CHAPTER 1

INTRODUCTION

An Online Vehicle Insurance management system is a web application is developed for tracking the details of the insurance policy, customer details and company details. This website is an online insurance Analysis and information management system that provides easy access to details when login into the Policy Holder module. This project is useful for any kind of insurance company to manage the insurance details, to sanctioned the insurance for the customer, process the insurance policy details and all kind of insurance process through online.

The Vehicle Insurance management system is a complete solution for organizations, which need to manage insurance for their vehicles, equipment, buildings, and other resources. This insurance management website has facilities like search tools for insurance awareness articles, guidelines, illustrations through images for visitors. This Vehicle Insurance management system can efficiently manage the company, records, provides instant access and improves productivity. In this online process the user enters into the website it will show details about insurance and its types, also it will show the details about different duration schemes to the corresponding insurance type or insurance policy.

The main objective of the developed system is to allow admin users to register insured persons with their name, date of birth, residence address and also policy details. This process contains the user registration form which is used to apply for an insurance policy online. It also helps the customer to view their own insurance status information. After giving registering all the insured persons, the website should provide management facilities like delete unwanted persons' data. And also, should provide awareness to the visitors about microinsurance through articles. If the user registered an insurance policy to this website, it will process that registration form by verification and immediately give the temporary policyholder ID to the user.

The website provides easy links for easy navigation in the site. After submission of the registration form, the admin will process to verify that particular details are registered by the customer and sanctioned the insurance policy. A visitor with minimum knowledge of web browsing/surfing can access the site very easily.

CHAPTER 2

REQUIREMENT SPECIFICATION

The requirement scan be broken down into two major categories namely Hardware and Software requirements. The formal specification the minimal hardware expected in a system in which the project has to be run. The latter specification the essential software needed to build and run the project.

2.1 Hardware Specification:

The hardware requirements are minimal and software can run with minimal requirements

- Ram : 1 GB or higher
- Hard-disk : 30 GB or more
- Processor : Dual core or Core 2 Due (i3 recommended)

2.2 Software Specification:

A. Operating System : Windows 7 or higher, Mac (Mountain Lion or higher), Linux

B. IDE or Text-Editor : Sublime, VSCode or Notepad ++

C. Browser : Chrome, Safari or Firefox

D. Other Software ; Xampp , Wamp or Mamp

2.3 Functional Requirements:

1. **User Management:**
    - Registration: Users (customers, agents, administrators) should be able to register with the system.
    - Login: Authenticated users should be able to log in securely.
2. **Policy Management:**
    - Create Policy: Agents should be able to create new insurance policies for customers.
    - Modify Policy: Agents and administrators should be able to modify existing policies.

- Cancel Policy: Agents and administrators should be able to cancel policies.

3. **Quote Generation:**
   - Calculate Premium: System should calculate insurance premium based on factors like vehicle type, age, usage, etc.
   - Provide Quotes: Users should be able to receive insurance quotes based on their inputs.
4. **Claims Management:**
   - Submit Claim: Customers should be able to submit insurance claims online.
   - Track Claim Status: Users should be able to track the status of their insurance claims.
   - Process Claims: Agents and administrators should be able to process and approve claims.
5. **Payments and Billing:**
   - Payment Processing: Users should be able to make premium payments securely online.
   - Billing History: Users should have access to their billing history and invoices.
6. **Vehicle Management:**
   - Register Vehicle: Customers should be able to register their vehicles for insurance.
   - Update Vehicle Information: Users should be able to update vehicle information such as registration number, model, etc.
7. **Reports and Analytics:**
   - Generate Reports: Agents and administrators should be able to generate various reports such as sales reports, claim reports, etc.
   - Analytics: The system should provide analytics on policy sales, claims processing times, etc.

2.4 Non-Functional Requirements:

1. **Security:**
   - Data Encryption: All sensitive data should be encrypted both in transit and at rest.

- Access Control: Role-based access control (RBAC) should be implemented to ensure that users only have access to the information they are authorized to view.
- Secure Authentication: Use of secure authentication methods like multi-factor authentication (MFA).

2. **Performance:**
   - Response Time: The system should respond to user requests within acceptable time limits.
   - Scalability: The system should be able to handle increased loads during peak times without significant degradation in performance.
3. **Reliability:**
   - High Availability: The system should be available 24/7 with minimal downtime for maintenance.
   - Disaster Recovery: There should be mechanisms in place to recover data in case of system failures or disasters.
4. **Usability:**
   - Intuitive Interface: The user interface should be user-friendly and intuitive, requiring minimal training for users to navigate.
   - Accessibility: The system should be accessible to users with disabilities, following accessibility standards.
5. **Compatibility:**
   - Cross-Platform Compatibility: The system should be compatible with different devices and operating systems.
   - Browser Compatibility: The system should work seamlessly across different web browsers.
6. **Regulatory Compliance:**
   - Compliance with Regulations: The system should comply with relevant regulations and standards in the insurance industry.
   - Data Protection: Compliance with data protection laws such as GDPR, ensuring user data privacy.
7. **Scalability:**
   - The system should be designed to handle a large number of users, policies, and claims efficiently.

