Conference on Data Engineering, 2007 pp .287-94.
- Libing He W, Wu Y, Jianqun D A novel interoperable model of distributed UDDI -proceedings of the 2008 IEEE International Conference on Networking, Architecture, and Storage - IEEE NAS 2008 Jun, pp .153-54.

• Nawaz F, Qadir K, Ahmad H F, and SEMREG-Pro: A Semantic based Registry forProactive Web Service Discovery using Publish Subscribe Model, Fourth International Conference on Semantics, Knowledge and Grid, IEEEXplore, 2008 Dec, pp .301-08.
• LIANG1 Q, CHUNG2 J A Federated UDDI System for Concurrent Access to ServiceData, IEEE International Conference on e-Business Engineering, ICEBE'08 - Workshops: AiR'08, EM2I'08, SOAIC'08, SOKM'08,BIMA'08, and DKEEE'08, 2008 Oct, pp .71-78.
• Vandan T, Nirmal D, InderjeetGarg S, Garg N, Soni P. An Improved Discovery Enginefor Efficient and Intelligent discovery of Web Service with publication facility, SERVICES 2009 - 5th 2009 World Congress on Services,2009, pp .63-70.
• Ma C, Song M, Xu K, Zhang X.Web Service Discovery Research and ImplementationBased on Semantic Search Engine, IEEE 2nd Symposium on Web Society, Beijing. 2010 Aug 16-17, pp .672-77.
• Rajendran T.Balasubramanie P, Flexible and Intelligent Architecture for Quality BasedWeb Service Discovery with an Agent- Based Approach, International Conference on Communication and Computational Intelligence(INCOCCI), Erode. 2010 Dec 27-29, pp .617-22.
• Johnsen F T, Hafsøe T, Eggen A, Griwodz C, Halvorsen P, Web Services Discoveryacross Heterogeneous Military Networks, IEEE Communications Magazine, 2010 Oct, 48(10), pp. 84-90.
• Ourania H, Georgios B, Mara N, Dimosthenis A, "A Specialized Search Engine for WebService Discovery" IEEE 19th International Conference on Web Services,USA. 2012 June 24-29, pp. 448-55.
• Raj R J R, Sasipraba T., Web Service Recommendation Framework Using Qos BasedDiscovery and Ranking Process 3rd

International Conference on Advanced Computing, ICoAC. Chennai. 2011Dec 14-16, pp .371-77.
• Ren X, Hou R, Extend UDDI Using Ontology for Automated Service Composition 2ndInternational Conferences on Mechanic Automation and Control Engineering, MACE, 2011 July, pp .298-301.

• PROBLEM STATEMENT DEFINITION

A customer problem statement outlines problems that your customers face. It helps youfigure out how your product or service will solve this problem for them.

The statement helps you understand the experience you want to offer your customers. Itcan also help you understand a new audience when creating a new product or service.

A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceiveyour product or service.

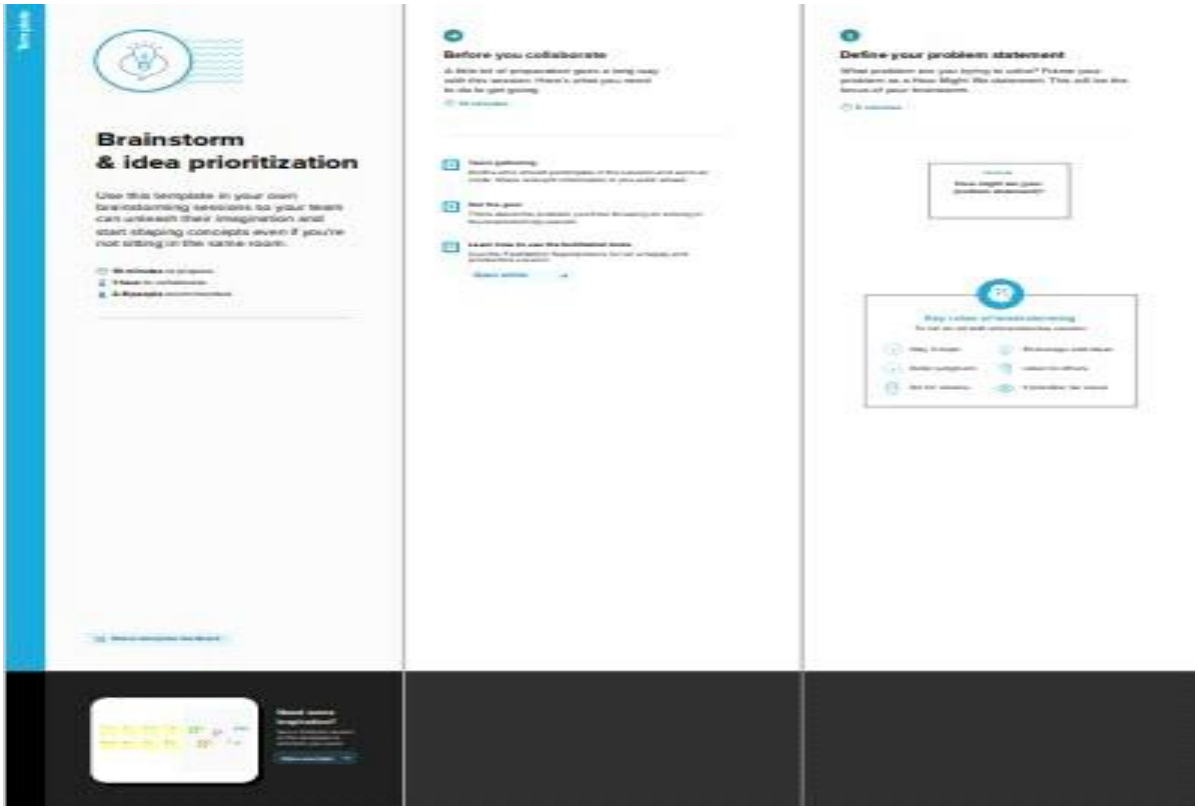
A Customer Problem Statement is a detailed description of an issue that needs to be addressed. This document thoroughly elaborates on the problem that your product or your service solves for your particular customers. It takes into consideration your customer's uniquepain points and how your product goals about solving their situation. A customer problem statement helps you and your team understand the detailed experience you are attempting to transform by analyzing and empathizing with your customers.

The customer problem statement is a critical component of a project. It benefits everyone involved with the project because it helps people understand why they're working onthe project, providing clarity on the reasons behind the product or service. Team members willconsider how your customers will be impacted by your project, what their thoughts and needs are, and thus come up with truly effective and valuable ways to improve their experience.

