PROJECT DOCUMENTATION

|  |  |
| --- | --- |
| **TEAM ID** | **PNT2022TMID24156** |
| **TEAM NAME** | **CCR** |
| **PROJECT NAME** | **CUSTOMER CARE REGISTRY** |

**Team Members:**

1. VIMITHA.U
2. VAISHNAVI.K
3. SNEHA.T
4. SARANYA.T

# INDEX

1. **INTRODUCTION**
   1. Project Overview
   2. Purpose

# LITERATURE SURVEY

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

# IDEATION & PROPOSED SOLUTION

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

# REQUIREMENT ANALYSIS

* 1. Functional requirements
  2. Non-Functional requirements

# PROJECT DESIGN

* 1. Data Flow Diagrams
  2. Solution & Technical Architecture

# PROJECT PLANNING & SCHEDULING

* 1. Sprint Planning & Estimation
  2. Sprint Delivery Schedule

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

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

# TESTING

* 1. Test Cases
  2. User Acceptance Testing

# RESULTS

* 1. Performance Metrics

# ADVANTAGES & DISADVANTAGES

1. **CONCLUSION**

# FUTURE SCOPE

1. **APPENDIX**
   1. Source Code

# INTRODUCTION:

* 1. **Project Overview:**

This Application has been developed to help the customer in processing their complaints.