CHAPTER 3

SYSTEM DESIGN

3.1 Entity-Relationship Diagram

The entity-relationship diagram, also known as the E-R Diagram, is a high level database design, which shows the database in diagrammatic approach. It consists of entities ,relationships, attributes and associations. The E-R Diagram for the project is shown in the figure 3.1 below:

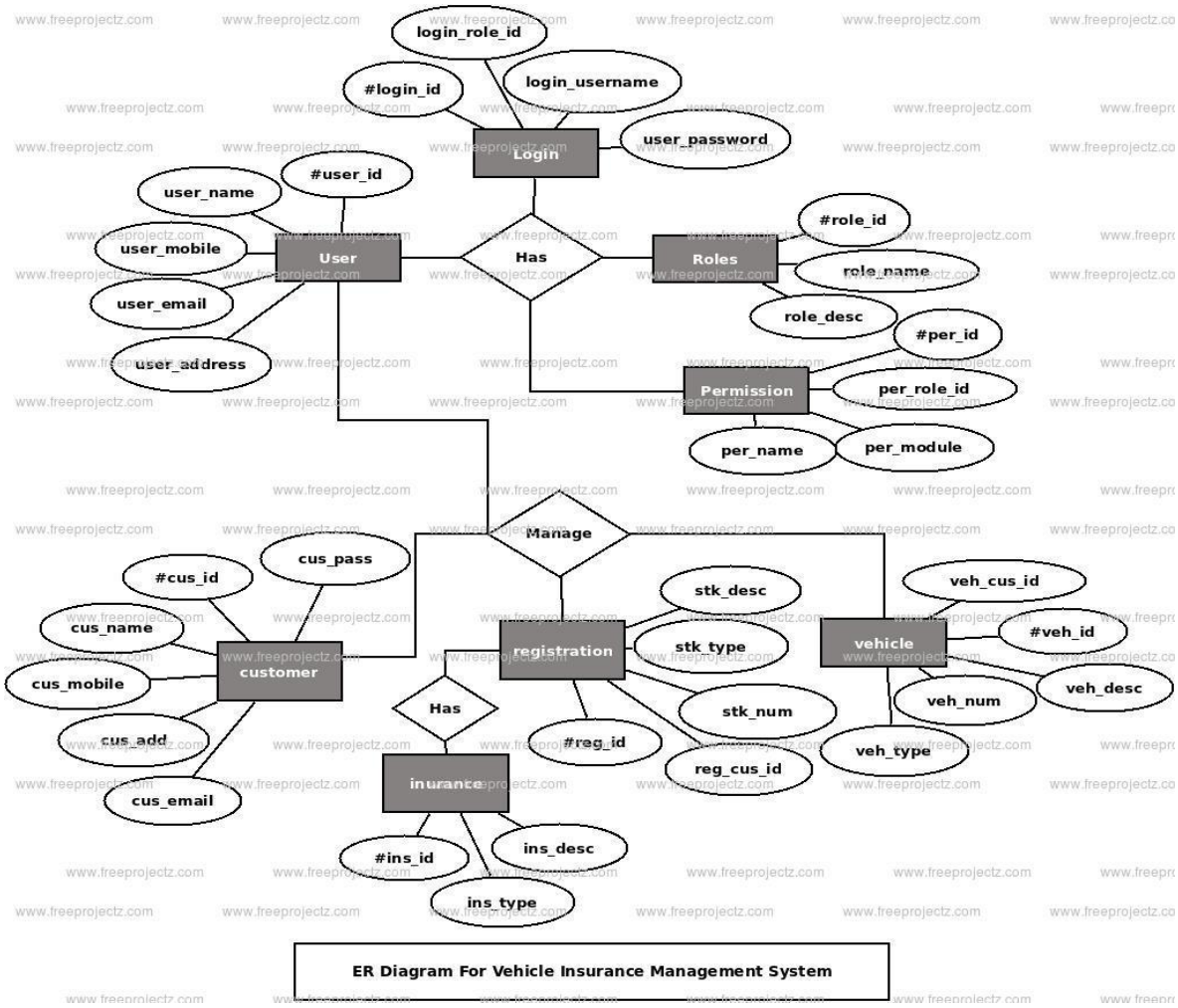


Figure 3.1: E-R Diagram of Vehicle Insurance Management system

3.2 Schema Diagram

A schema diagram is an illustrative display of most aspects of a database schema. A schema construct is a component of the schema, or an object within the schema. The schema diagram of the database system is illustrated in figure 3.2:

