CHINMAYA COLLEGE OF ARTS, COMMERCE AND SCIENCE

Layam road, Tripunithura – 682301

(Affiliated to Mahatma Gandhi University, Kottayam)

BACHELOR OF COMPUTER APPLICATION

MAIN PROJECT

ON

DETECTING FRAUD ON INSURANCE SYSTEM

Submitted by,

ANITTA GEORGE

Reg No: 180021094204

CHINMAYA COLLEGE OF ARTS, COMMERCE AND SCIENCE

Layam road, Tripunithura – 682301

(Affiliated to Mahatma Gandhi University, Kottayam)

CERTIFICATE

This is to certify that the Project report entitled

DETECTING FRAUD ON INSURANCE SYSTEM

has been submitted by

ANITTA GEORGE

Reg No: 180021094204

in partial fulfilment of the requirements for the award of the degree

BACHELOR OF COMPUTER APPLICATION

MAHATMA GANDHI UNIVERSITY

during the academic year 2020-2021

Submitted for the University Examination held on ……

Principal ***Head of the Department***

External Examiner ***Faculty Guide***

ACKNOWLEDGMENT

By blessing and permission of Almighty God, I was able to complete this work successfully.

My sincere thanks to our Principal, **Prof.Venugopal.B.Menon** for his overwhelming and moral support extended towards us.

I would like to thanks our head of the department and my faculty guide **Mrs. Nisha Sanjay** for her constant encouragements and support for the completion of my project.

I would also like to express my sincere thanks to all my teachers **Mrs. Andal V, Mrs. Sharmila Francis, Mrs. Remilda Rajan** and **Mr. Vishnu Mohanan** for their timely assistance and advice offered to us to make this project a success.

Finally, I thank my parents for their boundless support and for making our lives so easy and for helping to tackle all those difficulties in life.

Sincere thanks to all the other people who co-operated with me**.**

ANITTA GEORGE

DECLARATION

I **Anitta George** hereby declare that the project entitled '**DETECTING FRAUD ON INSURANCE SYSTEM'** submitted to **Mahatma Gandhi University, Kottayam** in partial fulfilment of the requirement for the award of degree of **Bachelor of Computer applications** is a record of original work done by me during the period of study at **Chinmaya College of Arts Commerce and Science, Tripunithura** under the supervision and guidance of **Mrs. Nisha Sanjay** (our faculty guide)department of Computer Applications and that this project work has not formed the basis for the award of any diploma/associates ship/fellowship or similar title to any candidate of any university.

PLACE: **ANITTA GEORGE**

DATE: **Reg No:180021094204**

SYNOPSIS

Detecting fraud on insurance system which named as Policy Craft is web application which is developed for tracking the details of the insurance police, customer details and company details. The project aims to get all the insurance companies under one umbrella. The system is a website that can register the insurance companies. The Companies add their own plan and policies to this website. The user can view and compare these plans and policies provided by deferent companies and they can select the company of their choice. It is possible to pay the premium amounts via online transaction. When an urgent situation arrives, the user can get the money without any difficulties. The website provides an interface to the companies to evaluate the performance of them. Agent can communicate to the customers via boxes. This website is online insurance Analysis and information management System that provides easy access of information regarding the people and resources of insurance. The project useful for any kind of insurance company to manage the insurance details to, sanctioned the insurance for customer, process the insurance policy details and all kind of insurance process through online. The insurance management system is a complete solution for organizations, which need to manage insurance for their vehicles, equipment building another resource. This insurance management website has facilities like search tools for insurance awareness articles, guideline, illustrations through images for visitors. The web site provides easy links for easy navigation in the site. After submission of registration form the admin will process to verify that particular details registered by the customer and sanctioned the insurance policy

TABLE OF CONTENTS

1.INTRODUCTION

1.1 PROJECT OVERVIEW

Detecting fraud on insurance system which named as Policy Craft is web application which is developed for tracking the details of the insurance police, customer details and company details. The head of insurance has taking a giant leap at the threshold of twentieth century. Insurance have become an integral part of life of an all over the globe. The project aims to get all the insurance companies under one umbrella. The system is a website that can register the insurance companies. The Companies add their own plan and policies to this website. The user can view and compare these plans and policies provided by different companies and they can select the company of their choice. It is possible to pay the premium amounts via online transaction. When an urgent situation arrives, the user can get the money without any difficulties. The website provides an interface to the companies to evaluate the performance of them. Our requirements related to insurance effortlessly compared to the current system. This website is online insurance Analysis and information management System that provides easy access of information regarding the people and resources of insurance. User can view their own personal details when log in into the user module. The project useful for any kind of insurance company to manage the insurance details to, sanctioned the insurance for customer, process the insurance policy details and all kind of insurance process through online. Policy Craft is a complete solution for organizations, which need to manage insurance for their vehicles, equipment building another resource. This insurance management system can efficiently manage the company, record, provides instant access and one that improves the 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 police. The main objective of the developed system is to allow admin user to register insurance person with their name, date of birth, residence address, medical history and also policy details. In this process contains the user registration form which is used to apply for insurance policy through online. Its anaesthesiology customer to view their own insurance status information. After giving registering all the insured person, website should provide management facilities like delete unwanted person data. The web site provides easy links for easy navigation in the site.

2.SYSTEM ANALYSIS

The software provides a user-friendly interface which can be operated by anyone with a minimum knowledge of the computer system. It stores the information needed by the portal in a data base which can be accessed by the administrator. In the required system, all the operations and activities related to the insurance management should be carried out efficiently. It should maintain a well-organized database for storing the resources that are provided by the system. This helps us to eliminate the entering of invalid data. Most problems of existing system can be solved by this proposed system. The system should cover almost all the functional areas of the portal. The insurance management system should be a database system that can store the information regarding keeping record of all customer and companies, their policy details etc.

System Analysis refers to the process of examining a situation with the intention of improving it through better process and methods. System analysis is therefore, the process of gathering and interpreting facts, diagnosing problem and using the information to recommend information in system or in other words, it means a detailed explanation or description. Before computerizing a system under consideration, it has to be analysed. We need to study how it functions currently, what are problems and what are requirements that proposed software should meet.

2.1 PROBLEM ANALYSIS

The first steps in the initial investigation are directed towards clarifying the problems of the existing system. Based on the initial investigation about the existing system it is found to have some issues such as the existing database for storing these details like customer details, request details etc and all are not that efficient and are not flexible. Updating and maintenance is cumbersome. Searching for a particular entry or a record is a bit time consuming task. Many users are having access to the database. Thus, it provides less security for the data being stored in the database.

The proposed system is thus overseeing all these issues so as to make recording of details within the organization more efficient and more secure.

2.1.1 EXISTING SYSTEM

The existing system is the manual system. The manual system is prone to error. It is time consuming. it is difficult to search for a data most of the insurance organization are not having any existing fully computerized system.it is very difficult for a person to produce the report. There are chances for changing the scheme report and do malpractice. They are managing the information in the form of Excel spread sheets. This system involves slot of manual entries with the entries with the applications to perform the desired task. Every member organization has its own data structure. Usage of papers in the payment process leads to less efficiency. less accuracy and less productivity. Due to lack of centralized data structure, it is very difficult to merge the data to analyses the statistic.

2.1.2 PROPOSED SYSTEM

The proposed system for making easier to manage policy holder details, policy details, claimant details and payment details. The proposed system is designed to eliminate the drawback of the existing system. it is designed by keeping to eliminate the drawback of the present system in order to provide a permanent solution to the problem. The primary aim of the new system is to speedup transaction. This insurance management system will be developed for managing the insurance management system. The overall system is control through the main menu. The report is prepared for the schemes and implemented by the concerned officials. In this project we have three modules the user management, company management and the administration.

BENEFITS OF PROPOSED SYSTEM

- To get all the insurance companies under one umbrella.
- To reduce data redundancy.
- Get information in any time.
- Reducing time.
- Reducing cost and energy.

2.1.3 FEASIBILITY STUDY

Feasibility is defined as the practical extent to which a project can be performed successfully. The objective of feasibility study is to establish the reasons for developing the software that is acceptable to the users, adaptable to changes and conformable to the established standards. Feasibility study lets the developer to foresee the future of the project and its usefulness. It is used for: Finding out whether a new system is required or not.

Feasibility study is test of system proposal according to its workability, impact on organization, ability to meet the needs, effective use of resources. During the study, the problem definition is crystallized and aspects of the problem to be included in this system are determined. The result of the feasibility study is a formal proposal. If the proposal is accepted, we continue with the project.

User needs a data-based system, which will remove all the mentioned problems that, the user is facing. The user wants a data-based system, which will reduce the bulk of paper work, provide ease of work, flexibility, fast record finding, modifying, adding, and removing.

We proposed our perception of the system, in accordance with the problems of existing system by making a full layout of the system on paper. We tallied the problems and needs by existing system and requirements. We were further updating in the layout in the basis of redefined the problems. In feasibility study phase we had undergone through various steps, which are described as under:

Cost:

The cost required in the proposed system is comparatively less to the existing system.

Effort:

Compared to the existing system the proposed system will provide a better working environment in which there will be ease of work and the effort required will be comparatively less than the existing system.

Time:

Record finding and updating will take less time than the existing system.

2.1.3.1 ECONOMICAL FEASIBILITY

The analysis raises financial and economic questions during the preliminary investigation to estimate the following:

- The cost to conduct a full system investigation.
- The cost of hardware and software for the class of application being considered.
- The benefits in the form of reduced cost.
- The proposed system will give the minute information, as a result the performance is improved which in turn may be expected to provide increased profits.

This feasibility checks whether the system can be developed with the available funds.

The System does not require enormous amount of money to be developed. Main part of the expenditure is for human resource and a part for a high-end server which can handle large amount of data. This can be done economically if planned judicially, so it is economically feasible. The cost of project depends upon the number of man hours required.

2.1.3.2 TECHNICAL FEASIBILITY

The project "DETECTING FRAUD ON INSURANCE SYSTEM" can be said to be technically feasible because there will be a smaller number of errors actually no errors because the whole project will be divided into modules such as customer's details, company details, payment details etc and so the errors if found, can be debugged very well and all the bugs can be removed.

Technical feasibility centres on the existing computer system and to what extend it can support the proposed system. It involves financial considerations to accommodate technical enhancements. If the budget is a serious constraint, then the project is judged not feasible. Here we need only a computer working in low speed to accomplish the task. Since the system uses database to implement, it is technically practical for all operators. The system can be implemented on the servers. The system requires no special expertise to operate, although some expertise will be required to code it.

2.1.3.3 BEHAVIORAL FEASIBILITY

Proposed projects are beneficial only if they can be turned into information systems that will meet the operating requirements of the organization. This test of feasibility asks if the system will work when it is developed satisfies all the operational conditions. It was the most difficult task for me, but met efficiently.

As this package is found to be feasible technically, economically and functionally, the system is judged feasible. Viewing the collected information, recommendation and justification, conclusions is made of the proposed system. Hence decision is taken to go on with the project.

People are inherently resistant to change and computer has been known to facilitate changes. An estimate should be made of how strong the user is likely to move towards the development of computerized system. These are various levels of users in order to ensure proper authentication and authorization and security of sensitive data of the organization. The software that is being developed is user friendly and easy to learn. In this way, the developed software is truly efficient and can work on any circumstances, tradition, locales. Operational study strives on ensuring that the equilibrium of the organization and status quotient in the organization neither are nor disturbed and changes are readily accepted by the users.

2.2 REQUIREMENT SPECIFICATION

A required specification is a structured document, which sets out the system services in details. This document, also called a functional specification, which should be precise. It may serve as a contract between the system buyer and the software developer. The requirement definition is developed into the requirement specification as detailed below: -

- Facility for administrator control
- Each user has unique identification
- The registrations would be done by customer
- Username and password are set by the users
- User must be capable for giving their requests

The requirement engineering process should normally involve writing requirements definition and then expanding this into a requirement specification. The software design is based directly on the requirements specification.