- **IDEATION & PROPOSED SOLUTION**

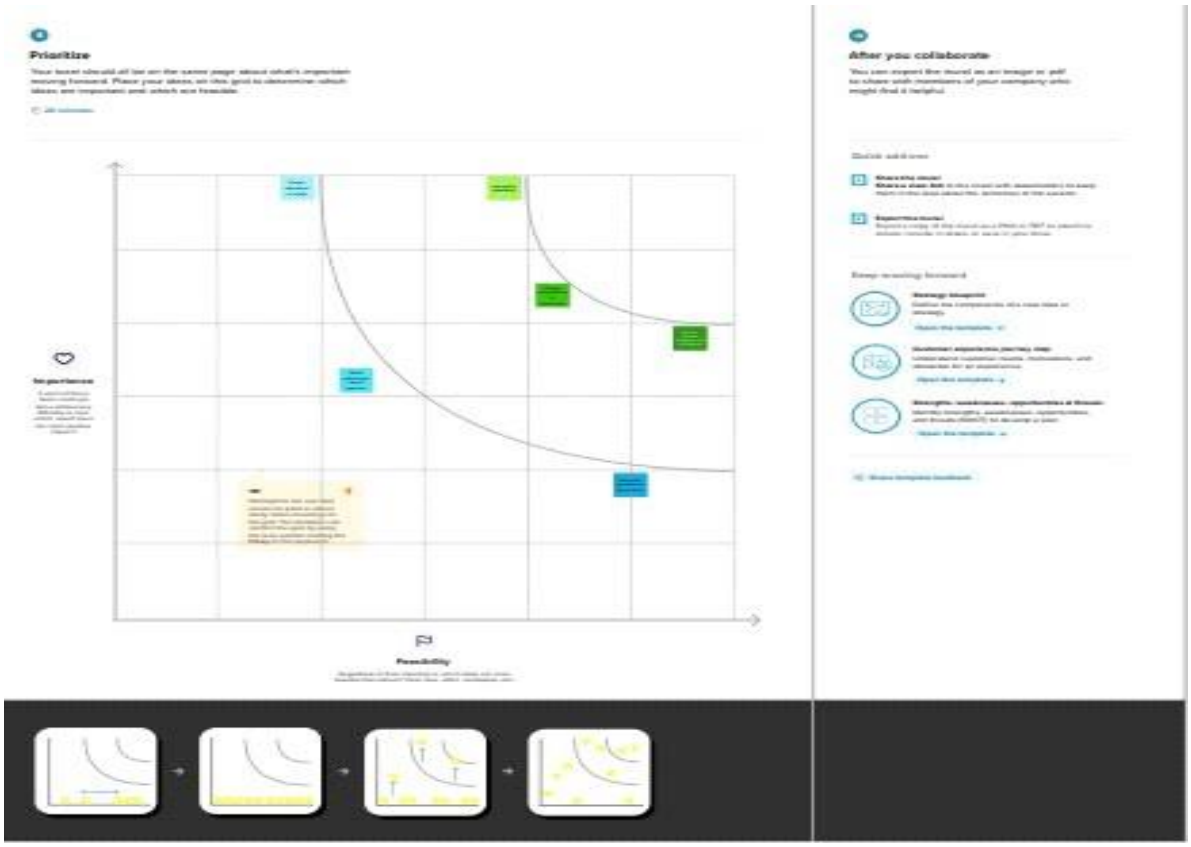
- **EMPATHY MAP CANVAS**

- **IDEATION & BRAINSTORMING**




Brainstorm


Group ideas
CUSTOMER'S EXPECTATIONS
SECURITY
SERVICES



- **PROPOSED SOLUTION**

S.NO.	PARAMETER	DESCRIPTION
01	Problem Statement (Problem to be solved)	To solve customer issues using Cloud Application Development.
02	Idea / Solution description	Assigned Agent routing can be solved by directly routing to the specific agent about theissue 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.
03	Novelty / Uniqueness	Assigned Agent Routing, Automated Ticket Closure, Status Shown to the Customer, and Backup data in case of failures.

S.NO.	PARAMETER	DESCRIPTION
04	Social Impact / Customer Satisfaction	Customer Satisfaction, Customer can track their status and Easy agent communication.
05	Business Model (Revenue Model)	● Key Partners are Third-party applications, agents, and customers. ● Activities held as Customer Service, System Maintenance. ● Key Resources support Engineers, Multi-channel. ● Customer Relationship have 24/7 Email Support, Knowledge-based channel. ● Cost Structure expresses Cloud Platform, Offices

S.NO.	PARAMETER	DESCRIPTION
06	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 gruntwork and more time on actually resolving critical customer issues

- **PROBLEM SOLUTION FIT**

PROJECT DESIGN PHASE –I (PROBLEM-SOLUTION FIT)

Problem-Solution fit canvas 2.0

Problem-Solution Fit

4

- **REQUIREMENT ANALYSIS**

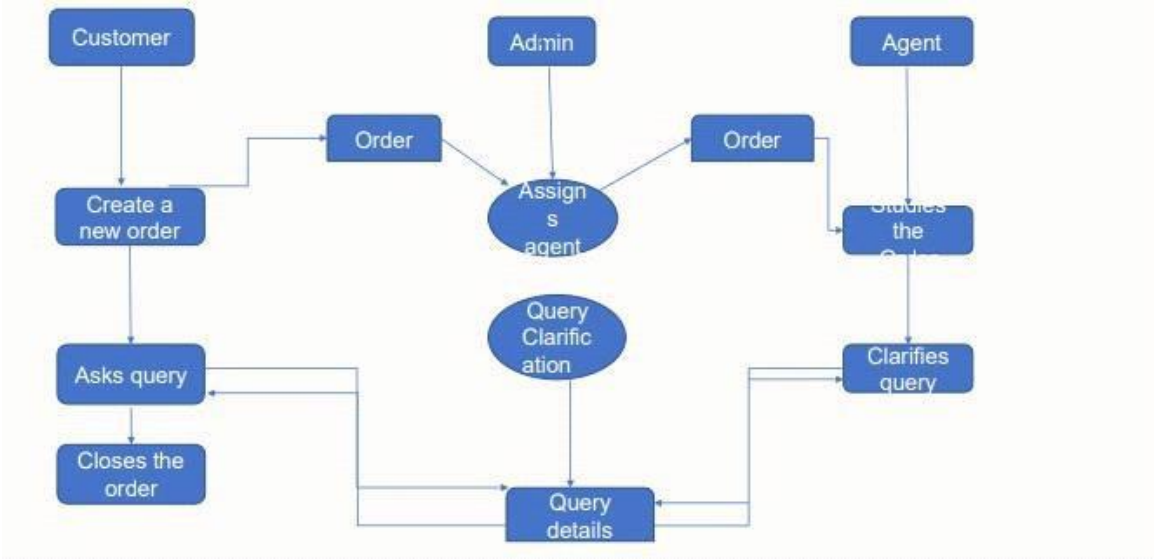
- **FUNCTIONAL REQUIREMENT**

FR No	Functional Requirement(Epic)	Sub Requirement(Story/ Sub-Task)
1	User Registration	Registration through Form Registration through Gmail Registration through Google
2	User Confirmation	Confirmation via Email Confirmation via OTP
3	User Login	Login via Google Login with Email id and Password
4	Admin Login	Login via Google Login with Email id and Password
5	Query Form	Description of the issues Contact information
6	E-mail	Login alertness
7	Feedback	Customer feedback

