|  |  |
| --- | --- |
|  | **SHRI SHAMBHUBHAI V. PATEL COLLEGE OF COMPUTER SCIENCE & BUSINESS MANAGEMENT** |

Affiliated To

**VEER NARMAD SOUTH GUJARAT UNIVERSITY,**

**SURAT**

A PROJECT REPORT

ON

**INSURANCE MANAGEMENT SYSTEM**

AS A PARTIAL REQUIREMENT FOR THE DEGREE

OF

BACHELOR OF COMPUTER APPLICATION

(B.C.A)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| **SUBMITTED BY:** | **GUIDED BY:** |
| Dhaduk Priyanshi G. [047] | Prof. Rekha Pichholiya |
| Dhaduk Vivek R . [049] |  |
| Desai Isha S. [044] |  |

|  |  |
| --- | --- |
|  | **SHRI SHAMBHUBHAI V. PATEL COLLEGE OF COMPUTER SCIENCE & BUSINESS MANAGEMENT** |

Certificate

This is to certify that the project titled **“ Insurance Management System ”** is the bonafide work carried out by **Dhaduk Priyansi G (047), Dhaduk Vivek R. (049), Desia Isha S (044),** students of TYBCA Sem-VI of Shri Shambhubhai V. Patel College Of Computer Science & Business Management, Surat affiliated to Veer Narmad South Gujarat University. He/she has successfully completed his/her project work in partial fulfilment of the requirements for the award of the degree of **"Bachelor of Computer Application"** during the academic year **2021-2022**. And that the project has not formed the basis for the award previously of any other degree, diploma, fellowship, or any other similar title.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| **SUBMITTED BY:** | **GUIDED BY:** |
| Dhaduk Priyanshi G. [047] | Prof. Rekha Pichholiya |
| Dhaduk Vivek R . [049] |  |
| Desai Isha S. [044] |  |

**ACKNOWELDGEMENTY**

* Our self Dhaduk Priyansi, Dhaduk Vivek, Desai Isha. Have the opportunity to express our knowledge. we would like to express our gratitude to all those who gave us the possibility to complete our project.
* Success is such a comprehensive project can not achieve single-handed. It is a team effort that sails the ship to the coast. so we would like to express our sincere thanks to all the dignitaries who were involved in making this project a great joy and turning it into a successful piece of work.
* **Prof. Rekha Pichholiya** – our professor and project co-ordinates has been very prudent to us threw out college studies. They are the person who has given this direction to our work and the shape to four imagination. we express our regard to them from the core of our hearts. we also like to thank the professors who are always ready to give the best guidance they are the person who gives solutions whenever needed.
* We would also like to acknowledge all the friends and colleagues, and team members for the help and encouragement by them from time to time. The constant support and encouragement of my friend are deeply appreciated. The project indeed gave challenging and exhilarating experience in designing and developing the required system.

From,

Dhaduk Priyansi G ,

Dhaduk Vivek R,

Desia Isha S.

**INDEX**

|  |  |
| --- | --- |
| **Chapter** | **Page No.** |
| **1. Introduction**   * 1. Project description   2. Project Profile   **2. Environment Description**   * 1. Hardware and Software Requirements   2. Technologies Used   **3. System Analysis and Planning**   * 1. Existing System and its Drawbacks   2. Feasibility Study   3. Requirement Gathering and Analysis   **4. Proposed System**  4.1 Scope  4.2 Project modules  4.3 Module vise objectives/functionalities Constraints  4.4 Expected Advantages  **5. Detail Planning**  5.1 Data Flow Diagram / UML  5.2 Process Specification / Activity Flow Diagram  5.3 Data Dictionary  5.4 Entity-Relationship Diagram / Class Diagram  **6. System Design**  6.2 Directory Structure  6.3 Input Design  6.4 Output Design  **7. Software Testing**  **8. Limitations and Future Scope of Enhancements References** | **1-4**  1  4  **5-13**  6  7  **14-19**  15  16  17  **20-24**  21  22  22  23  **25-56**  26  35  50  56  **57-00**  00  00  00  **00-00**  **00-00** |

1. **Introduction**

* **Project Description**
* **Project Profile**

* **Project Description**

**Insurance Management** is developed to see various kinds of insurance. This project is developed for the user to take the best insurance policy, the user to take a policy, and the admin who can manage both user and advisers.

Insurance management is an extensive and powerful script written in node js and react js. It gives the platform to manage insurance online. Users can see all policies and compare any policy to taking the best policy.

* **Features:-**
* Front page displays all types of policy.
* Insurance can create and manage by admin.
* Users can explore the various available policies of insurance companies after log-in in.
* User can change their password using Gmail OTP.
* Users can see all policies and compare any policy to taking the best policy.
* If users have any users queries about the policy they contact advisors.
* System admin handles all the policy, users, and advisers who are using the system.
  + 1. **User Panel Features: -**
       - User Registration.
       - User Login / Logout.
       - User Change Password.
       - User Compare policy.
       - User Search policy.
       - User Sort policy.
       - User live chat
       - User purchase policy
    2. **Admin Panel Features: -**
       - Admin Login.
       - Admin Logout.
       - Admin Insert Policy.
       - Admin View recent 10 User.
       - Admin Edit Policy.
       - Admin Delete Policy.
       - Admin Insert User In Some Cases.
       - Admin deletes User In Some Cases.
       - Admin view User
       - Admin view Policy

* **Project Profile:-**

**Project details**

|  |  |
| --- | --- |
| Project Title : | Auxilium Insura |
|  |  |
| Front-End : | React-JS |
|  |  |
| Back-End : | Node-JS |
|  |  |
| Browser : | All Browser Supported |
|  |  |
| Technology: | HTML, CSS, JavaScript, Bootstrap, MongoDBCompass, Express-JS |
|  |  |
| Guided By: | Prof. Rekha Pichholiya |
|  |  |
| Platform : | 16.14.0 |
|  |  |
| Tools Used For: | Visual studio, MongoDBCompass, Postman |
|  |  |
| Submitted To: | Shree Shambhubhai v. Patel college of computer science |
|  |  |
| Developed By: | Isha Sanjaybhai Desai  Priyansi Gabharubhai Dhaduk  Vivek Rajeshbhai Dhaduk |

1. **Environment Description**

* **Hardware and Software Requirements**
* **Technologies Used**

* **Hardware And Software Requirement :**

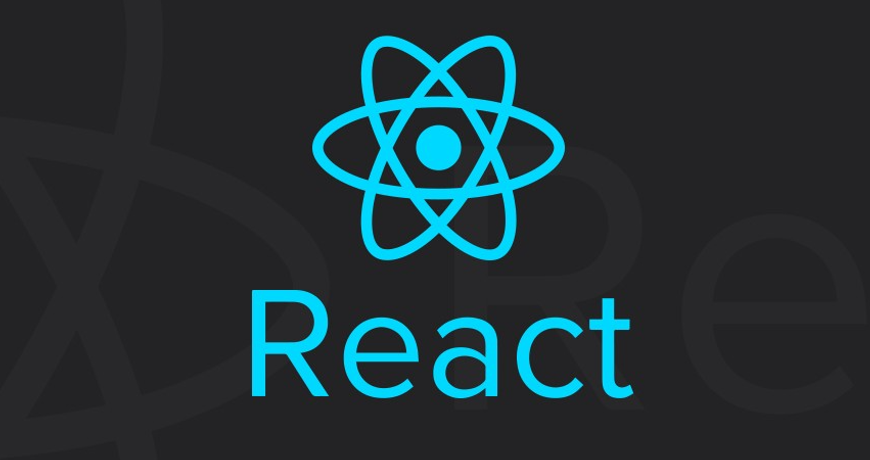
The efficient hardware and software configuration requires running the system is as suggested below. The configuration suggested is for better performance. The same functionality or higher configuration will always be better.

* **Client Side :**

* Internet-enabled device with a web browser.
* **Server side :**
* Node.js - 14.16.1
* MongoDB Database
* **Development Side :**
* **Processor:** Intel Core i5 11th generation
* **O.S:** Windows 10
* **Memory:** 8.00 GB
* **Hard disk:** 1TB
* **Web Browser:** Google Chrome (Recommended)

* **Technologies Used :**

* **Overview of React.js:-**



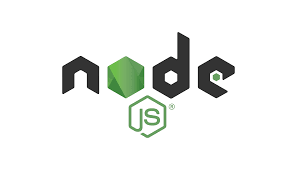
* React is a JavaScript library for building user interfaces.

.

* React is a library for building compassable user interfaces. It encourages the creation of reusable UI components, which present data that change over time.
* React abstracts away the DOM from you, offering a simpler programming model and better performance.
* React can also render on the server using Node, and it can power native apps using React Native.
* React implements one-way reactive data flow, which reduces the boilerplate and is easier to reason about than traditional data binding.

**React Advantages:-**

* React Uses virtual DOM which is a JavaScript object. This will improve apps performance since JavaScript virtual DOM is faster than the regular DOM.
* React Can be used on the client and server-sides as well as with other frameworks.
* React Components and data patterns improve readability, which helps to maintain larger apps.
* **Overview of Node.js :-**



* Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine).
* Node.js is an open-source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.
* Following are some of the important features that make Node.js the first choice of software architects.
* **Asynchronous and Event-Driven** :

All APIs of Node.js libraries are asynchronous, that is, non-blocking. It essentially means a Node. js-based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.

* **Very Fast** :

Being built on Google Chrome's V8 JavaScript Engine, the Node.js library is very fast in code execution.

* **Overview of JAVASCRIPT** :-



* JavaScript is a programming language that can be included on web pages to make them more interactive. You can use it to check or modify the Content of forms, and change images. Open new windows and write dynamic page content.
* This allows you to make parts of your web pages appear or disappear or move around on the page.
* JavaScript is a client-side, interpreted, object-oriented, high-level scripting language, while Java is a client-side, compiled, object-oriented high-level language.
* **JavaScript Feature**:-
  + JavaScript is a more flexible language.
  + JavaScript makes possible all type of validation and security.
  + JavaScript has a set of functions.
* **Overview of CSS** :-



* CSS is the language we use to style an HTML document.CSS describe how HTML elements should be displayed. This tutorial will teach you CSS from basic to advance.
* **Benefits of CSS**:-
* Greater typography and page layout controls
* With style sheets, you can specify traditional typography features that you could never do with HTML alone (even with its presentational extensions).
* Less Work
* Not only can format all similar elements in a document with a single style rule, but external style sheets also make it possible to edit the appearance of an entire site at once with a single style sheet edit.
* Potentially smaller documents
* Redundant font tags and nested tables make for bloated documents. Stripping Presentational HTML out of the document save on file size.
* **Overview of HTML**:-



* HyperText Markup Language (HTML) is the set of markup symbols or codes inserted into a file intended for display on the Internet. The markup tells web browsers how to display a web page's words and images.
* Each individual piece markup code (which would fall between "<" and ">" characters) is referred to as an element, though many people also refer to it as a tag.
* Some elements come in pairs that indicate when some display effect is to begin and when it is to end.
* HyperText Markup Language is the computer language that facilitates website creation.
* The language, which has code words and syntax just like any other language, is relatively easy to comprehend and, as time goes on, increasingly powerful in what it allows someone to create.
* **Overview of BootStarp**:-



* Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile-first projects on the web.
* It would be easy to send you over to their Getting Started page and call it a day. Their setup guide is indeed a host of useful information - links to CDNs, explanations on how to install with Bower, NPM, and Composer, information on integration with Auto prefixes and LESS, a bunch of templates, licenses, and translations.
* Bootstrap employs a handful of important global styles and settings that you’ll need to be aware of when using it, all of which are almost exclusively geared towards the normalization of cross-browser styles.
* It is a front-end framework used for easier and faster web development.
* It includes HTML and CSS-based design templates for typography, forms, buttons, tables, navigation, modals, image carousels, and many others. It can also use JavaScript plug-ins. It facilitates you to create responsive designs.
* **Overview of MongoDB:-**



* MongoDB is a [source-available](https://en.wikipedia.org/wiki/Source-available) [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [document-oriented database](https://en.wikipedia.org/wiki/Document-oriented_database) program. Classified as a [NoSQL](https://en.wikipedia.org/wiki/NoSQL) database program, MongoDB uses [JSON](https://en.wikipedia.org/wiki/JSON)-like documents with optional [schemas](https://en.wikipedia.org/wiki/Database_schema). MongoDB is developed by [MongoDB Inc.](https://en.wikipedia.org/wiki/MongoDB_Inc.) and licensed under the [Server Side Public License](https://en.wikipedia.org/wiki/Server_Side_Public_License) (SSPL).
* MongoDB Atlas also includes powerful features to enhance reliability for your mission-critical production databases, such as continuous backups and point-in-time recovery.
* MongoDB Atlas makes it easy to control access to your database. Your database instances are deployed in a unique Virtual Private Cloud (VPC) to ensure network isolation.
* MongoDB Atlas automates infrastructure provisioning, setup, and deployment so your teams can get the database resources they need when they need them. Patches and minor version upgrades are applied automatically.

1. **System Analysis and Planning**

* **Existing System and its Drawbacks**
* **Feasibility Study**
* **Requirement Gathering And Analysis**

* **Existing System and its Drawbacks :**

Our aim is “Customers want to be able to buy online insurance with confidence”. According to business experts, the news came that the entire insurance industry will adopt e-commerce. So the people who still refer to offline insurance to make them understandable, and easily adaptable about online insurance our research will help them.