System analysis includes two main procedures. They are: -

1. Preliminary analysis

2. Detailed analysis

The preliminary analysis stage begins when someone encounters a problem or limitation in an existing system, desires a modification to the existing system. The modification may be either change in existing system or proposing an entire new system. Detailed analysis expands the preliminary analysis to include a complete analysis of all possible alternative solutions to the problem and a complete explanation of what appears to be the most practical solution.

2.2.1 SOFTWARE REQUIREMENT SPECIFICATION

This documentation aims to define the overall software requirements for 'DETECTING FRAUD ON INSURANCE SYSTEM'. Efforts have been made to define the requirements and accurately. The final product will be having only features mentioned in this document and assumptions for any additional feature should not be made by any of the parties involved in developing/testing/implementing/using this product. In case it is required to have some additional features, a formal change request will need to be raised and subsequently a new release of this document and product will be produced.

This specification document describes the capabilities that will be provided by the software application 'DETECTING FRAUD ON INSURANCE SYSTEM'. It also states the various required constraints by which the system will abide. The intended audience for this document is the development team, testing team and end user of the product.

The software product ''DETECTING FRAUD ON INSURANCE SYSTEM' 'will be an application that will be used for detecting frauds on insurance system and make all insurance companies under one umbrella. The application allows the companies to register into the system and they can add their policy details. User will be able to login and the can view different policies provided by different companies.

The system will allow access only to authorized users with specific role (Administrator, Company, Customers). Depending upon the user's role he/she will be able to access only specific modules of the system.

A summary of the major functions that the system will perform (functional specification):

i. A login facility for enabling only authorized access to the system.

ii. User (with role company) will be able to add/view premium categories.

iii. User (with role company) will be able to add/view insurance amounts.

iv. User (with role company) will be able to add new policies and view policies of their company.

v. User (with role company) will be able to view payment and user details.

vi. User (with role customer) will be able to view policy details and company details.

vii. User (with role customer) will be able to apply for new policies.

viii. User (with role customers) will be able to view payment history.

ix. User (with role customers) will be able to make payments and rate the company.

x. User (with role administrator) will be able to view company and user details.

xi. User (with role administrator) will be able to accept/reject companies and customers.

xii. User (with role administrator) will be able to add/view Insurance category

User Interface functionality:

i. A login screen for entering the username and password will be provided. Access to different screens will be based upon the role of the user.
ii. There will be screen for capturing and displaying information about insurance policies.
iii. There will be a screen capturing and displaying information regarding the user and their policy details.

The following reports will be generated:

i. Customer Report: Report will be generated to show the list of people who are registered into the system.
ii. Policy Report: Report will be generated to show the details of policies of different companies.

iii. Company Report: Report will be generated to show the details of all companies which are registered.

iv. Category Report: Report will be generated to show all the categories of insurance policies

Major non-functional specifications:

i. Performance: The system excels in terms of performance. It can perform major functionalities in seconds.

ii. Reliability: In terms of reliability, since this is a web-based software there is not space for errors. Every bit of data in stored in a secure database.

iii. Usability: Anyone with a little knowledge in internet and computer can easily operate the system.

iv. Maintainability: Maintenance of the system can be easily done through software update; other maintenance is done in server side.

v. Portability: The system can be used in any device with an internet connection and a web browser.

2.3 HARDWARE AND SOFTWARE SELECTION AND JUSTIFICATION

HARDWARE SPECIFICATION

The selection of hardware is very important in the existence and proper working of any software. When selecting hardware, the size and capacity requirements are also important. Below is some of the hardware that is required by the system

Processor	:	Intel core i3 and above
Memory	:	Minimum 1 GB RAM
Hard Disk Drive	:	100 GB
Keyboard	:	QWERTY
Components	:	Scroll Mouse
Display	:	14inch Color Monitor

SOFTWARE SELECTION AND JUSTIFICATION

We require much different software to make the application which is in making to work efficiently. It is very important to select the appropriate software so that the software works properly.

Below are the software requirements.

Operating System	:	Windows7 and above
Front End	:	Django Framework for PYTHON
		HTML, JavaScript, CSS
Browser	:	Any One
IDE	:	Visual Studio Code
Back End	:	MYSQL
Documentation	:	Microsoft Word 2007

WINDOWS 7

Windows 7 is a personal computer operating system developed by Microsoft. It is a part of Windows NT family of operating systems. Development of Windows 7 started as early as 2006 under the codename "Blackcomb." Windows 7 was released to manufacturing on July 22, 2009 and became generally available on October 22, 2009 less than three years after the release of its predecessor, Windows Vista. Windows 7's server counterpart, Windows Server 2008 R2, was released at the same time. Windows 7 was primarily intended to be an incremental upgrade to the operating system, intending to address Windows Vista's critical reception (such as performance improvements), while maintaining hardware and software compatibility. Windows 7 continued improvements on Windows Aero (the user interface introduced in Windows Vista) with the addition of a redesigned taskbar that allows applications to be "pinned" to it, and new window management features. Other new features were added to the operating system, including libraries, the new file sharing system Homegroup, and support for multitouch input. A new "Action Centre" interface was also added to provide an overview of system security and maintenance information, and tweaks were made to the User Account Control system to make it less intrusive. Windows 7 also shipped with updated versions of

several stock applications, including Internet Explorer, Windows Media Player, and Windows Media Centre.

Python

Python is a dynamic, high level, free open source and interpreted programming language. It supports object-oriented programming as well as procedural oriented programming. In Python, we don't need to declare the type of variable because it is a dynamic typed language. For example, x=10, here x can be anything such as String, int etc.

Features in Python

There are many features in Python, some of which are discussed below –

1. **Easy to code**:

Python is high level programming language. Python is very easy to learn language as compared to other language like c, c#, java script, java etc. It is very easy to code in python language and anybody can learn python basic in few hours or days. It is also developer-friendly language.

2. **Free and Open Source:**

Python language is freely available at official website and you can download it. Since, it is open-source; this means that source code is also available to the public. So, you can download it as, use it as well as share it.

3. **Object-Oriented Language**:

One of the key features of python is Object-Oriented programming. Python supports object-oriented language and concepts of classes, objects encapsulation etc.

4. **GUI Programming Support:**

Graphical Users interfaces can be made using a module such as PyQt5, PyQt4, python or Tk in python. PyQt5 is the most popular option for creating graphical apps with Python.

5. **High-Level Language:**

Python is a high-level language. When we write programs in python, we do not need to remember the system architecture, nor do we need to manage the memory.

6. **Extensible feature:**

Python is an Extensible language. We can write some python code into c or c++ language and also, we can compile that code in c/c++ language.

7. **Python is Portable language:**

Python language is also a portable language. For example, if we have python code for windows and if we want to run this code on other platform such as Linux, Unix and Mac then we do not need to change it, we can run this code on any platform.

8. **Python is integrated language:**

Python is also an integrated language because we can easily integrated python with other language like c, c++ etc.

9. **Interpreted Language:**

Python is an Interpreted Language. Because python code is executed line by line at a time. Like other language c, c++, java etc. there is no need to compile python code this makes it easier to debug our code. The source code of python is converted into an immediate form called byte code.

10**. Large Standard Library:**

Python has a large standard library which provides rich set of module and functions so you do not have to write your own code for every single thing. There are many libraries present in python for such as regular expressions, unit-testing, web browsers etc.

11. **Dynamically Typed Language**

Python is dynamically-typed language. That means the type (for example- int, double, long etc.) for a variable is decided at run time not in advance. Because of this feature we don't need to specify the type of variable

Django

Django is a Python based free and open-source web framework, which follows the model-template-view (MTV) architectural pattern. Django's primary goal is to ease the creation of complex, database driven websites. The framework emphasizes reusability and plug ability of components, less code, low coupling, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

Despite having its own nomenclature, such as naming the callable objects generating the HTTP responses views, the core Django framework can be seen as an MVC architecture. It consists of an object-relational mapper (ORM) that mediates between data models (defined

as Python classes) and a relational database (Model), a system for processing HTTP requests with a web templating system (View), and a regular-expression-based URL dispatcher (Controller).

Also included in the core framework are:

- A lightweight and standalone web server for development and testing.
- A form serialization and validation system that can translate between HTML forms and values suitable for storage in the database.
- A template system that utilizes the concept of inheritance borrowed from object-oriented programming.
- A caching framework that can use any of several cache methods
- Support for middleware classes that can intervene at various stages of request processing and carry out custom functions
- An internal dispatcher system that allows components of an application to communicate events to each other via pre-defined signal.
- An internationalization system, including translations of Django's own components into a variety of languages
- A serialization system that can produce and read XML and/or JSON representations of Django model instances
- A system for extending the capabilities of the template engine
- An interface to Python's built-in unit test framework
- Django REST framework is a powerful and flexible toolkit for building Web APIs

Java Script

Java script is a scripting language that can be used to create client-side scripts and server-side scripts. Client-side scripts are executed in the browser while server-side scripts are executed on a server. That is JavaScript is an object-based scripting language for developing client based and server-based internet applications. We can insert JavaScript statements directly into an HTML page. When the page is displayed in the browser, the JavaScript statements are interpreted and executed by the browser. JavaScript statements can recognize and respond to user events such as mouse clicks or system generated events and so on. So, you can change the content and position of the elements on the page dynamically, in response to user interaction.

When the client requests an HTML page that includes a client-side Script, the server forwards the full content of the HTML document- the JavaScript statements and the HTML content. When the browser receives the document, it executes the HTML and JavaScript statements without any interaction with the server while both client-side JavaScript and server-side JavaScript have the same core language; each also has additional features relevant to the environment. That is, client-side JavaScript includes predefined objects that can be used only in the browser. Server-side JavaScript contain predefined objects that can be used in server-side application

Hyper Text Mark Up Language

An HTML file is a text file containing small markup tags. These tags tell the web browser how to display the page. An HTML file must have an htm or html file extension. An HTML file can be created by using a simple text editor. HTML documents are text files made up of HTML elements e.g.: <html>, <body>. HTML elements are defined using HTML tags.

HTML tags are used to markup HTML elements. The two characters surround HTML tags <and>. The surrounding characters are called angle brackets. HTML tags normally come in pairs like and . The first tag in a pair is the start tag: the second tag is the end tag. The text between the start and the end tag is the element content. HTML tags are not case sensitive; means the same as <B>.