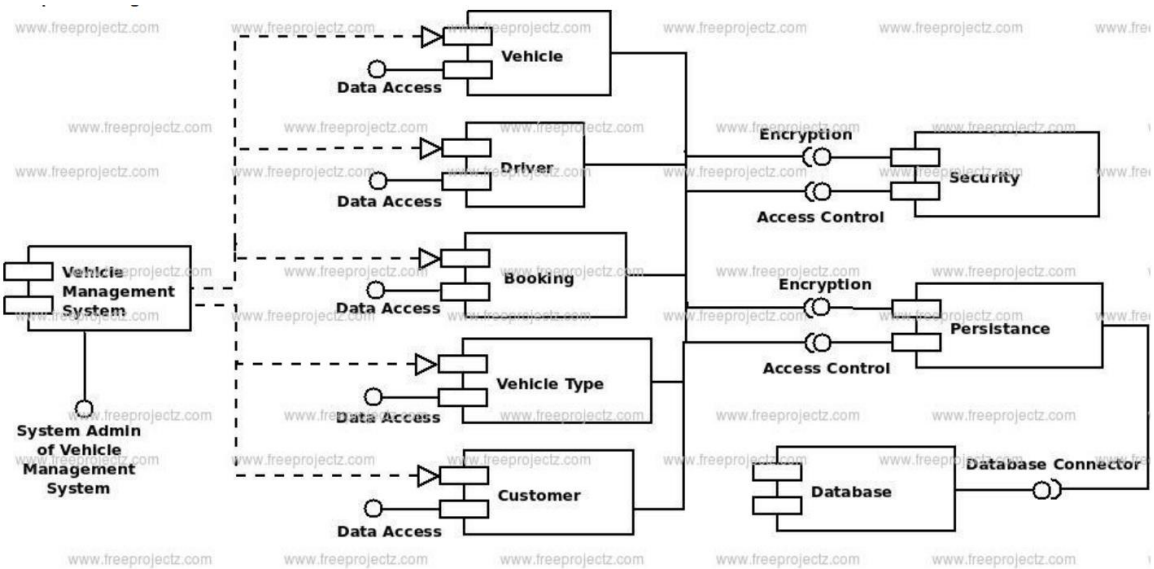


Figure 3.2: Schema Diagram of Vehicle Insurance Management system

CHAPTER 4

IMPLEMENTATION

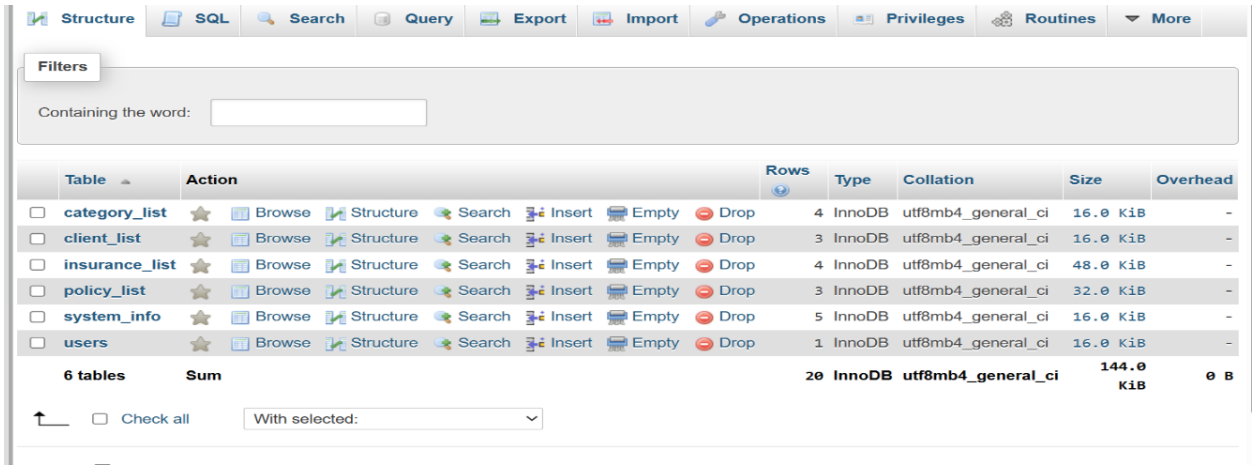
Databases are the storehouses of data used in the software systems. The data is stored intables inside the database. Several tables are created for the manipulation of the datafor the system. Two essential settings for a database are.

Primary key- the field that is unique for all the record occurrences. Foreign key-the field used to set relation between tables.

4.1 Creation of tables:

The Tables created are:

- Category_list
- Client_list
- Insurance_list
- Policy_list
- System_info
- User



Category_list

category_list

Column	Type	Null	Default	Links to	Comments	Media type
id *(Primary)*	int(30)	No				
name	text	No				
description	text	No				
status	tinyint(1)	No	1			
delete_flag	tinyint(1)	No	0			
date_created	datetime	No	current_timestamp()			
date_updated	datetime	Yes	*NULL*			

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	4	A	No	

Figure 4.1.1: Creation of Category_list

Client_list

client_list

Column	Type	Null	Default	Links to	Comments	Media type
id *(Primary)*	int(30)	No				
code	varchar(100)	No				
firstname	text	No				
middlename	text	No				
lastname	text	No				
gender	varchar(100)	No				
dob	date	No				
contact	text	No				
email	text	No				
address	text	No				
image_path	text	Yes	*NULL*			
status	tinyint(4)	No	1			
delete_flag	tinyint(4)	No	0			
date_created	datetime	No	current_timestamp()			
date_updated	datetime	Yes	*NULL*			

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	3	A	No	

Figure 4.1.2: Creation of Client_list

Insurance_list

insurance_list

Column	Type	Null	Default	Links to	Comments	Media type
id *(Primary)*	int(30)	No				
client_id	int(30)	No		client_list -> id		
policy_id	int(30)	No		policy_list -> id		
code	varchar(100)	No				
registration_no	text	No				
chassis_no	text	No				
vehicle_model	text	No				
registration_date	date	No				
expiration_date	date	No				
cost	float	No	0			
status	tinyint(4)	No	1			
date_created	datetime	No	current_timestamp()			
date_updated	datetime	Yes	*NULL*			

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	4	A	No	
client_id	BTREE	No	No	client_id	4	A	No	
policy_id	BTREE	No	No	policy_id	4	A	No	

Figure 4.1.3: Creation of insurence_list

Policy_list

policy_list

Column	Type	Null	Default	Links to	Comments	Media type
id *(Primary)*	int(30)	No				
category_id	int(30)	No		category_list -> id		
code	varchar(100)	No				
name	text	No				
description	text	No				
duration	float	No	0			
cost	float	No	0			
doc_path	text	Yes	*NULL*			
status	tinyint(4)	No	1			
delete_flag	tinyint(4)	No	0			
date_created	datetime	No	current_timestamp()			
date_updated	datetime	Yes	*NULL*			

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	3	A	No	
category_id	BTREE	No	No	category_id	3	A	No	

Figure 4.1.4: Creation of policy_list

System_info

system_info

Column	Type	Null	Default	Links to	Comments	Media type
id *(Primary)*	int(30)	No				
meta_field	text	No				
meta_value	text	No				

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	5	A	No	

Figure 4.1.5: Creation of system_info

Users

users

Column	Type	Null	Default	Links to	Comments	Media type
id *(Primary)*	int(50)	No				
firstname	varchar(250)	No				
middlename	text	Yes	*NULL*			
lastname	varchar(250)	No				
username	text	No				
password	text	No				
avatar	text	Yes	*NULL*			
last_login	datetime	Yes	*NULL*			
type	tinyint(1)	No	0			
status	int(1)	No	1		0=not verified, 1 = verified	
date_added	datetime	No	current_timestamp()			
date_updated	datetime	Yes	*NULL*			