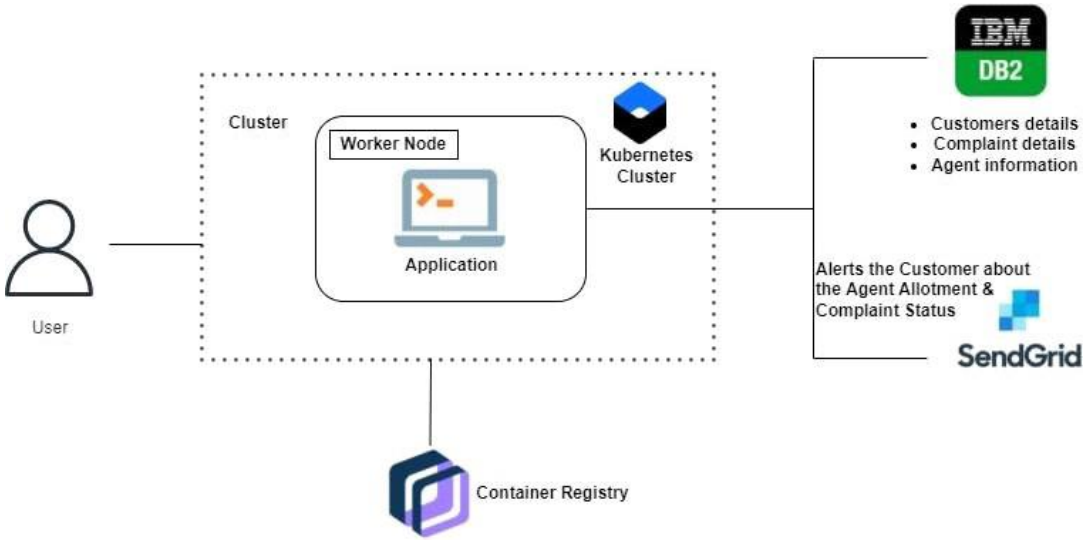
- NON-FUNCTIONAL REQUIREMENT

FR No	Non-Functional Requirement	Description
1	Usability	To provide the solution to the problem
2	Security	Track of login authentication
3	Reliability	Tracking of decade status through email
4	Performance	Effective development of web application
5	Availability	24/7 service
6	Scalability	Agents scalability as per the number of customers

- **DATA FLOW DIAGRAMS**
-
- **PROJECT DESIGN**



- **SOLUTION AND TECHNICAL ARCHITECTURE**



- **USER STORIES**

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile 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 orders raised by me.	I get all the info needed in my dashboard.	Low	Sprint-2
	Order creation	USN-4	As a customer, I can place my order with the detailed description of my query	I can ask my query	Medium	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 incase I forgot my old password.	I get access to my account again	Medium	Sprint-4
	Order details	USN-7	As a Customer ,I can see the current stats of order.	I get abetter 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 the order details assigned to me by 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/er dobuts	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 (Mobile user)	Login	USN-1	As a admin, I can login to the appliaction by entering Correct email and password	I can access my account/dashboard	High	Sprint-1
	Dashboard	USN-2	As an admin I can see all the orders raised in the entire system and lot more	I can assign agents by seeing those order.	High	Sprint-1
	Agent creation	USN-3	As an admin I can create an agent for clarifying the customers queries	I can create agents.	High	Sprint-2
	Assignment agent	USN-4	As an admin I can assign an agent for each order created by the customer.	Enable agent to clarify the queries.	High	Sprint-1
	Forgot password	USN-5	As an admin I can reset my password by this option in case I forgot my old password.	I get access to my account.	High	Sprint-1

- **PROJECT PLANNING & SCHEDULE**

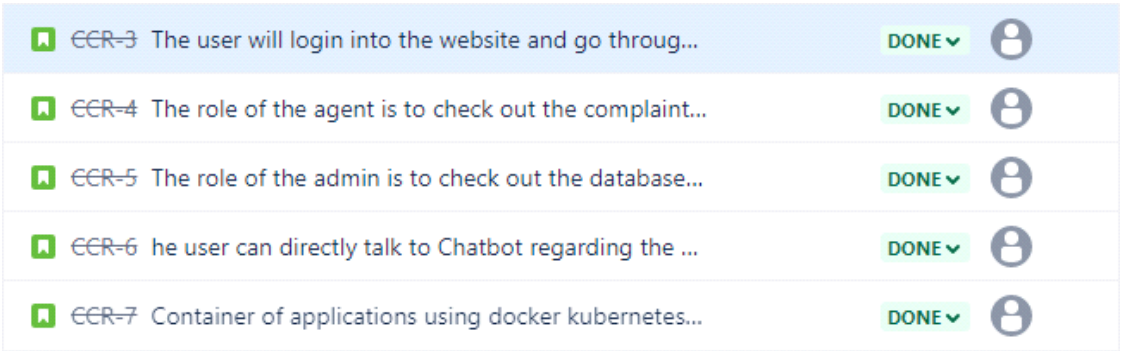
- **SPRINT PLANNING &ESTIMATION**

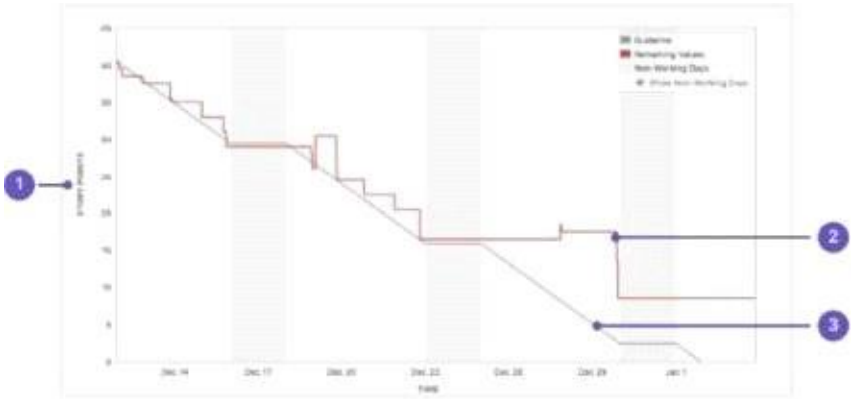
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	User Panel	USN-1	The user will login into the website and go through the services available on the webpage.	20	Medium	Shyam R Sujindhar C
Sprint-2	Agent Panel	USN-2	The role of the agent is to check out the complaint tickets and to contact the user and solve the complaint they raise.	20	High	Shankar P Jayasurya C
Sprint-3	Admin Panel	USN-3	The role of the admin is to check out the database about the availability and have a track of all the things that the users are going to experience and manage the agent and complaint tickets.	20	High	Shankar P
Sprint-4	Chat Bot	USN-4	The user can directly talk to Chatbot regarding the services. Get the recommendations based on information provided by the user.	20	High	Jayasurya S
Sprint-5	Final Delivery	USN-5	Container of applications using docker kubernetes and deployment the application. Create the documentation and final submit the application	20	High	Sujindhar C Shyam R Shankar P Jayasurya S