MySQL

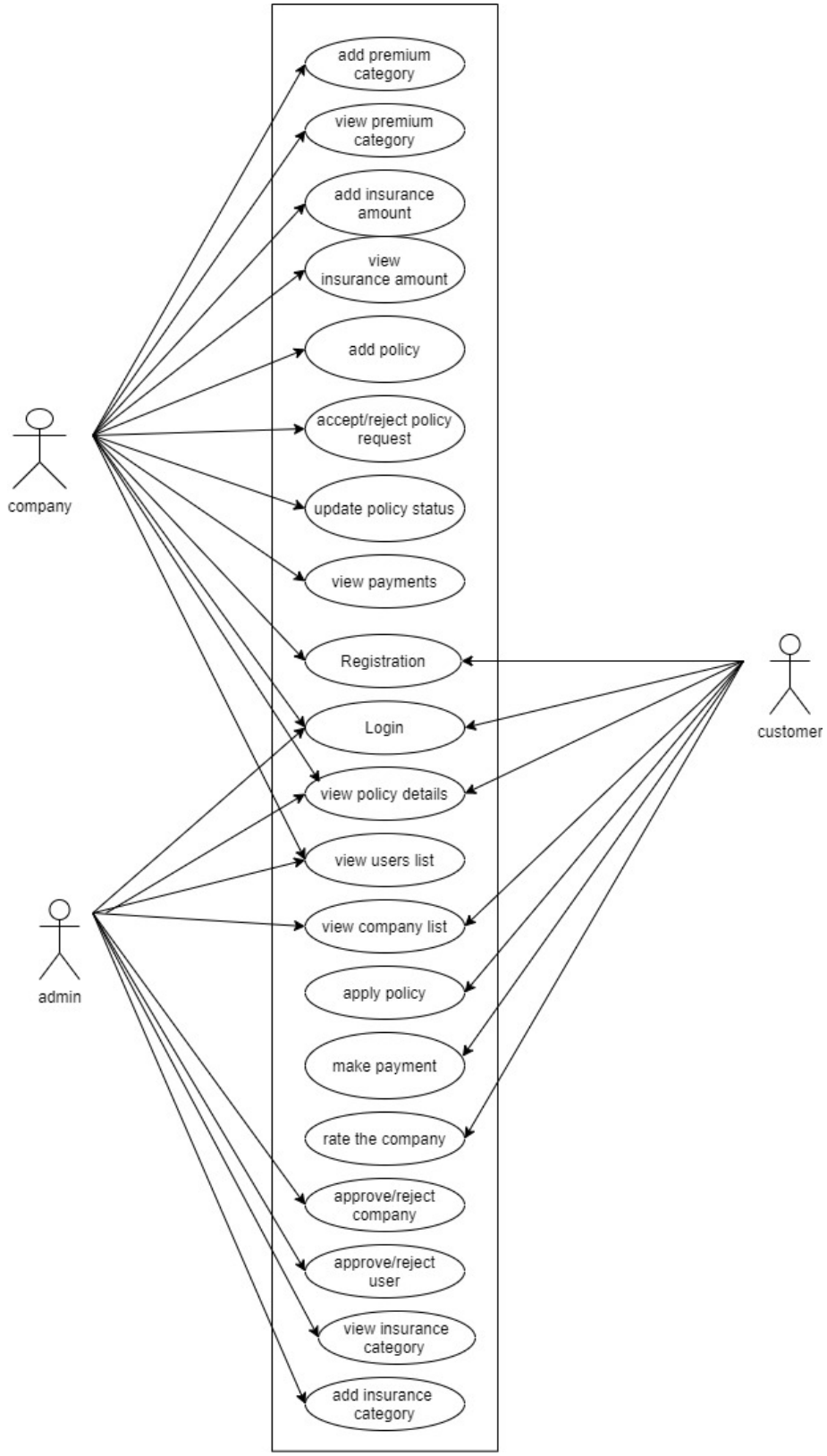
Relational database systems are the most important database systems used in the software industry today. One of the most outstanding systems is MySQL.

The important aspects of SQL Server are:

- MySQL is easy to use.
- Embedded database library.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack (and other "AMP" stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python." Free-software-open-source projects that require a full-featured database management system often use MySQL.

2.4 USE CASE DIAGRAM

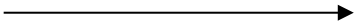


2.5 DATAFLOW DIAGRAM

The DFD also known as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data and the output data generated by the system.

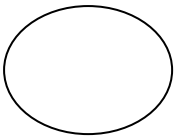
Data flow diagram (DFD) is used to show how data flows through the system and the processes that transform the input data into output. Data flow diagrams are a way of expressing system requirements in a graphical manner. Four simple notations are used to complete a DFD. The notations are given below:

DATA FLOW



A directed arc or an arrow is used as a Data Flow Symbol. This represents the data flow occurring between two processes or between an external entity and a process; in direction of the Data Flow Arrow. Data flow Symbols are annotated with corresponding data names.

PROCESS



A function is represented using a circle. This symbol is called a process or a bubble. Bubbles are annotated with the names of corresponding functions.

EXTERNAL ENTITY

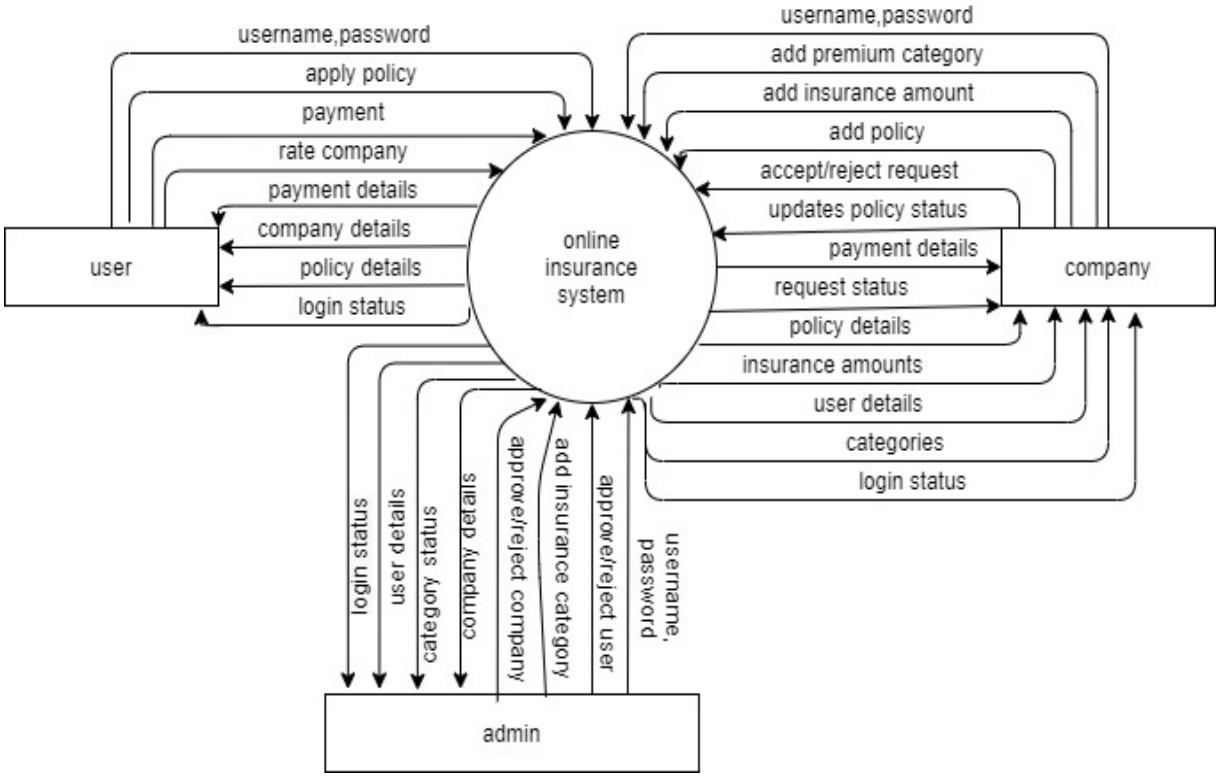
An external entity such as a user, project manager etc. is represented by a rectangle. The external entities are essentially those physical entities external to the application system, which interact with the system by inputting data to the system or by consuming the data

produced by the system. In addition to the human users the external entity symbols can be used to represent external hardware and software such as application software.

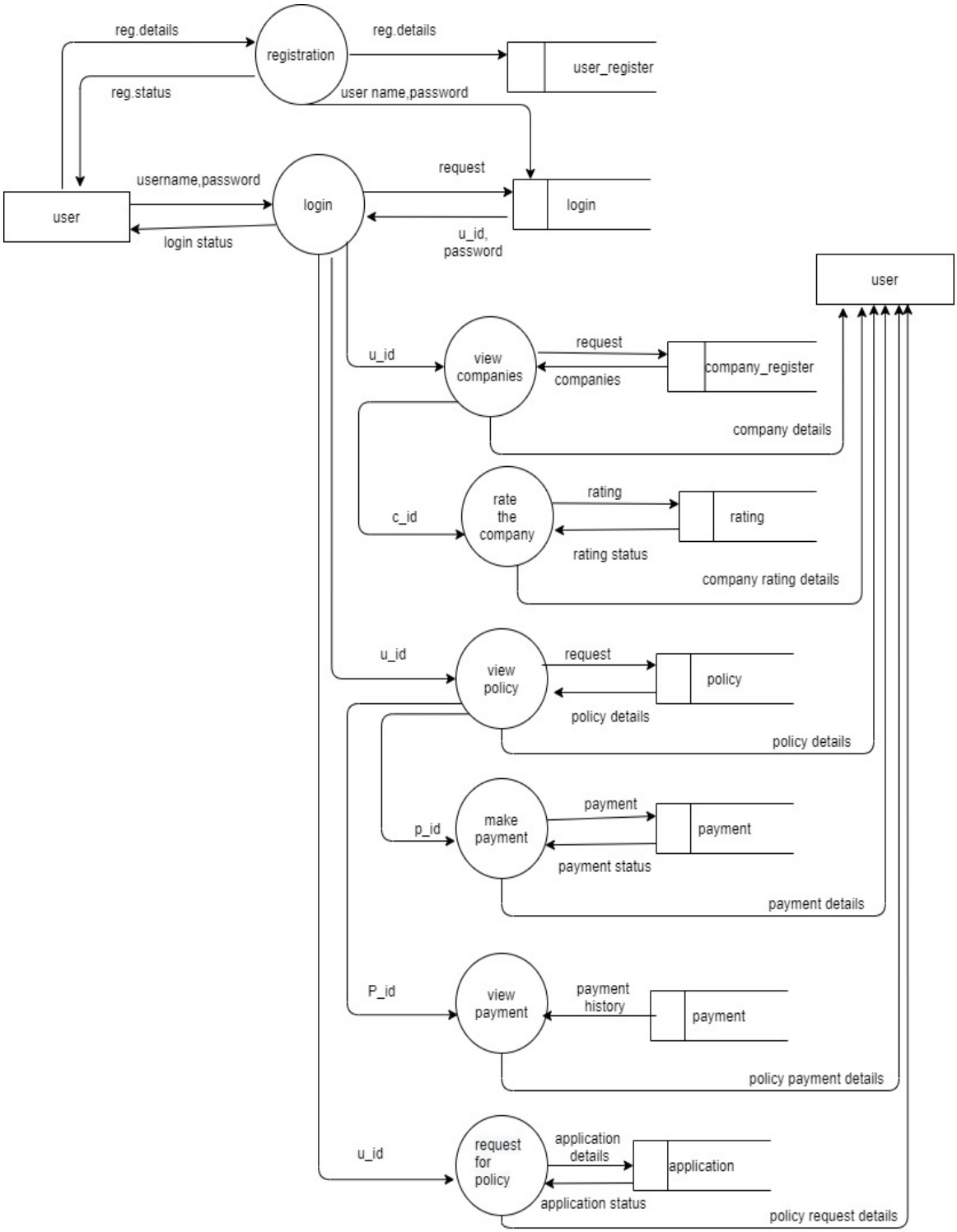
DATA STORE

A Data Store represents a logical file; it is represented using two parallel lines. A logical file can represent either Data Store Symbol, which can represent either data structure or a physical file on disk. Each data store is connected to a process by means of a Data Flow Symbol. The direction of the Data Flow Arrow shows whether data is being read from or written into a Data Store. An arrow flowing in or out of a data store implicitly represents the entire area of the Data Store and hence arrows connecting to a data store need not be annotated with the names of the corresponding data items.

