ATMIYA UNIVERSITY

RAJKOT

A

Report On

ELECTROPEDIA

Under subject of

PROJECT

B.TECH, Semester– VII

(Computer Engineering)

Submitted by:

1. JENIL VACHHANI 190002112
2. RAYJADA DHRUVRAJSINH 190002098

Prof. Nirali Borad

(Faculty Guide)

Prof. Tosal M.Bhalodia

(Head of the Department)

Academic Year

(2022-23)

CANDIDATE'S DECLARATION

We hereby declare that the work presented in this project entitled "**ELECTROPEDIA"** submitted towards completion of project in **7th Semester** of B.Tech (Computer Engineering) is an authentic record of our original work carried out under the guidance of "**Prof. Nirali Borad".**

We have not submitted the matter embodied in this project for the award of any other degree.

Semester: 7th

Place: Rajkot

Signature:

Rayjada Dhruvrajsinh
(180002098)

Vachhani Jenil
(190002112)

ATMIYA UNIVERSITY

RAJKOT

CERTIFICATE

Date:

This is to certify that the "**ELECTROPEDIA**" has been carried out by **Rayjada Dhruvrajsinh** under my guidance in fulfillment of the subject Project in COMPUTER ENGINEERING (7thSemester) of Atmiya University, Rajkot during the academic year 2022.

Prof. Nirali Borad
(Project Guide)

Prof. Tosal M.Bhalodia
(Head of the Department)

ATMIYA UNIVERSITY

RAJKOT

CERTIFICATE

Date:

This is to certify that the **"ELECTROPEDIA"** has been carried out by **Vachhani Jenil** under my guidance in fulfillment of the subject Project in COMPUTER ENGINEERING (7thSemester) of Atmiya University, Rajkot during the academic year 2022.

Prof. Nirali Borad

(Project Guide)

Prof. Tosal M.Bhalodia

(Head of the Department)

ACKNOWLEDGEMENT

We have taken many efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to Prof. Nirali Borad for their guidance and constant supervision as well as for providing necessary information regarding the Mini Project titled **"ELECTROPEDIA".** We would like to express our gratitude towards staff members of Computer Engineering Department, Atmiya University for their kind co- operation and encouragement which helped us in completion of this project.

We even thank and appreciate to our colleague in developing the project and people who have willingly helped us out with their abilities.

Rayjada Dhruvrajsinh
(180002098)

Vachhani Jenil
(190002112)

ABSTRACT

Our project is about predicting how much capacity any device (inverter, generator) would require for users' homes, offices, or any place. If there is some shortage or regular cuts of electricity in some area and user want an inverter then by using our website and by just adding the items they are using like (AC, Fans, Lights, T.V, etc.) They would know how much capacity of the inverter they have to buy. And by doing so, he/she don't have to go for different sites and pages, here in one website we will provide this facility.

INDEX

LIST OF FIGURES

LIST OF TABLES

INTODUCTION

1.1 Introduction

The major objective is to solve the issues faced with electricity cuts and what to buy for resolving that issue. Another problem is that we don't know what product the client is searching for, and hence we made a web for that. A point that displays a huge list of products from different orders, similar to electronics, mobiles, clothes, or books, needs to be suitable to identify what the client is searching for. A client can be searching for any generator or any machine then just by adding the items and the used machine in the particular office/area, they can find the machine to be used and which is best for them. The machines included in our web are UPS, Solar, Solar Plates, Inverter, Generator, etc.

1.2 PURPOSE

The electricity can be at any required voltage; in particular, it can operate AC equipment designed for mains operation, or rectified to produce DC at any desired voltage. For example, if your DVD player draws 100 watts and your laptop another 100 watts, a minimum 300-watt inverter is recommended. If the item is motor-driven, it requires additional start-up (surge) wattage (typically 2-3 times the continuous wattage required) to start the device.

1.3 SCOPE

It will provide a user friendly environment for buying any products via using our site. It has an inbuilt calculator which will provide any item of there need by just adding the number of appliances into the form Admin can add/delete product, devices, category.

1.4 PROJECT OBJECTIVE

In literature survey we look into the details of other systems which are built like our project but in that we don’t have the sections of what to buy for the particular area and so we made a project which can decide the item according to the area or according to the requirement. Here we try to reduce the disadvantages of the systems and tried to improve the performance and the efficiency of the new system. The existing Electropedia is an online application that can be accessed throughout the organization and outside as well with proper login This system can be used as an application for the ADMIN of any area that have to handle office or any other large industries to manage the system information with regards to handle this type of large area system .

CHAPTER 2

SOFTWARE REQUIREMENTS SPECIFICATION

2.1 Hardware Requirements

Table 2.1.1 Hardware Requirements

Number	Description
1	PC with 250 GB or more Hard disk.
2	PC with 2 GB RAM
3	PC with Pentium 1 and Above.

2.2 Software Requirements

Table 2.2.1 Software Requirements

Number	Description	Type
1	Operating System	Windows XP / Windows
2	Language	Python
3	Database	MySQL
4	IDE	Pycharm, VSCODE
5	Browser	Google Chrome

CHAPTER 3

PLANNING & DESIGN

3.1 PROJECT PLANNING

Project management report is an essential project management tool. It provides a summary overview of the project's status that you can share with stakeholders, clients and team members. Ideally, the project report is just a page or two long.

The project planning phase refers to:

- Developing a project to make it ready for investment.
- Determines the jobs/tasks required to attain project objectives.
- Identifying essential project sponsors and stakeholders in order to establish the project's scope, budget, and timeline for completion.
- Upon enlisting the stakeholder requirements, prioritizing/setting project objectives.
- Identifying the project deliverables required to attain the project objectives.
- Creating the project schedule.
- Identifying and developing appropriate mitigation plans for project risks, if any.
- Communicating and presenting the project plan to stakeholders.

3.1.1 PROJECT DEVELOPMENT APPROACH AND JUSTIFICATION

I have started Project Development from deciding name of the website, after that I have decided color scheme and logo of the website. Then I have created wireframes for front end designing. After that, I started the front-end development by HTML, CSS and JavaScript, after that I have done database connection with the help of PHP and XAMPP server also used Python Django.