- **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	10	6 Days	24 Oct 2022	29 Oct 2022	10	29 Oct 2022
Sprint-2	6	6 Days	31 Oct 2022	05 Nov 2022	7	05 Nov 2022
Sprint-3	6	3 Days	07 Nov 2022	09 Nov 2022	6	09 Nov 2022
Sprint-4	5	3 Days	09 Nov 2022	12 Nov 2022	5	12 Nov 2022
Sprint-5	8	6 Days	13 Nov 2022	19 Nov 2022	8	19 Nov 2022

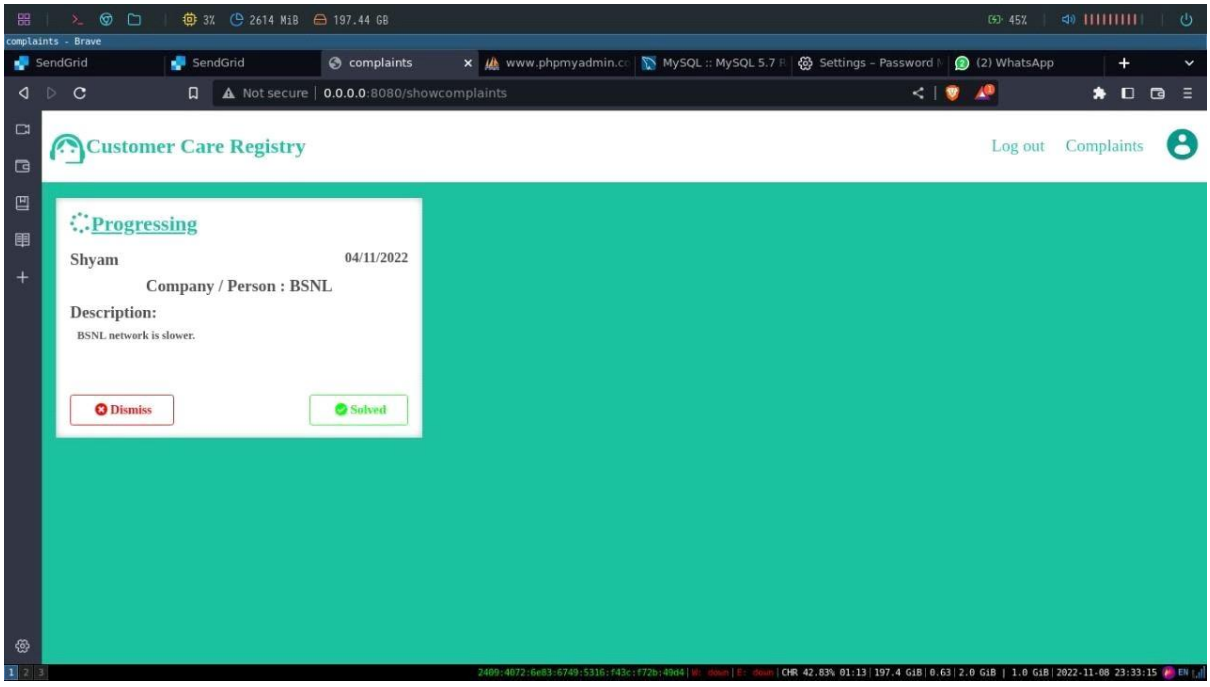
- **REPORTS FROM JIRA**



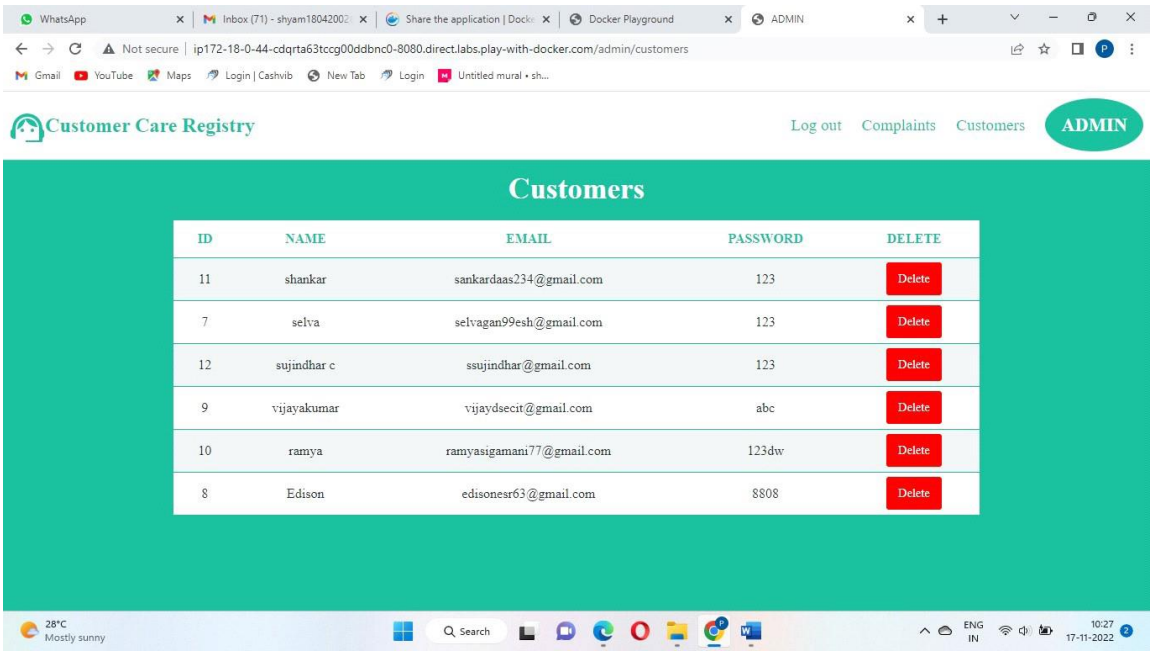


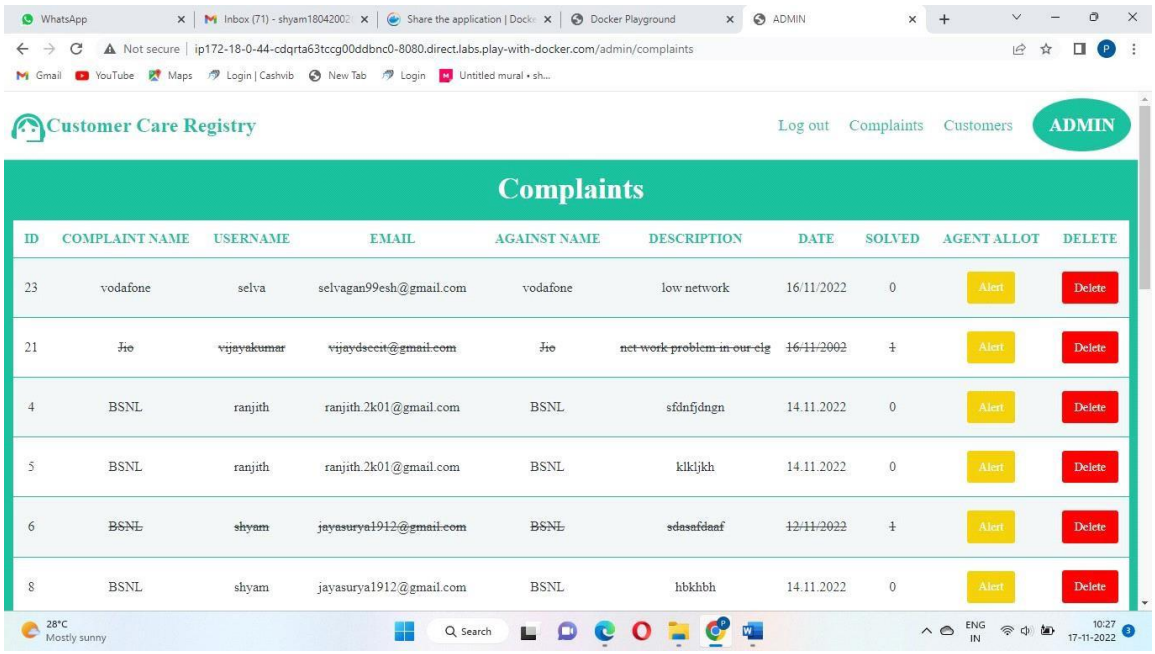
Burndown Graph

- **FEATURE 1**
-
- **CODING & SOLUTIONING**



- **FEATURE 2**





- **DATABASE SCHEMA**

Table definition

COMPLAINTS

No statistics available.

Name	Data type	Nullable	Length	Scale
ID	INTEGER	N		0
USERNAME	VARCHAR	N	150	0
EMAIL	VARCHAR	N	150	0
AGAINST_PERSON	VARCHAR	N	150	0
DES	VARCHAR	N	150	0

Table definition

COMPLAINTS

No statistics available.

Name	Data type	Nullable	Length	Scale
AGAINST_PERSON	VARCHAR	N	150	0
DES	VARCHAR	N	150	0
DATE	VARCHAR	N	150	0
SOLVED	VARCHAR	N	150	0

Table definition

CUSTOMERDEATILS

No statistics available.

Name	Data type	Nullable	Length	Scale
ID	INTEGER	N		0
USERNAME	VARCHAR	N	150	0
EMAIL	VARCHAR	N	150	0
PASSWRD	VARCHAR	N	150	0

VARCHAR

- **TEST CASES**
-
- **TESTING**

  - **FUNCTIONAL TESTING**

    Functional test can be defined as testing two or more modules together with the intent of finding defects, demonstrating that defects are not present, verifying that the module performs its intended functions as stated in the specification and establishing confidence that a program does what it is supposed to do.

  - **WHITE BOX TESTING:**

    Testing based on an analysis of internal workings and structure of a piece of software. This testing can be done sing the percentage value of load and energy. The tester should know what exactly is done in the internal program. Includes techniques such as Branch Testing and Path Testing. Also known as Structural Testing and Glass Box Testing.

  - **BLACK BOX TESTING:**

Testing without knowledge of the internal workings of the item being tested. Tests are usually functional. This testing can be done by the user who has no knowledgeof how the shortest path is found.

- **USER ACCEPTANCE TESTING**

Acceptance testing can be defined in many ways, but a simple definition is the succeeds when the software functions in a manner that can be reasonable expected by the customer. After the acceptance test has been conducted, one of the two possible conditions exists. This is to fine whether the inputs are accepted by the database or other validations. For example accept only numbers in the numeric field, date format data in the date field. Also the null check for the not null fields. If any error occurs then show the error messages. The function of performance characteristics to specification and is accepted. A deviation from specification is uncovered and a deficiency list is created. User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