For a new generation of working professionals, online insurance is the bridge that connects the digital age and the challenges of adult life. With internet access is rising and a young generation of working middle-class professionals reaching the cusp of adult life, the online insurance business is gathering momentum.

Our research reveals that, while of course there are significant variations in customer attitudes and behaviors around the globe, driven by the diverse economic, demographic, competitive, and regulatory environments, there are some underlying themes that are remarkably consistent. Listen to the voice of the customer.

* + **Existing System**
    - Policybazzar.com
    - Turtlemint.com
    - Licindia. in
    - Digitinsurance.com

* **Feasibility Study :**

The feasibility of our project has been judged on the time, technology, resources available, and project length.

* Time:- This project takes at least 3 months to be completed if we take the help of reused components; otherwise it may take more than 5 months to be completed.
* Technology:- This project is built on React.js & Node.js platform which is worldwide already standard. Therefore the technical risk is not so high hence feasible
* Resources:- Good technology professionals are a basic requirement. Internet and favorable environment, etc. the resources were fully available.

**1. Economic Feasibility:**

The prime requirement of this project is an internet connection as the project been developed is a Website. Moreover, it also demands a good server for data redirection and high speed and efficient CPU ram and processing and managing the online transaction done by the online Jobseeker & Recruiter

**2. Technical Feasibility:**

This website requires high and up-to-date technology. It needs database feedback and also needs to be accessed through the web and the internet. This can be done easily, so online payment should be facilitated. We provide our clients with the right worker for their job. So that clients can do their work easily and with proper use of their time. We provide our workers with the right clients for their job. So that worker can do their work easily and with proper use of their time.

**3. Management Feasibility:**

Management feasibility ratio and aspect of management news. Here all level of management rise of related basic feasibility and gives their decision for all those feasibilities to use or not is comes in basic feasibility of the system, bus how the requirement of that all, after that all manager.

**4. Operational Feasibility:**

The web application can be beneficial only if it satisfies the organization’s requirement; in such a way that resource utilization & the optimum outcome is justified. A web application should not only be robust but should also be able to work simultaneously with other systems. Operational feasibility means that web applications should not affect any existing system during the development phase or even in the implementation phase.

**5. behavioural feasibility:**

The new system can be beneficial only if it satisfies the organization’s requirements. In such a way that resource utilization and the optimum outcome are justified. A new system should not only be robust but should also be able to work simultaneously with other systems. Operational feasibility means that a new system should not affect any existing system during the development phase or even in the implementation phase. Following are some points underlying the operational feasibility of the system. As the development proceeded many doubts got clear doubt.

* Our project guide- Asst. Prof. Mansi Shah guided us to take proper care and check for the operational feasibility of the system.
* Efforts were made to optimize the human efforts in data collection, storage, retrieval, security, and presentation.
* The proposed system made its best efforts in achieving the necessary function and performance. As required by the user and keeping in mind some infrastructure constraints. Some of the following issues raised are to test the operational feasibility of a project including the following.
* Is there sufficient support for the management from the user?
* Will the system be used and work properly if it is being developed and implemented?
* Will there be any resistance from the user that will undermine the possible application benefits?
* This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration so there is no question of resistance from the user that can undermine the possible application benefits.

* **Requirement Gathering and Analysis:-**

* **Requirement Gathering :**
* Before developing an elegant application, the best way to identify basic needs and functionality is to take an interview with the director and management who are working in that specific area addressed by the proposed application.
* For gathering basic needs and functionalities, we had a meeting with people personally to understand required functionalities and also noted down some useful features to be included.
* Every successful system passes through requirement gathering because without any requirement the development process is like adding bugs to an empty system.
* Requirement gathering and analysis make the whole picture of the system and because of that, we can identify the features, the modules, and the functionality of the system.
* **Requirement Analysis :**
* System should be able to provide a registration facility.
* System should provide a safe login facility.
* System should provide forgot password facility.
* System should manage email Contact, skills.
* System should be able to add or remove user, policy and also manage its information.
* We analyzed our gathered information and we have decided our system should have the following functionality:-
  + **Common models**
    - Registration
    - Login
    - Forget Password (User )
    - Logout

### Module 1 ( USER )

* + User Registration.
  + User Login / Logout.
  + User Change Password.

### Module 2 ( ADMIN )

* + Admin Login.
  + Admin Logout.
  + Admin Insert policy.
  + Admin Send Email To User.
  + Admin Add policy.
  + Admin Delete policy.
  + Admin View Top 10 User.
  + Admin Edit policy.
  + Admin Delete policy.
  + Admin Insert User In Some Cases.

1. **System Analysis and Planning**

* **Project Scope**
* **Project modules**
* **Module wise objectives/functionalities Constraints.**
* **Expected Advantages**

* **Project Scope:-**

In the current project of WEB ANALYSIS, we have undertaken, the corresponding scope of our intranet website is as follows: For

This is a user-friendly layout & easy to understand for every user or service provider.

* It will allow the user to recover the password.
* It Also allows Service Providers to recover password
* It is able to Give information about registered Users and also edit that information.
* It allows sending mails.
* This system provides the insurance policy detail of all companies.
* It provides the main branch address of all insurance companies so the client can easily find the location of the company and contact it
* It also provides a comparison of all insurance policies.
* .pdf file download facility

* **Project Modules:-**

1. **User**
2. **Admin**

* **Modules Wise Objectives/functionalities Constraints:-**

The main objective of the Insurance Agency Management system is to Provide a Service insurance Policy to the customer.

**4.3.1 Admin**

* Admin can upload/delete/edit insurance policy for the insurance company
* Admin can manage user
* Admin can manage the insurance policy

**4.3.2 User**

* Users can register Itself
* Users can log in
* Users can view the policy
* Users get a quote
* Users can change their password

* **Expected Advantages:-**

* **High Security**

There is a high-security mechanism provided by the proposed system. No unauthorized person can make a change in the data that are stored in the application. Only an authorized person can change & also can make a login. Also, systems provide two-step authentication for security reasons.

* **Easy and Fast Response**

All user’s data are stored in an online centralized database. So, users access data from anywhere and on any device. No need for any other device to run the application on the device. Separate login for separate users like faculty, student, and admin.

* **High Amount of Accuracy**

In our application there are less amount of human activities are involved in that case which will have less effect on our accounts. Before the Insurance Company registers, we can verify the insurance company details for security.

* **Easy to use**

Auxilium Insura is giving an easy and responsive interface for mobile and tablet to online insurance services through the web. We are providing a helpline number so that users can easily communicate with the system

1. **Detail Planning**

* **Data flow diagram/UML**
* **Process specification / activity flow diagram**
* **Data dictionary**
* **Entity-relationship diagram / class diagram**

* **Data Flow Diagram/UML:-**

* A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and Outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.

### Strict landing page policy :

* A data flow diagram needs to be simple because users have to go through it and understand suggest corrections or changes.
* External entity
* Data flow
* Process
* Datastores

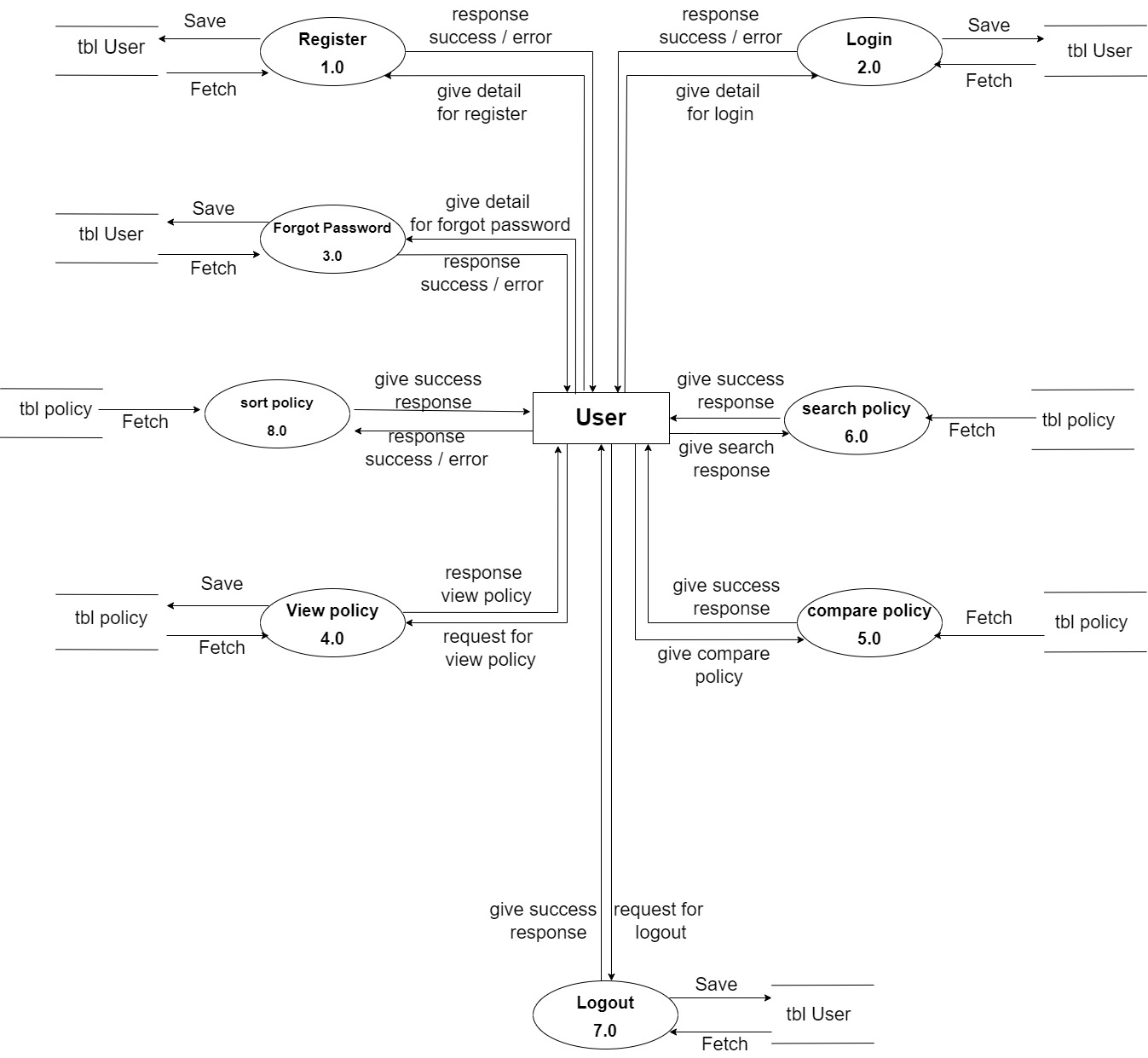
### Symbols used in DFD:

* + 1. **Process**: -
    - Flow of data is transformed.
    - A Process represents some amount of work using the behavior of data.
    - Transformation of data from one form to another.
    - A circle represents a process.
    - The area of the circle is divided into two parts.
    - The process number is written in the top portion and the process name is written in the bottom portion.
    - E.g. verify credits, update inventory files, etc.
    1. **External Entity:-**
    - A source or destination of data that is external to the system.
    - As the name suggests it line outside the context of the system. It is represented by solid squares.
    - The people/department that will be giving the data and/or receiving information.
    - It can be referred to as external entities as they do not accomplish any word done by the system.
    - The data coming in and the report produced is referred to as the input and output of the system respectively.
    1. **Data Flow:-**
    - Line is a packet of data. it may be in form of documentation, letters, etc.
    - A data flow portrays an interface among different components in the flow diagram.
    - It represents the data of the data as it flows through a system. An arrow represents it.
    - The arrowhead point in the direction in which the data moves.
    1. **Data Stores:-**
    - Any stores data but no reference to the physical method of storing.
    - If there is a logical requirement for the data to be stored, it is held in a data store.
    - A data store there for, is a repository of the data.