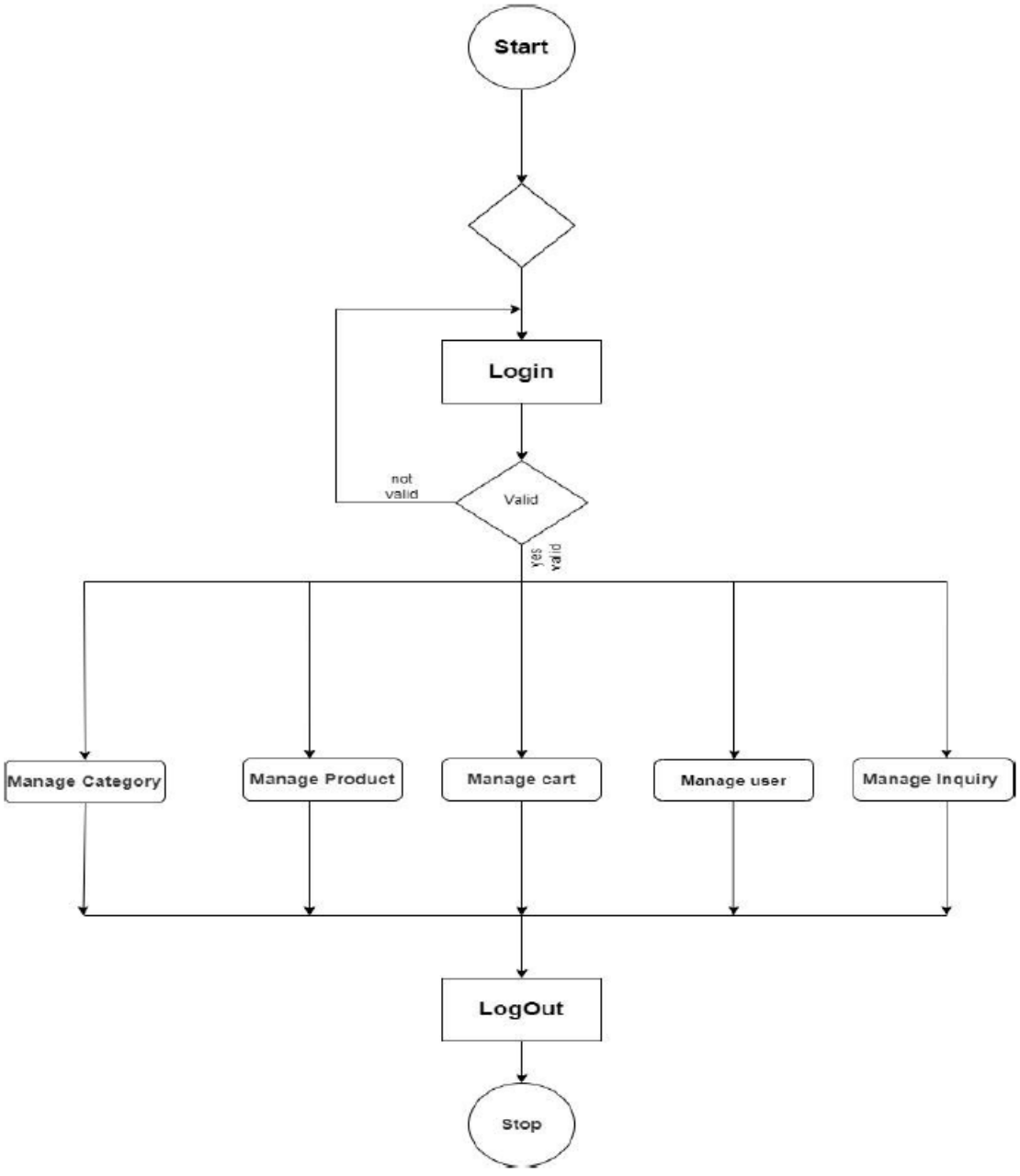
3.2 SYSTEM DESIGN

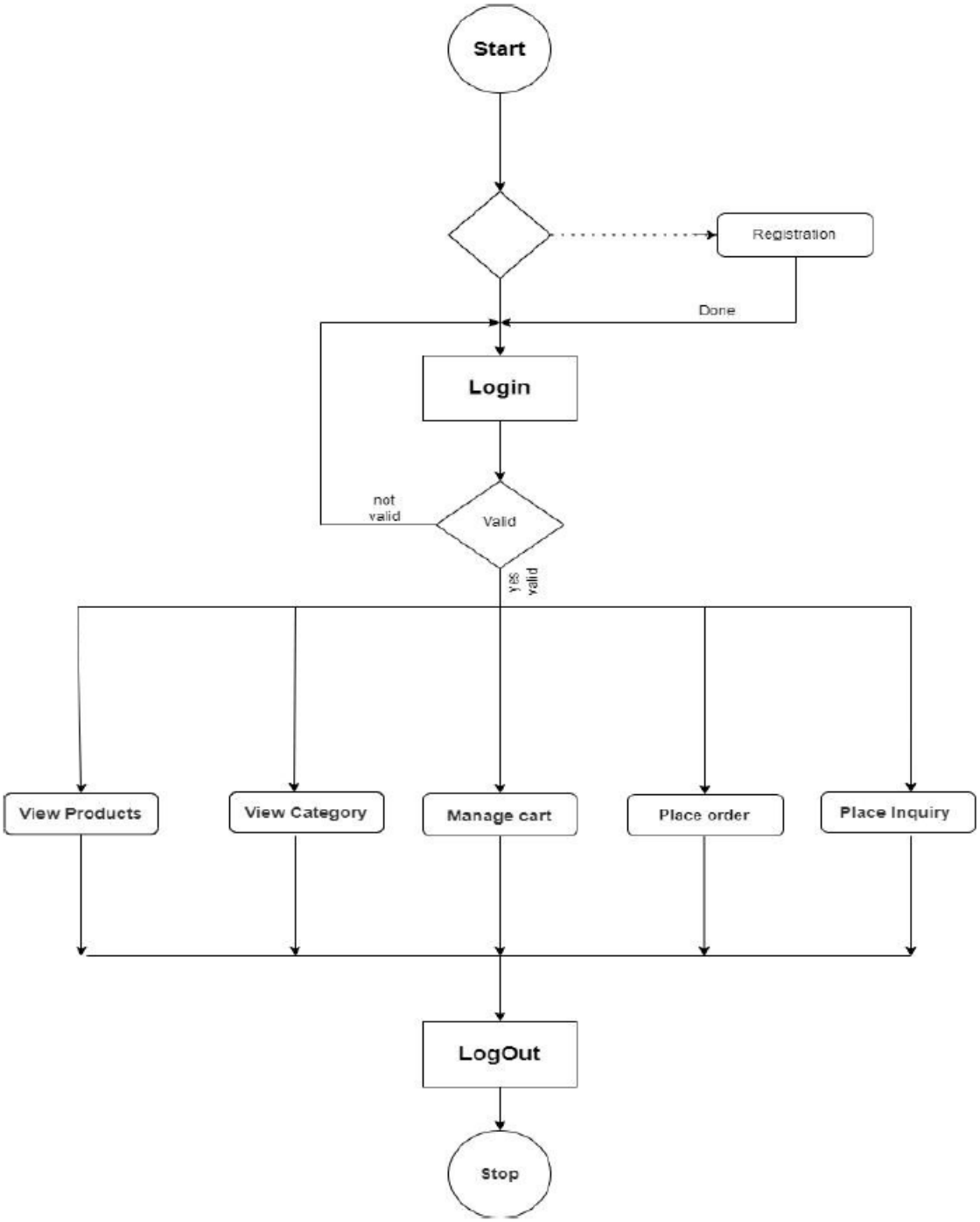
System design is started by making various UML diagrams of the project.

The list of diagrams is:

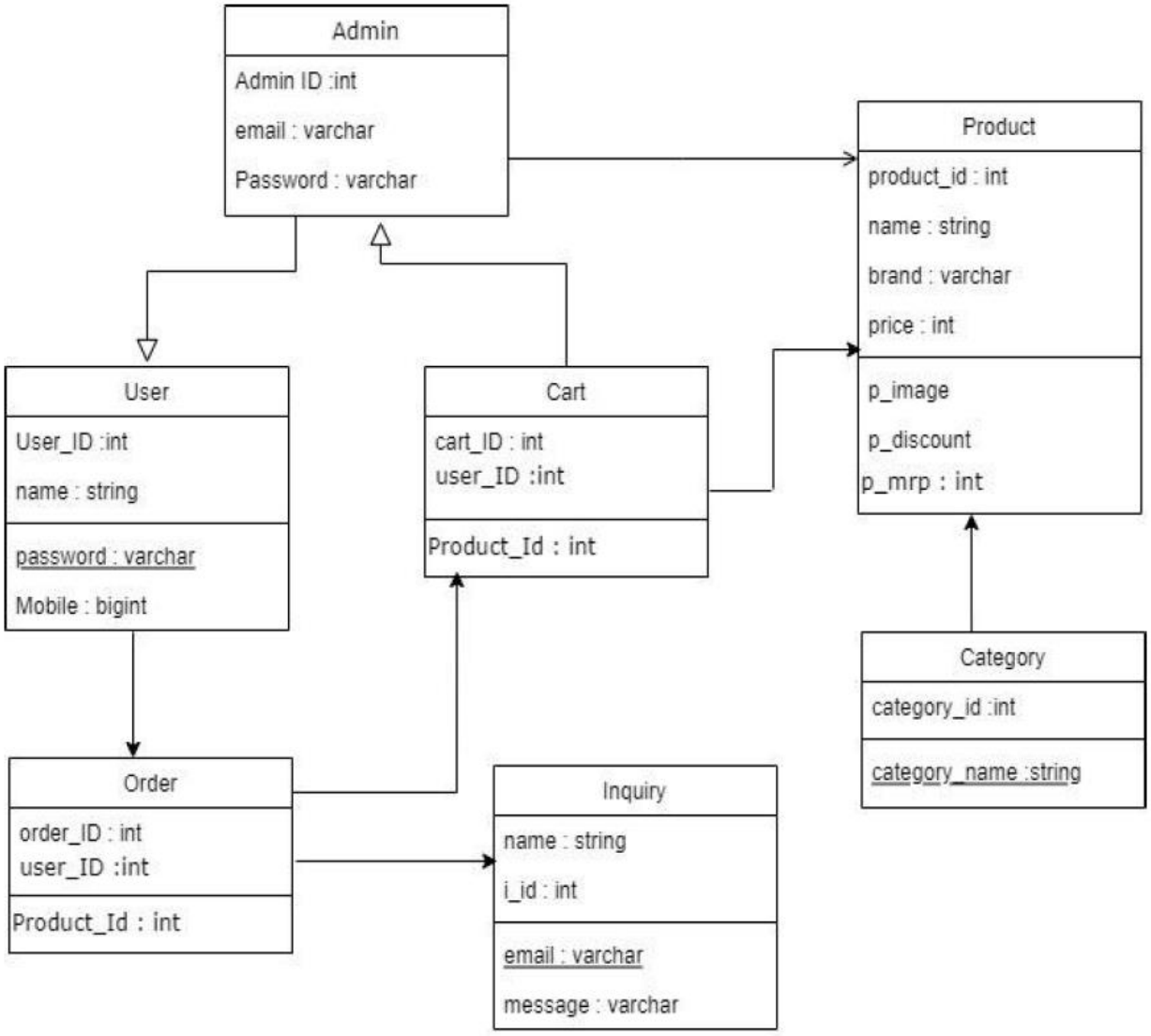
- Activity Diagram
- Class Diagram
- E-R Diagram
- Use a case Diagram
- Data Dictionary Diagram