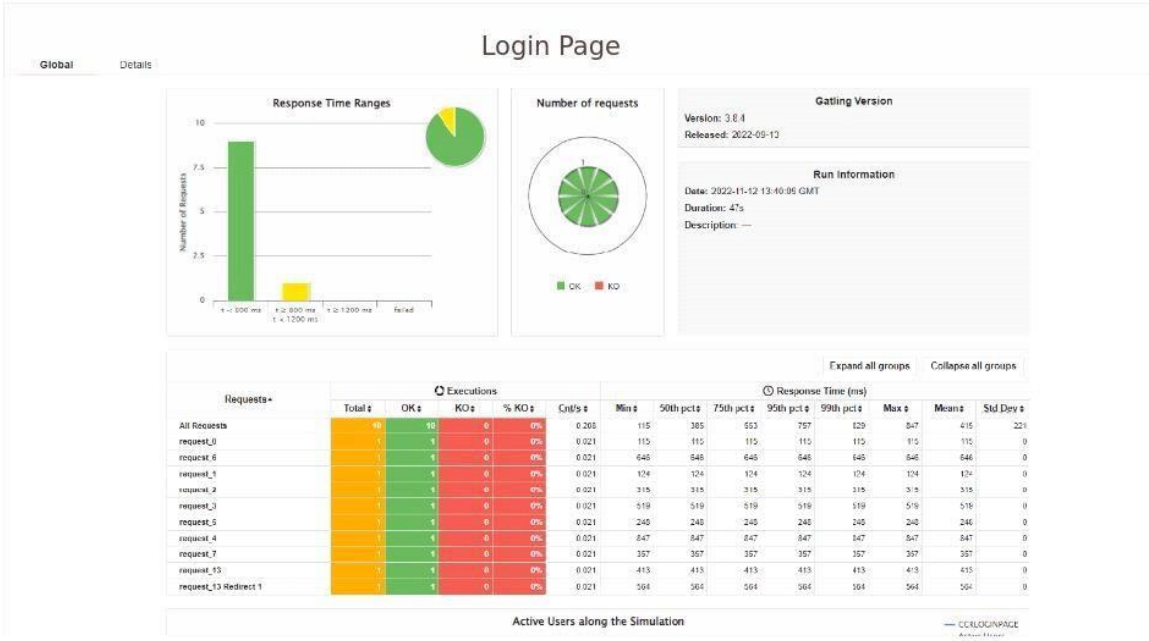
- **TEST RESULTS**

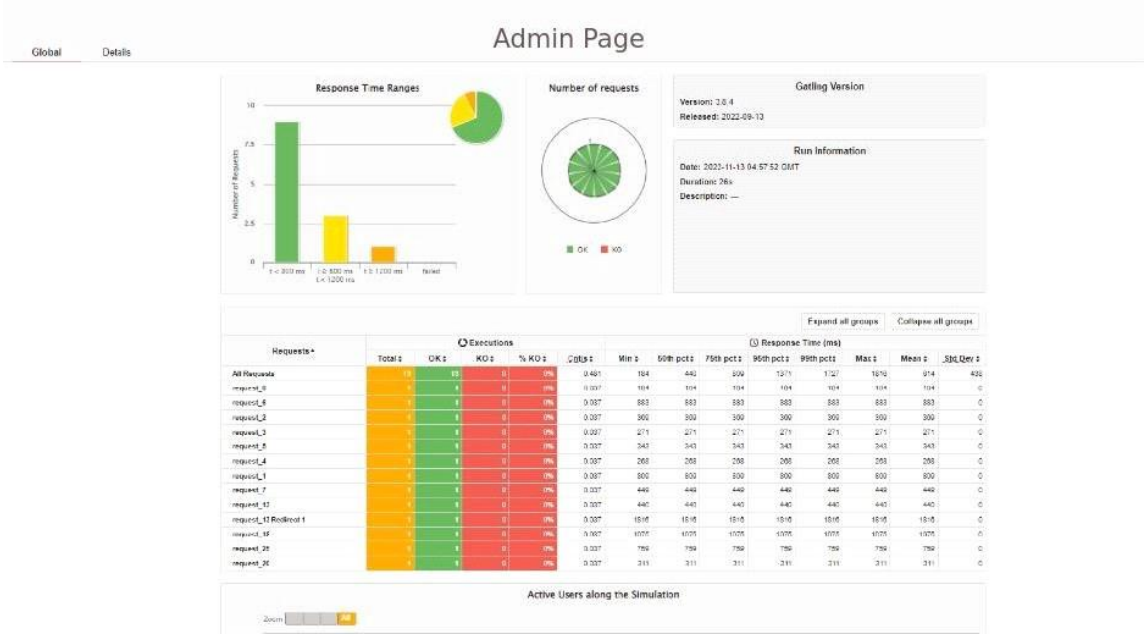
A	B	C	D	E	F	G	H	I
Sprint 1 UI/UX								
Testcase	Type	Component	Scenario	Step to execute	Expected result	Actual result	Status	Executed by
1	UI	login/signup page	clicking on site link	click in site link	login/signup page loads	page load	PASS	Shyam
2	Functional	login/signup page	login in to user acco	enter credentials	login to home page	home page loads	PASS	
3	Functional	login/signup page	signup a user	enter user details	login to home page	home page loads	PASS	
4	Functional	home page	logout of the home p	logout the user	back to login page	login page loads	PASS	
5	Functional	login/signup page	login with unregister	redirect to signup pa	back to signup page	signup page loads	PASS	
6	Functional	login/signup page	signup a existing em	use a existing user (	back to sign up page	signup page loads	PASS	
7	Functional	wrong password	sign in with wrong pa	login with wrong pas	back to sign up page	signup page loads	PASS	
8	Functional	wrong email	signin with wring em	login with wrong em	back to signup page	signup page loads	PASS	
Spirnt 2 db2								
1	Functional	complaint page	display registerd cor	click on complaint ta	complaint list down	complaint list dow	PASS	
2	Functional	complaint page	clicking on solve	click on solve button	solve the complaint	complete the com	PASS	
3	Functional	complaint page	click on the dismiss	click dismiss button	delete the complaint	deletes the compl	PASS	
4	Functional	complaint page	fill up the complaint f	click submit	create the complaint	create a new com	PASS	

9.1 PERFORMANCE METRICES

-

- **RESULTS**


Login Page
Global
Details
Response Time Ranges
Number of requests
Gatling Version
Version: 3.8.4
Released: 2022-09-13
Run Information
Date: 2022-11-12 13:40:06 GMT
Duration: 47s
Description: —
Expand all groups
Collapse all groups
Requests
Executions
Response Time (ms)
Total
OK
KO
% KO
Cnt/s
Min
50th pct
75th pct
95th pct
99th pct
Max
Mean
Std Dev
All Requests
request_0
request_6
request_1
request_2
request_3
request_5
request_4
request_7
request_13
request_13 Redirect 1
Active Users along the Simulation


Admin Page
Global
Details
Response Time Ranges
Number of requests
Gatling Version
Version: 3.8.4
Released: 2022-09-13
Run Information
Date: 2022-11-13 04:57:52 GMT
Duration: 26s
Description: —
Expand all groups
Collapse all groups
Requests
Executions
Response Time (ms)
Total
OK
KO
% KO
Cnt/s
Min
50th pct
75th pct
95th pct
99th pct
Max
Mean
Std Dev
All Requests
Active Users along the Simulation

Advantage

-
- **ADVANTAGES &DISADVANTAGES**

□ Flow sheet is a powerful tool to monitor clinical data and track trends
□ Provides a dashboard of who needs what
□ Provides total population data reporting with no chart abstraction

- Generates revenue (it shows when services are needed)
- Provides outreach information at fingertips
- Improves team-based care
- Smaller software package than EHRs
- Creating loyal customers through good customer service can provide businesseswith lucrative long-term relationships.
- Customer loyalty. Loyal customers have many benefits for businesses

Disadvantage

- Disease-specific, not longitudinal
- Does not include information necessary for billing
- Requires hardware, software and maintenance
- Requires data entry and data maintenance
- Parallel documentation system (i.e., some information has to be entered in twosystems)
- Can't stand alone, must have an additional documentation system.