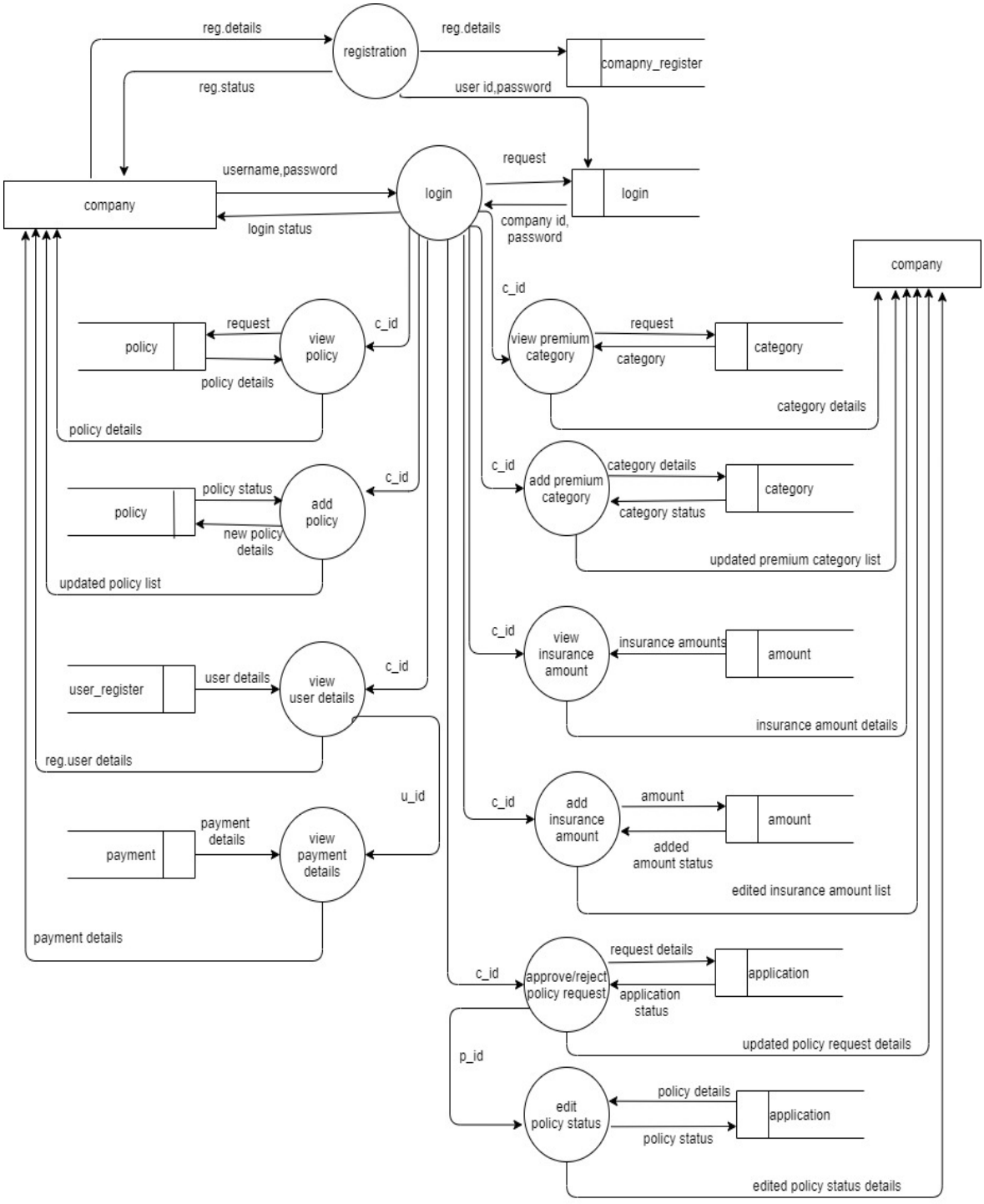
2.5.1 CONTEXT LEVEL DIAGRAM



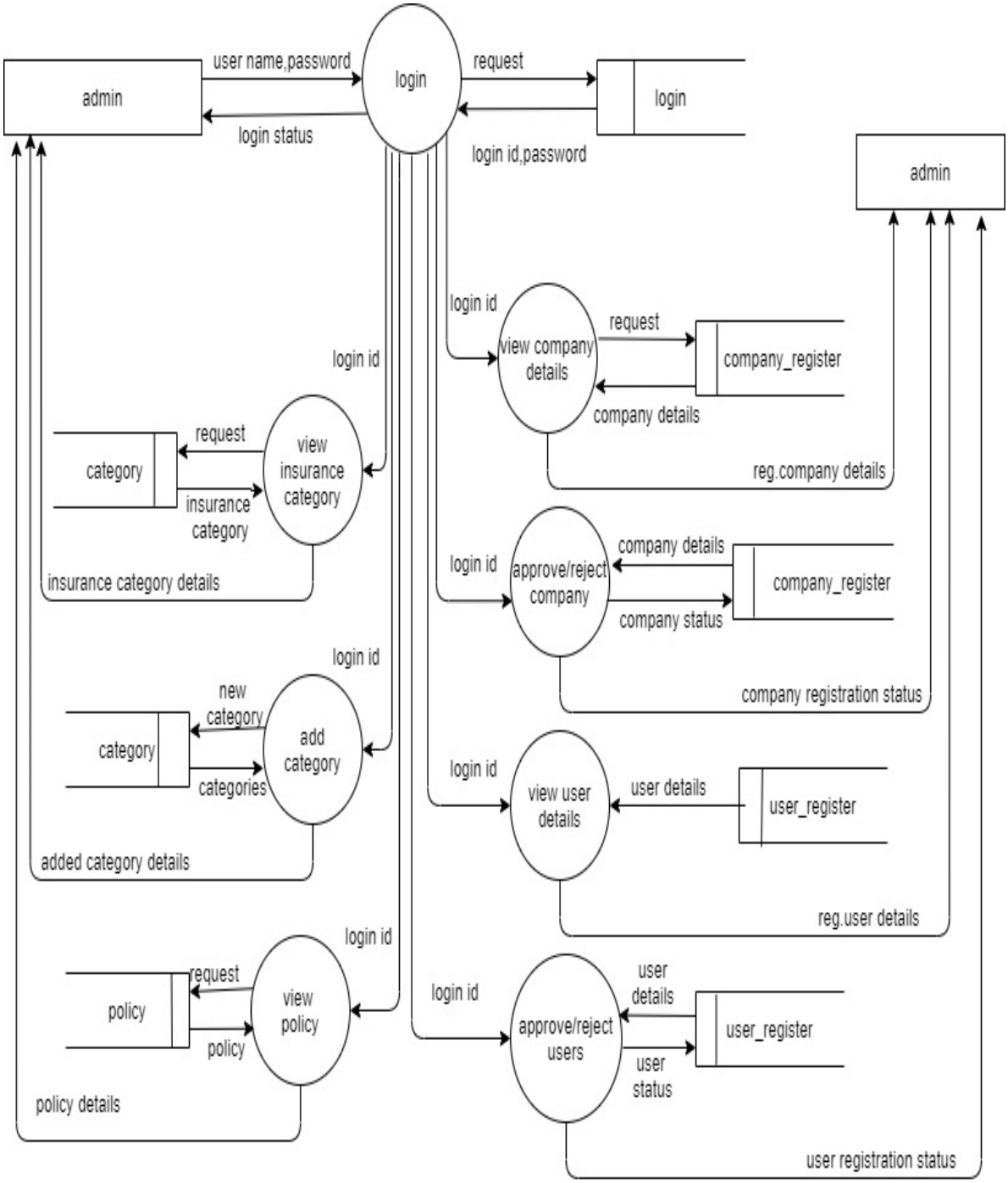
2.5.2 FIRST LEVEL DATA FLOW DIAGRAM FOR CUSTOMER



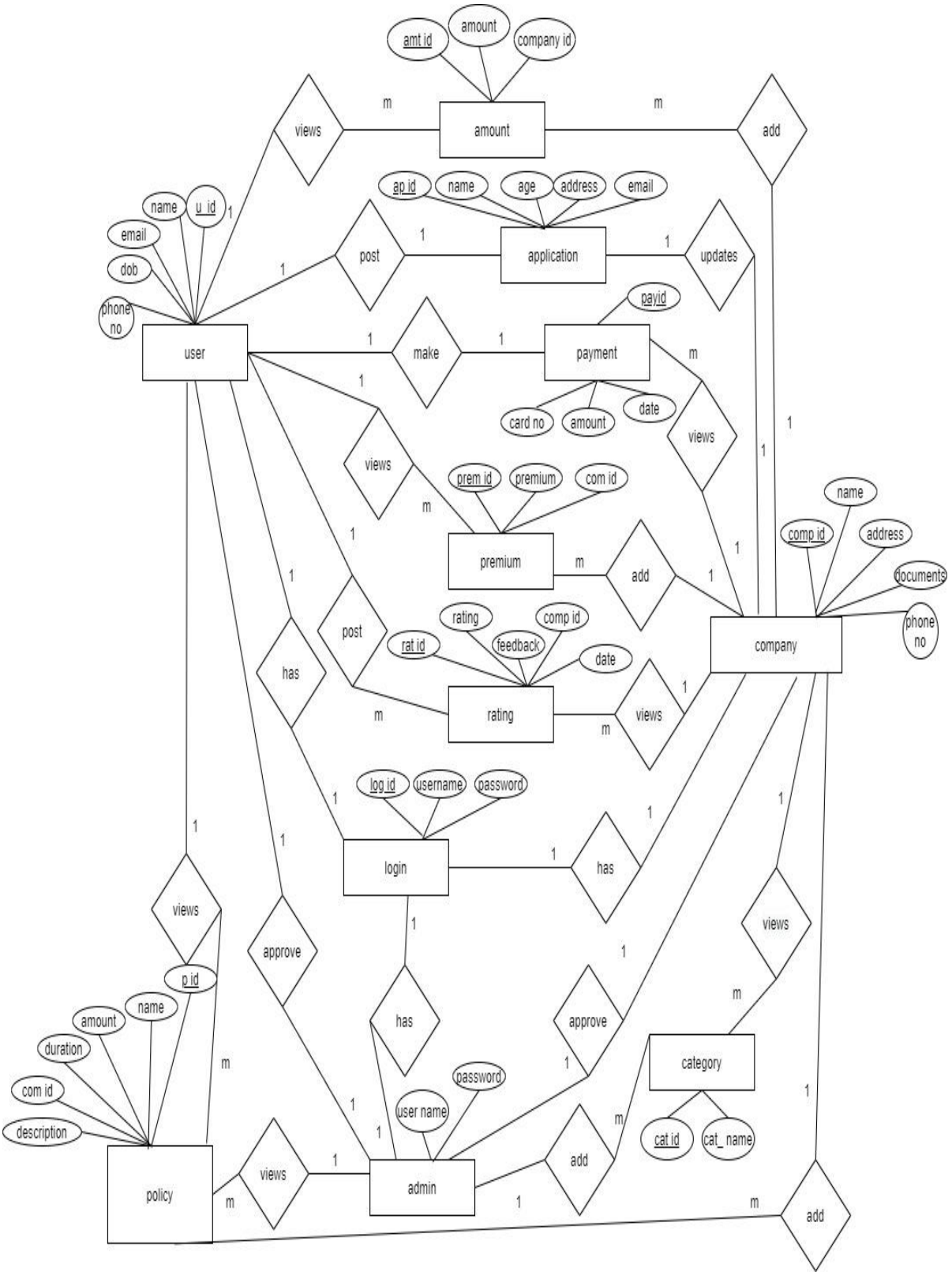
2.5.3 FIRST LEVEL DATA FLOW DIAGRAM FOR COMPANY



2.5.4 FIRST LEVEL DATA FLOW DIAGRAM FOR ADMIN



2.6 ER DIAGRAM



3. SYSTEM DESIGN

The most creative and challenging phase of the system development is system design, is a solution to how to approach to the creation of the proposed system. It refers to the technical specification that will be applied. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Design goes through the logical and physical stages of development. At an early stage in designing a new system, the system analyst must have a clear understanding of the objectives, which the design is aiming to fulfil. The first step is to determine how the output is to be produced and in what format. Second input data and master files (database) have to be designed to meet the requirements of the proposed output. The operational (processing) phases are handled through program construction and testing.