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	2	A	No	

Figure 4.1.6: Creation of user

4.2 Insertion of values:

Category_list

```
SELECT * FROM `category_list`
```

id	name	description	status	delete_flag	date_created	date_updated
1	Commercial	Lorem ipsum dolor sit amet, consectetur adipiscing...	1	0	2022-02-03 08:52:50	2022-02-03 08:53:09
2	2 wheeler	Integer auctor at mauris a dapibus. Donec id posue...	1	0	2022-02-03 08:53:32	*NULL*
3	3 Wheeler	Sed at leo vel felis pellentesque scelerisque. Nun...	1	0	2022-02-03 08:54:52	*NULL*
4	4 Wheeler	Quisque at erat at ipsum mollis viverra. Quisque i...	1	0	2022-02-03 08:56:25	*NULL*

Figure 4.2.1: Insertion of Category_list

Client_list

```
SELECT * FROM `client_list`
```

id	code	firstname	middlename	lastname	gender	dob	contact	email	address	image_path	status	delete_flag	date_created	date_updated
1	202202-00001	Mark	D	Cooper	Male	1997-06-23	09123456789	mcooper@sample.com	Block 6 Lot 23, Here Subd., Over There, Anywhere 2...	uploads/clients/1.png?v=1643856423	1	0	2022-02-03 10:45:56	2022-02-03 10:47:03
2	202202-00002	Claire	C	Blake	Male	1997-10-14	09456789312	cblake@sample.com	This is her sample address.	uploads/clients/2.png?v=1643859659	1	0	2022-02-03 11:40:59	2022-02-03 11:40:59
3	202202-00003	Test12312	Test	Test	Male	2022-02-14	0956465798798	test@samnple.com	Test	*NULL*	0	1	2022-02-03 11:41:25	2022-02-03 11:42:47

Figure 4.2.2: Insertion of Client_list

Insurance_list

SELECT * FROM `insurance_list`

id	client_id	policy_id	code	registration_no	chassis_no	vehicle_model	registration_date	expiration_date	cost	status	date_created	date_updated
1	1	2	202202-00001	GBN-23141507	123654789	Sample Vehicle 101	2022-02-03	2025-02-03	4999.99	1	2022-02-03 13:49:19	2022-02-03 14:43:54
3	1	1	202202-00002	654985158	5489798558	Sample 74844158	2022-02-08	2023-02-08	1500	1	2022-02-03 15:16:15	NULL
4	2	3	202202-00003	8798754564	8789564687	Sample Vehicle	2022-01-29	2023-01-29	4000	1	2022-02-03 15:18:47	NULL
5	2	3	202202-00004	8798754564	8789564687	Sample Vehicle 101	2015-02-18	2016-02-18	4000	1	2022-02-03 15:19:13	NULL

Figure 4.2.3: Insertion of insurence_list

Policy_list

SELECT * FROM `policy_list`

id	category_id	code	name	description	duration	cost	doc_path	status	delete_flag	date_created	date_updated
1	2	123456789	Policy101	Sed eget egestas dolor, id mattis ante. Duis non d...	1	1500	uploads/policies/1.pdf?v=1643853360	1	0	2022-02-03 09:34:32	2022-02-03 09:56:00
2	4	23061415	Policy102	In bibendum lacus eget purus luctus varius. Intege...	3	4999.99	uploads/policies/2.pdf?v=1643853724	1	0	2022-02-03 10:02:04	2022-02-03 10:02:04
3	1	987654321	Commercial Policy1001	Integer scelerisque sapien non molestie gravida. V...	1	4000	uploads/policies/3.pdf?v=1643853870	1	0	2022-02-03 10:04:30	2022-02-03 10:04:30

Figure 4.2.4: Insertion of policy_list

System_info

SELECT * FROM `system_info`

id	meta_field	meta_value
1	name	Vehicle Insurance Management System
6	short_name	VIMS - PHP
11	logo	uploads/logo-1643849196.png
13	user_avatar	uploads/user_avatar.jpg
14	cover	uploads/cover-1643849034.png

Figure 4.2,5: Insertion of system_info

Users

Showing rows 0 - 0 (1 total, Query took 0.0004 seconds.)

SELECT * FROM `users`

id	firstname	middlename	lastname	username	password	avatar	last_login	type	status (Email verified, 1 = verified)	date_added	date_updated
1	Administrator	NULL	Admin	admin	0192023a7bbd73250516f069df18b500	uploads/clients/1.png?v=1643856356	NULL	1	1	2021-01-20 14:02:37	2022-02-03 10:45:56

Figure 4.2.6: Insertion of user

4.3 Qureies:

1.Retrieve the details of Category_list

SELECT * from category_list WHERE id= 1;

id	name	description	status	delete_flag	date_created	date_updated
1	Commercial	Lorem ipsum dolor sit amet, consectetur adipiscing...	1	0	2022-02-03 08:52:50	2022-02-03 08:53:09

2. Updating Client_list .

UPDATE client_list SET client_list.status=2 where client_list.id=1;

Before updation:

SELECT * FROM `client_list`

id	code	firstname	middlename	lastname	gender	dob	contact	email	address	image_path	status	delete_flag	date_created	date_updated
1	202202-00001	Mark	D	Cooper	Male	1997-06-23	09123456789	mcooper@sample.com	Block 6 Lot 23, Here Subd., Over There, Anywhere 2...	uploads/clients/1.png?v=1643856423	1	0	2022-02-03 10:45:56	2022-02-03 10:47:03
2	202202-00002	Claire	C	Blake	Male	1997-10-14	09456789312	cblake@sample.com	This is her sample address.	uploads/clients/2.png?v=1643859659	1	0	2022-02-03 11:40:59	2022-02-03 11:40:59
3	202202-00003	Test12312	Test	Test	Male	2022-02-14	0956465798798	test@samnple.com	Test	NULL	0	1	2022-02-03 11:41:25	2022-02-03 11:42:47