Getting customers and know how to keep them are parts of the business that create income and the rest of all the activities of the business. Usually it create costs, that is a business aims and objectives is to make profits and create income and for this the business must care for their customers. Customers are not only concerned about the services and product they are getting but they are also concerned with the treatment they are getting. Today there is lot of competition and the customers is left with different alternatives that is different kinds of services and products which the customers might usually choose. A customer is the most important person in the company. Customers are corporate or individuals upon whom the companies depend, whereas the customers do not depend upon any particular company. They form part of the purpose for which companies are working for. If any problem facing by the customer, it is company’s responsibility to look into the problems and give solution for the customers. That is , if there is no customer there is no business.

# Purpose:

The customers can raise the ticket with a detailed description of the issue.

An Agent will be assigned to the Customer to solve the problem.

Whenever the agent is assigned to the customer they will be notified with an email alert.

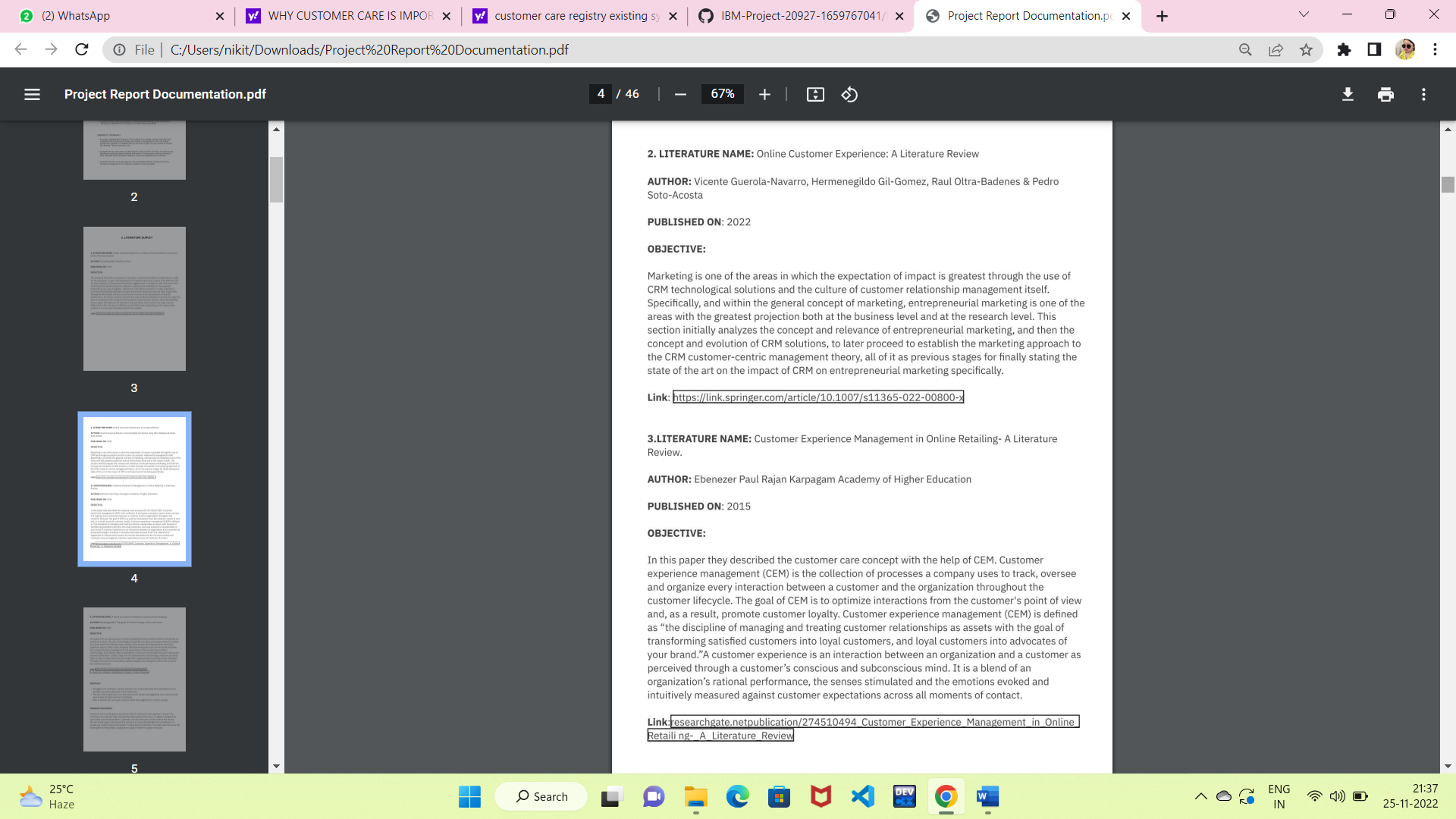
Customers can view the status of the ticket till the service is provided.