3.3 ACTIVITY DIAGRAM

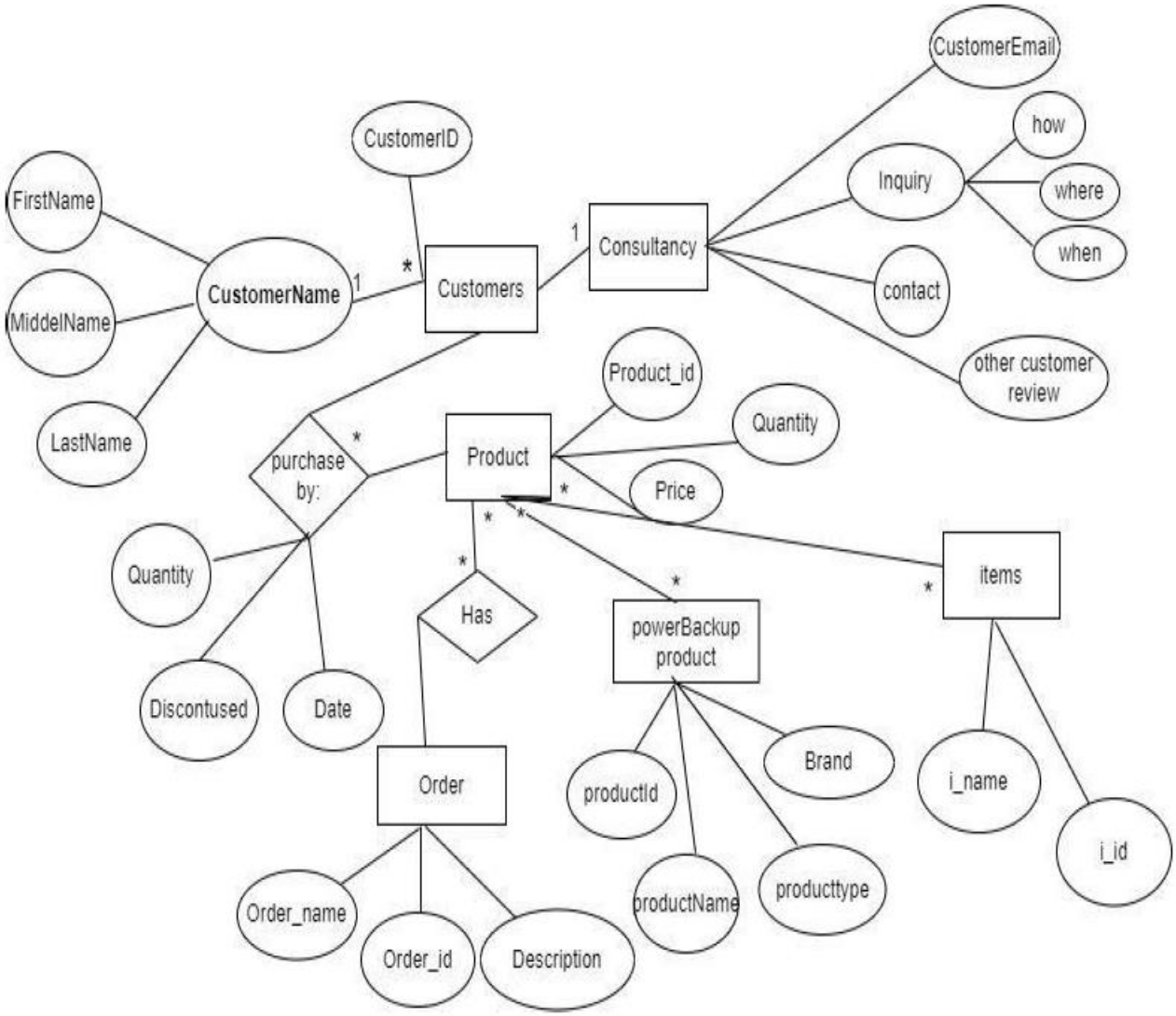



Start
Registration
Done
Login
not
valid
Valid
yes
valid
View Products
View Category
Manage cart
Place order
Place Inquiry
LogOut
Stop