After updation:

id	code	firstname	middlename	lastname	gender	dob	contact	email	address	image_path	status	delete_flag	date_created	date_updatec
1	202202-00001	Mark	D	Cooper	Male	1997-06-23	09123456789	mcooper@sample.com	Block 6 Lot 23, Here Subd., Over There, Anywhere 2...	uploads/clients/1.png?v=1643856423	2	0	2022-02-03 10:45:56	2024-03-06 17
2	202202-00002	Claire	C	Blake	Male	1997-10-14	09456789312	cblake@sample.com	This is her sample address.	uploads/clients/2.png?v=1643859659	1	0	2022-02-03 11:40:59	2022-02-03 11
3	202202-00003	Test12312	Test	Test	Male	2022-02-14	0956465798798	test@samnple.com	Test	NULL	0	1	2022-02-03 11:41:25	2022-02-03 11

4.4 Triggers:

A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.

```
<?php require_once('../config.php') ?>
<!DOCTYPE html>
<html lang="en" class="" style="height: auto;">
 <?php require_once('inc/header.php') ?>
<body class="hold-transition ">
  <script>
    start_loader()
  </script>
  <style>
    html, body{
      height:calc(100%) !important;
      width:calc(100%) !important;
    }
    body{
      background-image: url("<?php echo validate_image($_settings->info('cover')) ?>");
      background-size:cover;
      background-repeat:no-repeat;
    }
    .login-title{
      text-shadow: 2px 2px black
    }
    #login{
      flex-direction:column !important
    }
    #logo-img{
        height:150px;
```

```
      width:150px;
      object-fit:scale-down;
      object-position:center center;
      border-radius:100%;
    }
    #login .col-7,#login .col-5{
      width: 100% !important;
      max-width:unset !important
    }
  </style>
  <div class="h-100 d-flex align-items-center w-100" id="login">
    <div class="col-7 h-100 d-flex align-items-center justify-content-center">
      <div class="w-100">
        <center><img src="<?= validate_image($_settings->info('logo')) ?>" alt="" id="logo-img"></center>
        <h1 class="text-center py-5 login-title"><b><?php echo $_settings->info('name') ?></b></h1>
      </div>
    </div>
    <div class="col-5 h-100 bg-gradient">
      <div class="d-flex w-100 h-100 justify-content-center align-items-center">
        <div class="card col-sm-12 col-md-6 col-lg-3 card-outline card-primary rounded-0 shadow">
          <div class="card-header rounded-0">
            <h4 class="text-purle text-center"><b>Login</b></h4>
          </div>
          <div class="card-body rounded-0">
            <form id="login-frm" action="" method="post">
              <div class="input-group mb-3">
                <input type="text" class="form-control" autofocus name="username" placeholder="Username">
                <div class="input-group-append">
                  <div class="input-group-text">
```

```
                <span class="fas fa-user"></span>
              </div> </div>
    <div class="input-group mb-3">
              <input type="password" class="form-control" name="password" placeholder="Password">
              <div class="input-group-append">
                <div class="input-group-text">
                  <span class="fas fa-lock"></span>
                </div></div></div>
             <div class="row">
              <div class="col-8">
              </div>
              <!-- /.col -->
              <div class="col-4">
                <button type="submit" class="btn btn-primary btn-block btn-flat">Sign In</button>
              </div>
              <!-- /.col -->
            </div></form></div></div></div></div></div></div></div></div>
        </div></div></div></div>
<!-- jQuery -->
<script src="plugins/jquery/jquery.min.js"></script>
<!-- Bootstrap 4 -->
<script src="plugins/bootstrap/js/bootstrap.bundle.min.js"></script>
<!-- AdminLTE App -->
<script src="dist/js/adminlte.min.js"></script>
<script>
  $(document).ready(function(){
    end_loader();
  })
</script><body>
</html>
```