# LITERATURE SURVEY

* 1. **Existing System:**

The existing system is a semi-automated at where the information is stored in the form of excel sheets in disk drives. The information sharing to the Volunteers, Group members, etc. is through mailing feature only. The information storage and maintenance is more critical in this system. Tracking the member's activities and progress of the work is a tedious job here. This system cannot provide the information sharing by 24x7 days.

# Reference

****

# 2.1.1 Limitations:

The following are the limitations of the existing systems

1. The context formed if the customers did not find solution for their problem.
2. The choices, or even interpretations, must be made by the designer, which is not necessarily obvious, and therefore requires the use of an expert
3. The application which uses self-reports to gain information to provide reliable results can face one of the main issues with self-reports are that the participants can lie.

# Problem Statement:

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

Admin : The main role and responsibility of the admin are to take care of the whole process.  Starting from Admin login followed by the agent creation and assigning the customer's complaints.  Finally, He will be able to track the work assigned to the agent and a notification will be sent to the customer.

User: They can register for an account.  After the login, they can create the complaint with a description of the problem they are facing.  Each user will be assigned with an agent.  They can view the status of their complaint.

# IDEATION AND PROPOSED SOLUTION :

* 1. **Empathy Map Canvas :**

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

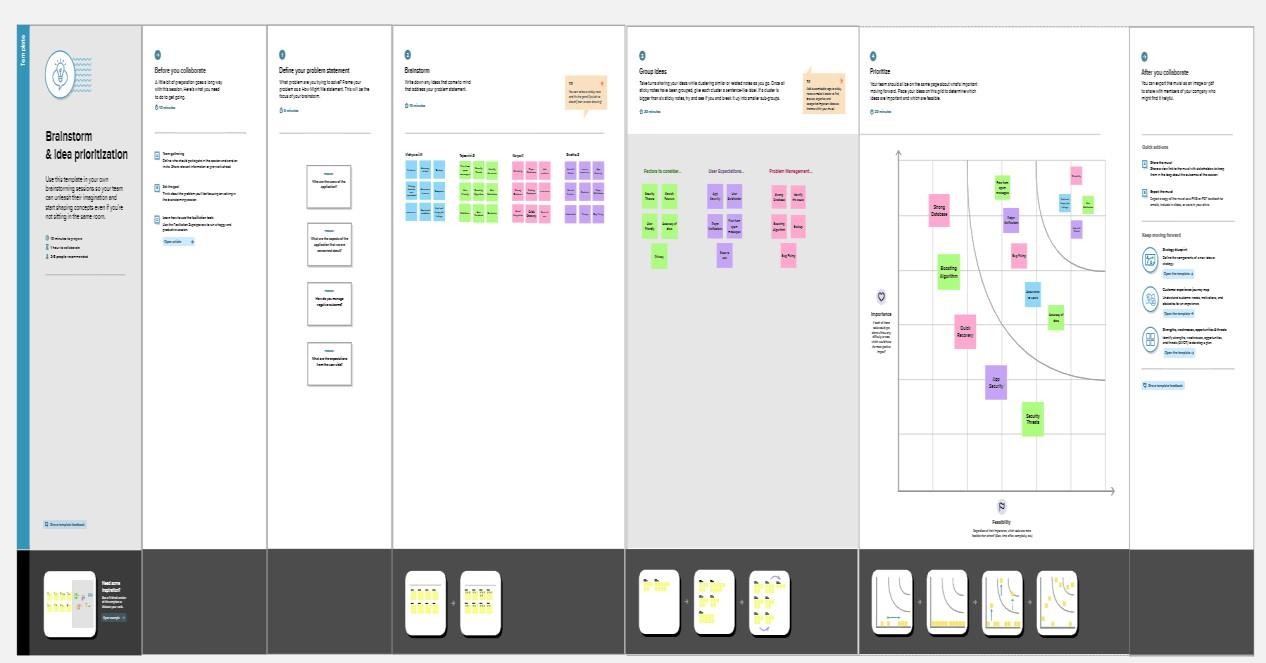
An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community



# Ideation And Brainstorming :

*Ideation is the process where you generate ideas and solutions through sessions such as Sketching, Prototyping, Brainstorming, Brainwriting, Worst Possible Idea, and a wealth of other ideation techniques. Ideation is also the third stage in the Design Thinking process.* Ideation is often the most exciting stage in a Design Thinking project, because during Ideation, the aim is to generate a large quantity of ideas that the team can then filter and cut down into the best, most practical or most innovative ones in order to inspire new and better design solutions and products.