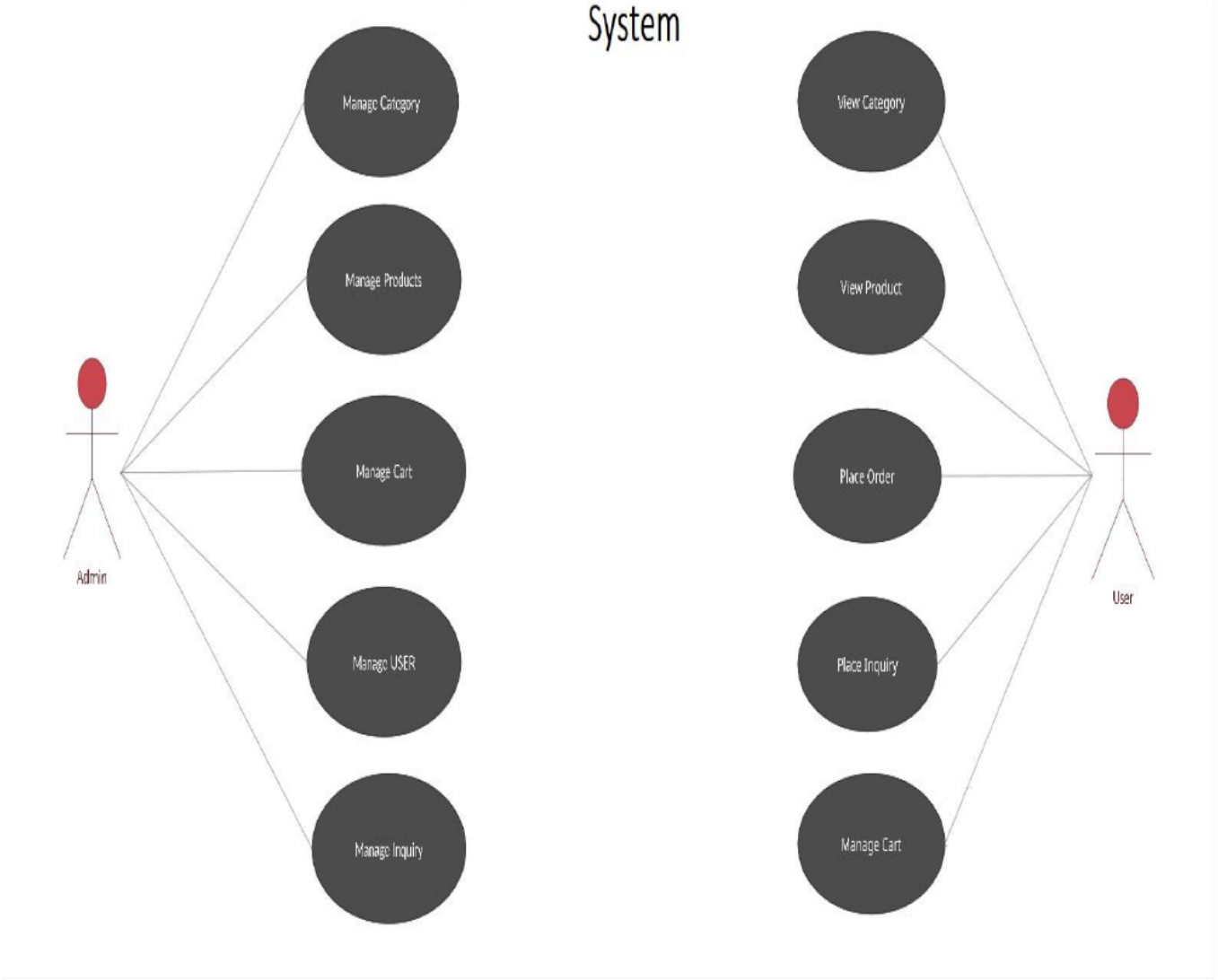
3.4 CLASS DIAGRAM



3.5 E-R DIAGRAM



3.6 USE CASE DIAGRAM



3.7 Input /Output Interface

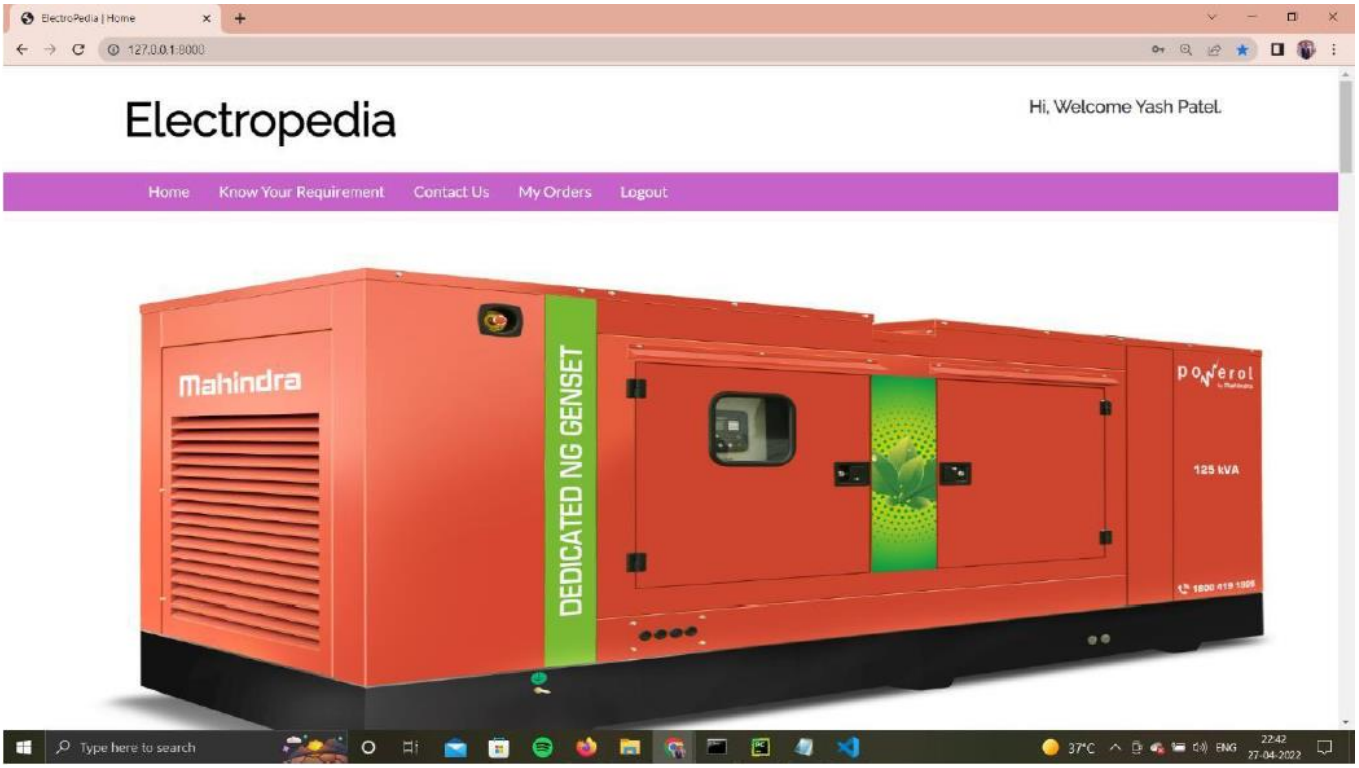


Fig.3.7.1.1 Home Page

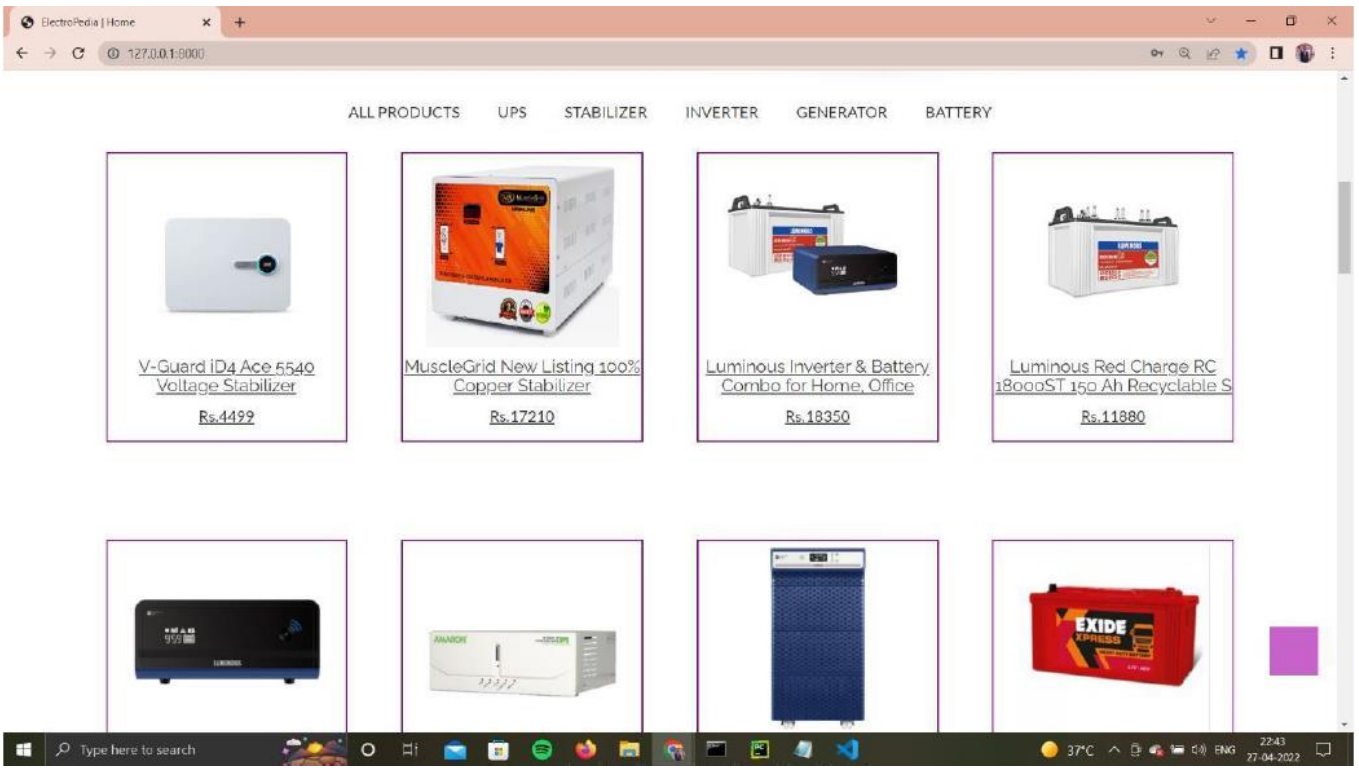