- Experience burnout and stress. Working as a customer service representative requires you to maintain a friendly demeanour at all times, regardless of howcustomers act or how you personally feel

• CONCLUSION

Companies today are modernizing customer care, using advanced AI to ensure a positive customer experience starting from the first interaction and throughout the buyer's journey. To properly manage customer care, companies must understand how they are succeeding and what needs improvement. This requires establishing key performance indicators (KPIs) for customer service and creating a system of gathering metrics across channels. In conclusion, customer care, involves the use of basic ethics and any company who wants to have success and grow, needs to remember, that in order to do so, it must begin with establishing a code of ethics in regards to how each

employee is to handle the dealing with customers. Customers are at the heart of the company and its growth or decline. Customer care involves, the treatment, care, loyalty, trust the employee should extend to the consumer, as well in life. This concept can be applied to so much more than just customer care. People need to treat others with respect and kindness; people should try to take others into consideration when making any decision. If more people were to practice this policy, chances are the world would be a better, more understanding place for all to exist. Thereby, the customer care registry would be far helpful and approachable. It offers easy tracking, recording and notification than any other means.

• FUTURE SCOPE

The current state of customer care registery, in so many companies, lookssomething like this:

- Customer acquisition is prioritised over retention
- Customer service investment projects are sidelined.
- Departmental efficiency is of highest priority.
- Businesses see employees in the customer service department as short-term anddisposable. They are there to fulfil a specific, repetitive, purpose.
- Employees are considered unskilled and leaders hire accordingly.
- New agents view customer service as a ‘last resort’ or ‘short term’ job. Peopleoften see careers in customer support as unambitious.
- Agent training rarely goes beyond product and people skills.

In the next 3-5 years, we expect to see these **future customer care registrytrends**:

- The shift from a primarily 'cost centre' to primarily 'growth centre' worldview.

- The job desk for a customer care registry director will focus more on leadership,innovation, and ability to drive company-wide improvement.
- Customer service will shift to become a strategic partner of marketing, sales, andproduct development. CS will help with direction, project prioritisation, and impact.
- A need for customer service leaders to take a highly strategic seat at the table.They'll need to argue for investment in talent, technology, and innovation.
- A shift in performance metrics. Forget of resolved tickets. In the future, we'll measure performance based on of customers saved from the precipice of churn.
- A career in customer care registry will not be a last resort. Top graduates willprioritise getting an education in strategic customer interaction.
- Focus on ticket deflection will reduce because brands will view each customer interaction as an opportunity to learn, build a relationship, and grow profits. Theydeserve a well-trained, human touch.