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

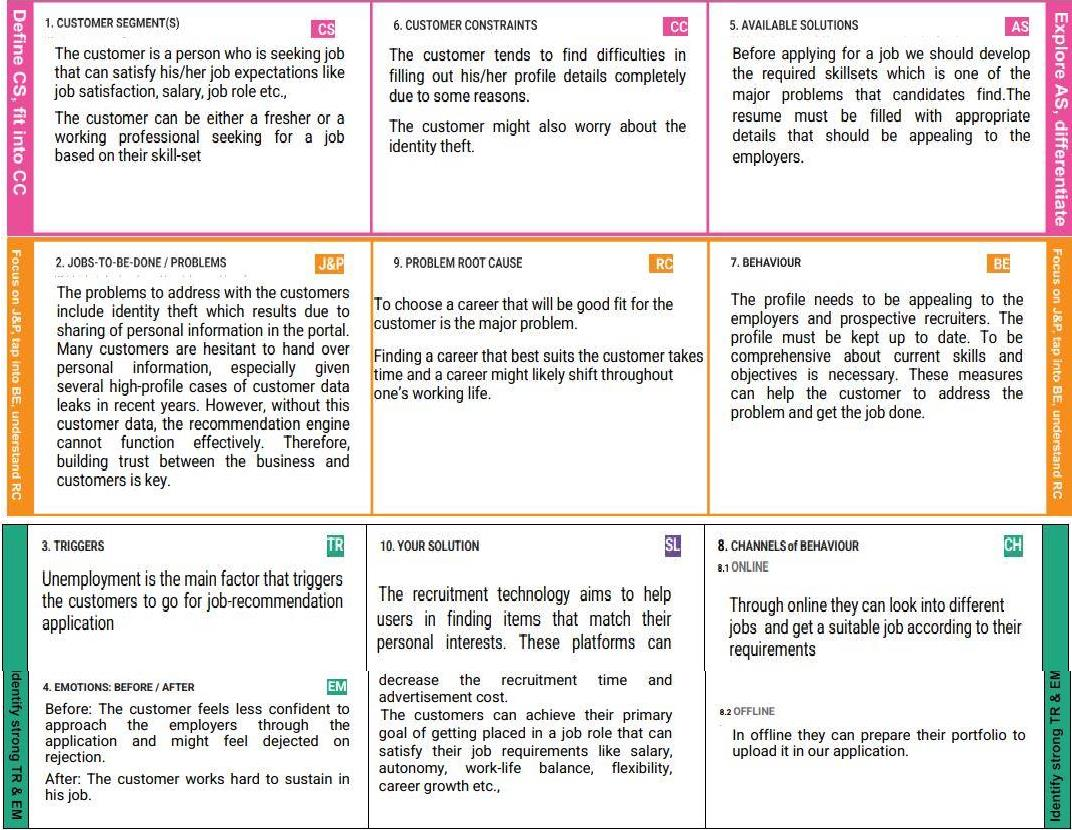


# Proposed Solution :

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
| 1. | Problem Statement (Problem to besolved) | The customers can raise the ticket with a detailed description of the issue.  An Agent will be assigned to the Customer to solve the problem.  Whenever the agent is assigned to the customer they will be notified with an email alert.  Customers can view the status of the ticket till the service is provided. |
| 2. | Idea / Solution description | The user should enter their login  Details and will enter inside the site.  Can ask their queries and can get solution for it. Can also ask solutions  For their questions to be solved. Agents will be available for each customers and proper answer will be given by them and will guide the customer to find out his solutions. |
| 3. | Novelty / Uniqueness | Agents allocation for each customer and guiding customer through messages, calls etc is one of the best advantages for the customer. |

|  |  |  |
| --- | --- | --- |
| 4. | Social Impact /Customer Satisfaction | The customer gets satisfied by this application as he can get his replies through calls or through texts , based on his need. Each customer has a agent , and this will guide them rightly. |
| 5. | Business Model (Revenue Model) | Social media is one of the ways by which we can advertise about our application. |
| 6. | Scalability of the Solution | Scalable recruiting is that ability to fluctuate with hiring demands is a complicated issue. It involves many building block   * Establish targeted goals * Build an attractive employer brand * Build a pipeline of talent * Generate accuratjob descriptions |

* 1. **Problem Solution Fit :**



# REQUIREMENT ANALYSIS:

* 1. **Functional Requirements:**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirements (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through Form  Registration through Gmail |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | User Login | Login via Email Login via OTP |
| FR-4 | Employer Login | Login via Email Login via OTP |
| FR-5 | User Profile Details | Fill the profile details   * Name * Number * Email-id |
| FR-6 | Agent Profile Details | * Name * Agent number |