3.1 STRUCTURED DESIGN METHODOLOGIES

Design methodology refers to the development of a system or method for a unique situation. Design methodology stresses the use of brainstorming to encourage innovative ideas and collaborative thinking to work through each proposed idea and arrive at the best solution. Meeting the needs and requirements of the end user is the most critical concern. To employ design methodology various analyses and testing have been done so as to meet the desired user needs. Every input that the user input is being tested in this software. That means the validity of each data is being checked and if found invalid necessary warning and prompting messages are displayed. The output forms are also tested in detail to see whether the desired output is met or not. Also, the output forms are made clearer and more meaningful for the user to understand.

3.2 USER INTERFACE DESIGN

User interface design is one of the major functions in developing a system. It is a good understanding of the user needs very clearly. Because the user is the person who has to interact with system being developed. So that it should seek the needs of the user before developing it. The system is designed in a very user-friendly manner that makes the user with little knowledge of computer and of the organization can work very easily with the system.

The input forms that are used to enter the data are made clearer and easier to understand. Every time the user enters data the system is designed to check the validity of the data and if found as

invalid meaningful prompting and warning messages are displayed. This makes the user comfortable to interact with the system.

Also, when a user login to the system it checks the username and password entered to see whether it is valid user or not. It ensures security of the system and database. The data storage and data processing are made more efficient so that accurate results are being displayed on the output forms. And also, the retrieval of specific records as demanded by the user is made very faster that saves the user time. The input forms for the project are given in Appendix B for reference.

Main Input Forms are as follows: -

Login: This form is used to enter the details of login. The details include username and password

Company registration: This form is used to enter the company details. It may include name, locality, phone number etc…

Policy request: This form is used to give request for new policy.

Rating: This form is used to give rating for the company.

3.3 OUTPUT DESIGN

Output design is used to provide outputs to the users of the system. Computer output is the most important direct source of information to the user. Efficient intelligible output design improves the system relationships with the user and help in decision making major form of the output is the hardcopy from the printer and the screen reports. The output devices to consider depend on factors such as compatibility of the devices with the system, expected print quality and number of copies needed.

Here, in this case, I make use of forms which contains the tables to show the outputs of the processed data. The output design has been done so that the results of processing should be communicated to the user. Effective output design will improve the clarity and performance of outputs. Output is the main reason for developing the system and the basis on which they will evaluate the usefulness of the application. Output design phase of the system is concerned with

the Convergence of information to the end user in a friendly manner. The output design should be efficient, intelligible so that system relationship with the end user is improved and thereby enhancing the process of decision making. The output forms for the project are given in Appendix B for reference.

The major output forms are as follows:

User details: This form shows the details of all users

Category details: This form shows the different categories of insurance.

The major output reports are as follows:

Company Report: This report shows the details of all companies that are registered into the system.

Policy Report: This report shows the policies of different companies that are accessed by customers.

Payment Report: This report shows the payment history of users.

The reports for the project are given in Appendix C for reference.

3.4 DATABASE DESIGN

A database is a collection of interrelated data stored within minimum redundancy to serve many users quickly and efficiently. It is a process of designing the database file, which is the key source of the information in the system. The objective of database is to design is to provide storage and it contributes to the overall efficiency of the system. The file should properly design and planned for collection, accumulation, editing and retrieving the required information.

3.4.1 DATA & INTEGRITY CONSTRAINTS

The primary objective of a database design is fast response time to inquiries, more information at low cost, control of redundancy, clarity and ease of use, accuracy and integrity of the system, fast recovery and availability of powerful end-user languages. The theme behind a database is to handle information as an integrated whole thus the main objective is to make information as access easy, quick, inexpensive and flexible for the users. In this project, we mainly concentrated into relational databases.

Relational database stores data in tables, which is turn, are composed of rows also known as records, columns also known as fields. The fields in the relational model are: -

Primary Key

The key which is uniquely identify records. They also notify the not null constraints.

Foreign Key

The key which references the primary key, is the data inserted in the primary key column of the table.

Normalization

After the conceptual level, the next level of process of database design to organize the database structure into a good shape called Normalization. The normalization simplifies the entries, removing redundancies from the system data and finally builds a data structure, which is both flexible and adaptable to the system. The different normal forms obtained during the database design are given below:

In the database design, we create a database with different tables that is used to store the data. We normalize the data in the table. **Database normalization** is the process of organizing the fields and tables in a relational database to minimize redundancy and dependency. Normalization usually involves dividing large tables into smaller (and less redundant) tables and defining relationships between them. The objective is to isolate data so that additions, deletions, and modifications of a field can be made in just one table and then propagated through the rest of the database via the defined relationships. In the project I have made used of the 2nd normal form, Third Normal Form (3NF) is a property of database tables. A relation

is in third normal form if it is in Second Normal Form and there are no functional (transitive) dependencies between two (or more) non-primary key attributes. The overall objective in the development of database technology has been to treat data as an organizational resource and as an integrated whole. Database Management System allows data to be protected and organized separately from other resources. Database is an integrated collection of data. This is the difference between logical and physical data.

In my project, I have made use of tables which are stored in the database named policy craft. The tables are used to store the values that are generated by the application.

3.4.2 TABLE DESIGN

Table No:1

Table Name: login
Description: To store login values

Log_id	Int	Primary key
Reg_id	Int	Foreign key
Email	Varchar []	
Password	Varchar []	
status	Varchar []	
count	Int	
user type	Varchar []	

Table No:2

Table Name: user_register
Description: To store user details

reg_id	int	Primary Key
U_name	Varchar []	
U_address	Varchar []	
U_phone	Int	
U_dob	date	
U_gender	Varchar []	
U_email	Varchar []	

Table No:3

Table Name: company

Description: To store company details

c_id	int	Primary key
c_name	Varchar []	
C_regno	Varchar []	
C_email	Varchar []	
C_address	Varchar []	
C_phone	int	
C_doc	Varchar []	

Table No:4

Table Name: amount

Description: To store insurance amount

Amt_id	int	Primary key
amount	Varchar []	
C_id	int	Foreign key

Table No:5

Table Name: application

Description: To store insurance request details

App_id	int	Primary key
Name	Varchar []	
gender	Varchar[]	
age	Varchar[]	
Dob	date	
Phone no	int	
Address	Varchar []	
email	Varchar []	

P_id	int	Foreign key
C_id	Int	Foreign key
Cat_id	int	Foreign key
U_id	Int	Foreign key
status	Varchar []	

Table No:6

Table Name: payment

Description: To store payment details

pay_ id	int	Primary key
U_id	Int	Foreign key
C_id	int	Foreign key
P_id	int	Foreign key
cardno	int	
Date	date	
Amt	int	
App_id	int	Foreign key

Table No:7

Table Name: category

Description: To store insurance categories

cat_id	int	Primary key
Cat_Name	Varchar []	

Table No:8

Table Name: premium

Description: To store insurance premium

Prem_id	int	Primary key
Prem_type	Varchar[]	
C_id	int	Foreign key

Table No:9

Table Name: rating

Description: To store ratings

rating_id	int	Primary key
u_id	int	Foreign key
C_id	int	Foreign key
P_id	int	Foreign key
rating	int	
date	Date	
Feedback	Varchar[]	

Table No:10

Table Name: policy

Description: To store policy details

p_id	int	Primary key
P_name	Varchar[]	
P_duration	Varchar[]	
P_description	Varchar[]	
cat_id	int	Foreign key
amt_id	Int	Foreign key
Prem_id	int	Foreign key
P_doc	Varchar[]	
payment	int	
C_id	int	Foreign key

4.CODING

When considered as a step-in software engineering, coding is viewed as a natural consequence of design. However, programming language characteristics and coding style can profoundly affect software quality and maintainability. The coding step translates a detail design representation into a programming language realization. The translation process continues when a compiler accepts source code as input and produces machine-independent object code as output. The initial translation step in detail design to programming language is a primary concern in the software engineering context. Improper interpretation of a detail design specification can lead to erroneous source code. Style is an important attribute of source code and can determine the intelligibility of a program. The elements of a style include internal documentation, methods for data declaration, procedures for statement construction, and I/O coding and declaration. In all cases, simplicity and clarity are key characteristics. An offshoot of coding style is the execution time and/or memory efficiency that is achieved. Coding is the phase in which we actually write programs using a programming language. In the coding phase, design must be translated into a machine-readable form. If design is performed in a detailed manner, coding can be accomplished mechanistically. It was the only recognized development phase in early or unsystematic development processes, but it is just one of several phases in a waterfall process. The output of this phase is an implemented and tested collection of modules.

In my project, I have made use of the python to code the whole project and have made use of the MySQL server to act as a database to store the results of the processed data which is the output of the project. The coding for the project is given in Appendix B for reference.

5.IMPLEMENTATION OF SECURITY

The software quality assurance is comprised of a variety of tasks associated with seven major activities.

- Application of technical methods
- Conduct of formal technical reviews.
- Software testing.
- Enforcement of standards.
- Record keeping and reporting.

The quality begins with a set of technical methods and tools that help the analyst to achieve high quality specification and the designer to develop high quality design. The next activity involves assessment for quality for the design that is created which is the formal technical review. Software testing combines a multi-step strategy with a series of test case design methods that help ensure effective error detection.

5.1 DATA SECURITY

The software maintains a well-organized database for storing the details that are provided by the user. This helps us to eliminate the entering of invalid data. Data is not accessible to unauthorized users. The system analyst will provide the test data, specially designed to show that the system will operate successfully in all its aspects and produce expected results under expected conditions. Preparation of test data and the checking of results should be carried out in conjunction with the appropriate users and operational departments.
Also, the extent to which the system should be tested must be planned.

5.2 USER AND ACCESS RIGHTS

Admin: She/he will be having all the control of the system and admin can view all the records in the software.

Company: She/he will add/remove policy and also accept/reject policy request.

User: She/he will view company and policy details and request for policy.

6.SYSTEM TESTING

System testing is the stage of implementation highly aimed at ensuring that the system works accurately and efficiently before the live operation commences. Testing is vital to the success of the system. The primary objective of testing is to derive a set of tests that has the highest like hood for uncovering defects in then software. The system test in implementation should conform that all is correct and an opportunity to show the users that the system works as expected. It accounts the largest percentage of technical effort in the software development process. Testing phase in the development cycle validates the code against the functional specification

The performance of the system is measured in this phase. Testing is a set activity that can be planned and conducted systematically. Testing begins at the module level and works towards the integration of entire computers-based systems. Nothing is complete without testing, as it is vital success of the system. The testing can be a set of verification and validation process.

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

Two types of Validations are present, that are

- Client-side validation
- Server-side validation

Client-Side Validation

Client-side validation is something that will happen on users' browser. The validation will occur before the data gets posted back to server. It is a good idea to have client-side validation as the user gets to know what needs to be changed immediately. JavaScript is most widely used to perform client-side validation.

Server-Side Validation

Server-side validation occurs at server. The benefit of having server-side validation is that if the user somehow bypasses the client-side validation (accidentally or deliberately), then we can catch the problem on the server side. So, having server-side validation provides more security

and ensures that no invalid data gets processed by the application. Server-side validation is done by writing our custom logic for validating all the input.

The different types of testing are as follows:

1.1 Whitebox Testing

White box testing (also known as Clear box testing, Open box testing, Glass box testing, Transparent box testing, Code-Based testing or Structural testing) is a testing technique that takes into account the internal mechanism of a system. In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code, white box testing is often used for verification.