Fig.3.7.1.2 Home Page

Fig.3.7.1.3 Home Page

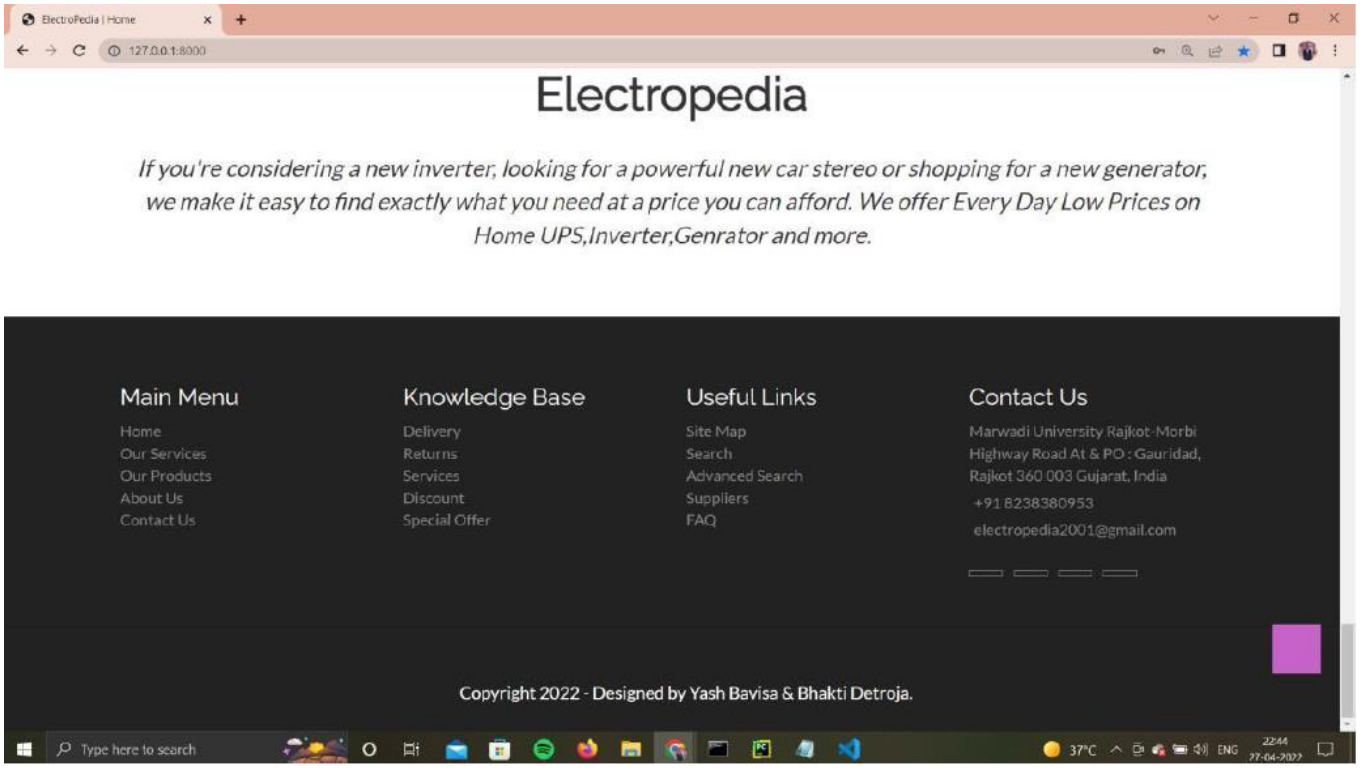


Fig.3.7.1.4 Home Page

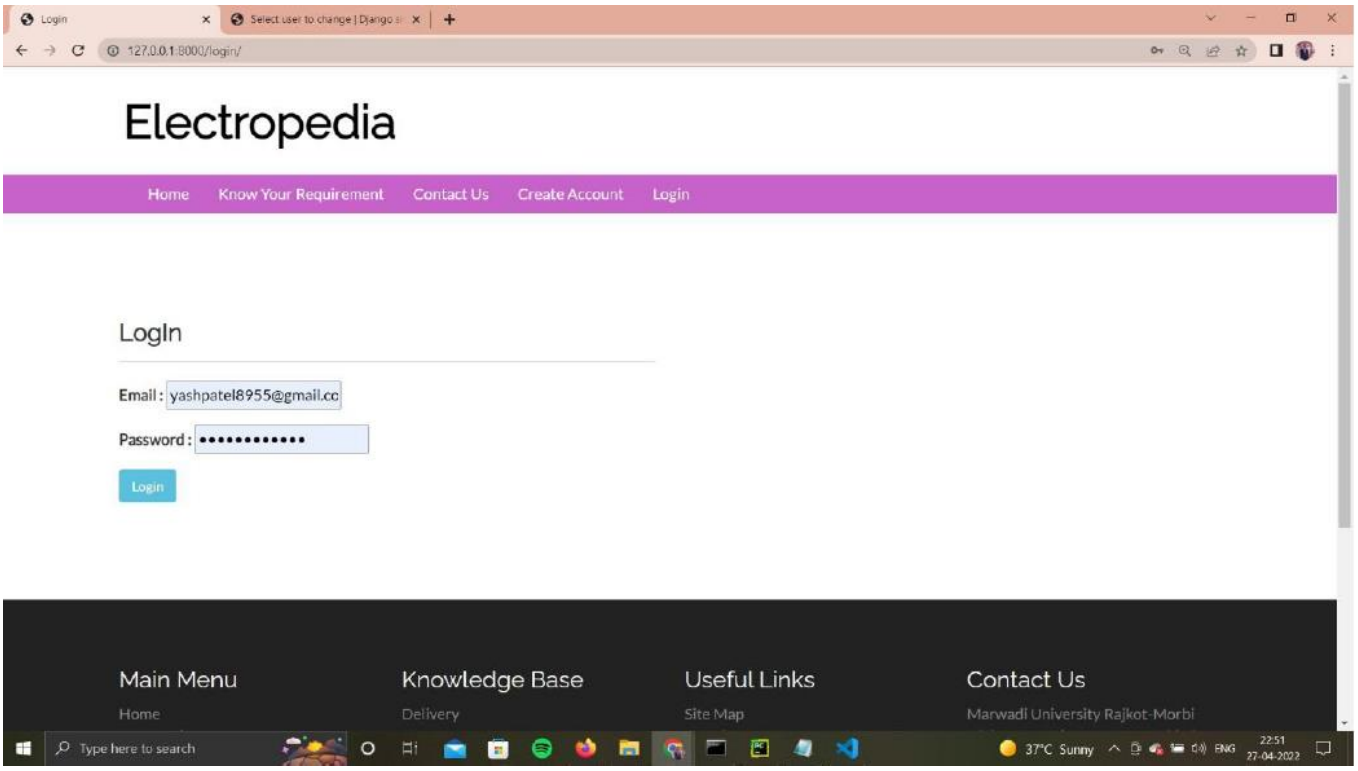


Fig.3.7.2 Login Page

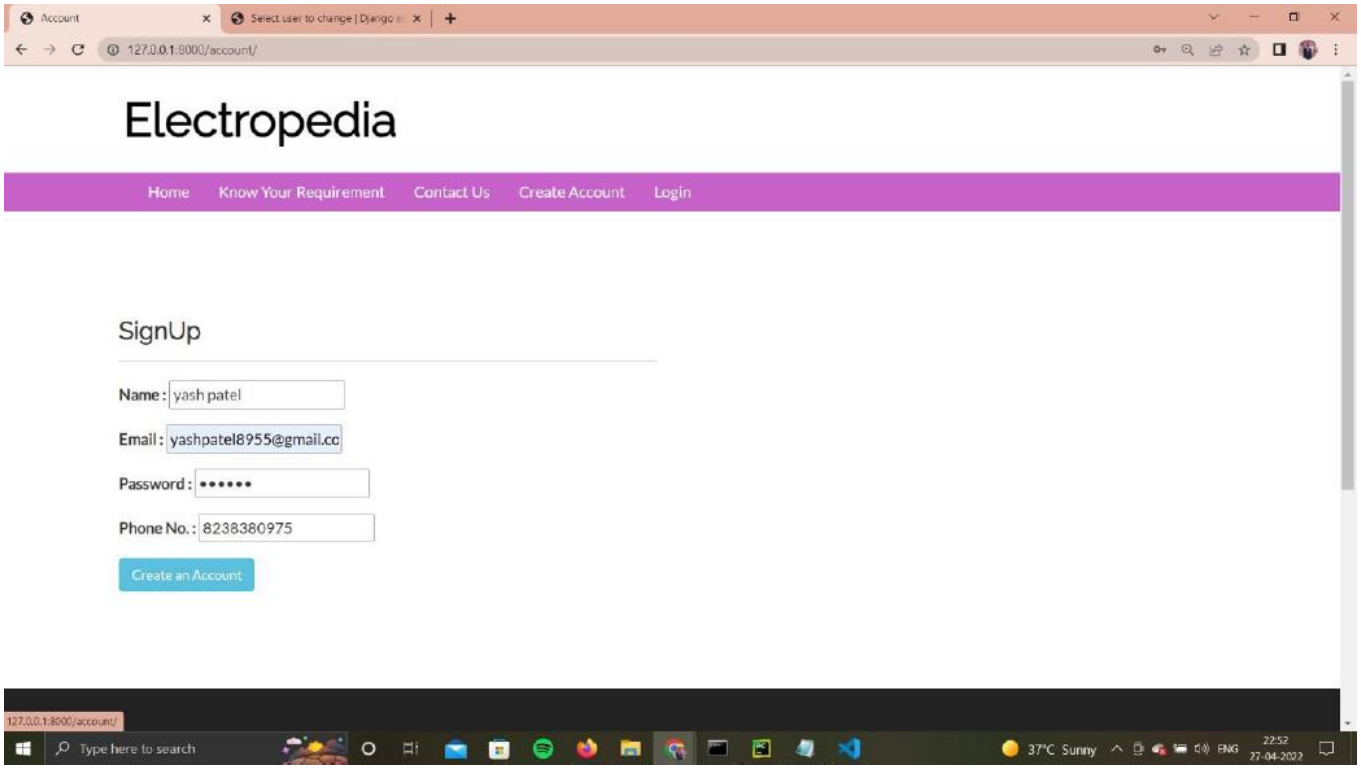


Fig.3.7.3 Create An Account Page