  Modern and developing technology enables this future to exist. With newtechnology, administrative tasks will tend toward zero.

- The sole purpose of the customer service is to meet the expectations of the customers so that they are satisfied with the outcome. These services are also available to understand the queries of the customers and ensure that they enjoy acost-effective experience after purchasing any product from the respective company.

SOURCE CODE

-
- **APPENDIX**

```
from flask import
Flask,render_template,request,url_for,session,redirectfr
om flask_mysqldb import MySQL
from sendmail import
sendemail,forget_password_mail,updated_password_mail,solve_mail
imp
ort
json
imp
ort
ibm
_db
imp
ort
re
from random
import randint
from datetime
import date

app = Flask(__name__)
# http://remotemysql.com/
```

```
# dsn_hostname = "b0aebb68-94fa-46ec-a1fc-
1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud"

# dsn_uid = "dmt13873"

# dsn_pwd =

"740yZ1Yq8Uj2E4qm"#

dsn_database = 'bludb'

# dsn_port = 31249

conn =
ibm_db.connect("DATABASE=bludb;HOSTNAME=b0aebb68-94fa-
46ec-a1fc-
1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;
P ORT=3124
9;SECURITY=SSL;SSLServerCertificate=src/DigiCertGlobalRootC
A. crt;UID=d mt13873;PWD=740yZ1Yq8Uj2E4qm","") # type:
ignore

print(conn)

print("connection successful...")


# database configuration

# app.config['MYSQL_HOST'] =

'sql12.freesqldatabase.com'#

app.config['MYSQL_USER'] = 'sql12552843'

# app.config['MYSQL_PASSWORD'] =

'zWIzHmXNi8'# app.config['MYSQL_DB'] =

'sql12552843' app.secret_key = "super secret

key"

# mysql = MySQL(app)
```

```
@a
pp.r
out
e
('/')
def
ho
me(
):
    today = date.today()
    current_date =
    today.strftime('%d/%m/%Y')i
    f "google_token" in session:
        session["current_date"] =
        current_datereturn
        render_template('home.htm
        l')
    if "username" in session:
        session["current_date"] =
        current_datereturn
        render_template('home.htm
        l')
    return render_template('index.html')
```

```
# manually registration
@app.route('/register',methods=[
"POST"])def register():
    if request.method ==
        'POST': name =
        request.form['una
        me']

        mail =
        request.form['m
        ail']pwd =
        request.form['p
        wd']
        cpwd = request.form['confirmpwd']
        if not
            re.match(r'[^@]+@[^@]+\.[^@]+'
            , mail):msg = 'Invalid email
            address !'
            return
        render_template('index.html',signupmsg=ms
        g)if pwd != cpwd:
            msg = 'Please enter correct confirm password'
```

```
        return
    render_template('index.html',signupmsg=ms
    g)# check account is exists or not
    # cursor =
    mysql.connection.cursor()
    rCheckQuery = ''
    result = ibm_db.exec_immediate(conn,f"SELECT * FROM
customerdeatilsWHERE email LIKE '{mail}'")
    # cursor.execute('SELECT * FROM customerdeatils WHERE email
    LIKE
% s',[mail])
    # existing_user =
    cursor.fetchone()#
    cursor.close()
    existing_user =
    ibm_db.fetch_row(result)
    #exits
    if existing_user:
        msg = 'Account already exists please login.'
        return
    render_template('index.html',signupmsg =
    msg)# not exists


    # cursor = mysql.connection.cursor()
```

```
        # cursor.execute('INSERT INTO customerdeatils
VALUES(null,% s,% s,%s)',(name,mail,pwd))
        #
        mysql.connection.
        commit()#
        cursor.close()
        regInsertQuery = f"INSERT INTO customerdeatils
(username,email,passwrd) VALUES('{name}','{mail}','{pwd}')"
        insertflag =
        ibm_db.exec_immediate(conn,regInsertQuer
        y) msg = 'Your registration successfully
        completed.'
        # send mail
        sendemail(mail,'Accoun
        t_creation')

    return
render_template('index.html',signupmsg =
msg)# admin page
@app.route('/admin
/<which>')def
admin(which):
    if which == 'customers':
        # cursor = mysql.connection.cursor()
```

```
        result = ibm_db.exec_immediate(conn,'SELECT * FROM
        customerdeatils')data = []
        while ibm_db.fetch_row(result):

            temp =
[ibm_db.result(result,0),ibm_db.result(result,1),ibm_db.result(result,2),ibm_
db. re sult(result,3)]
            data.append(temp
        ) return
    render_template('admin.html',customers=data,complaints=Non
    e)if which == 'complaints':
        # cursor = mysql.connection.cursor()

        result = ibm_db.exec_immediate(conn,'SELECT * FROM
        complaints')data = []
        while ibm_db.fetch_row(result):

            temp =
[ibm_db.result(result,0),ibm_db.result(result,1),ibm_db.result(result,2),ibm_
db. re sult(result,3),ibm_db.result(result,4),ibm_db.result(result,5)]
            data.append(temp
        ) return
render_template('admin.html',customers=None,complaints=da
ta) # admin delete
@app.route('/Delete/<ty
pe>/<id>')def
Delete(type,id):
    if type == 'customers':
```

```
        # cursor = mysql.connection.cursor()
        result = ibm_db.exec(conn,f'DELETE FROM customerdeatils
WHERE id ="{id}"')
        #
        mysql.connection.
        commit()#
        cursor.close()
        return
    redirect(url_for('admin',which='customers'
    ))if type == 'complaints':
        # cursor = mysql.connection.cursor()
        result = ibm_db.exec_immediate(conn,f'DELETE FROM
complaintsWHERE id = {id}')
        #
        mysql.connection.
        commit()#
        cursor.close()
        return
redirect(url_for('admin',which='complaints'))
# manually login
@app.route('/login',methods=['POST','GET'])

def login():
    if request.method
        == 'POST': mail
```

```
=
request.form['ma
il1']
password =
request.form['pwd1']#
login is admin or not
if mail == "admin" and password ==
   'admin@1810': return
   redirect(url_for('admin',which='customers')
   )
# check account is exists or not
# cursor = mysql.connection.cursor()
query = "SELECT * FROM customerdeatils WHERE
email=? ANDpasswrd=?"
stmt = ibm_db.prepare(conn, query) #
type:ignore
ibm_db.bind_param(stmt,1,mail) #
type:ignore
ibm_db.bind_param(stmt,2,password) #
type:ignoreibm_db.execute(stmt) #
type:ignore
user = ibm_db.fetch_assoc(stmt)
# type:ignoreprint(user,password)
```

```
        #
        e
        x
        i
        s
        t
        s

        i
        f

        u
        s
        e
        r
        :
            session["username"] =
            user['USERNAME']
            session['mail'] = mail
            return
render_template('home.html',username=session["username"],mail=session["ma
il"
])
        else:
            msg = 'mail or password is not valid.'
```

```
            return render_template('index.html',signinmsg=msg)
    if request.method ==
        "GET": return
        redirect(url_for('ho
        me'))

# logout
method
@app.rout
e('/logout'
)def
logout():
    if "username" in
        session:
        session.pop("