# Non-functional Requirements:

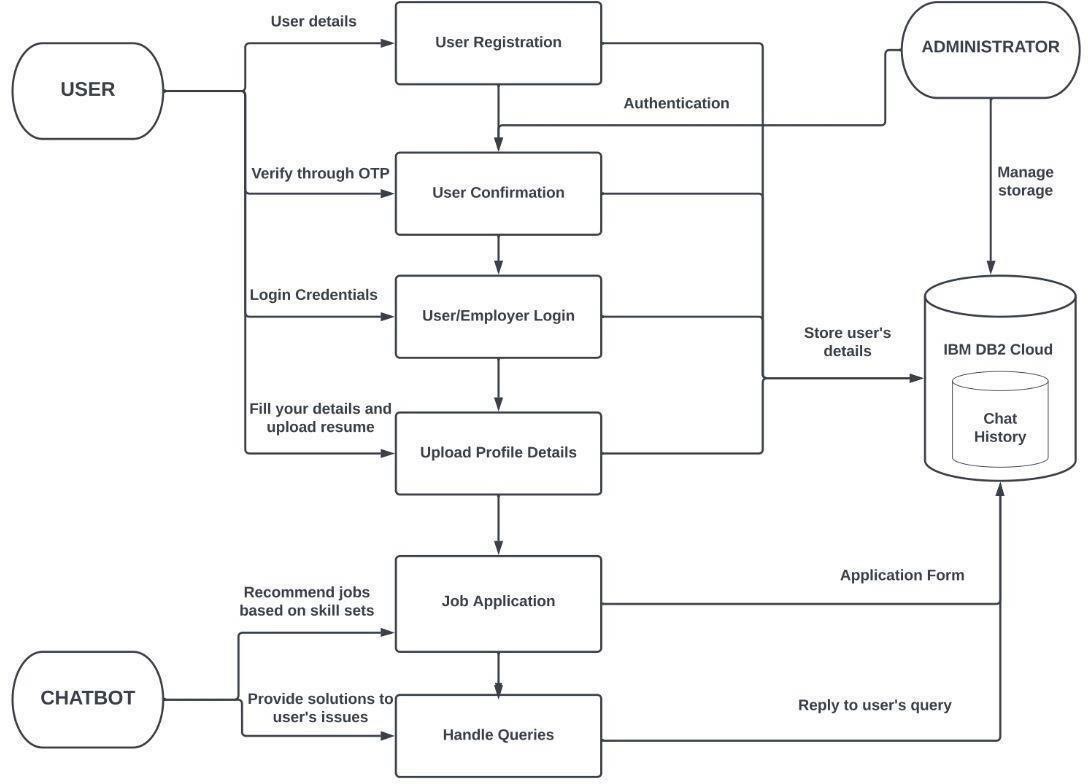
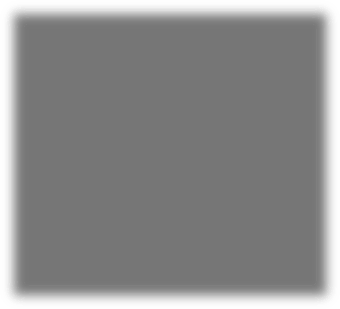
Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirements** | **Description** |
| NFR-1 | **Usability** | It involves watching a group of users interacting with your website or application in order to see what works and what doesn’t.  Doing so will allow you to make small  tweaks that often go overlookedin having any impact at all. |
| NFR-2 | **Security** | Providing a secure platform to users is crucial as itinvolves employers and employees upload their profile details  on the application |
| NFR-3 | **Reliability** | The application exhibits the quality of  being trustworthy or of performing consistently well. |
| NFR-4 | **Performance** | Networking ensures better performance as it an effective of finding a new job. It enables for the employees to connect with the employers workingin the organization of their interest to get more  information of the job and the roles  available. |
| NFR-5 | **Availability** | To make sure that the application is easily available and ready to use for the  users. |
| NFR-6 | **Scalability** | The measure of the application’s ability to increaseor decrease in performance and cost in response to changes in the application and system  processing demands. |

# PROJECT DESIGN

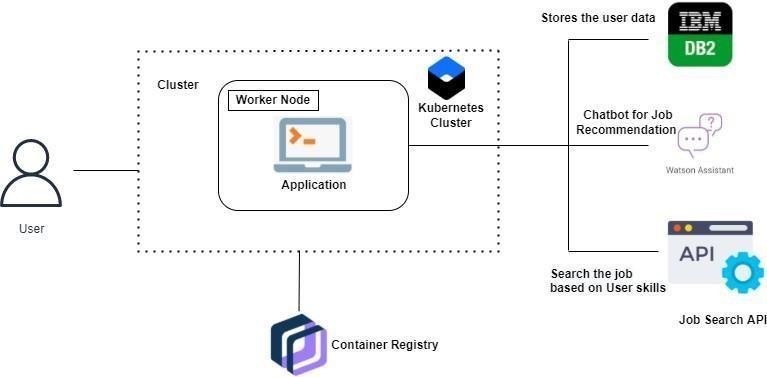
* 1. **Data Flow Diagrams**

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



# Solution & Technical Architecture

To develop an end-to-end web application capable of displaying the current job openings based on the user skillset. The user and their information are stored in the Database. An alert is sent when there is an opening based on the user skillset. Users will interact with the chatbot and can get the recommendations based on their skills. We can use a job search API to get the current job openingsin the market which will fetch the data directly from the webpage.