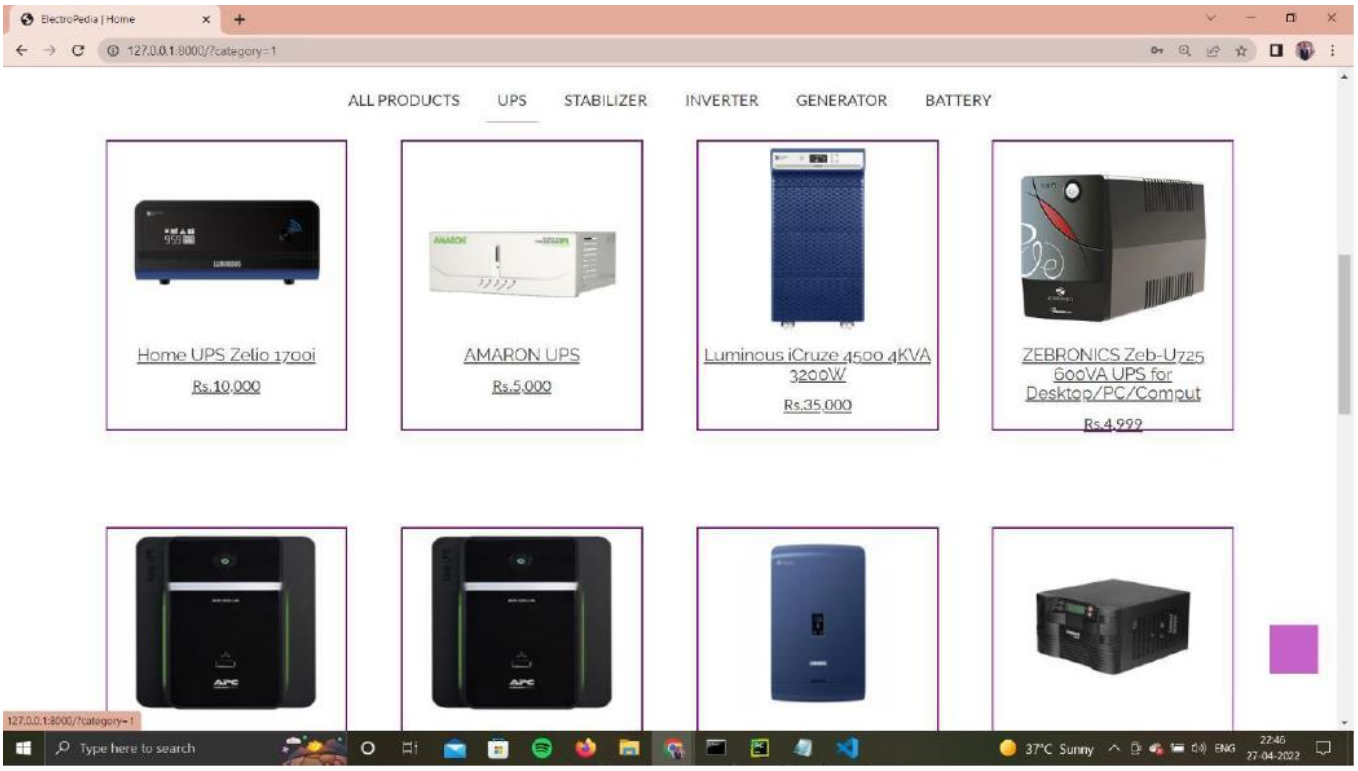


Fig.3.7.4 UPS Categories Page

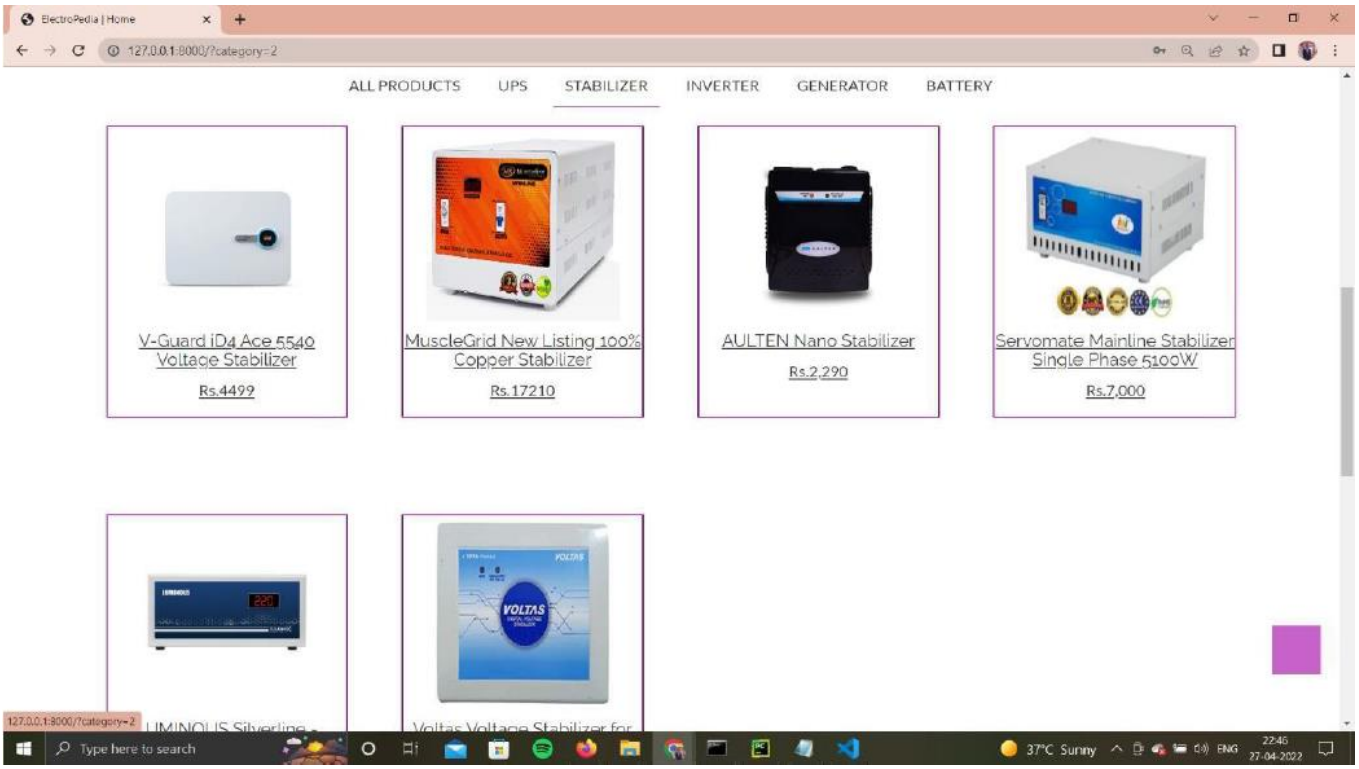


Fig.3.7.5 Stabilizer Categories Page

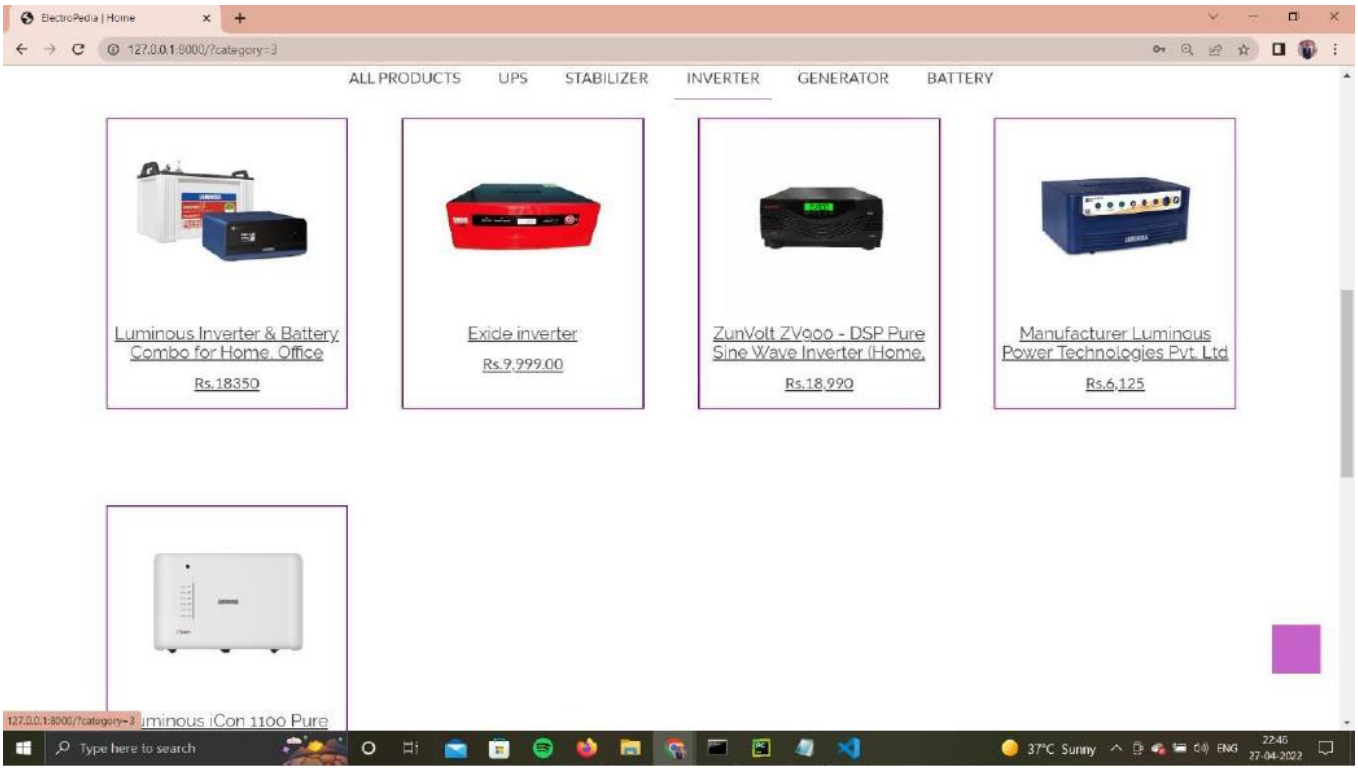


Fig.3.7.6 Inverter Categories Page