### Context Level DFD

### 

### 1st Level DFD ( User )



### 1st Level DFD ( Admin )

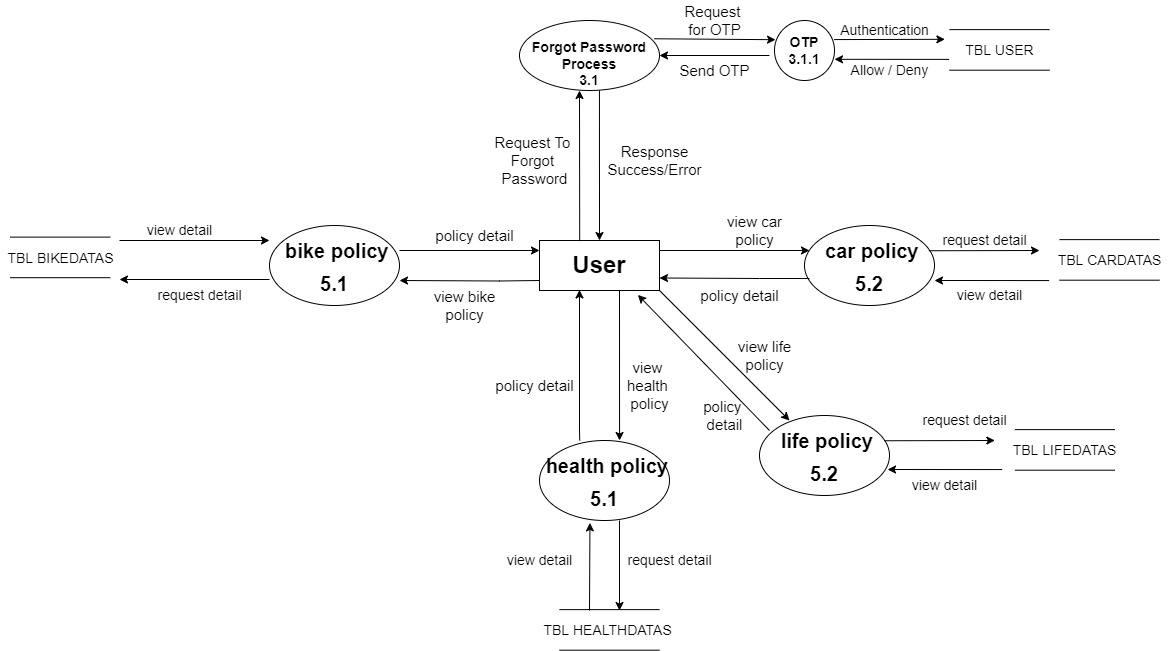
### 

### 1st Level DFD ( policy )

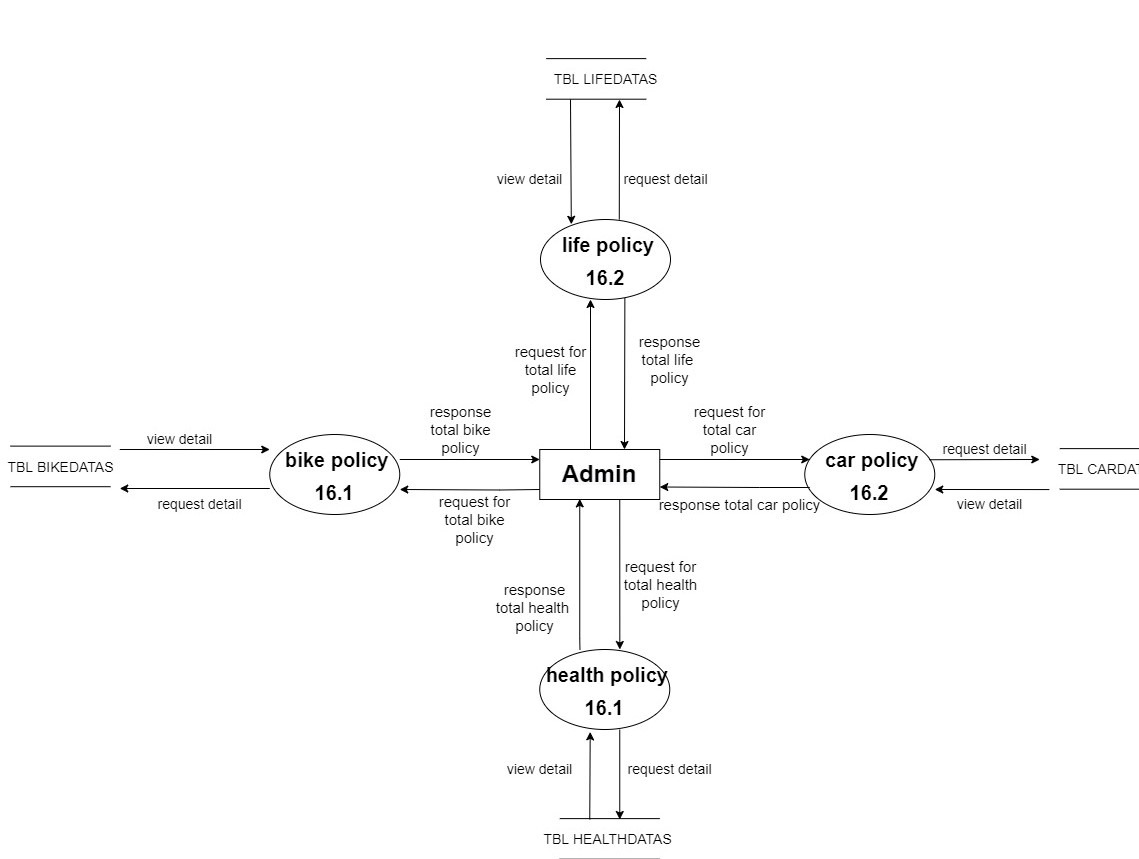
### 

### 

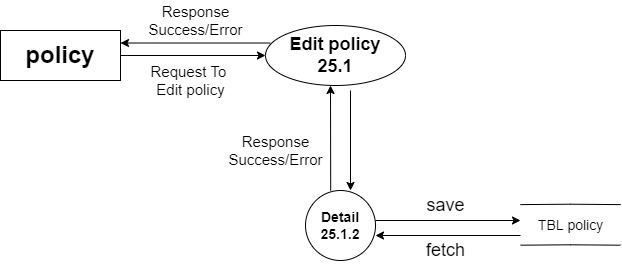
* **2st Level DFD ( User )**



* **2st Level DFD ( Admin )**



* **2st Level DFD ( policy )**



* **Process Specification:-**

### User Register

### 

### User Login

### 

### User Forgot Password

### 

### User View Policy

### 

### User compare Policy

### 

### User search policy

### 

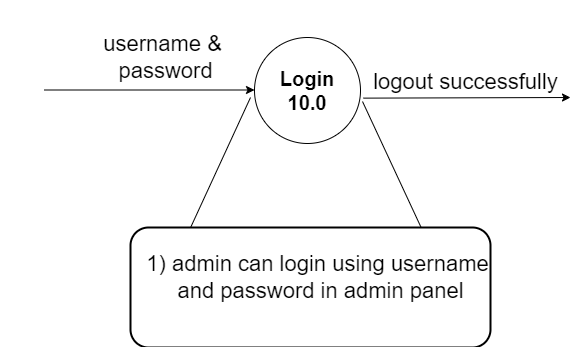
### User sort policy

### 

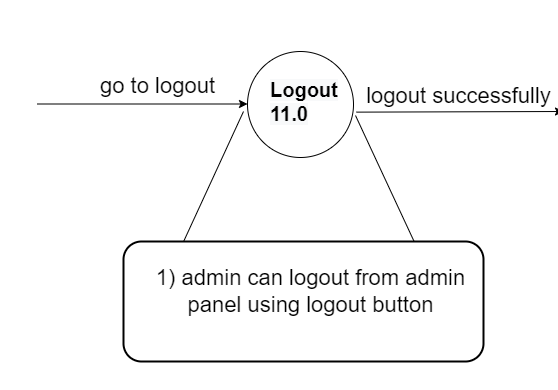
### User Logout

### 

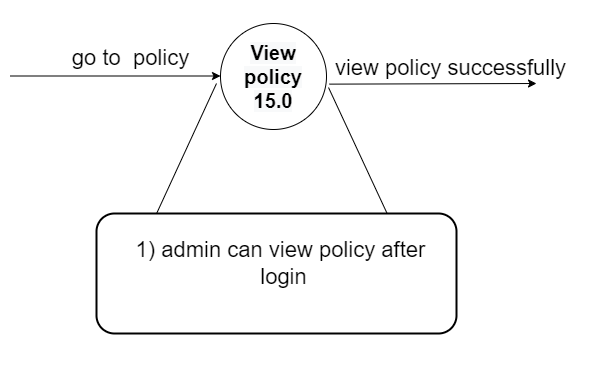
* **Admin Login**



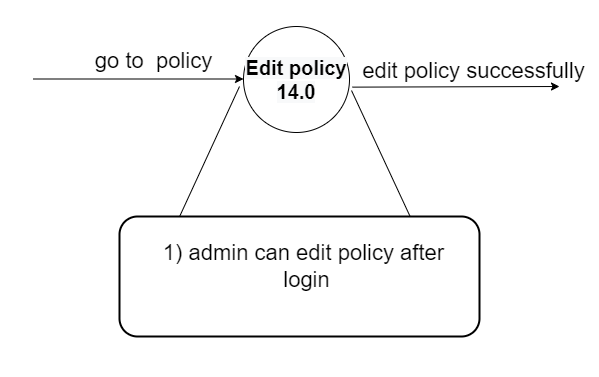
* **Admin Logout**



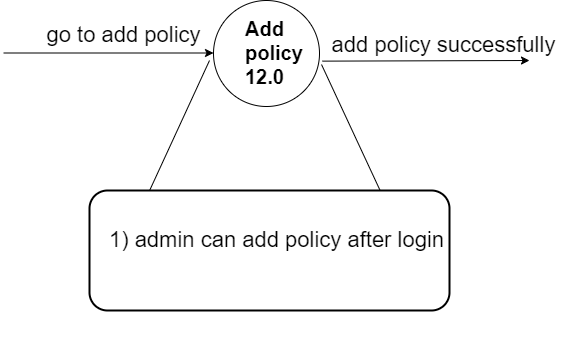
* **Admin View Policy**



* **Admin Edit Policy**



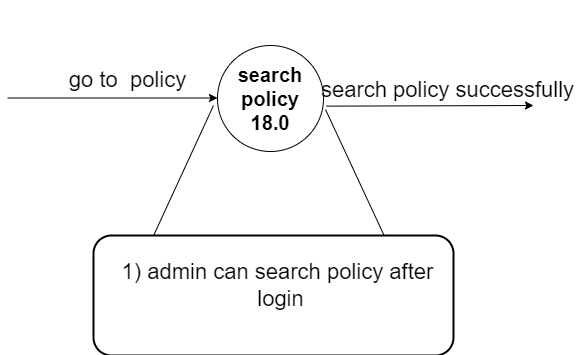
* **Admin Add policy**



* **Admin delete policy**



* **Admin search policy**



* **Data Dictionary:-**

* **User:-**

|  |  |  |  |
| --- | --- | --- | --- |
| Data Name | Type | Characteristics | Description |
| User\_id | Autogenerate | Only number allow | User\_id is the unique id of user data. |
| Name | String | Only string allow | It displays the user’s name. |
| Email | Email form | Only email allow | It displays the user’s email. |
| Gender | String | Only string allow | It displays the user’s gender. |
| Phone | String | Only number allow | It displays the user’s phone number. |
| Pan\_no | String | Only string allow | It displays the user’s pan\_no. |
| Password | Password (encrypted) | password | It displays the user’s password. |
| Age | Number | Only number allow | It displays the user’s Age. |
| Token | String  (Autogenerate) | - | It displays the user’s login token. |
| Date & Time | String  (Autogenerate) | Date | It displays the user’s registration date and time. |

* **Admin:-**

|  |  |  |  |
| --- | --- | --- | --- |
| Data Name | Type | Characteristics | Description |
| Admin\_id | Autogenerate | Only string allow | Admin\_id is the unique id of admin data. |
| Name | String | Only string allow | It displays the admin’s name. |
| Email | Email form | Only email allow | It displays the admin’s email. |
| Gender | String | Only string allow | It displays the admin’s gender. |
| Phone | String | Only number allow | It displays the admin’s phone number. |
| Pan-no | String | Number and string allow | It displays the admin’s Pan-no. |
| Password | Password (encrypted) | password | It displays the admin’s password. |
| Token | String  (Autogenerate) | - | It displays the admin’s login token. |
| Date & Time | String  (Autogenerate) | Date | It displays the admin’s register date and time. |