1.2 Black box Testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing. Black box testing is often used for validation.

6.1 Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

6.2 Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

6.3 User Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

The system was tested and found to be working as expected. There was no abnormal behaviour reported during the testing of the program. Testing is a method by which we try reducing the testing efforts and bringing out the maximum output. Testing helps us in knowing whether the logical assumptions that we have taken for the system are correct, and if they are correct, we

have obtained our goal. We test the system to know the errors, to check the validity of the information, to also group the modules with the aim that we meet the system requirements according to the system needs.

Testing is vital to the success of the system. System testing makes logical assumption that if all the parts of the system are correct, we have achieved the mission successfully. System testing is the stage of implementation that is aimed at assuring that the system works accurately and efficiently before the live operation commences.

6.4 Test Case Design

Function tested	Test condition	Expected result	Actual result	Status
Name	Entered non characters	Not allowed	Not allowed	Pass
Phone number	Entered more than 10 digits	Not allowed	Not allowed	Pass
Login	Invalid username or password	Not allowed	Not allowed	Pass
E-mail	Entered invalid email	Not allowed	Not allowed	Pass

6.5 Test Report and Debugging

Testing means verifying correct behaviour. Testing can be done at all stages of module development: requirements analysis, interface design, algorithm design, implementation, and integration with other modules. In the following, attention will be directed at implementation testing. Implementation testing is not restricted to execution testing. An implementation can also be tested using correctness proofs, code tracing, and peer reviews, as described below.

Debugging is a cyclic activity involving execution testing and code correction. The testing that is done during debugging has a different aim than final module testing. Final module testing

aims to demonstrate correctness, whereas testing during debugging is primarily aimed at locating errors. This difference has a significant effect on the choice of testing strategies.

- **Report error conditions immediately** - Much debugging time is spent zeroing in on the cause of errors. The earlier an error is detected, the easier it is to find the cause. If an incorrect module state is detected as soon as it arises then the cause can often be determined with minimal effort. If it is not detected until the symptoms appear in the client interface then may be difficult to narrow down the list of possible causes.

- **Maximize useful information and ease of interpretation** - It is obvious that maximizing useful information is desirable, and that it should be easy to interpret. Ease of interpretation is important in data structures. Some module errors cannot easily be detected by adding code checks because they depend on the entire structure. Thus, it is important to be able to display the structure in a form that can be easily scanned for correctness.

- **Minimize useless and distracting information** - Too much information can be as much of a handicap as too little. If you have to work with a printout that shows entry and exit from every procedure in a module then you will find it very difficult to find the first place where something went wrong. Ideally, module execution state reports should be issued only when an error has occurred. As a general rule, debugging information that says "the problem is here" should be preferred in favour of reports that say "the problem is not here".

- **Avoid complex one-use testing code** - One reason why it is counterproductive to add module correctness checks for errors that involve the entire structure is that the code to do so can be quite complex. It is very discouraging to spend several hours debugging a problem, only to find that the error was in the debugging code, not the module under test. Complex testing code is only practical if the difficult parts of the code are reusable.

7.SYSTEM IMPLEMENTATION AND MAINTENANCE

Implementation is the stage in the project where the theoretical stage is turned into a working system and is giving confidence on the new system for the users that it will work efficiently and effectively. It involves care full planning, Investigation of the current system and its constraints on implementation, design of methods to achieve the changeover, an evaluation, of change over methods. Apart from planning, major task of preparing the implementation are education and training of users. The more complex system being implemented, the more involved will be the system analysis and the design effort required just for implementation. An implementation coordinating committee based on policies of individual organization has been appointed. The implementation process begins with preparing a plan for the implementation of the system.

Implementation is the final and important phase. The most critical stage in achieving a successful new system and in giving the users confidence that the new system will work and be effectives. The system can be implemented only after thorough testing is done and if it is found to be working according to the specification. This method also offers the greatest security since the old system can take over if the errors are found or inability to handle certain type of transactions while using the new system. The implementation process begins with preparing the plan for the implementation of the system. Once the planning has been completed, the major effort in the computer department is to ensure that the programs in the system are working properly.

The following are the steps involved in the implementation plan:

- Test system with sample data
- Detection and correction of errors
- Make the necessary changes in the system
- Check the existing system
- Installation of hardware and software utilities
- Training and involvement of user personals

8. SCOPE OF PROJECT

The "DETECTING FRAUD ON INSURANCE SYSTEM" is a desktop application project, which is developed with an intention for making all the policy companies under one umbrella. The "DETECTING FRAUD ON INSURANCE SYSTEM "is a time-saving and efficient project. The System can use it to efficiently store all the data in a secure database. It is less prone to errors as the program checks the data entered before saving it to database. If it finds any data to be unsatisfactory it shows a warning to the user to correct the error.

It is extremely simple to use and quite powerful at the same time. It takes the load off the company and user in the "DETECTING FRAUD ON INSURANCE SYSTEM". The system is very flexible and changes can be made without much difficulty. The future extension in the system can be made in such a way that addition of new modules can be done without much difficulty. The reconstruction of the system will increase the flexibility of the system.

Our main aim of the project is to find people who made malpractice for getting insurance and to reduce human efforts. The user can maintain all the records about the user, policies, payments etc and save it in the database. The goals that wish to be achieved are:

1. System manages to save all the user records accurately and computerized.
2. System able to avoid the data redundancy on user details.
3. System able to display policy details and company details

9.FUTURE ENHANCEMENTS

The project developed using python Django and MySQL, based on the requirement specification of the user and analysis of the existing system, with flexibility for future enhancements. Any system which has been in use for a number of years gradually decays and become less effective because of change in environment to which it has to be adapted. For the time being it is possible to overcome problems by amendments and minor modifications to acknowledge the need of fundamental changes.

"DETECTING FRAUD ON INSURANCE SYSTEM "satisfies the requirements of the management. The system is developed in a user-friendly manner. It has one module for manipulating the database. The application can be enhanced in the future with the needs of the management. The database and the information can be updated to the latest coming versions. There are also possibilities for enhancing and further developing the project with the latest information and needs of the management, since the coding are in procedural block formats, altering the code is also made easy.

All the functions have been done carefully and successfully in the software, and if any development is necessary, in future it can be done without affecting the design by adding additional modules to the system. Some of the enhancements that can increase the value of this application are the following:

- The entire process of the firm can be computerized.
- It can be integrated with the web for universal access.
- Upgrading the performance: This system is now implemented at the client machine only but as a future enhancement we can modify the system in such a way to make it work on a client- server network. The system can be even more enhanced by making it an internet-based system.

10.CONCLUSION

The project "DETECTING FRAUD ON INSURANCE SYSTEM "has been created with the intention of providing a user with application which will suffice all needs for the details and other updates. All the requirements specifications were followed as for as possible and few additional features were added that can make the application more user friendly and less complicated. The project was successfully completed within the time span allotted. All the modules are tested separately and put together to form the main system. Finally, the system is tested with real data and it worked successfully. Thus, the system has fulfilled the entire objective defined.

The project "DETECTING FRAUD ON INSURANCE SYSTEM" has been developed with the proper guidance. A fully fledged user manual for this system is provided to the user for future working and functional references. We hope the "DETECTING FRAUD ON INSURANCE SYSTEM" fulfils all the needs in possible manner. The system has been developed in an interactive manner; the system is flexible, user friendly and has its own full data security and all data recovery facility.

The Major Advantages Are:

- Easy retrieval of data available in database.
- Quick implementation of results.
- Very user friendly.
- Does not require large amount of memory.
- Very less manual work is needed.
- Very cost effective

11.BIBLIOGRAPHY

Elias. M. Awad, "*System Analysis and Design*", Galgoeia publications – Second Edition2005.

Henry Korth, "*Data Base Design Concept*", Mc grew hill – Fifth Edition 2001.

Roger.S.Pressman, "*Software Engineering*", Mc grew hill – Fifth Edition 2006.