Figure 4.4.1: Login page

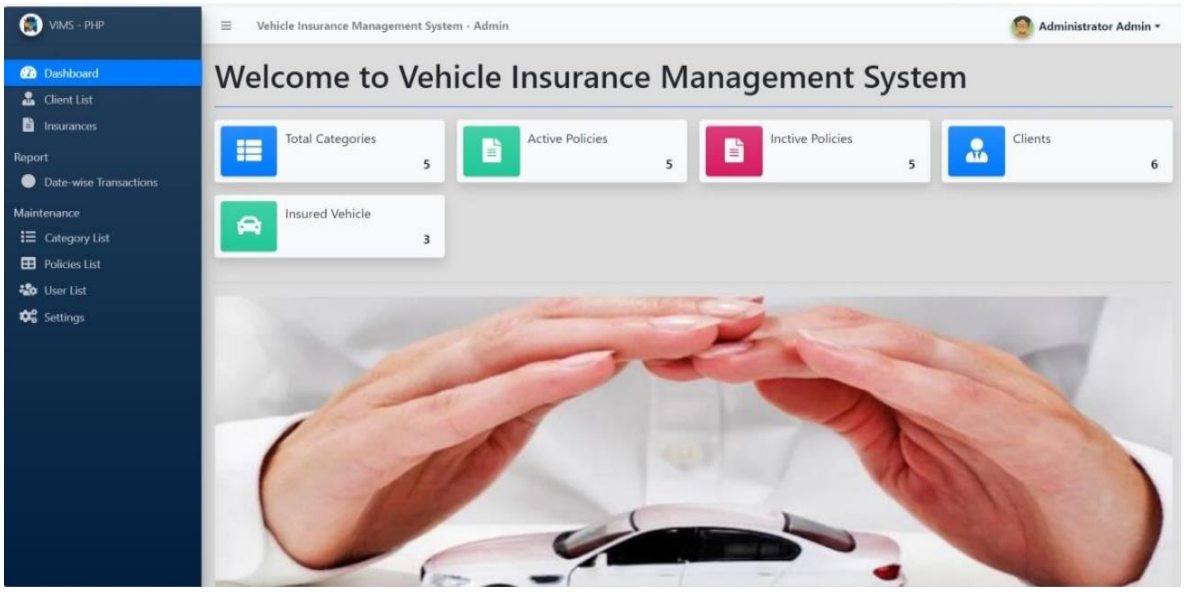


Figure 4.4.2: Dashboard

4.5 Front-End and Back End Details:

HTML:

- HTML is a markup language that web browsers use to interpret and compose text, images, and other material into visible or audible web pages.
- Default characteristics for every item of HTML markup are defined in the browser, and these characteristics can be altered or enhanced by the web page designer's additional use of CSS.

CSS:

- Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML).
- CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.CSS is designed to enable the separation of content and presentation, including layout, colours, and fonts.

JavaScript :

- JavaScript is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm. Alongside HTML and CSS, JavaScript is one of the three core technologies of the World Wide Web JavaScript is most commonly used as a client-side scripting language. This means that JavaScript code is written into an HTML page.
- There is no reason why JavaScript couldn't be used to write real, complex programs. However, this site exclusively deals with the use of JavaScript in web browsers.

PHP:

- PHP is an open-source, server-side programming language that can be used to create websites, applications, customer relationship management systems and more. It is a widely-used general-purpose language that can be embedded into HTML. the full form of PHP is Hypertext Preprocessor.

BOOTSTRAP:

- Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains HTML, CSS and JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

MySQL:

- MySQL is an Oracle-backed open source relational database management system (RDBMS) based on Structured Query Language (SQL)... Although it can be used in a wide range of applications, MySQL is most often associated with web applications and publishing SQL stands for Structured Query Language, and it is a programming language designed for querying data from a database.
- MySQL is a relational database management system, which is a completely different thing.

CHAPTER 5:

RESULTS

1. The Fig 5.1 contain the dashboard it has the total categories, active policies, inactive policies, clients and insured vehicles.