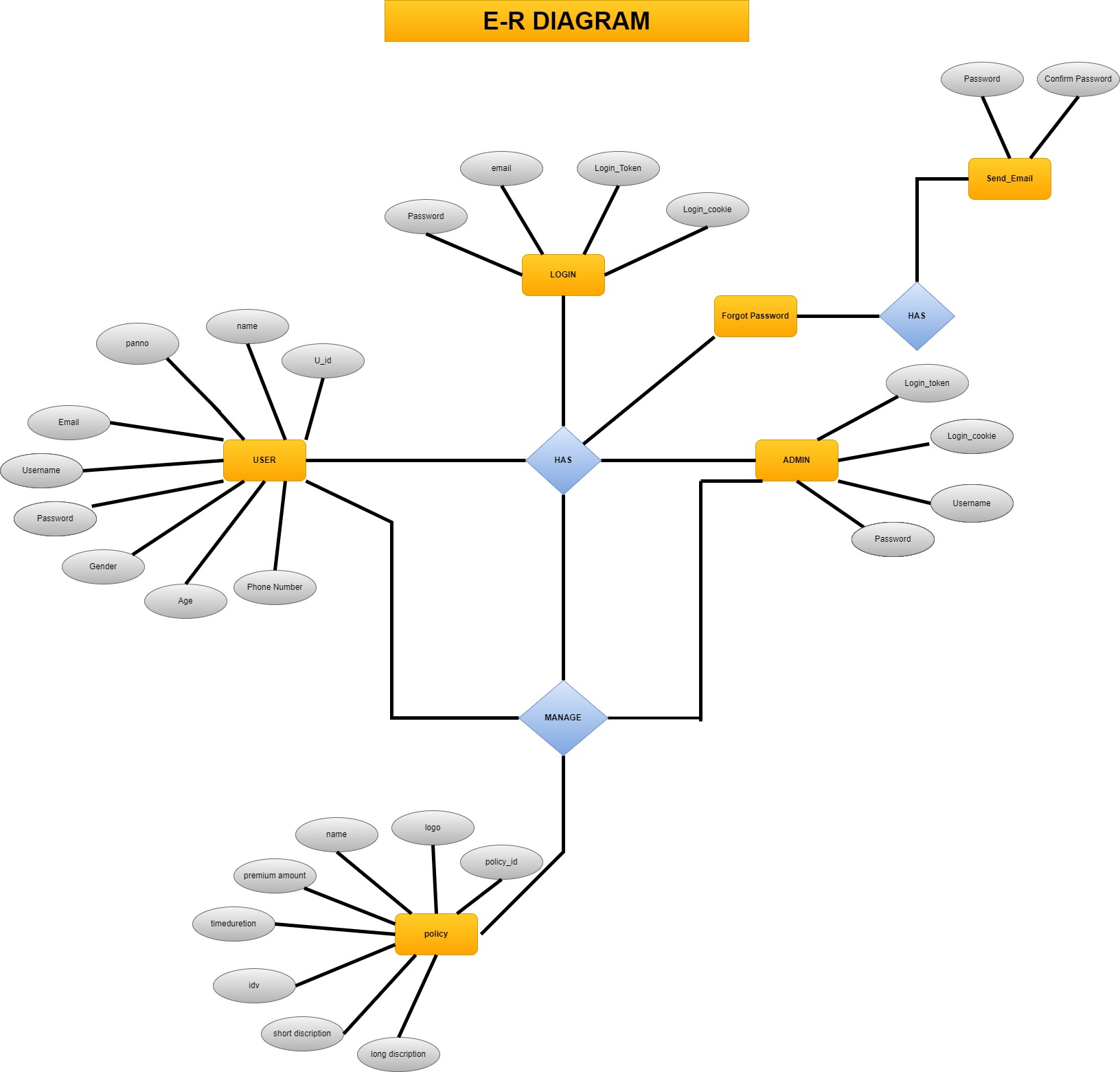
* **Policy:-**

|  |  |  |  |
| --- | --- | --- | --- |
| Data Name | Type | Characteristics | Description |
| policy\_id | Autogenerate | Only number allow | policy\_id is the unique id of policy data. |
| name | String | Only string allow | It displays the policy name. |
| Logo | file | file | It displays the policy logo file. |
| Premium amount | Number | Only number allow | It displays the policy Premium amount |
| Short description | String | Only string allow | It displays a policy Short description. |
| Long  Description | String | Only string allow | It displays policy long descriptions. |
| Time duration | String | Only number string allow | It displays policy time duration |
| idv | Number | Only number allow | It displays policy idv |
| Date & Time | String  (Autogenerate) | date | It displays policy created date and time. |

* **Forgot Password:-**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Characteristics | Description |
| User\_id | Autogenerate | Only number allow | User\_id is the unique id of user data. |
| Email | Email form | email | It displays user emails they save in the user database. |
| OTP | String | Only number allow | It displays unique OTP for security |
| Password | Password  (encrypted) | password | It displays the new password of the user. |
| Confirm Password | Password  (encrypted) | password | It displays the new confirm password of the user. |
| Date & Time | String  (Autogenerate) | date | It displays user data updated date and time. |

* **Entity-Relationship Diagram**



**6 System Design**

* **Database Design**
* **Directory Structure**
* **Input Design**
* **Output Design**

* **Database Design:-**

* **User:-**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Description |
| User\_id | Autogenerate | Null | Primary | User\_id is the unique id of user data. |
| Name | String | Not Null |  | It displays the user’s name. |
| Email | Email form | Not Null |  | It displays the user’s email. |
| Gender | String | Not Null |  | It displays the user’s gender. |
| Phone | Number | Not Null |  | It displays the user’s phone number. |
| Pan\_no | String | Not Null |  | It displays the user’s pan\_no. |
| Password | Password (encrypted) | Not Null |  | It displays the user’s password. |
| Age | String | Null |  | It displays the user’s Age. |
| Token | String  (Autogenerate) | Null |  | It displays the user’s login token. |
| Date & Time | String  (Autogenerate) | Null |  | It displays the user’s register date and time. |

* **Admin:-**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Description |
| Admin\_id | Autogenerate | Null | Primary | Admin\_id is the unique id of admin data. |
| Name | String | Not Null |  | It displays the admin’s name. |
| Email | Email form | Not Null |  | It displays the admin’s email. |
| Gender | String | Not Null |  | It displays the admin’s gender. |
| Phone | Number | Not Null |  | It displays the admin’s phone number. |
| Pan-no | String | Not Null |  | It displays the admin’s Pan-no. |
| Password | Password (encrypted) | Not Null |  | It displays the admin’s password. |
| Token | String  (Autogenerate) | Null |  | It displays the admin’s login token. |
| Date & Time | String  (Autogenerate) | Null |  | It displays the admin’s register date and time. |

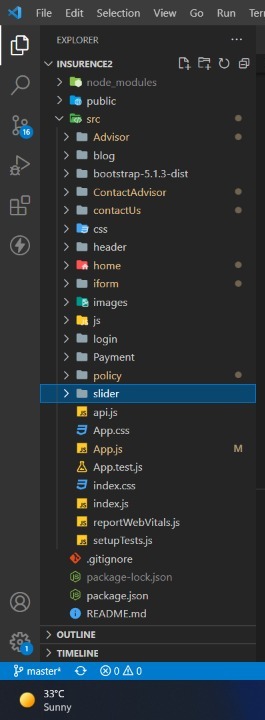
* **Policy:-**

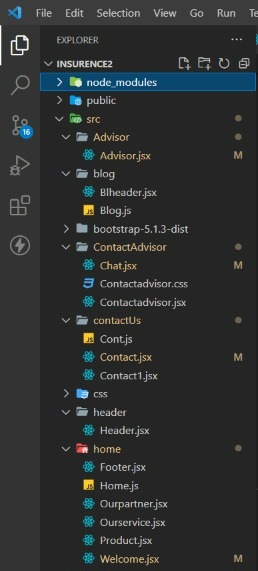
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Description |
| policy\_id | Autogenerate | Null | Primary | policy\_id is the unique id of policy data. |
| name | String | Not Null |  | It displays the policy name. |
| Logo | file | Not Null |  | It displays the policy logo file. |
| Premium amount | Number | Not Null |  | It displays the policy Premium amount |
| Short description | String | Not Null |  | It displays a policy Short description. |
| Long  Description | String | Not Null |  | It displays policy long descriptions. |
| Time duration | String | Null |  | It displays policy time duration |
| idv | Number | Null |  | It displays policy idv |
| Date & Time | String  (Autogenerate) | Null |  | It displays policy created date and time. |

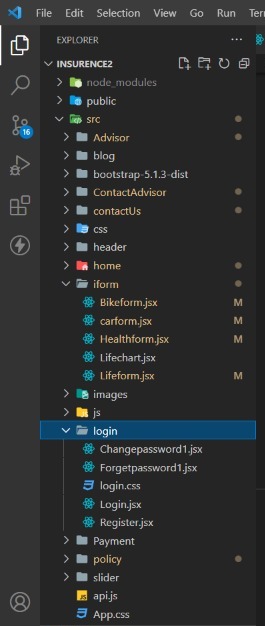
* **Forgot Password:-**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Description |
| User\_id | Autogenerate | Null | Foreign | User\_id is the unique id of user data. |
| Email | Email form | Not Null |  | It displays user emails they save in the user database. |
| OTP | String | Not Null |  | It displays unique OTP for security |
| Password | Password  (encrypted) | Not Null |  | It displays the new password of the user. |
| Confirm Password | Password  (encrypted) | Not Null |  | It displays the new confirm password of the user. |
| Date & Time | String  (Autogenerate) | Null |  | It displays user data updated date and time. |