# Table-1: Components & Technologies:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.no** | **Component** | **Description** | **Technology** |
| 1. | User Interface | How user interacts with application e.g.,Web UI, Mobile  App, Chatbot etc. | HTML, CSS,  JavaScript,Flask, Python |
| 2. | Application Logic-1 | Logic for a process in the  application | Python or Java |
| 3. | Chatbot | To provide job recommendation  and to solve user queries related to job | IBM Watson Assistance |
| 4. | Cloud Database | To store user data and job  related data | IBM DB2. |
| 5. | File Storage | To store user data like resumes  and job posts | IBM Cloud Object  Storage |
| 6. | External API-1 | Purpose of External API used in  the application | IBM Weather API, etc. |
| 7. | External API-2 | Purpose of External API used in  the application | Search API |
| 8. | Cloud Server | To Deploy the application | Kubernetes |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.no** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | List the open-source frameworks used | HTML, CSS,  JavaScript, Flask, Kubernetes, Docker |
| 2. | Security Implementations | List all the security / access  controlsimplemented, use of firewalls etc. | IBM DB2  IBM Cloud Object Storage |
| 3. | Scalable Architecture | Justify the scalability of  architecture (3 – tier, Microservices) | Kubernetes IBM DB2 |
| 4. | Availability | Justify the availability of application | Kubernetes |

1. **PROJECT PLANNING AND SCHEDULING**

# 6.1. Sprint planning and Estimation:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-1 | Confirmation | USN-2 | As a user, I will receive confirmation email once I have registered for the application. | 3 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-1 |  | USN-3 | As a user, I can register for the application  through Google account | 2 | Low | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-1 | Login | USN-4 | As a user, I can login to the  application by entering email and password | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-1 | User Profile  Details | USN-5 | As a user, I can enter my profile details in the application. | 3 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-1 | Dashboard | UNS-6 | As a user, I can see my personal details and  activities. | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement** | **User Story**  **Number** | **User Story / Task** | **Story**  **Points** | **Priority** | **Team Members** |
| Sprint-2 | User Profile | USN-7 | As a user, I can update and edit my details | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-2 | Database | USN-8 | As an administrator, I can store the details of the user in IBM database. | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-2 | Cloud Storage | USN-9 | Can store customer details | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement** | **User Story**  **Number** | **User Story / Task** | **Story**  **Points** | **Priority** | **Team Members** |
| Sprint-3 | SendGrid Service | USN-12 | As a user, will get notification from the site giving solutions | 5 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-3 | Docker | USN-13 | As a user, I can access the application. | 5 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-3 | Kubernetes | USN-14 | As a user, I can access the application. | 5 | Medium | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-3 | Customer care service | USN-15 | As a user, I can get technical  Support through customer care to handle any queries. | 5 | High | Vaishnavi, Vimitha  Saranya, Sneha |

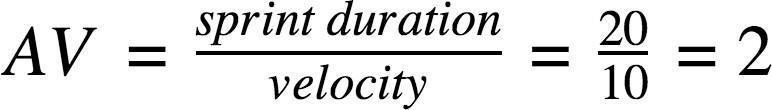
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement** | **User Story**  **Number** | **User Story / Task** | **Story**  **Points** | **Priority** | **Team Members** |
| Sprint-4 | Unit Testing | USN-16 | As a user, I can access and utilize the application without any interruption. | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-4 | Integration Testing | USN-17 | As a user, I can access and utilize the application without any  interruption. | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-4 | System Testing | USN-18 | As a user I can access and utilize the application without any  interruption. | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-4 | Acceptance Testing | USN-19 | As a user, I can access and utilize  The application without any interruption. | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |
| Sprint-4 | Deployment | USN-20 | Deploying the application | 4 | High | Vaishnavi, Vimitha  Saranya, Sneha |

* 1. **Sprint Delivery Schedule:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date(Planned)** | **Story Points Completed (as onPlanned End Date)** | **Sprint Release Date(Actual)** |
| Sprint-1 | 20 | 6 Days | 24Oct 2022 | 29 Oct 2022 | 20 | 29 Oct 2022 |
| Sprint-2 | 20 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 20 | 05 Nov 2022 |
| Sprint-3 | 20 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 20 | 12 Nov 2022 |
| Sprint-4 | 20 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 20 | 19 Nov 2022 |

**Velocity:**

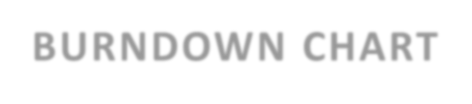
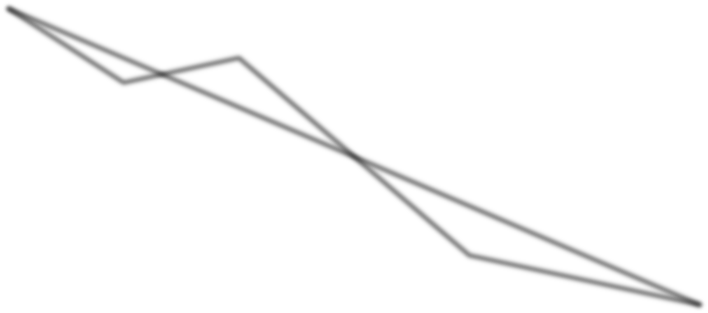
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) periteration unit (story points per day)