www.stackoverflow.com

www.W3Schools.com

www.tutorialspoint.com

APPENDIX -I

CODING

Views

```
from django.shortcuts import render

from django.http import (HttpResponse,HttpResponseRedirect)

from django.shortcuts import render

import MySQLdb

import datetime

now = datetime.datetime.now()

import simplejson as json

from django.core.files.storage import FileSystemStorage

from django.views.decorators.cache import cache_control

import smtplib

import urllib.request

conn = MySQLdb.connect("localhost","root","","policy_craft")

c = conn.cursor()

# Create your views here.

def index(request):

    msg = ""

    if 'u_register' in request.POST:

        u_name = request.POST.get("u_name")

        u_last_name = request.POST.get("u_last_name")

        u_email = request.POST.get("u_email")

        u_address = request.POST.get("u_address")

        u_phone = request.POST.get("u_phone")

        u_gender = request.POST.get("u_gender")

        u_dob = request.POST.get("u_dob")
```

```
        password = request.POST.get("password")
u_full_name = u_name.strip()+" "+u_last_name.strip()
        print(u_full_name)
         c_count = "select count(*) from user_register where u_email = '"+str(u_email)+"' and  u_phone = '"+str(u_phone)+"'"
 c.execute(c_count)
        conn.commit()
        data = c.fetchone()
        if data[0] == 0:
            s = "insert into user_register(`u_name`,`u_email`,`u_address`,`u_phone`,`u_gender`,`u_dob`) values('"+str(u_full_name).strip()+"','"+str(u_email)+"','"+str(u_address)+"','"+str(u_phone)+"','"+str(u_gender)+"','"+str(u_dob)+"')"
            c.execute(s)
            conn.commit()
   s1 = "insert into login(`reg_id`,`email`,`password`,`type`) values((select max(u_id) from user_register),'"+str(u_email)+"','"+str(password)+"','user')"
            c.execute(s1)
            conn.commit()
            msg = "User Registered Successfully"
            return render(request,"index.html",{"msg":msg})
        else:
            msg = "Account Already Exists"
            return render(request,"index.html",{"msg":msg})
  if 'c_register' in request.POST:
        c_name = request.POST.get("c_name")
        c_reg_no = request.POST.get("c_reg_no")
        c_email = request.POST.get("c_email")
```

```
        c_address = request.POST.get("c_address")
        c_phone = request.POST.get("c_phone")
        # c_doc = request.POST.get("c_doc")
        password = request.POST.get("password")
        myfile = request.FILES["c_doc"]
        fs = FileSystemStorage()
        filename = fs.save(myfile.name, myfile)
        uploaded_file_url = fs.url(filename)
    c_count = "select count(*) from company_register where c_email = '"+str(c_email)+"' and c_reg_no = '"+str(c_reg_no)+"'"
        c.execute(c_count)
        conn.commit()
        data = c.fetchone()
        if data[0] == 0:
            s = "insert into company_register(`c_name`,`c_reg_no`,`c_email`,`c_address`,`c_phone`,`c_doc`) values('"+str(c_name)+"','"+str(c_reg_no)+"','"+str(c_email)+"','"+str(c_address)+"','"+str(c_phone)+"','"+str(uploaded_file_url)+"')"
            c.execute(s)
            conn.commit()
    s1 = "insert into login(`reg_id`,`email`,`password`,`type`) values((select max(c_id) from company_register),'"+str(c_email)+"','"+str(password)+"','company')"
            c.execute(s1)
            conn.commit()
            msg = "Company Registered Successfully"
            return render(request,"index.html",{"msg":msg})
        else:
            msg = "Account Already Exists"
```

```
        return render(request,"index.html",{"msg":msg})
 if 'login' in request.POST:
      try:
        user_name = request.POST.get("user_name")
        password = request.POST.get("password")
        s = "select * from login where email='"+str(user_name)+"' and password='"+str(pass
word)+"' and status='1' "
        # print(s)
        c.execute(s)
        conn.commit()
        log_count = c.fetchone()
        ci=log_count[6]
        sta=log_count[7]
        print(log_count,ci,sta,"looooooooooooooo")
        if log_count[5] == "admin":
            return HttpResponseRedirect("/admin_home")
        elif log_count[5] == "user":
            if sta !="blocked":
                request.session["u_id"]=log_count[1]
                # request.session["username"]=log_count[2]
                # request.session["type"]=log_count[4]
                return HttpResponseRedirect("/user_home")
            else:
                msg="You are blocked"
                return render(request,"index.html",{"msg":msg})


        elif log_count[5] == "company":
```

```
            request.session["com_id"]=log_count[1]
            print(log_count[1])
            # request.session["username"]=log_count[2]
            # request.session["type"]=log_count[4]
            return HttpResponseRedirect("/com_home")
        else:
            msg = "Invalid User"
            return render(request,"index.html",{"msg":msg})
    except:
        msg = "Invalid User"
        return render(request,"index.html",{"msg":msg})
  return render(request,"index.html",{"msg":msg})
#-----------------Admin page---------------#
def admin_header_footer(request):
  return render(request,"admin_header_footer.html")
def admin_home(request):
  return render(request,"admin_home.html")
def approve_company(request):
  msgg = ""
  s = "select * from company_register c , login l where c.c_id = l.reg_id and l.status = 0"
  c.execute(s)
  conn.commit()
  data = c.fetchall()
  print(s)
  if not bool(data):
    msgg = "No new Company registrations..."
```

```
    return render(request,"approve_company.html",{"data":data,"msgg":msgg})
def approve_com(request):
    reg_id = request.GET.get("reg_id")
    s = "update login set status = '1' where reg_id = '"+str(reg_id)+"'"
    c.execute(s)
    conn.commit()
    return HttpResponseRedirect("/approve_company")
def view_company(request):
    msgg = ""
    s = "select * from company_register c , login l where c.c_id = l.reg_id and l.status = 1"
    c.execute(s)
    conn.commit()
    data = c.fetchall()
    print(s)
    if not bool(data):
        msgg = " No Company list to show..."
    return render(request,"view_company.html",{"data":data,"msgg":msgg})
def remove_com(request):
    reg_id = request.GET.get("reg_id")
    s = "delete from company_register where c_id = '"+str(reg_id)+"'"
    # s = "delete login set status = '1' where reg_id = '"+str(reg_id)+"'"
    c.execute(s)
    conn.commit()
    s1 = "delete from login where reg_id = '"+str(reg_id)+"'"
    c.execute(s1)
    conn.commit()
```

```
    return HttpResponseRedirect("/view_company")
def approve_user(request):
    msgg=""
    s = "select * from user_register u , login l where u.u_id = l.reg_id and l.status = 0"
    c.execute(s)
    conn.commit()
    data = c.fetchall()
    print(s)
    if not bool(data):
        msgg = "No new User registrations..."
    return render(request,"approve_user.html",{"data":data,"msgg":msgg})
def approve_us(request):
    reg_id = request.GET.get("reg_id")
    s = "update login set status = '1' where reg_id = '"+str(reg_id)+"'"
    c.execute(s)
    conn.commit()
    return HttpResponseRedirect("/approve_user")
def view_user(request):
    msgg = ""
    s = "select * from user_register u , login l where u.u_id = l.reg_id and l.status = 1"
    c.execute(s)
    conn.commit()
    data = c.fetchall()
    print(s)
    if not bool(data):
        msgg = " No User List to show..."
```

```
    return render(request,"view_user.html",{"data":data,"msgg":msgg})
def block(request):
    reg_id = request.GET.get("reg_id")
    st="blocked"
    s = "update  login set bstatus='"+str(st)+"' where reg_id = '"+str(reg_id)+"'"
    # s = "delete login set status = '1' where reg_id = '"+str(reg_id)+"'"
    c.execute(s)
    conn.commit()
     return HttpResponseRedirect("/view_user")
def remove_us(request):
    reg_id = request.GET.get("reg_id")
    s = "delete from user_register where u_id = '"+str(reg_id)+"'"
    # s = "delete login set status = '1' where reg_id = '"+str(reg_id)+"'"
    c.execute(s)
    conn.commit()
    s1 = "delete from login where reg_id = '"+str(reg_id)+"'"
    c.execute(s1)
    conn.commit()
    return HttpResponseRedirect("/view_company")
def add_category(request):
    msg = ""
    qry1 = "select * from category"
    c.execute(qry1)
    cat_data = c.fetchall()
    if 'add_category' in request.POST:
        category = request.POST.get("category")
```

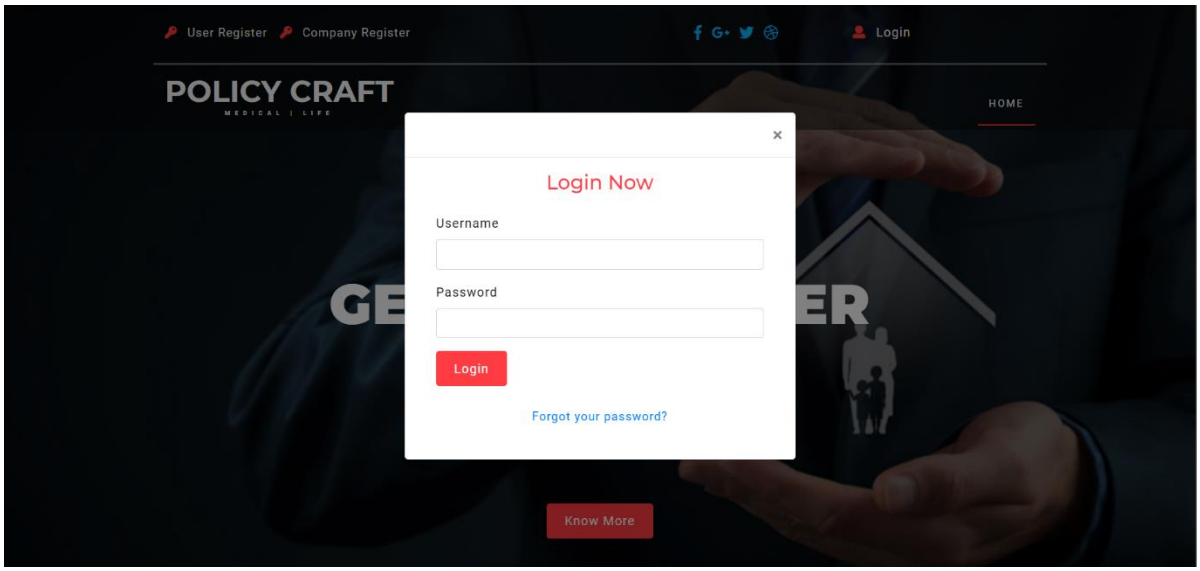
```
        ss = "select count(*) from category where cat_name= '"+str(category)+"'"
        c.execute(ss)
        data = c.fetchone()
        if data[0] == 0 :
            print(category)
            s = "insert into category(`cat_name`) values('"+str(category)+"')"
            c.execute(s)
            conn.commit()
            msg = "Category Added Successfully"
            return render(request,"add_category.html",{"msg":msg,"cat_data":cat_data})
      else:
            msg = "Category already Exists"
            return render(request,"add_category.html",{"msg":msg,"cat_data":cat_data})
    return render(request,"add_category.html",{"msg":msg,"cat_data":cat_data})
def admin_view_policy(request):
    msgg = ""
    print("admin")
    qry1 = "select * from category"
    c.execute(qry1)
    cat_data = c.fetchall()
    qry2 = "select * from company_register"
    c.execute(qry2)
    com_data = c.fetchall()
    if 'submit' in request.POST:
        category = request.POST.get("category")
        company = request.POST.get("company")
```

```
        s= "SELECT * FROM policy p ,premium pr, amount a WHERE p.cat_id = "'+str(categ
ory)+"' AND p.com_id = "'+str(company)+"' AND p.`com_id`= pr.`com_id` AND a.`com_id`
= p.`com_id` AND p.premium_id = pr.`pre_id` AND a.`amt_id` = p.`amount_id`"
        print(s)
        c.execute(s)
        conn.commit()
        data = c.fetchall()
        print(data)
        if not bool(data):
            msgg = " No Policy List to show..."
        return render(request,"admin_view_policy.html",{"cat_data":cat_data,"com_data":com_
data,"data":data,"msgg":msgg})
     return render(request,"admin_view_policy.html",{"cat_data":cat_data,"com_data":com_data
})
def logout(request):
    return render(request,"logout.html")
```