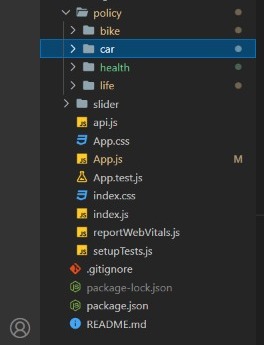
* **Directory Structure:-**





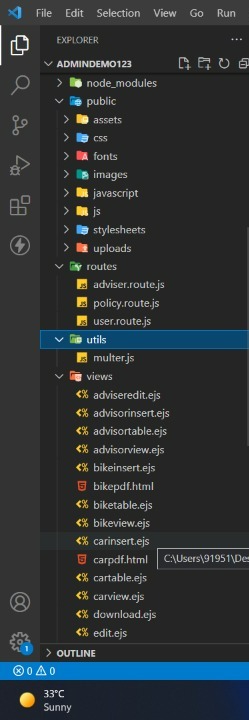


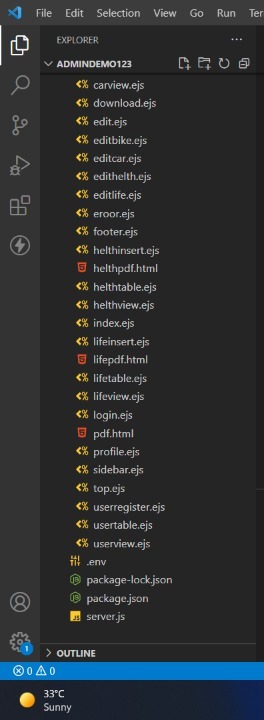






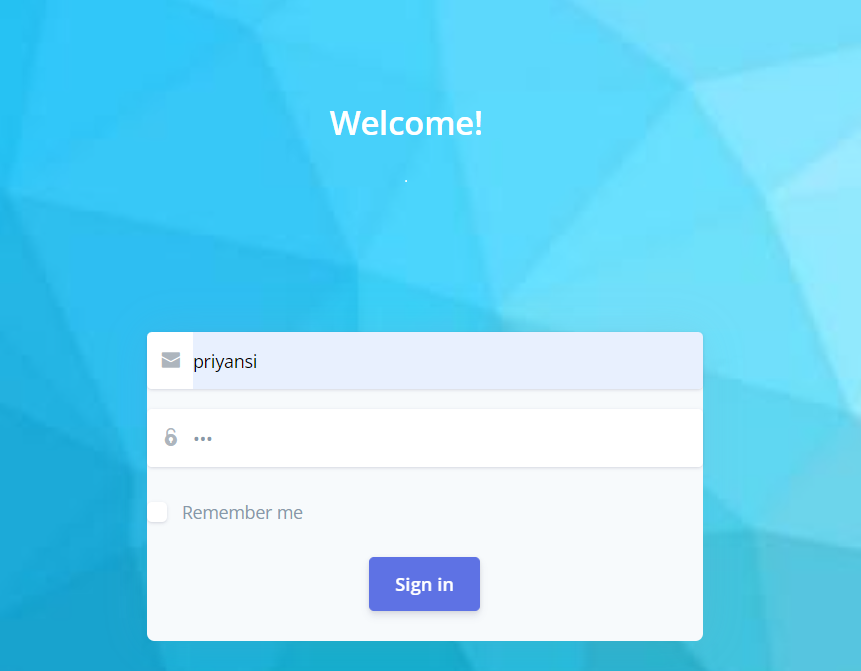


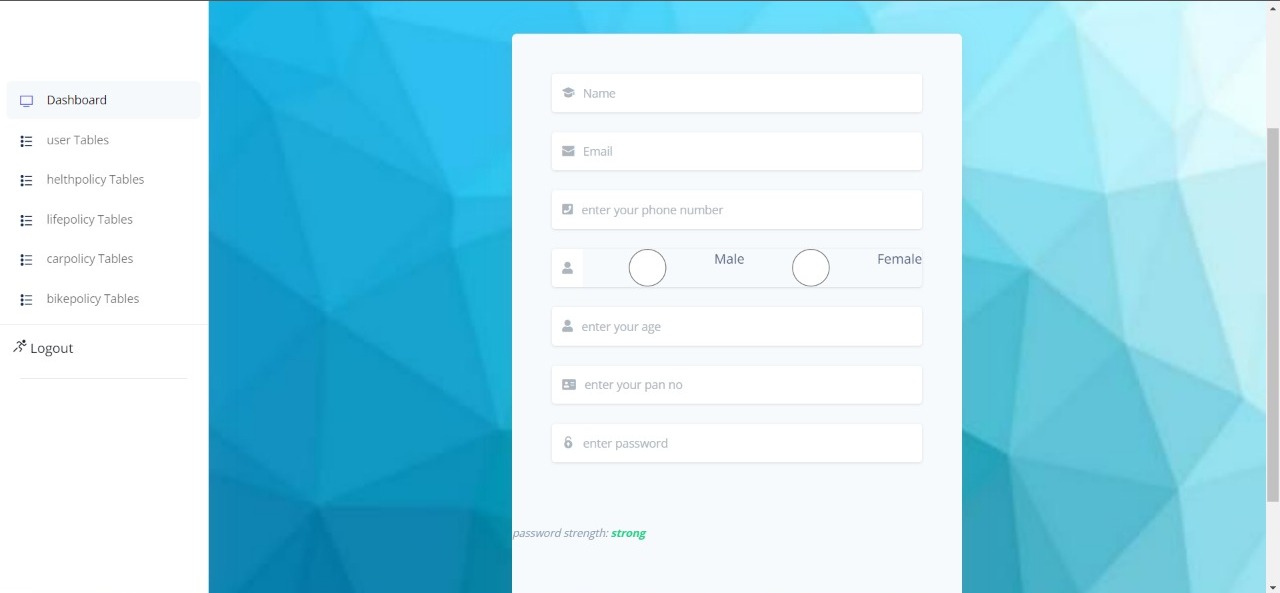


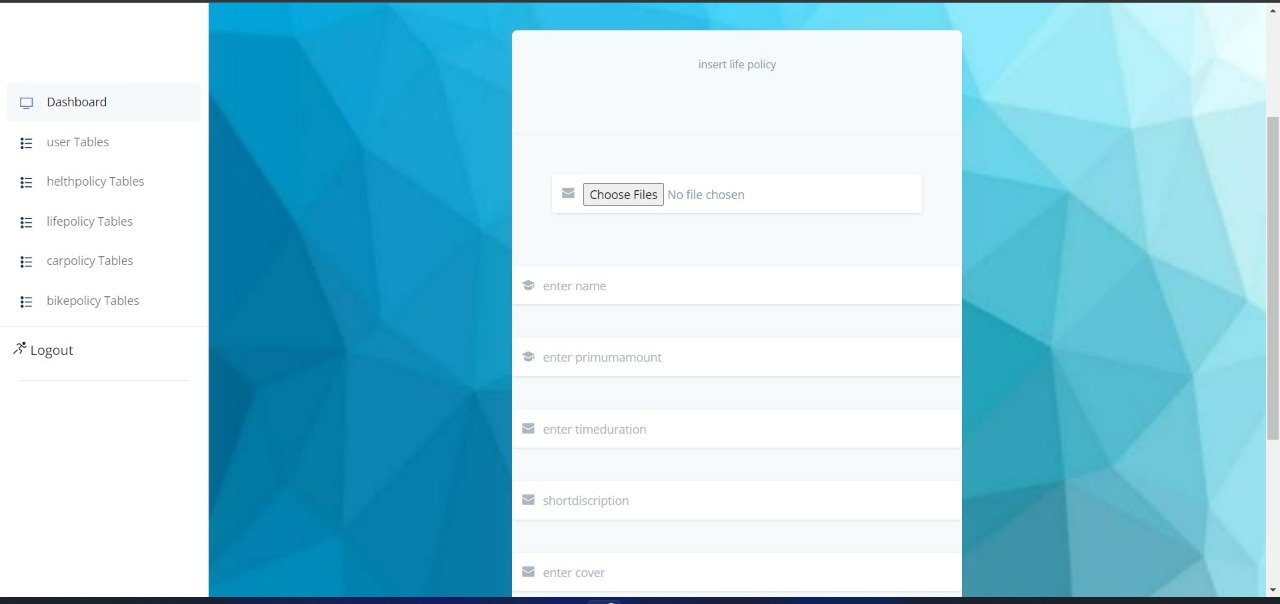


* **Input Design:-**

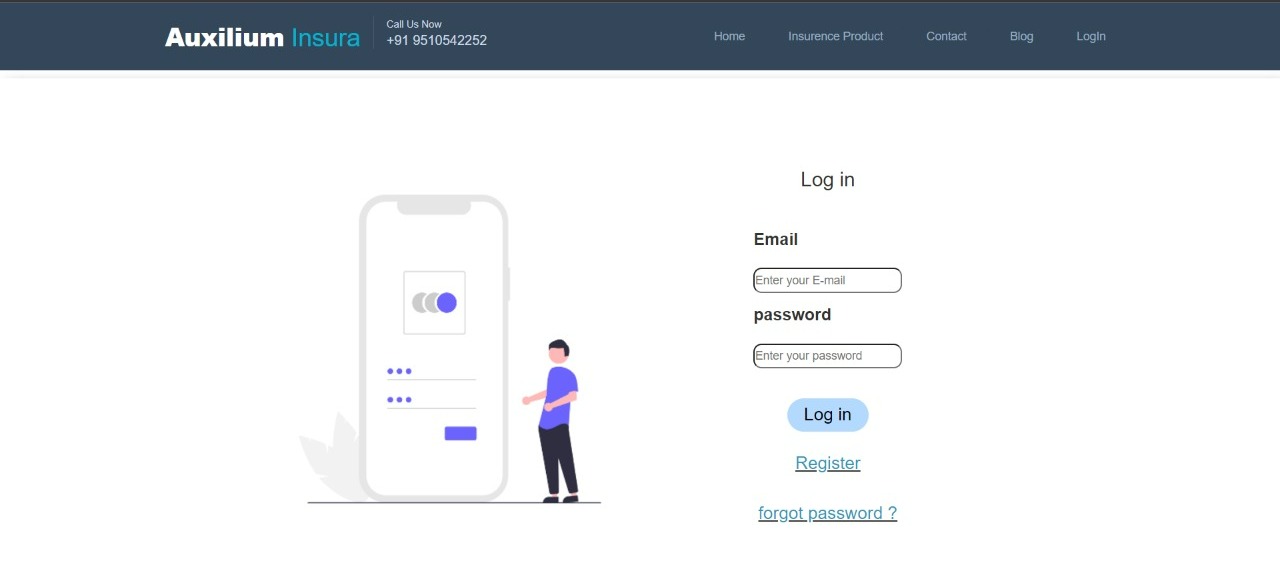
* **Admin-side input design**

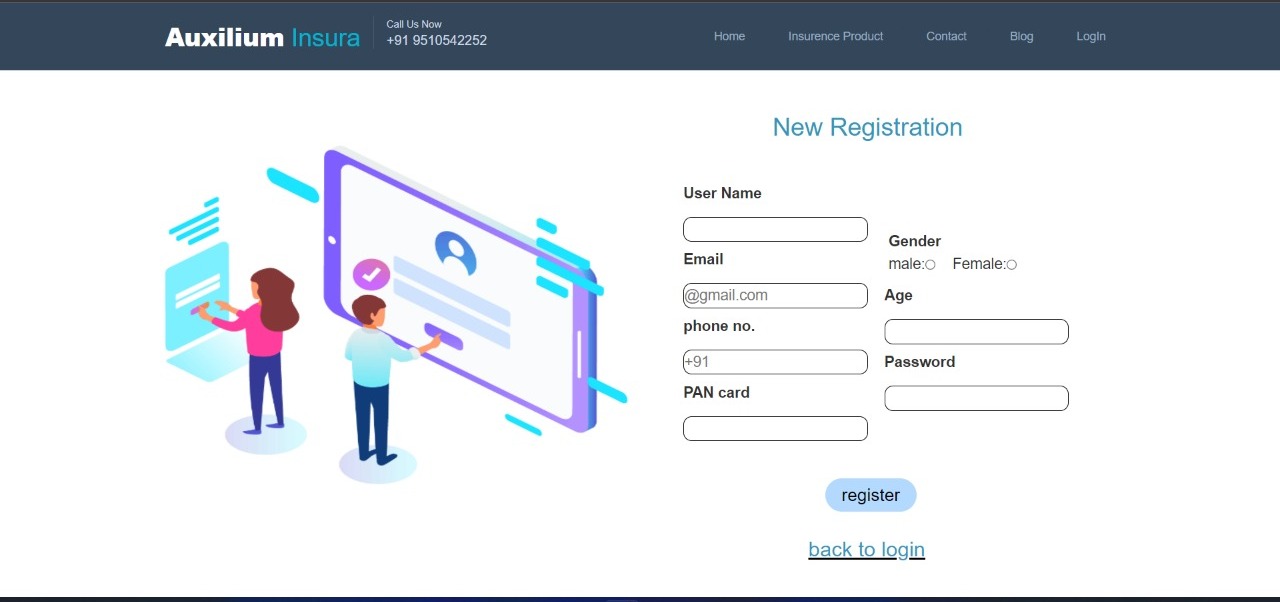


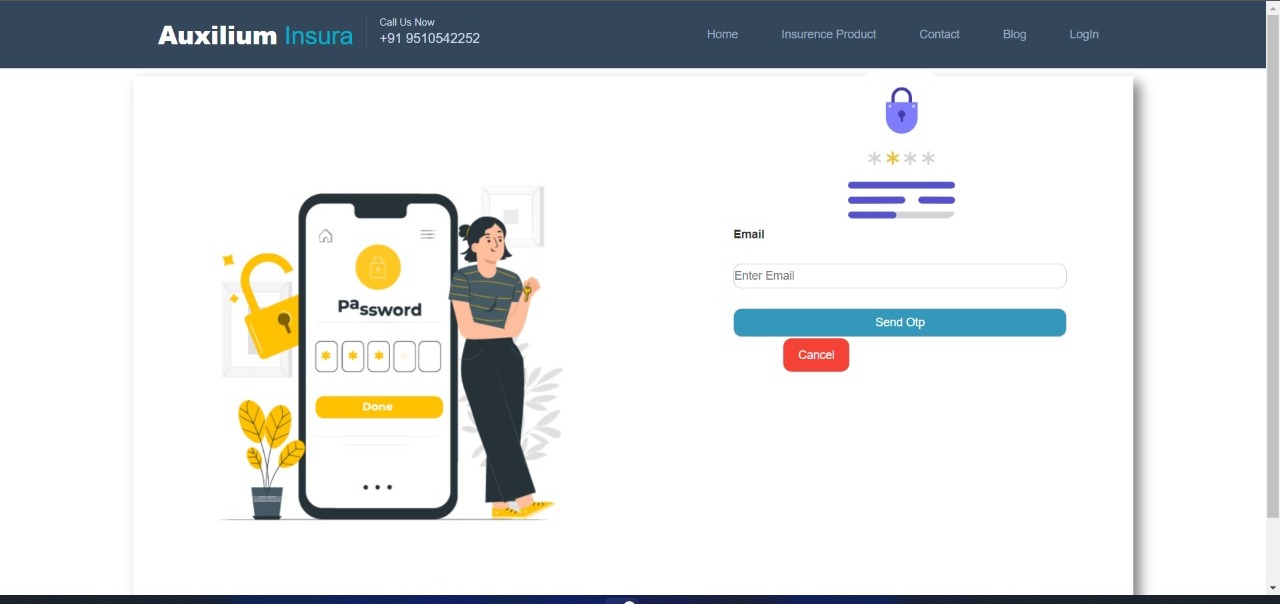


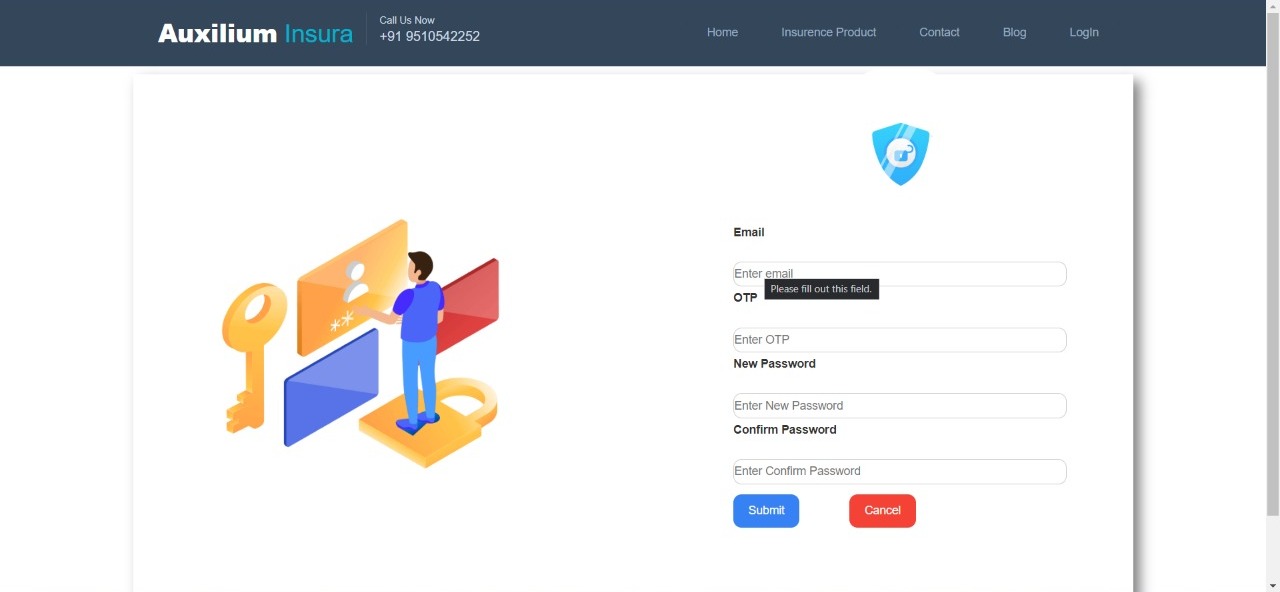


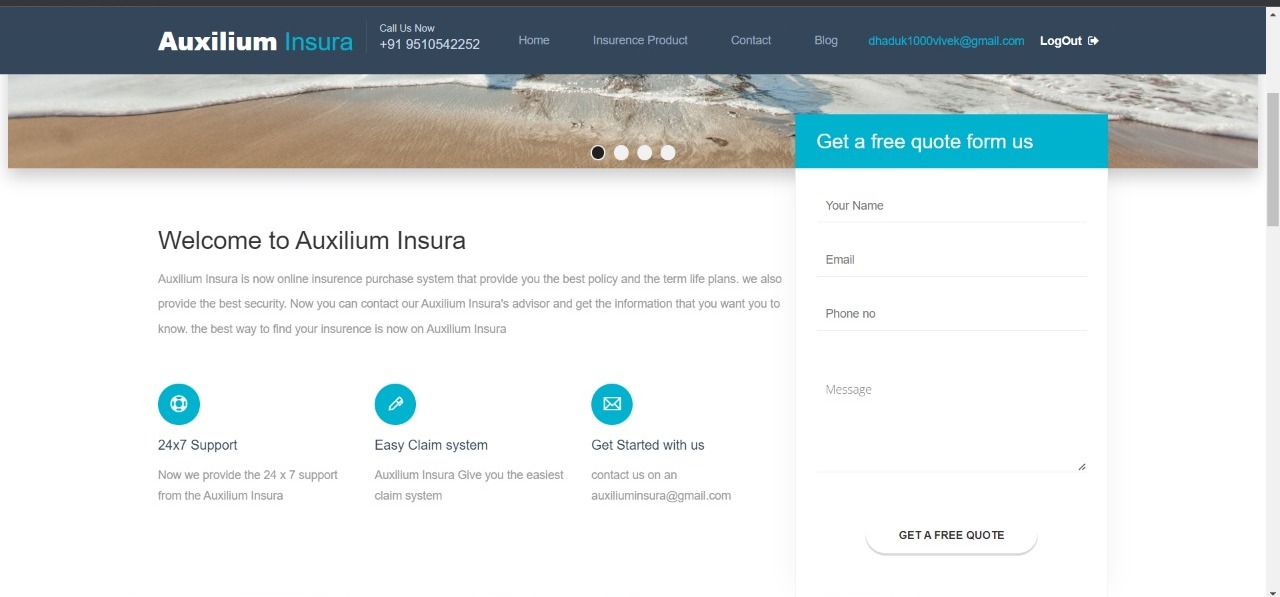
* **User-side input design**

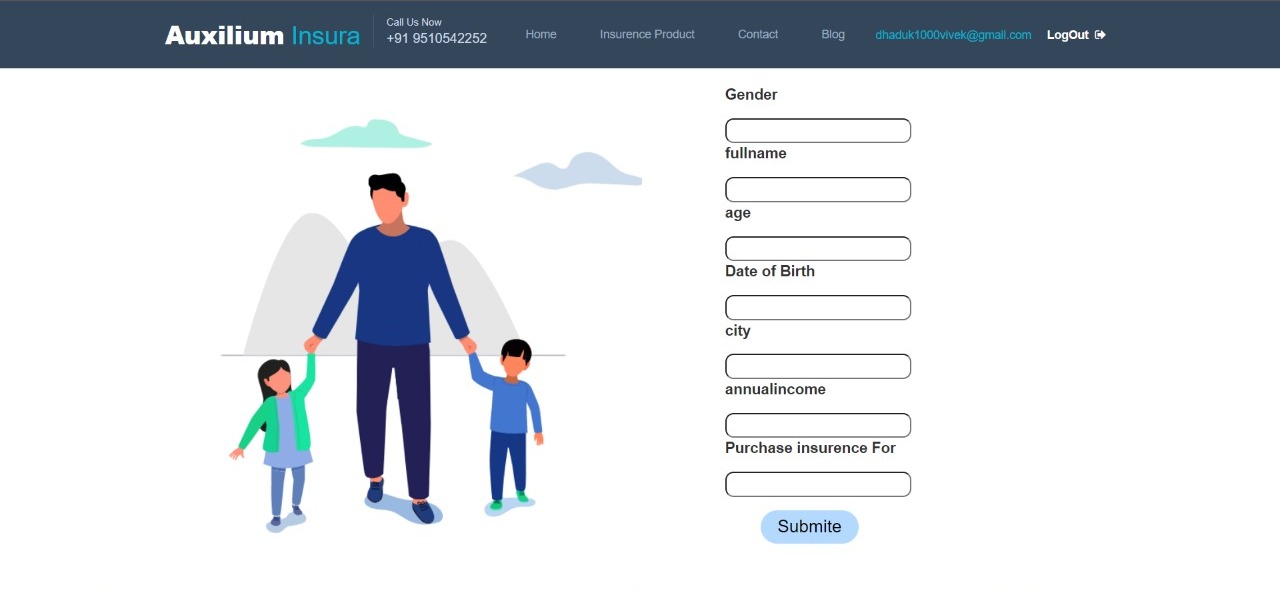






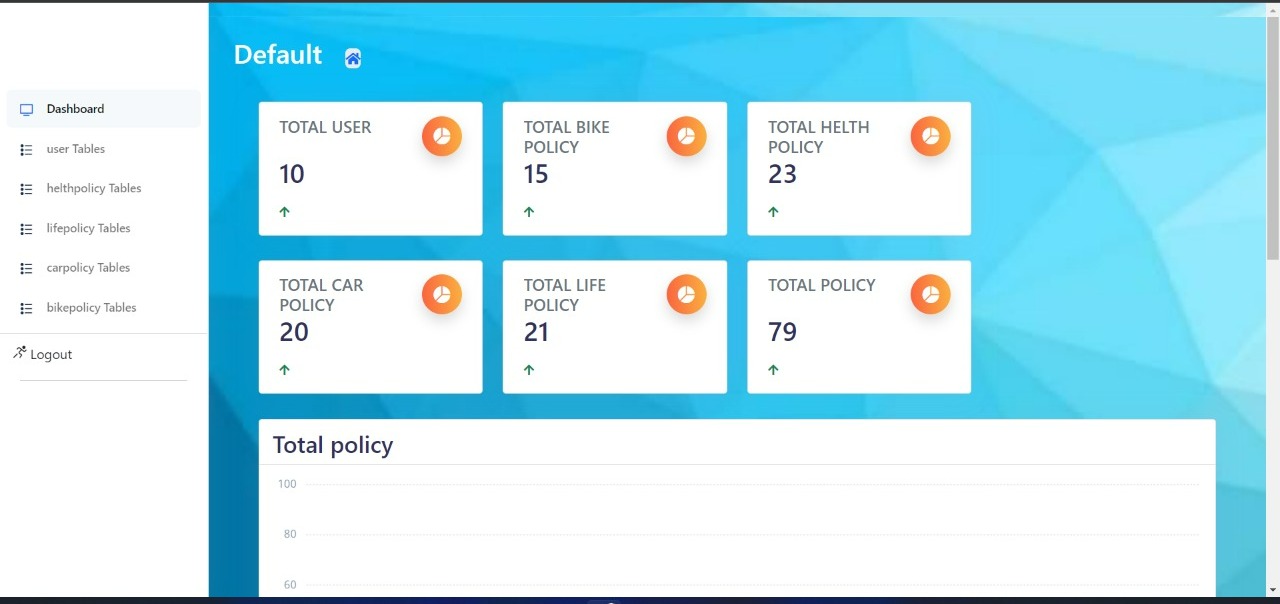


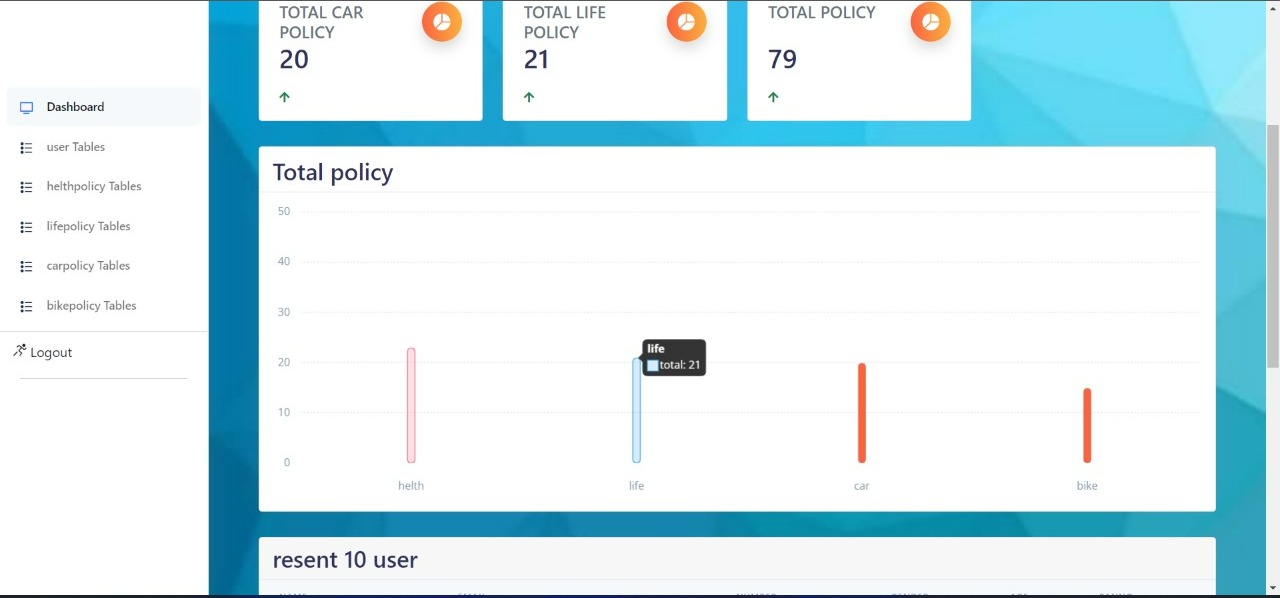


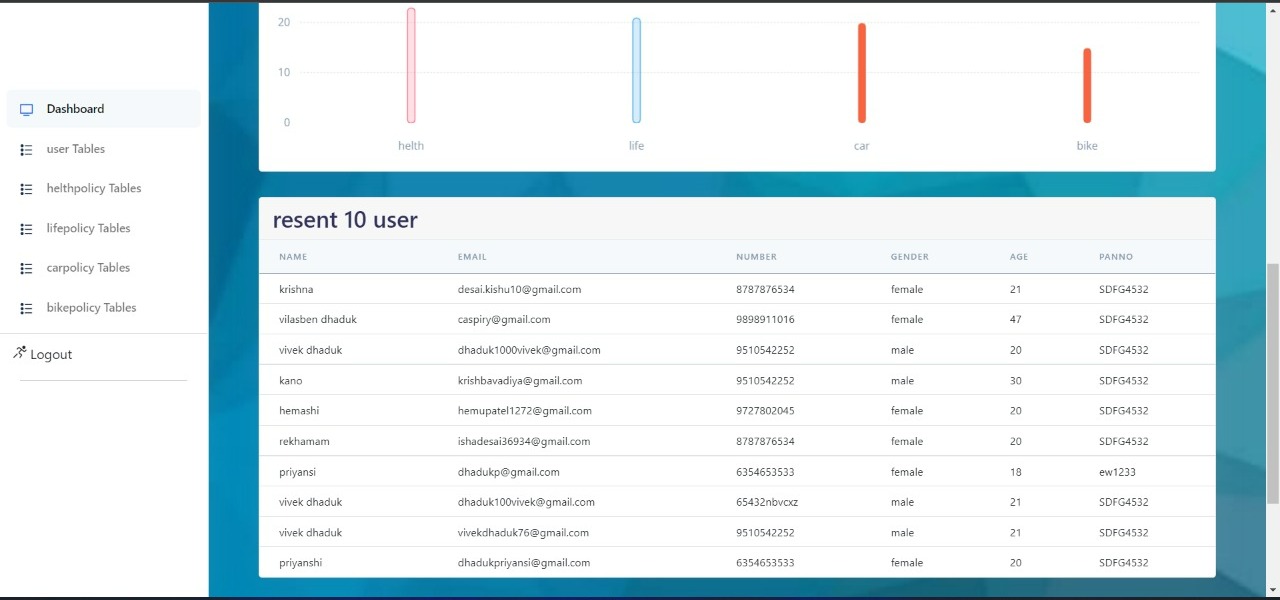


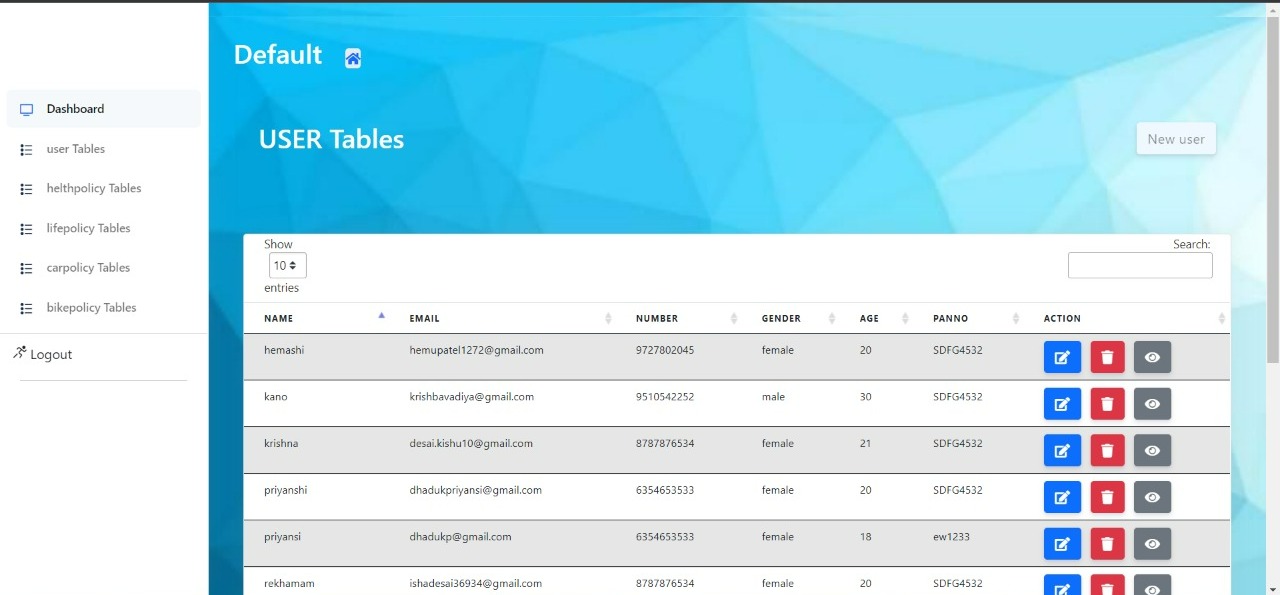
* **Output Design:-**

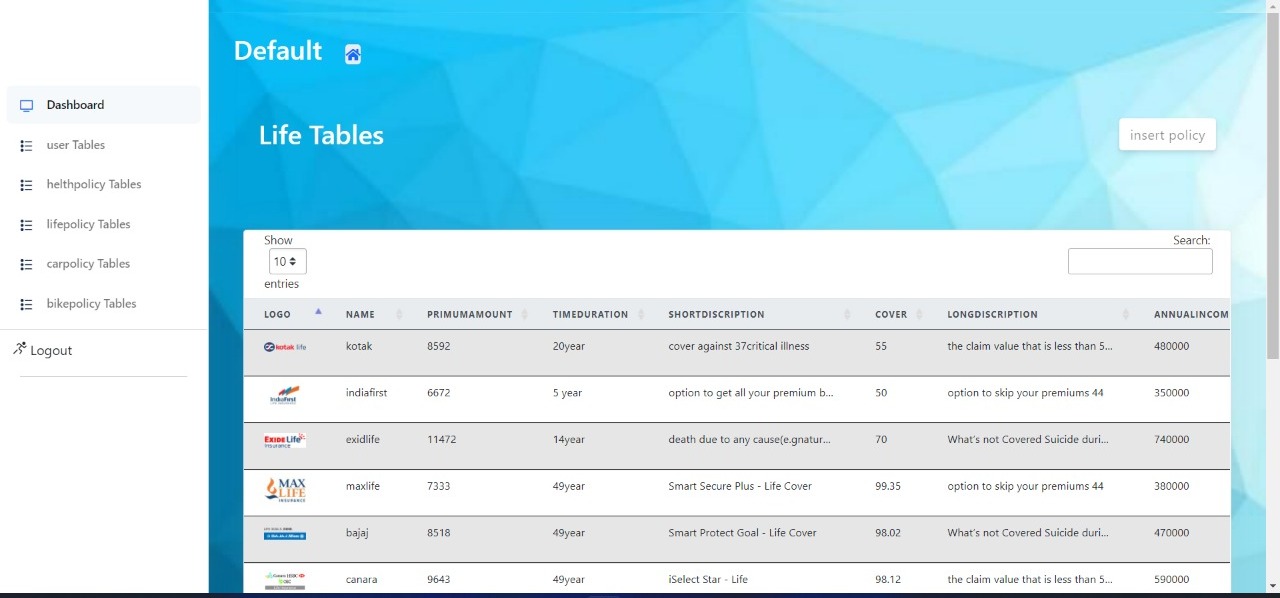
* **Admin-side output design**



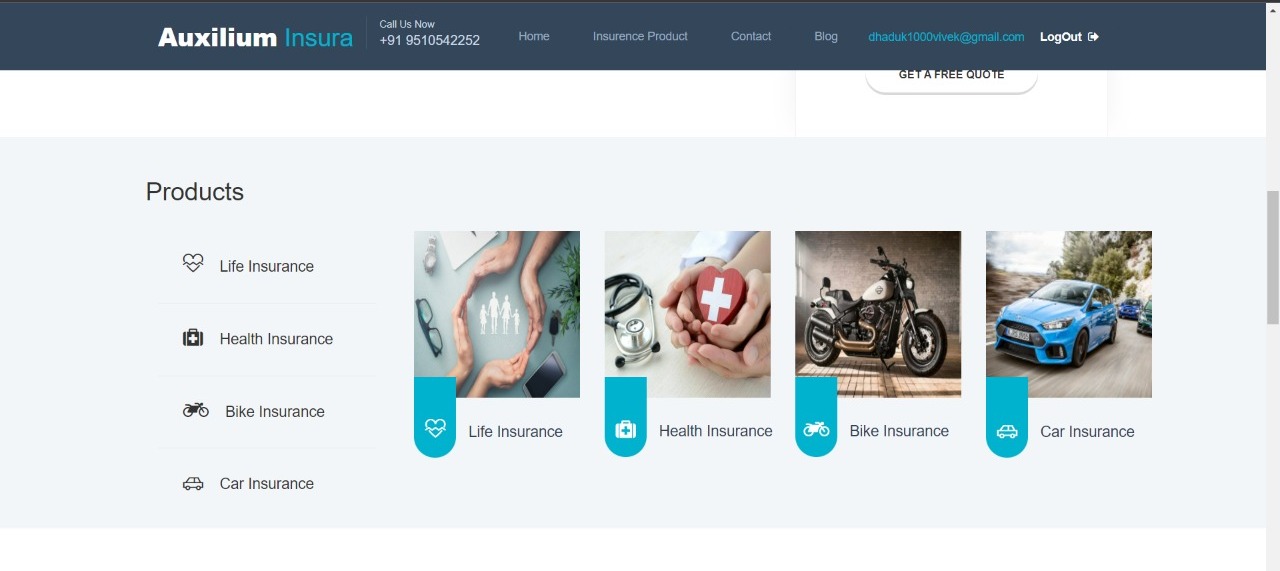


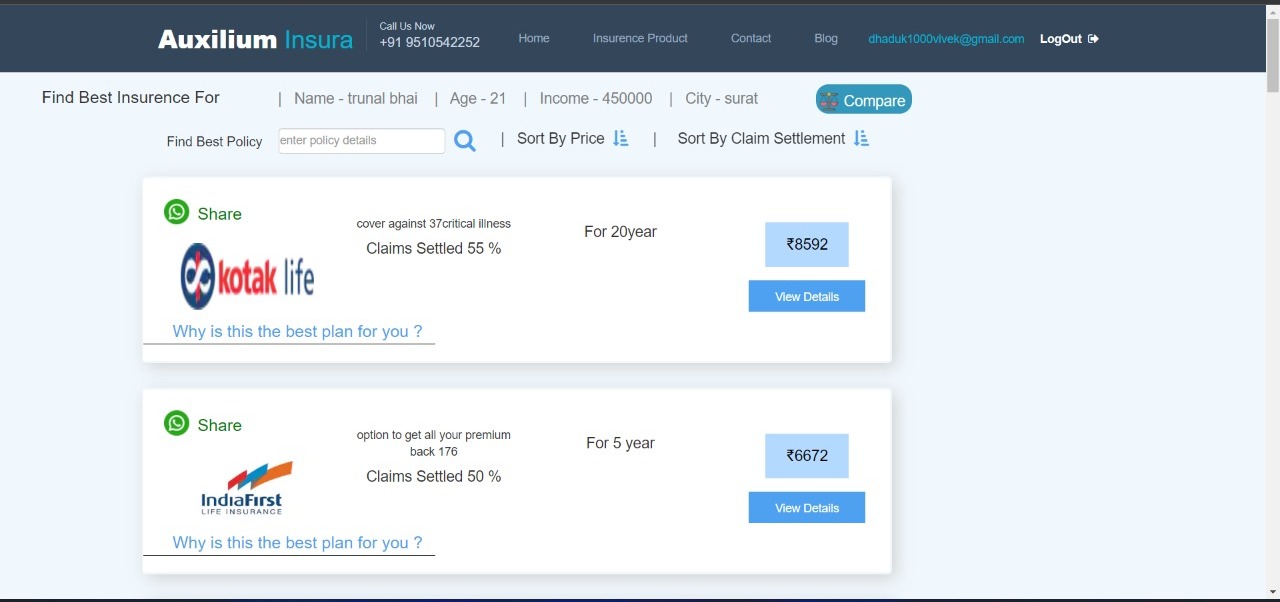


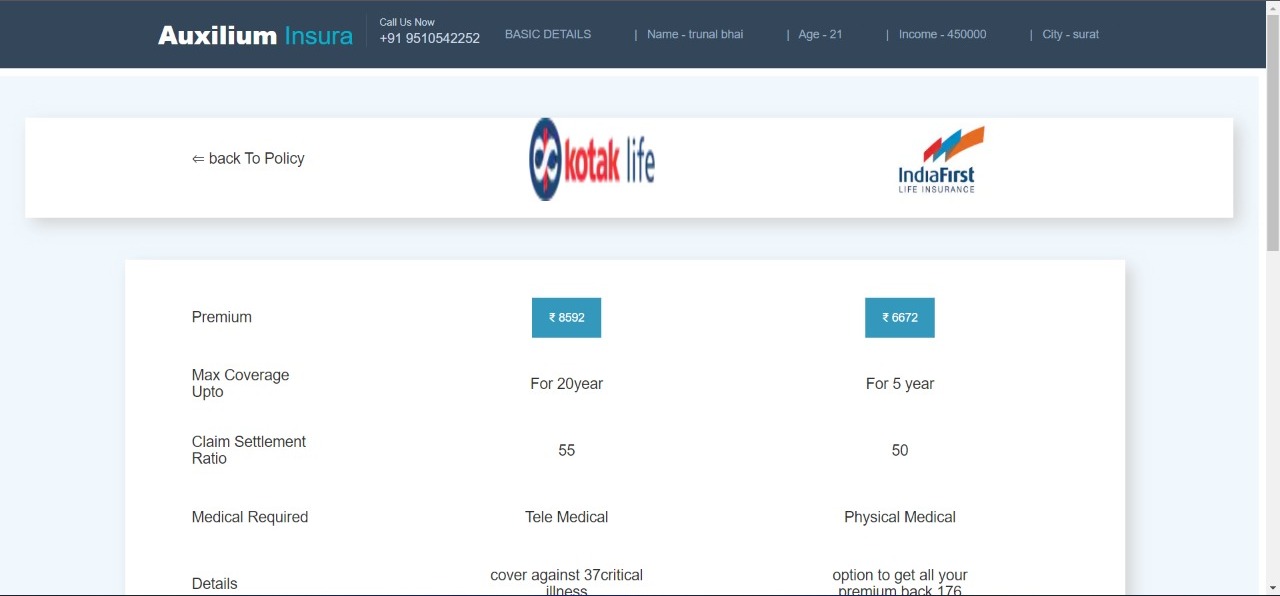


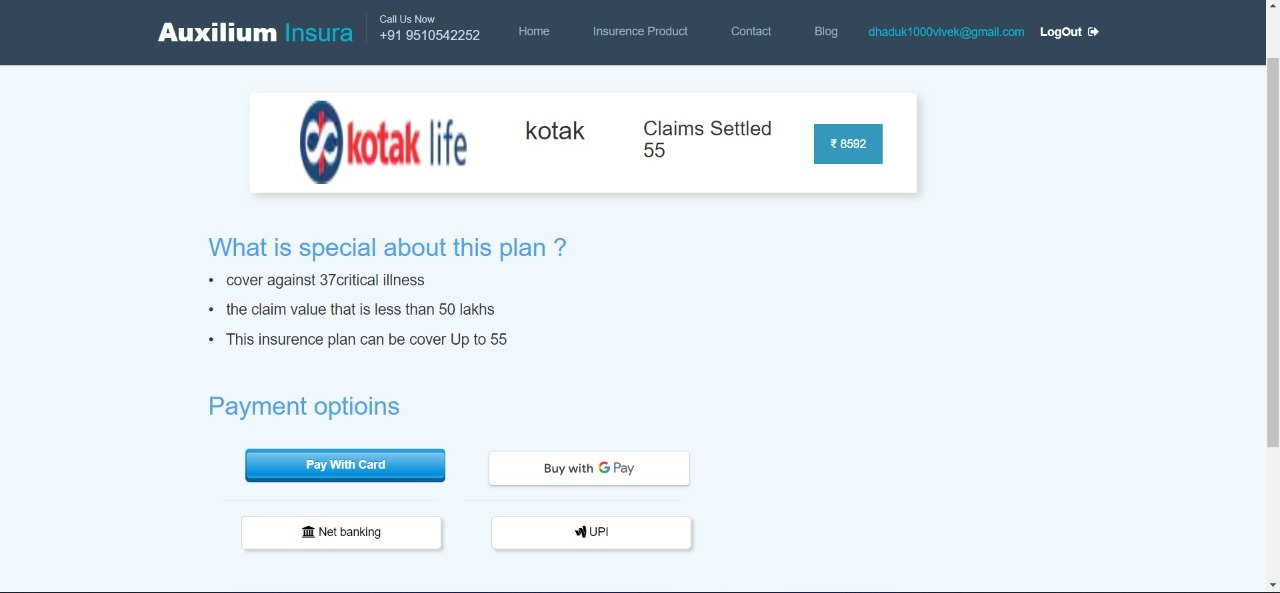


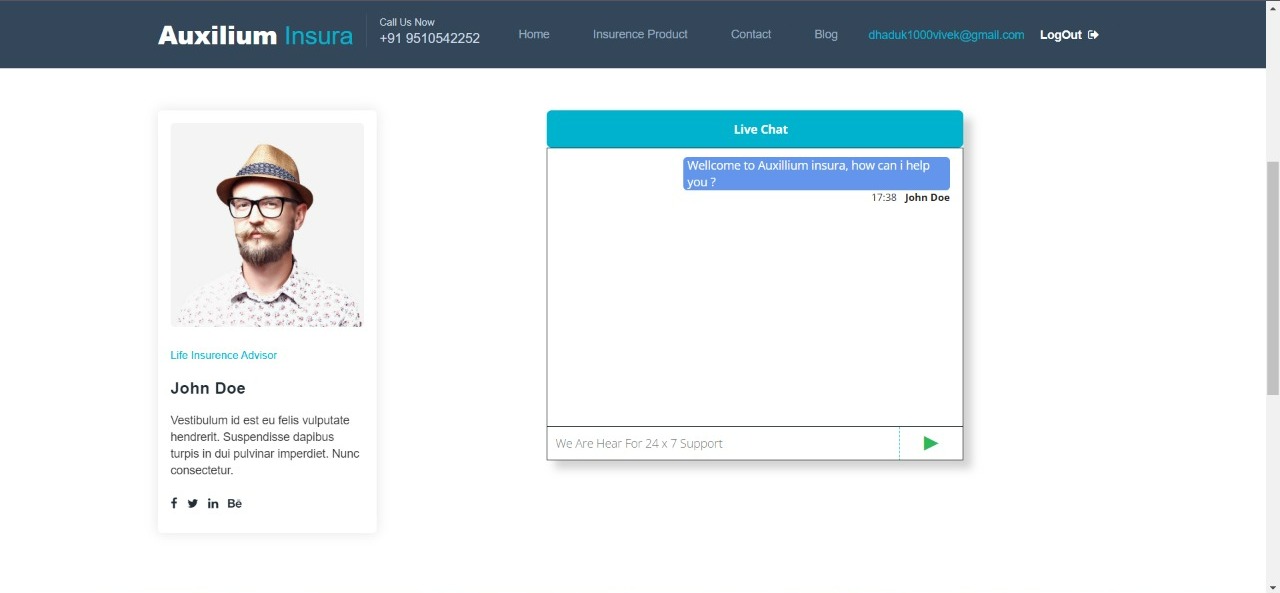
* **User-side output design**

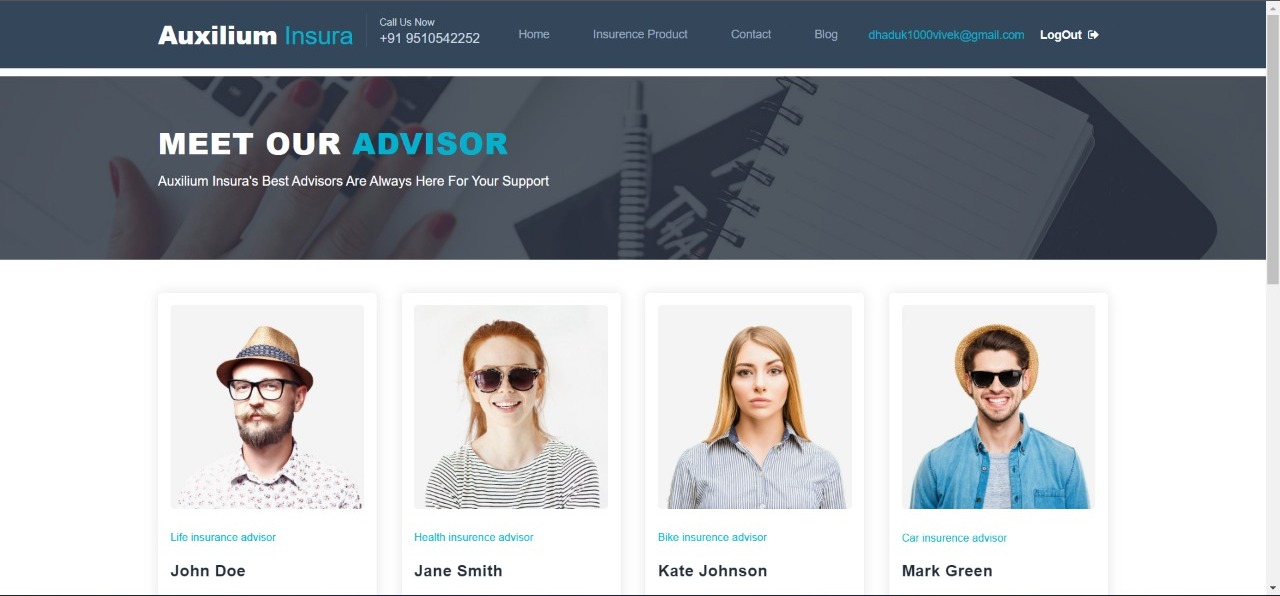


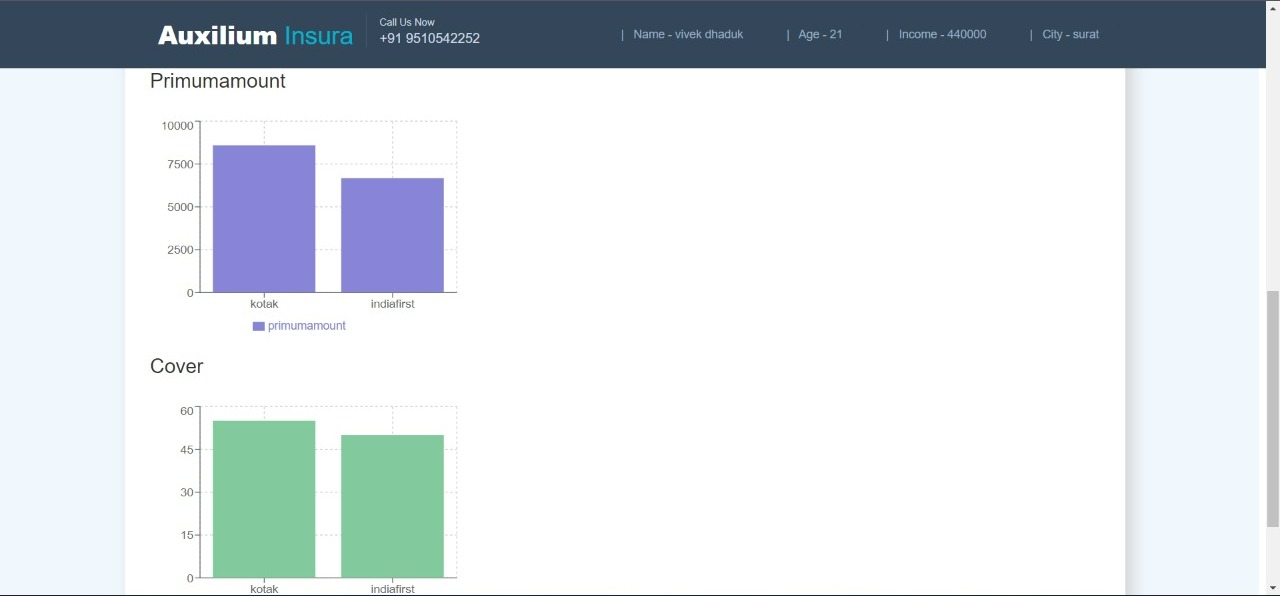










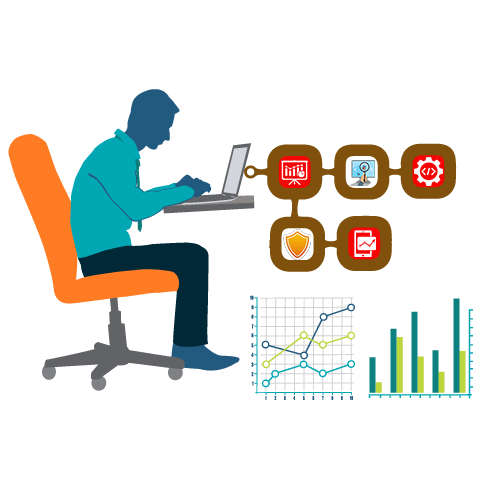


**6 .Software Testing**

* **Software Testing**

* **Software Testing:-**

* Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing is an exposure of a system to trial input to see whether the software meets the correct output. Testing cannot be determined whether the software meets the user’s needs, only whether it appears to conform to requirements. Testing can show that a system is free of errors, only that it contains errors. Testing finds errors, it does not correct errors. Software success is a quality product, on time and within cost. Testing can reveal critical mistakes. Testing should, therefore,
* Validate Performance.
* Detects errors.
* Identify.
* Inconsistencies.



* **Test Objective:**