**Burndown Chart:**

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile [software development](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies suchas [Scrum](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/). However, burn down charts can be applied to any project containing measurable progress over time.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Initial  Estimate | 24-Oct | 25-Oct | 26-Oct | 27-Oct | 28-Oct | 29-Oct |
| Sprint  number | Day 0 | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 |
| Sprint-1 | 20 | 0 | 10 | 5 | 3 | 1 | 1 |
| Sprint-2 | 20 | 2 | 10 | 4 | 1 | 1 | 2 |
| Sprint-3 | 20 | 5 | 5 | 5 | 5 | 0 | 0 |
| Sprint-4 | 20 | 3 | 3 | 3 | 3 | 3 | 5 |
|  | | | | | | | |
| Task planned | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| Task Actual | 7 | 5.5 | 6 | 4 | 2 | 1.5 | 1 |



8

**BURNDOWN CHART**

7

6

5

4

3

2

1

0

Task Planned

Task Actual

1. **CODING AND SOLUTIONING**

# Feature 1:

When you open the healthcare chatbot, the authenticate page c appears first where you can authenticate with your credentials. If you are an incipient utilizer then the system prompts to engender an incipient profile asking for designation, mobile number, and password and email id. Your profile is engendered and the information is safe. It then redirects you to the authenticate page. The web application is designed in an alluring manner where the robot is moving and the colors of the background are effulgent which amends one’s mood. This will magnetize a sizably voluminous number of users. This is achieved by utilizing HTML for the format and CSS for styling and FLASK was utilized for connecting the code to the web application and present it to the utilizer. Once you have authenticated in, the chatbot greets you and interaction is commenced. It will ask you a bunch a question and then depending on your symptom it explicates the disease briefly, prescribes the medicine and whether to consult a medico or not. We had accentuated on the conception that we wanted to avail unlettered people so if the utilizer is inculcated to only some extent the chatbot has text to verbalization function which reads the conversation to the utilizer for better

understanding. We have utilized the chatterbot package; it utilizes the verdant Bayesian algorithm to determine if the input verbal expression meets a particular set of criteria that warrant a replication to be engendered from that logic adapter. The chatbot functions in multiple languages for people from different regions to utilize it. Google API is utilized (gTTS) for this.

# FEATURE 2:

To utilize the chatbot the utilizer must have a computer or phone to access it. It is asking to other chatbots in authenticate in and the for incipient users, they have to engender an incipient account so their details are stored safely with us for future references and it will be more facile for them to authenticate again if compulsory. After authenticate in the chatbot will ask a series of questions following which it can provide the felicitous diagnosis. For the chatbot we have not used traditional datasets, we have inscribed .yml files and have them indicted for the chatbot manually.

# FLOW CHART WORKING

1. Start the chatbot application.
2. Enter the authenticate in credentials.
3. If new user create account. Then Step2
4. After authenticate in conversation commences.
5. Chat with the chatbot.
6. Few questions will be asked by the chatbot.
7. The chatbot processes the input from the utilizer.
8. Comparison of symptoms mentioned by the utilizer and the database.
9. Provide precise and copacetic prescription and report.
10. We now ken what to do about the symptoms and then close the application.

# TESTING:

* 1. **Test Cases:**

# UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

# INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

# FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

* **Valid Input** : identified classes of valid input must be accepted.
* **Invalid Input** : identified classes of invalid input must be rejected.
* **Functions** : identified functions must be exercised.
* **Output** : identified classes of application outputs must be exercised.
* **Systems/Procedures** : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

# SYSTEM TESTING

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

# WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

# BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a

definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

# FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

# ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

# TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

# SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

# User Acceptance Testing:

Final Stage, before handling over to the customer which is usually carried out by the customer where the test cases are executed with actual data. The system under consideration is tested for user acceptance and constantly keeping touch with the prospective system user at the time of developing and making changes whenever required. It involves planning and execution of various types of tests in order to demonstrate that the implemented software system satisfies the requirements stated in the requirement document.