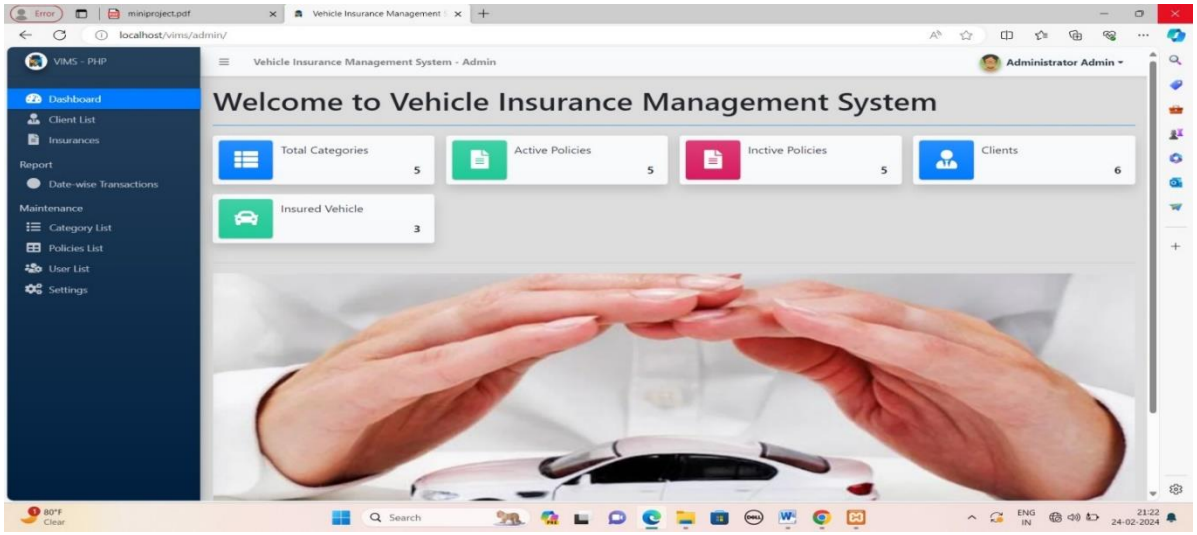


Figure 5.1: Insertion of user

5.1: Dashboard

2. The Fig 5.2 contain the Client list it show the list of clients if you want to add the new client you can do.

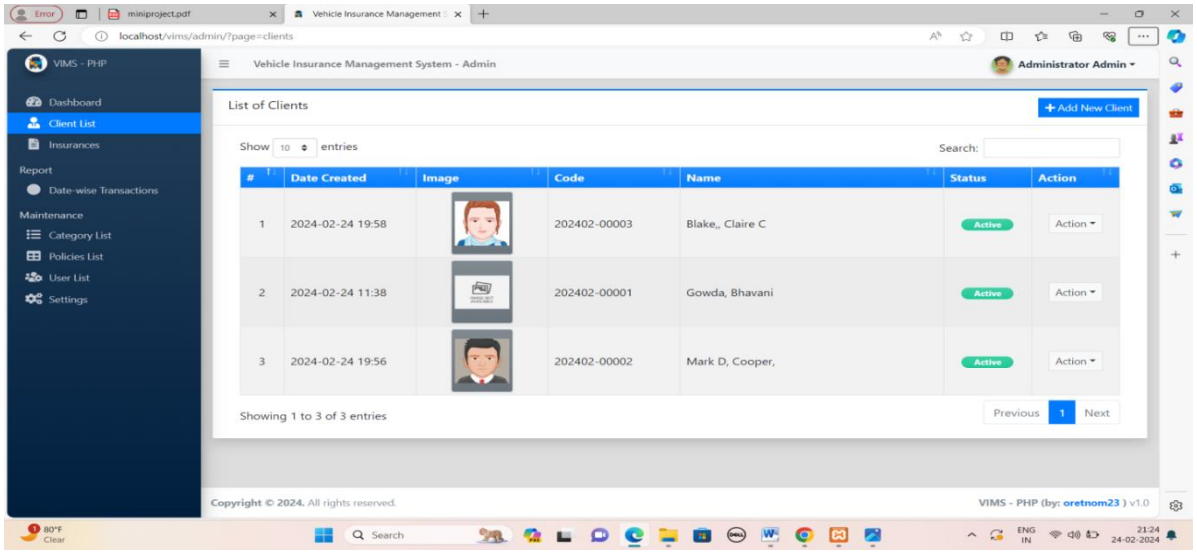


Figure 5.2:Client list

3. The Fig 5.3 contain the Insurances it shows the list of insurances if you want to add the new insurances you can do.

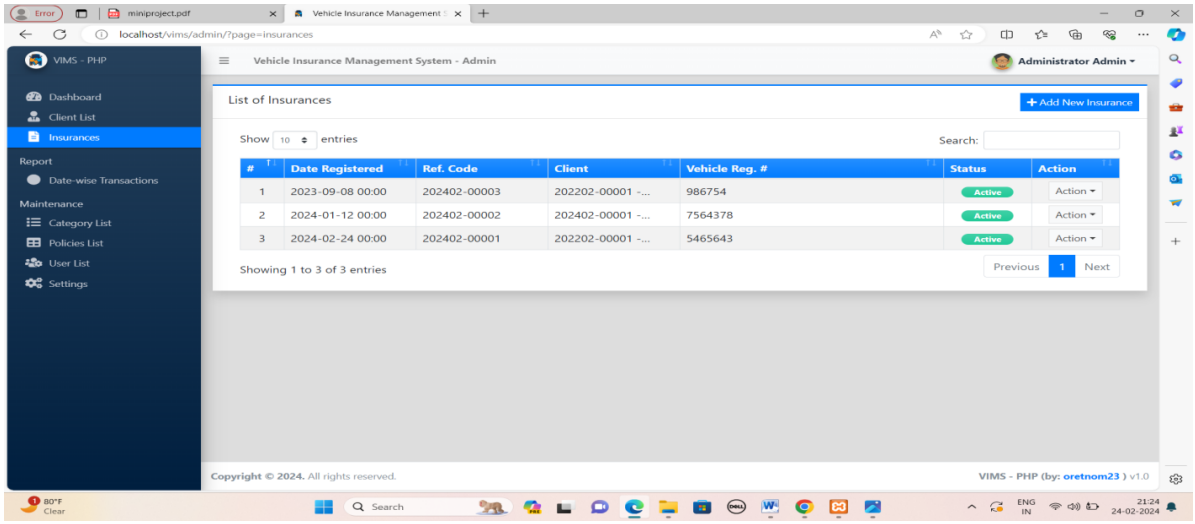


Figure 5.3: Insurances

4. The Fig 5.4 contain the Category list it shows the list of category if you want to add the new category you can do.

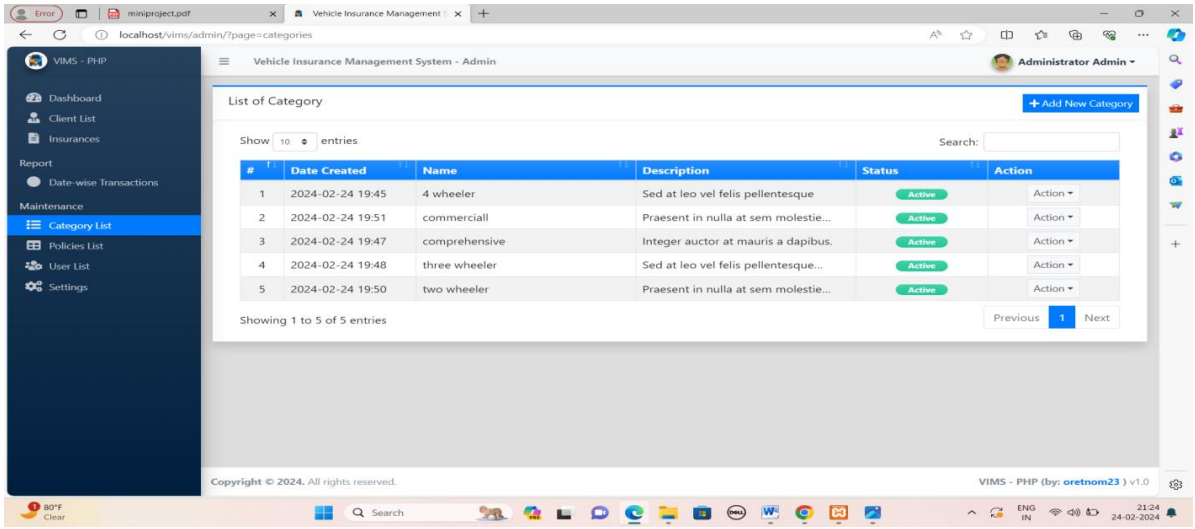


Figure 5.4: Category List

5. The Fig 5.5 contain the Policies List it shows the List of policies if you want to add the new policies you can do.

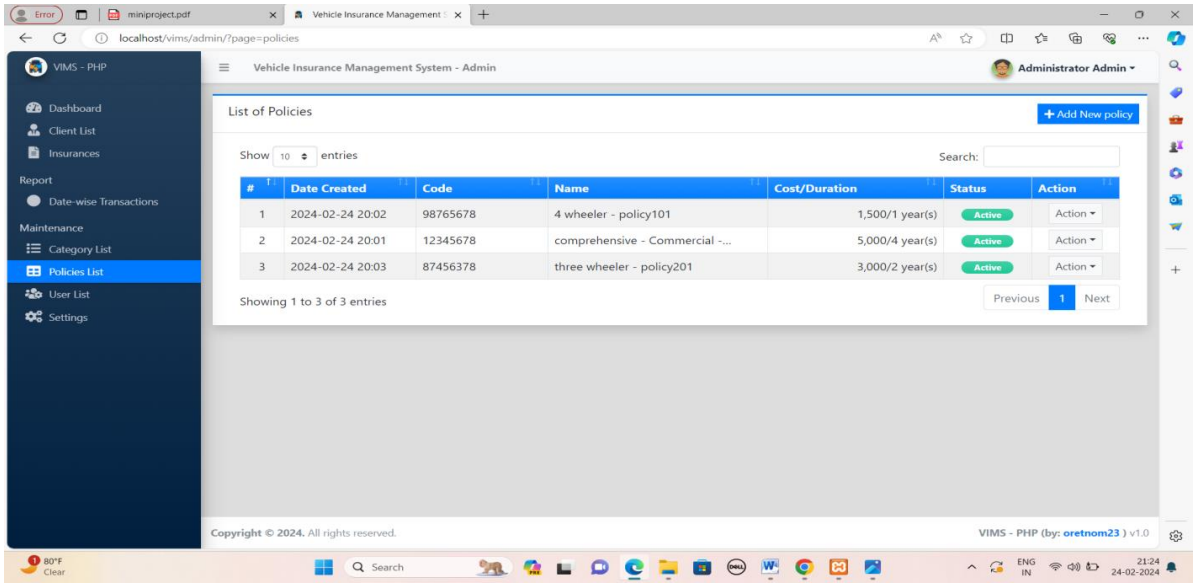


Figure 5.5: Policies List

6. The Fig 5.6 contain the user list it shows the list of user if you want to add the new user you can do.

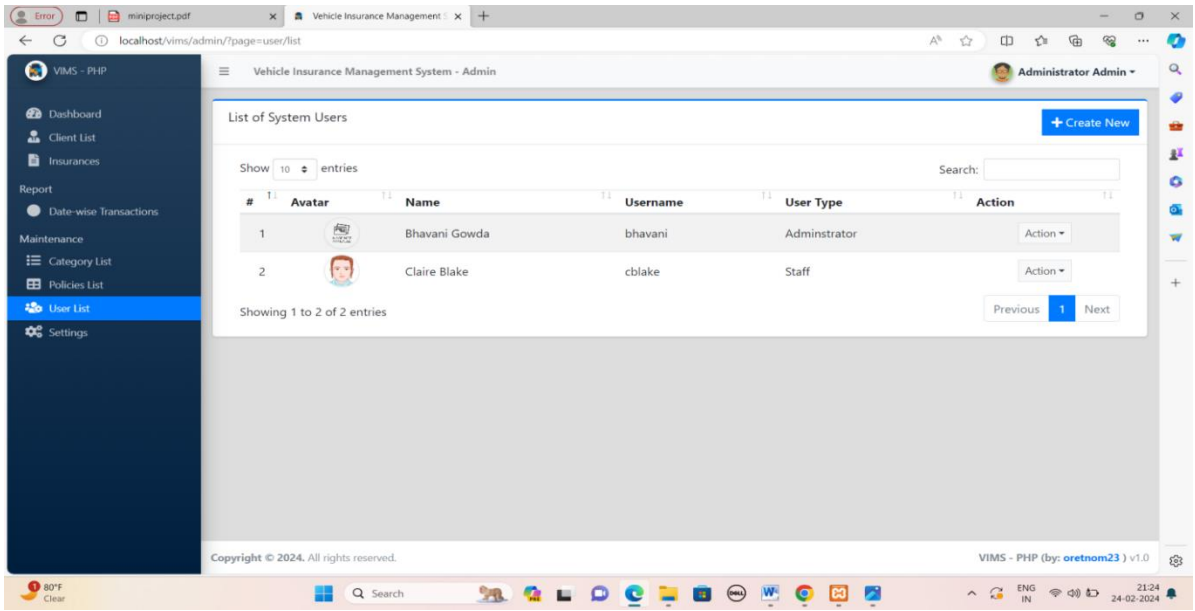


Figure 5.6: User List

CONCLUSION

In the present situation where the technology is the buzzword and has revolutionized the way we work and live, we would be the losers if we do not keep up with the changing world. Moreover, it makes a world of difference and a whole of sense to break up from the age-old work culture and embrace the effective, cost, and it saving ways of looking and working at things.

This is precisely where the Online Vehicle Insurance supports and improves many of the core functionality of the insurance organization i.e. insurance project helps in quick easy monitoring of the reports that have been automatically generated as and when the admin and policy agent perform transactions in the system. Using such a system helps the organization in minimizing the time consumed in fulfilling the day-to-day functionality's and cutting down the expenses incurred on the same .

REFERENCES

1. https://m.indiamart.com/proddetail/camera-rental-services-19999275512.html
2. https://wallpapercave.com/wp/wp4390828.jpg
3. https://wwe.freeprojectz.com for setting reminders
4. https://aitckm.in reference for creating the website
5. https://www.tutorialstonight.com code for adding text next to the image
6. https://www.dezven.com to create a logo for our website
7. https://codedamn.com CSS code for styling website
8. https://images.app.goo.gl/zk26aeKKVvTX9x1L9 copying logo.