* There Is Strong Evidence That Effective Requirement Management Leads to Overall Project Cost Savings. The Three Primary Reasons For This Are:
* Errors in requirement typically cost over 10 times more to repair than other errors.
* Requirement errors typically comprise over 40% of all errors in a software project.
* The Testing Procedure Should Care For All Of These, As Well, In Order To Attain A Flawless, Error-Free, And Efficient Functioning System; Too, Software Testing Is An Important Phase Of Any Software Development Life Cycle. Various Reports and Data Used For The Same Are The Core Of The System. The Testing, Therefore, Becomes Important In Order To Maintain The Cost As Well As Improve Performance And Consistency. The Testing Procedure For The System Has Been Divided Into Various Parts Ranging For Single Unit Testing To Entire System Testing.
* **Testing Principles:**
* All tests should be traceable to customer requirements.
* Tests should be planned long before testing begins.
* The Pareto principle applies to software testing.
* Testing should begin “in the small” and progress toward testing “in the large”.
* Exhaustive testing is not possible.
* **Unit Testing:**
* Unit Testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of the software. It usually has one or a few inputs and usually a single output.
* **Integration Testing:**
* **Integration Testing** is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.
* **Top-down integration:**
* Top-down integration testing is an integration testing technique used in order to simulate the behavior of the lower-level modules that are not yet integrated. Stubs are the modules that act as a temporary replacement for a called module and give the same output as that of the actual product.
* **Bottom-down integration:**
* Bottom-up testing is a specific type of integration testing that tests the lowest components of a codebase first. More generally, it refers to a middle phase in software testing that involves taking integrated code units and testing them together, before testing an entire system or code base.