This final stage is important in getting software ready for deployment and delivery.

Two set of acceptance test to be run:

1. Those developed by quality assurance group.
2. Those developed by customer.

# RESULTS :

* 1. **Performance Metrics :**

The following are the key points for a performance metrics

# User Satisfaction / Apdex Scores

The application performance index, or Apdex score, has become an industry standard for tracking the relative performance of an application. It works by specifying a goal for how long a specific web request or transaction should take.

# Average Response Time

The fore mentioned user satisfaction Apdex scores as a preferred way to track overall performance. That said, averages are still a useful application performance metric.

# Error Rates

The last thing you want your users to see are errors. Monitoring error rates is a critical application performance metric. It is common to see thousands of exceptions being thrown and ignored within an application. Hidden application exceptions can cause a lot of performance problems.

# Count of Application Instances

If your application scales up and down in the cloud, it is important to know how many server/application instances you have running. Auto-scaling can help ensure your application scales to meet demand and saves you money during off-peak times. This also creates some unique monitoring challenges.

# ADVANTAGES AND DISADVANTAGES :

* 1. **Advantages :**

A good customer care service have many benefits for the businesses. It increases profits.

Customers can stick around if they get satisfied with the customer service , by giving proper solutions for their queries and by solving their questions.

The customers can raise the ticket with a detailed description of the issue.

An Agent will be assigned to the Customer to solve the problem.

Whenever the agent is assigned to the customer they will be notified with an email alert.

Customers can view the status of the ticket till the service is provided.

# Disadvantages :

1. There will be loads of queries and problems customer arises. But not all the time we get answer to those questions. Sometimes, Not knowing the answer to the customers queries is possible.
2. It can be difficult to measure their effectiveness since not all online recruitment services offer an in-depth analysis of the user’s questions.
3. Sometimes, we cannot meet the customers expectation.

# CONCLUSION:

In conclusion, the three main objectives of our apps is to satisfy customer needs. The application we are creating also provide the user with chatting assistant which enables them to get a more accurate solutions based on their problems. it has a successful usage in e-commerce applications to deal with problems . Customers can get their solutions and can ask their queries at any time needed.

# Future Scope:

Future directions of our work will focus on performing a more exhaustive evaluation considering a greater amount of methods and data as well as a comprehensive evaluation . As part of our ongoing research, we aim to build a application for the customer to ask their queries and get solutions for their queries and problems.

# APPENDIX

* 1. **Source Code Registration page**

<html>

<head>

<title>Registration</title>

<link rel="stylesheet" href="{{url\_for('static',filename='css/bootstrap.min.css')}}">

<link rel="stylesheet" href="{{url\_for('static',filename='myfont/css/all.min.css')}}">

<script src="{{url\_for('static',filename='js/jquery.min.js')}}"></script>

<style>

.shadow{

box-shadow:3px 3px 10px black; padding:30px;

}

.button{

background-color:#0161FF ; color:white;

padding:20px 10px; text-align:center; font-size:15px;

cursor:pointer; border-radius:50%;

}

</style>

</head>

<body style="margin:0;

background-image:url('/static/job.png');

background-repeat:no-repeat; background-position:center; background-attachment:fixed;">

<div class="container" style="margin-top:50px">

<div class="col-md-offset-3 col-md-5">

<div class="shadow">

<form method="POST" action="/register" class="page-align">

<h2 class="page-header text-primary text- center">Registration</h2>

<div class="form-group">

<label>Name</label>

<input type="text" class="form-control" name="name"

required>

</div>

<div class="form-group">

<label>Mail</label>

<input type="text" class="form-control" name="mail" required>

</div>

<div class="form-group">

<label>Contact</label>

<input type="text" class="form-control" name="contact"

required>

</div>

<div class="form-group">

<label>Password</label>

<input type="password" class="form-control" name="password"

required>

</div>

<div class="form-group">

<input type="submit" class="btn btn-success btn-block" value="Register">

<a href="{{url\_for('index')}}" class="btn btn-primary btn- block">Back to Home</a>

</div>

</form>

</div>

</div>

</div>

</body>

</html>

<script>

$(document).ready(function(){

$("form").attr("autocomplete","off");

});

</script>