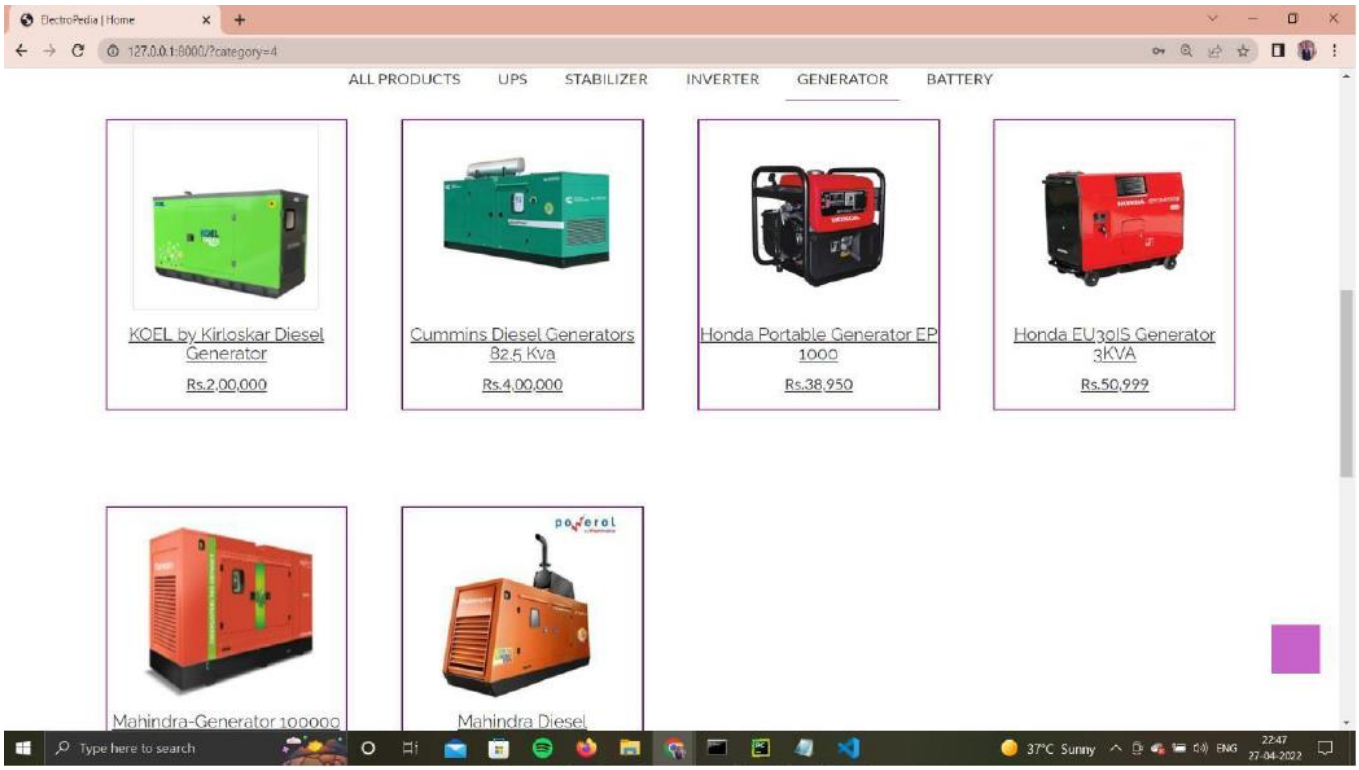


Fig.3.7.7 Generator Categories Page

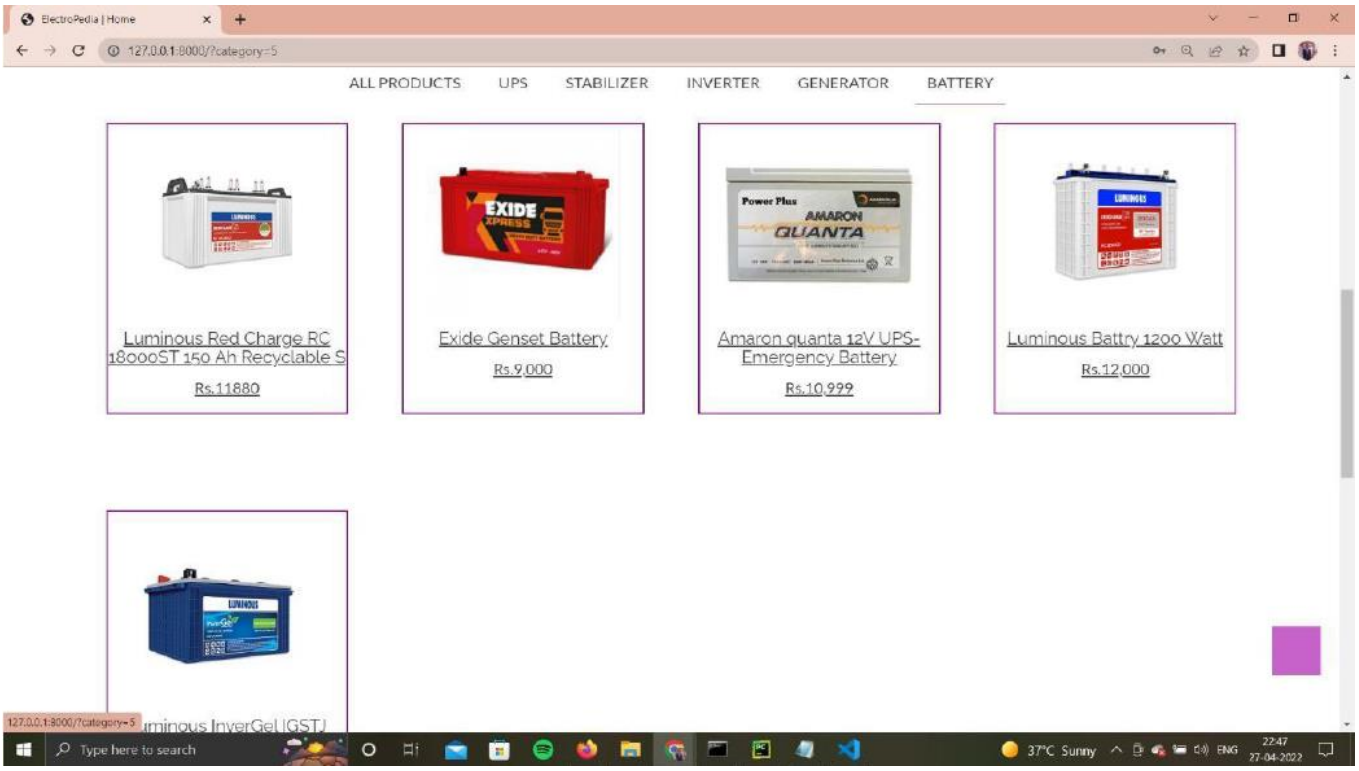


Fig.3.7.8 Battery Categories Page

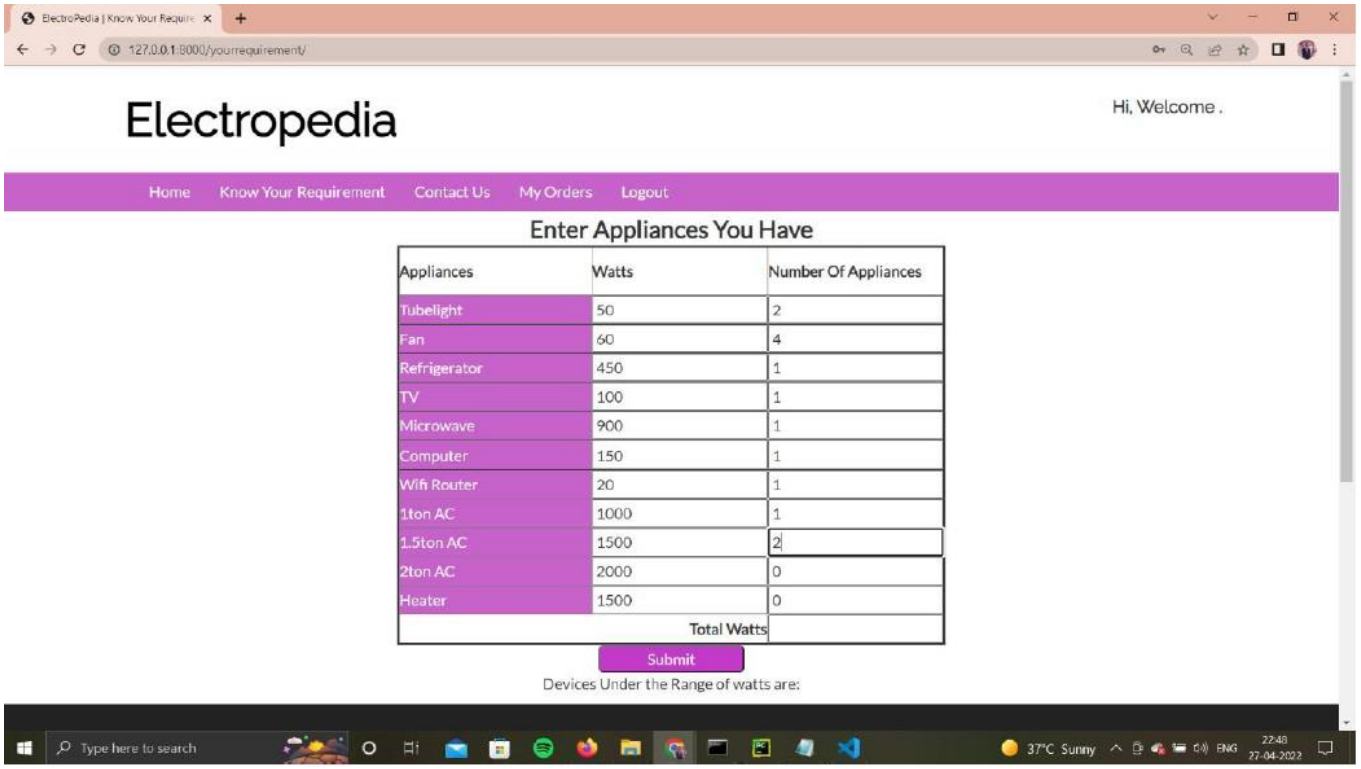


Fig.3.7.9.1 Know Your Requirement Page

Fig.3.7.9.2 Know Your Requirement Page