* **Validation Testing :**
* The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements.
* Validation Testing ensures that the product actually meets the client's needs. It can also be defined as demonstrating that the product fulfills its intended use when deployed in an appropriate environment.
* **System Testing:**
* System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.
* **Recovery Testing :**
* Recovery testing is a type of non-functional testing technique performed in order to determine how quickly the system can recover after it has gone through a system crash or hardware failure. Recovery testing is the forced failure of the software to verify if the recovery is successful.
* **Security Testing :**
* Security testing is a process intended to reveal flaws in the security mechanisms of an information system that protect data and maintain functionality as intended.
* **Stress Testing :**
* Stress testing is a Non-Functional testing technique that is performed as part of performance testing. During stress testing, the system is monitored after subjecting system to overload to ensure that the system can sustain the stress.
* **Sanity Testing :**
* Sanity testing is the subset of regression testing and it is performed when we do not have enough time for doing testing. Sanity testing is surface-level testing where the QA engineer verifies that all the menus, functions, and commands available in the product and project are working fine.
* **White Box Testing :**
* White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing [software](https://en.wikipedia.org/wiki/Software) that tests the internal structures or workings of an application, as opposed to its functionality (i.e. [black-box testing](https://en.wikipedia.org/wiki/Black-box_testing)).
* In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases.
* The tester chooses inputs to exercise paths through the code and determines the expected outputs.