        username")
    if "google_token" in
        session:
        session.pop("goo
        gle_token")
        session.pop("mail
        ")
    if "mail" in
        session:
```

```
        session.p
        op("mail
        ")
    return redirect(url_for('home'))


# complaint register
@app.route('/complaint',methods=
[' POST'])def complaint():
    if request.method == 'POST':
        complaint_name =
        request.form['complaint_name']name =
        request.form['name']
        mail = request.form['email']
        against_person =
        request.form['against_person']date =
        request.form["date"]
        des =
        request.form['complaint_
        des'] # cursor =
        mysql.connection.cursor()

        if not name == session["username"] or not mail ==
            session["mail"]:msg = "please don't change username
            and mail."
```

```
            return render_template('home.html',msg=msg)

        result = ibm_db.exec_immediate(conn,f"INSERT INTO
complaints(username,email,against_person,des,date,solved)
VALUES('{name}','{mail}','{against_person}','{des}','{date}'
,'{'0'}')")

        #

        mysql.connection.

        commit()#

        cursor.close()

        sendemail(mail,'complaint_creation')

        msg = 'Complaint registerd you check

        out complaints section.'return

        render_template('home.html',msg=msg)


# show complaints

and progress

@app.route('/showc

o mplaints') def

showcomplaints():

    # cursor = mysql.connection.cursor()


    # cursor.execute("SELECT * FROM complaints WHERE
username= % sAND email=%
s",(session["username"],session["mail"]))
```

```
    # details =
    cursor.fetchall()
    # cursor.close()
    query = "SELECT * FROM complaints WHERE username=?
    AND email=?"stmt = ibm_db.prepare(conn, query) # type:ignore
    ibm_db.bind_param(stmt,1,session["username"]) # type:ignore
    ibm_db.bind_param(stmt,2,session['mail']) # type:ignore
    ibm_db.execute(stmt)
    data = []
    while

        ibm_db.fetch_row(stmt):

        temp =

[ibm_db.result(stmt,0),ibm_db.result(stmt,1),ibm_db.result(stmt,2),ibm_db.resu
lt (stmt,3),ibm_db.result(stmt,4),ibm_db.result(stmt,5),ibm_db.result(stmt,6)]
        print(te
        mp)
        data.ap
        pend(te
        mp)
    return render_template('complaints.html',complaints=data)


# update complaint
@app.route('/solve',methods=
["POST"])def
solve_complaint():
```

```
    if request.method
        == "POST":c_id
        =
        request.form['c_i
        d']


        print(c_id)
        # cursor = mysql.connection.cursor()
        # cursor.execute("UPDATE complaints SET solved = % s
WHERE id = %s",('1',c_id,))
        query = "UPDATE complaints SET solved =
        '1' WHERE id = ?"#
        mysql.connection.commit() stmt =
        ibm_db.prepare(conn, query) # type:ignore
        ibm_db.bind_param(stmt,1,c_id) #
        type:ignore ibm_db.execute(stmt)
        detail =
        ibm_db.result(stmt
        ,0)print(detail)
        # cursor.execute("SELECT * FROM complaints WHERE id =
        % s",[c_id])query2 = "SELECT * FROM complaints WHERE
        id = ?"
        stmt1 = ibm_db.prepare(conn, query2) # type:ignore
```

```
        ibm_db.bind_param(stmt1,1,c_id) #
        type:ignoreibm_db.execute(stmt1)
        details =
        ibm_db.result(stmt1,
        0)# cursor.close()
        print(details)
        # solve_mail(session['mail'],'user')
        return
    redirect(url_for('showcomplaints'
    ))return
    redirect(url_for('showcomplaint
    s'
    ))


# # admin agent
allot #
@app.route('/solve_admin',methods=["
POST"])# def solve_admin():
#     if request.method == "POST":
#         c_id = request.form['c_id']
#         # cursor = mysql.connection.cursor()
#         cursor.execute("SELECT * FROM complaints WHERE id =
% s",[c_id])#   query = "SELECT * FROM complaints WHERE id =
?"
```

```
#         details =
cursor.fetchone()#
          cursor.close()
#         solve_mail(details[3],'admin')
#         return
redirect(url_for('admin',which='complaints'))
#
       return
redirect(url_for('admin',which='complaints'
)) # remove complaint
@app.route('/dismiss',methods=[
"POST"])def
dismiss_complaint():
    if request.method == "POST":

        c_id = request.form["c_id"]
        # cursor = mysql.connection.cursor()
        # cursor.execute("DELETE FROM complaints WHERE id =
        % s",[c_id])# mysql.connection.commit()
        # cursor.close()
        query = "DELETE FROM complaints
        WHERE id = ?"stmt =
        ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt,1,c_id) #
        type:ignore ibm_db.execute(stmt)
```

```
        return
    redirect(url_for('showcomplaint
    s'
    ))return
    redirect(url_for('showcomplaint
    s'
    ))
# send otp in user mail id
@app.route('/send_otp',methods=["POST
","GET"])def send_otp():
    if request.method
        == "POST":mail
        =
        request.form["m
        ail"]
        cursor = mysql.connection.cursor()
        cursor.execute("SELECT * FROM customerdeatils WHERE
email = %s",[mail])
        temp =
        cursor.fetcho
        ne()
        cursor.close()
        if not temp:
            return
render_template('forget.html',type='otp',msg1='Your account
doesn\'t exist please register')
```

```
        otp = randint(10 **
        5,10**6)
        forget_password_ma
        il(mail,otp)
        session["otp"] = otp

        return
render_template('forget.html',type='update_password',tempmail=m
ail) # forget password method
@app.route('/forgetpassword/<type>',methods=["POST","GET"])
def
    forgetpassw
    ord(type):if
    type ==
    'otp':
        return
    render_template('forget.html',type=typ
    e)if request.method == "POST":
        mail =
        request.form["m
        ail"]otp =
        request.form["ot
        p"]
```

```
        pwd =
        request.form["passwor
        d "] c_pwd =
        request.form["con_pwd
        "]
        print(otp,session['otp'])
        if not pwd == c_pwd:
            msg = 'Please Enter Password
            properly' return
        render_template('forget.html',type='updatePassword',msg=ms
        g)if not otp == str(session['otp']):
            msg = "Your OTP is Incorrect."
            return
        render_template('forget.html',type='updatePassword',msg=ms
        g)cursor = mysql.connection.cursor()
        cursor.execute("UPDATE customerdeatils SET passwrd = %
s WHEREemail = % s",(pwd,mail))
        mysql.connectio
        n.commit()
        cursor.close()
        msg = 'password updated
        successfully'
        updated_password_mail(m
        ail)
```

```
        return  render_template('forget.html',type='updatePassword',msg=msg)


if__name__== '__main__':
    app.run(host = '0.0.0.0',port = 8080,debug=True)
```

GitHub & Project Demo Link

GitHub link: https://github.com/IBM-EPBL/IBM-Project-12303-1659447004

Demo link: **https**://youtu.be/XVklWCRjMCo