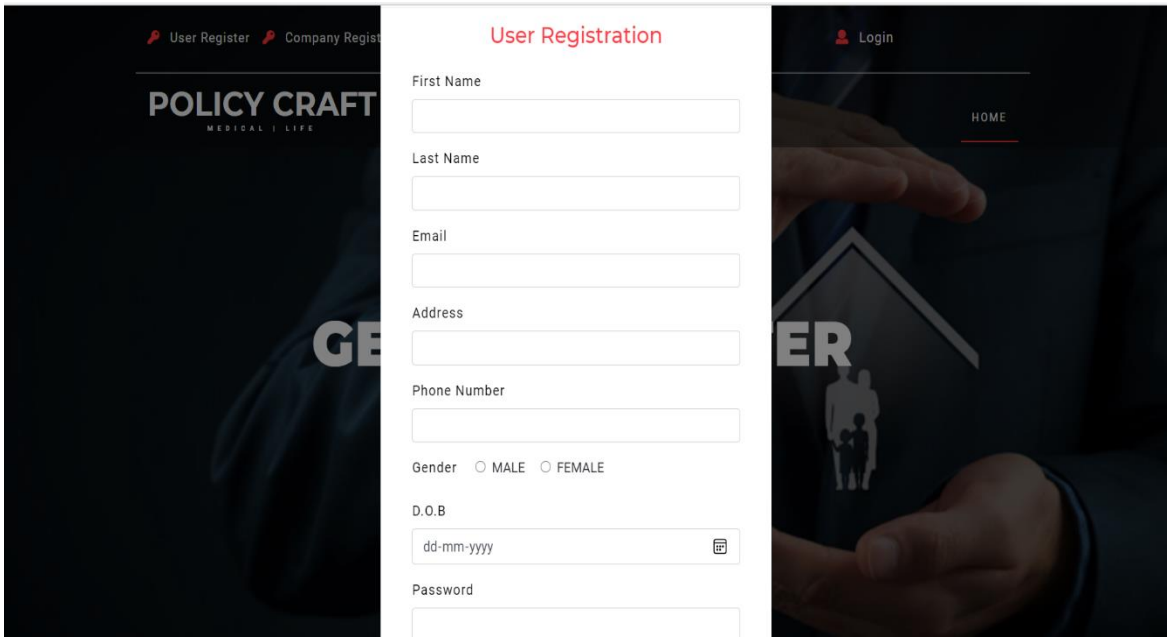
APPENDIX-II

FORMS

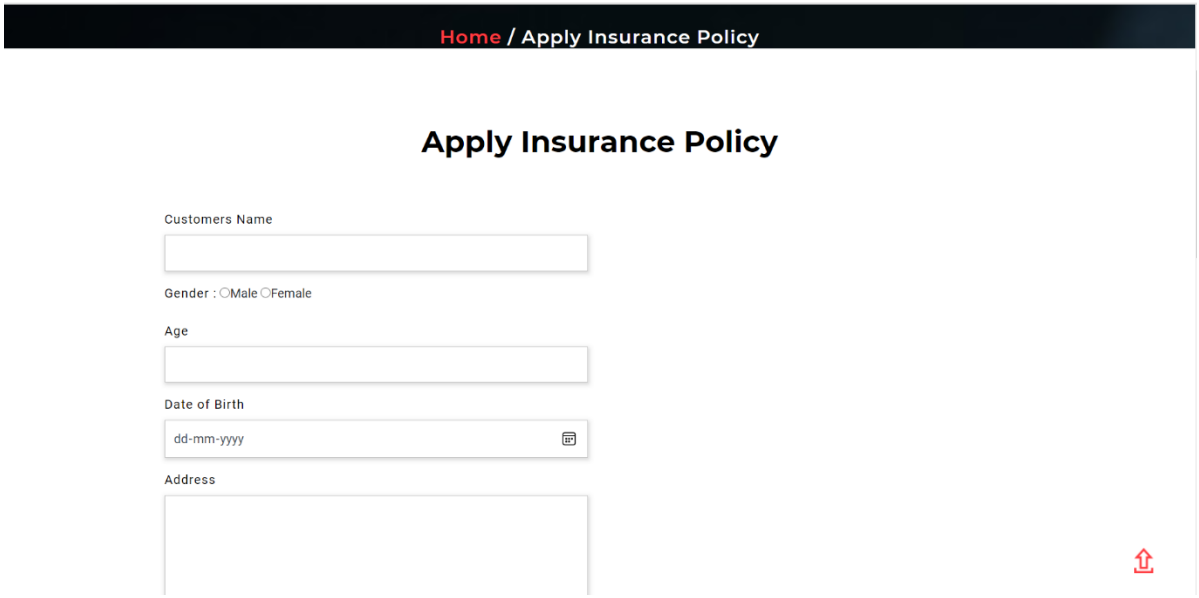
Login form



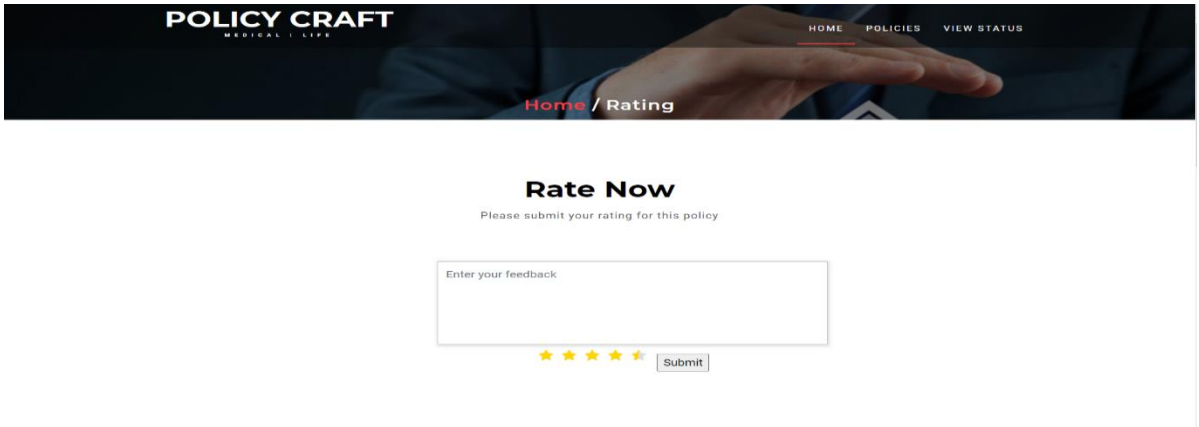
Customer registration



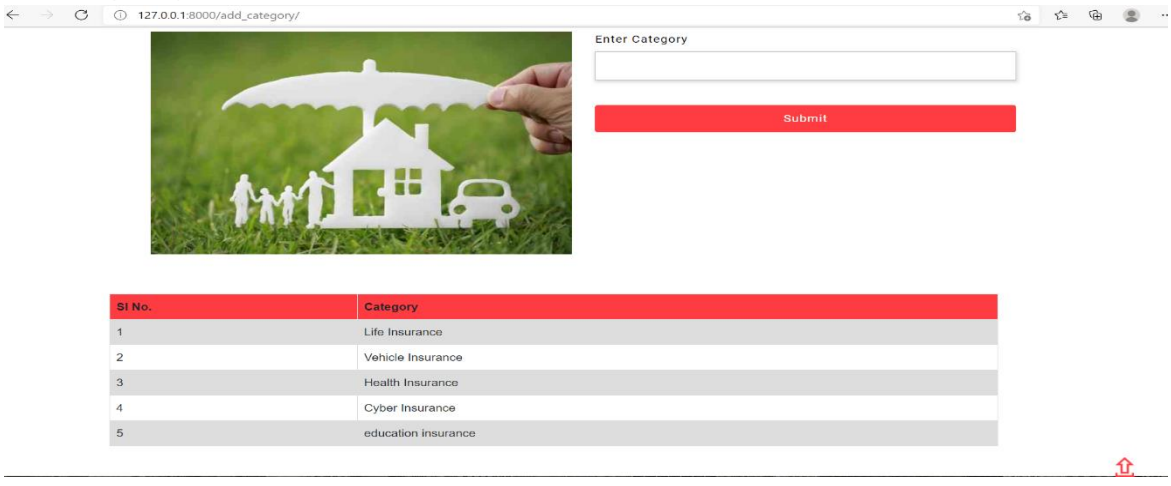
Application Form



Rating



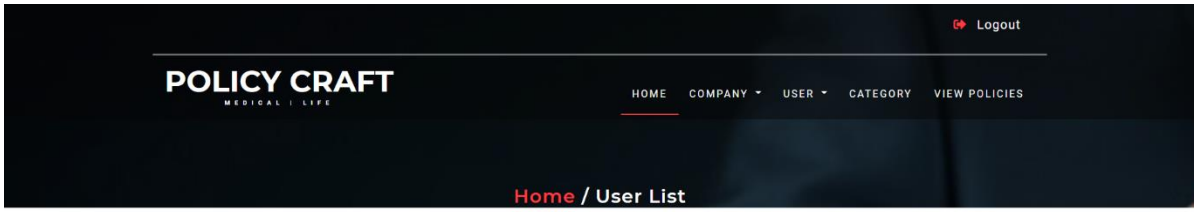
Category Form



APPENDIX- III

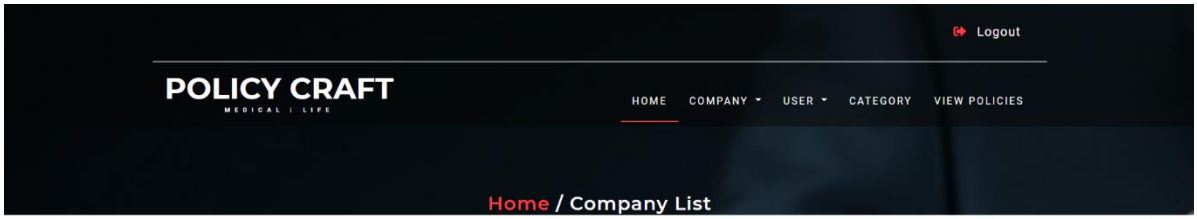
REPORTS

Users Report



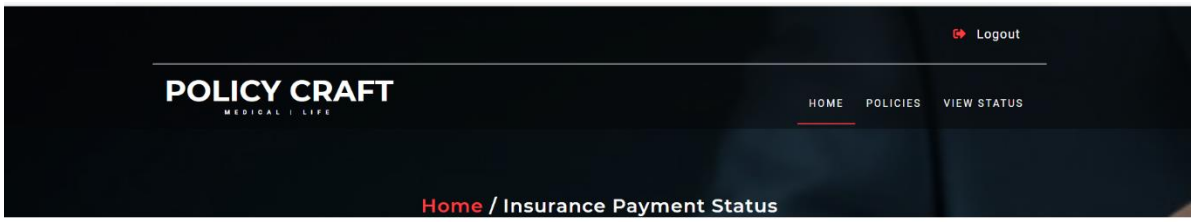
Sl No.	User Name	Gender	Email ID	Address	Phone	Count	Remove	Block
1	priya jose	female	priya@gmail.com	kalapparambath	8891878989	0	⊘	
2	aneeja joy	female	aneeja@gmail.com	fsfsggf	8089186044	0	⊘	
3	anitta george	female	anitta@gmail.com	KALATHIPARAMBIL HOUSE	7736465991	1	⊘	

Company Report



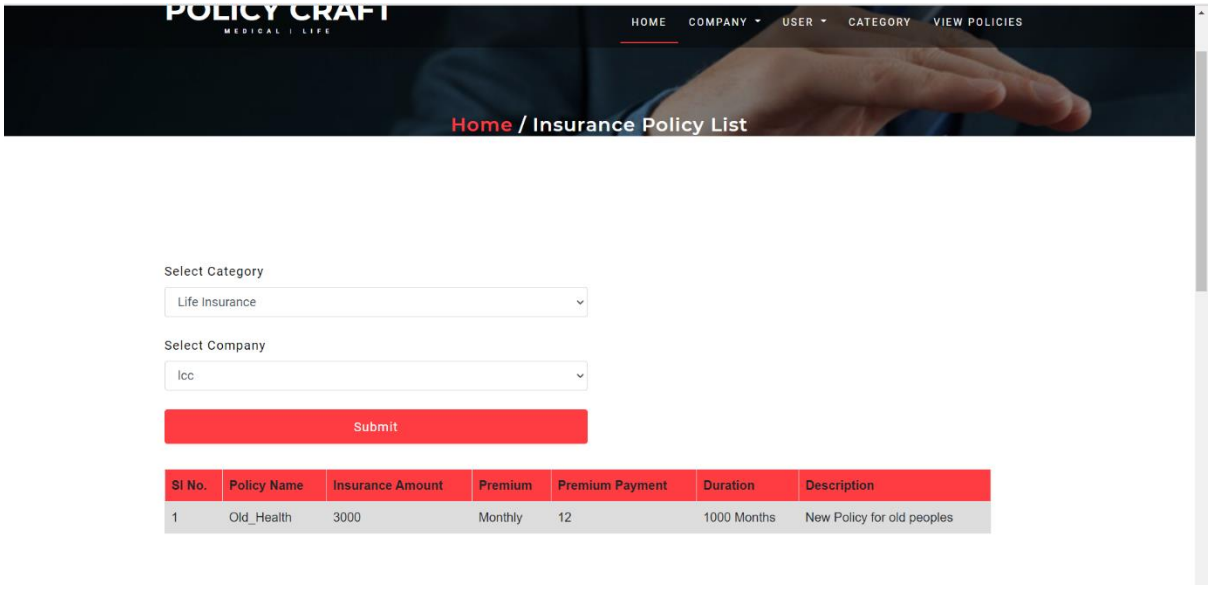
Sl No.	Company Name	Company Reg no.	Email ID	Address	Phone	Documents	Remove
1	lcc	9999999999999999	google@email.com	scfadsfv	9874563210	View Documents	⊘
2	bajaj	8990889990777878	bajaj@gmail.com	aluva	8909787786	View Documents	⊘
3	hdfc	124709038434343	hdfc@gmail.com	hdfc ,mumbai	9633654778	View Documents	⊘

Payment Report



Sl No.	Date	Amount
1	2020-01-06	1000
2	2020-01-06	1000
3	2020-01-06	1000

Policy Report



INTRODUCTION

SYSTEM ANALYSIS

SYSTEM DESIGN

CODING

IMPLEMENTATION OF SECURITY

SYSTEM TESTING

SYSTEM IMPLEMENTATION

AND

MAINTENANCE

SCOPE OF THE PROJECT

FUTURE ENHANCEMENTS

CONCLUSION

BIBLIOGRAPHY

APPENDIX A – CODING

APPENDIX B - FORMS

APPENDIX C – REPORTS