* This is analogous to testing nodes in a circuit, e.g. [in-circuit testing](https://en.wikipedia.org/wiki/In-circuit_test) (ICT). White-box testing can be applied at the [unit](https://en.wikipedia.org/wiki/Unit_testing), [integration](https://en.wikipedia.org/wiki/Integration_testing), and [system](https://en.wikipedia.org/wiki/System_testing) levels of the software testing process.
* Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today.
* It can test paths within a unit, paths between units during integration, and between subsystems during a system-level test.
* Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

1. **Limitations and future Scope of Enhancements, References**

* **Limitations of the system:-**

* Since every system has some limitations so our proposed system is also not untouchable in this regard.
* The main limitations of our system are:
* Excel export has not been developed for the policy.
* admin cannot update logo image once it is uploaded.

* **Future scope of enhancements:-**

* The social networking site has a wide range to change as per time requirements. That is the reason that not a particular project is ever considered complete because the demand of the user and thinking always change day by day.
* Always do more than what have.
* Some of the enhance which we have thought of are:
* We can give more advanced software for online insurance management systems including more facilities.
* We will host the platform on an online server to make it accessible worldwide.
* Integrate multiple load balancers to distribute the load of the server.
* Implement the backup mechanism for taking backup of codebase and database on regular basis on different servers.
* Create a master and slave database structure to reduce the overload of the database queries.
* User can create his own business page for advertisement.
* Payment of the advertisement in online mode.
* Development of mobile application which can run on multiple OS and devices.

* **References:-**

* <https://stackoverflow.com/>