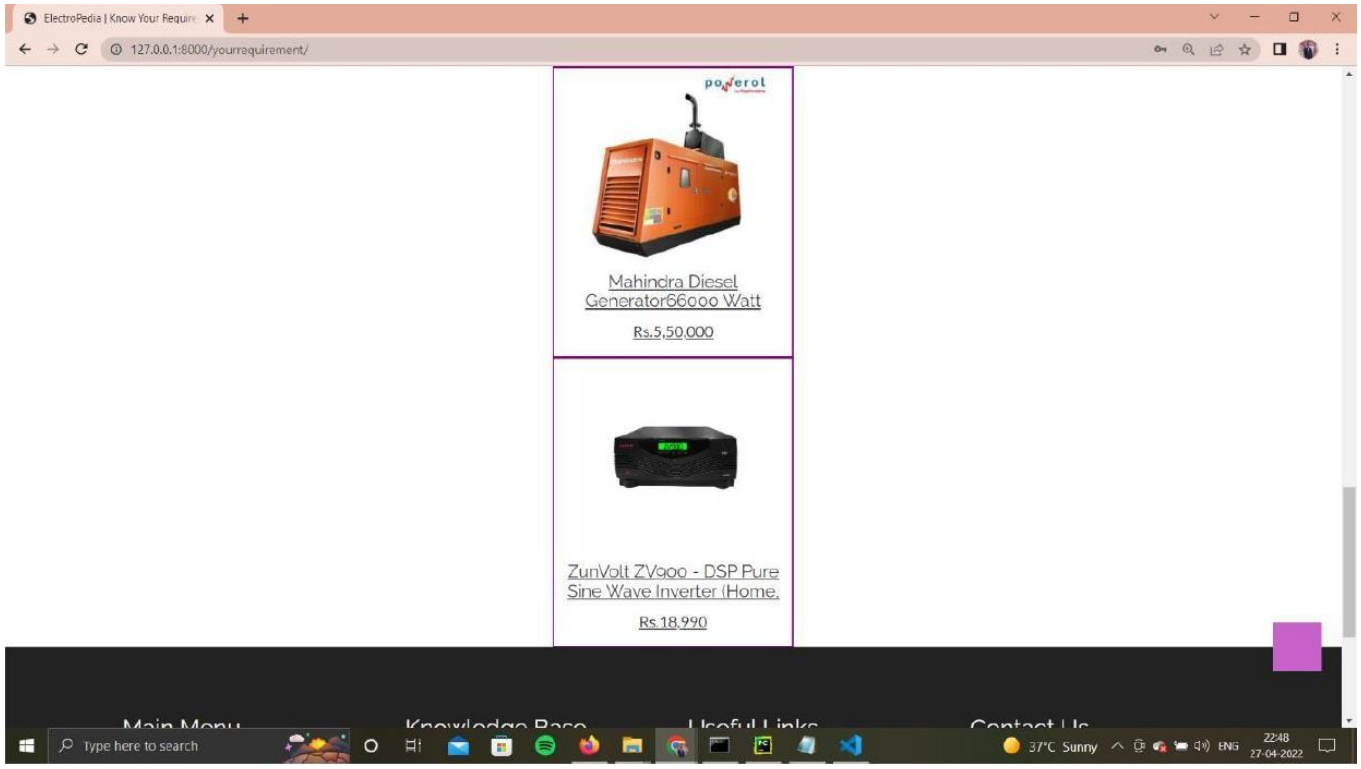


Fig.3.7.9.3 Know Your Requirement Page

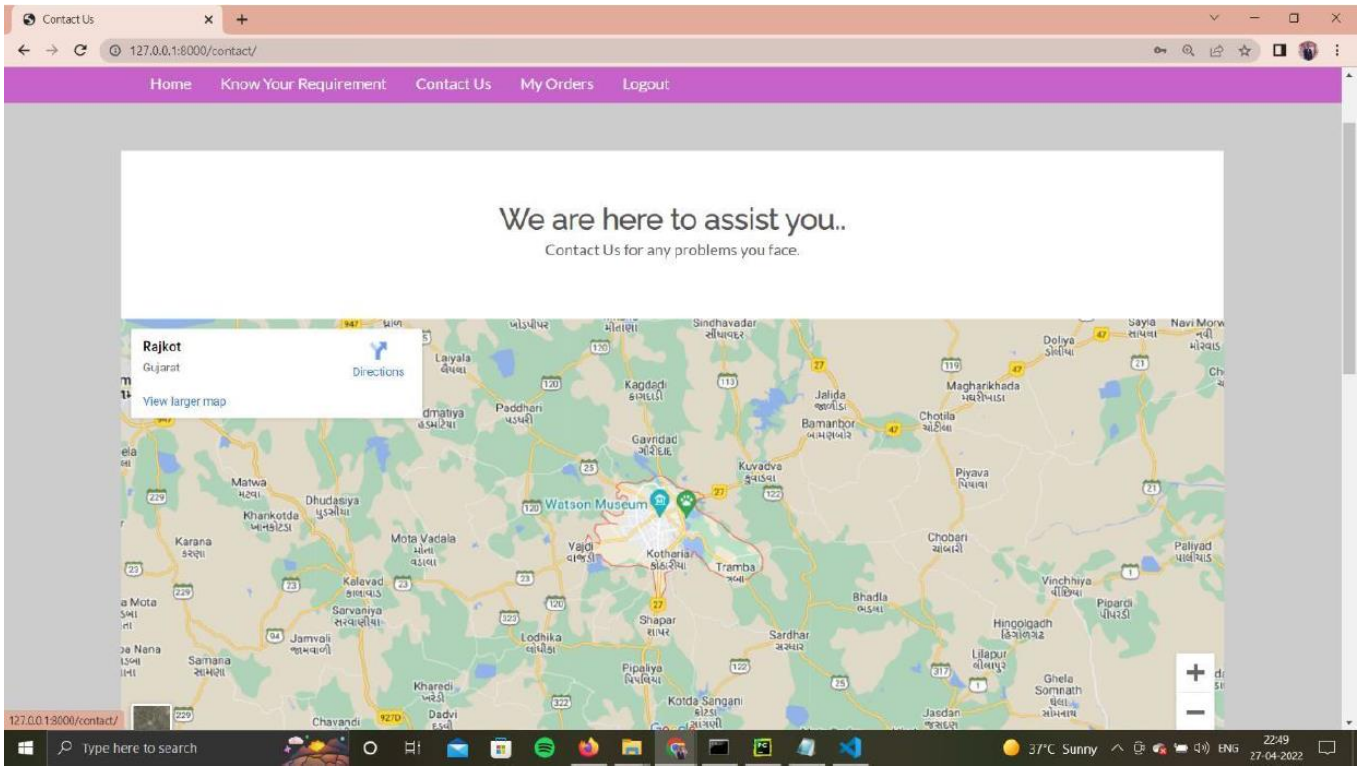


Fig.3.7.10.1 Contact Us Page

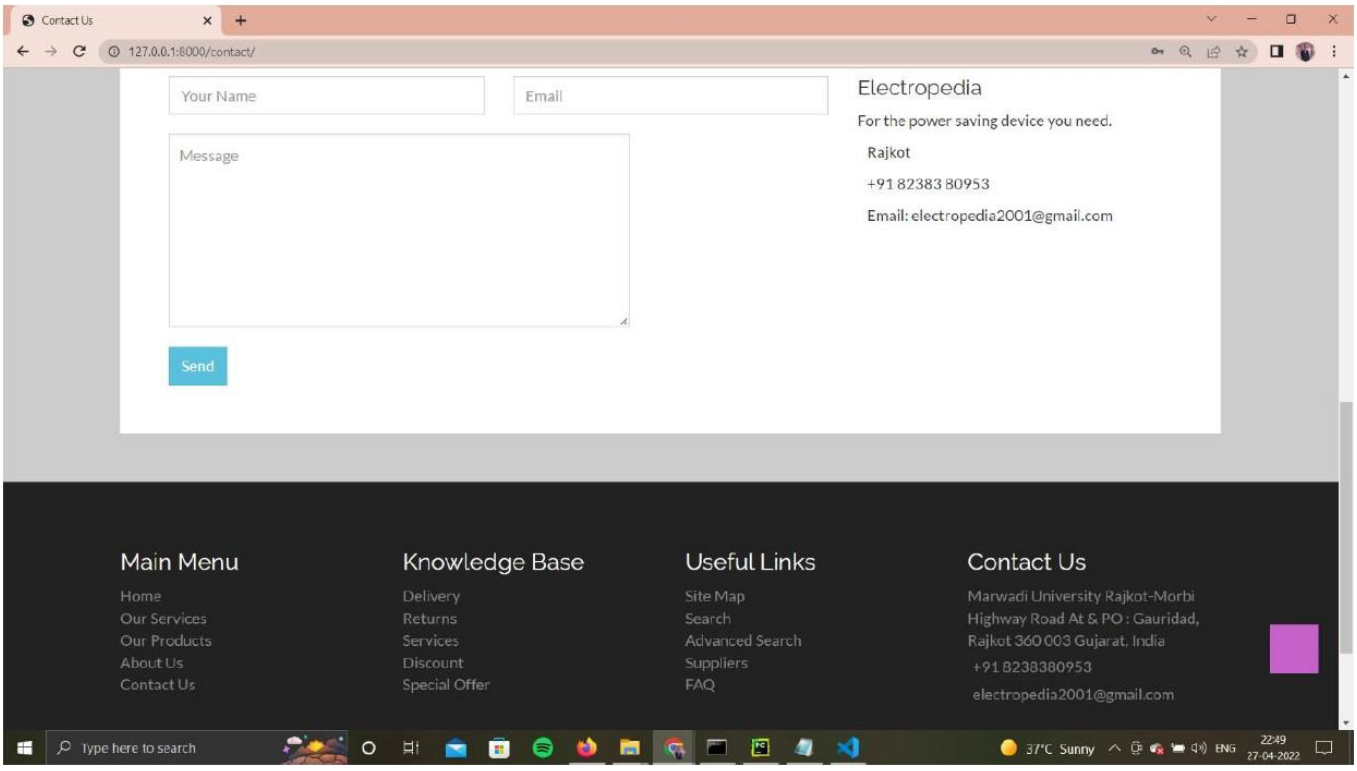


Fig.3.7.10.2 Contact Us Page

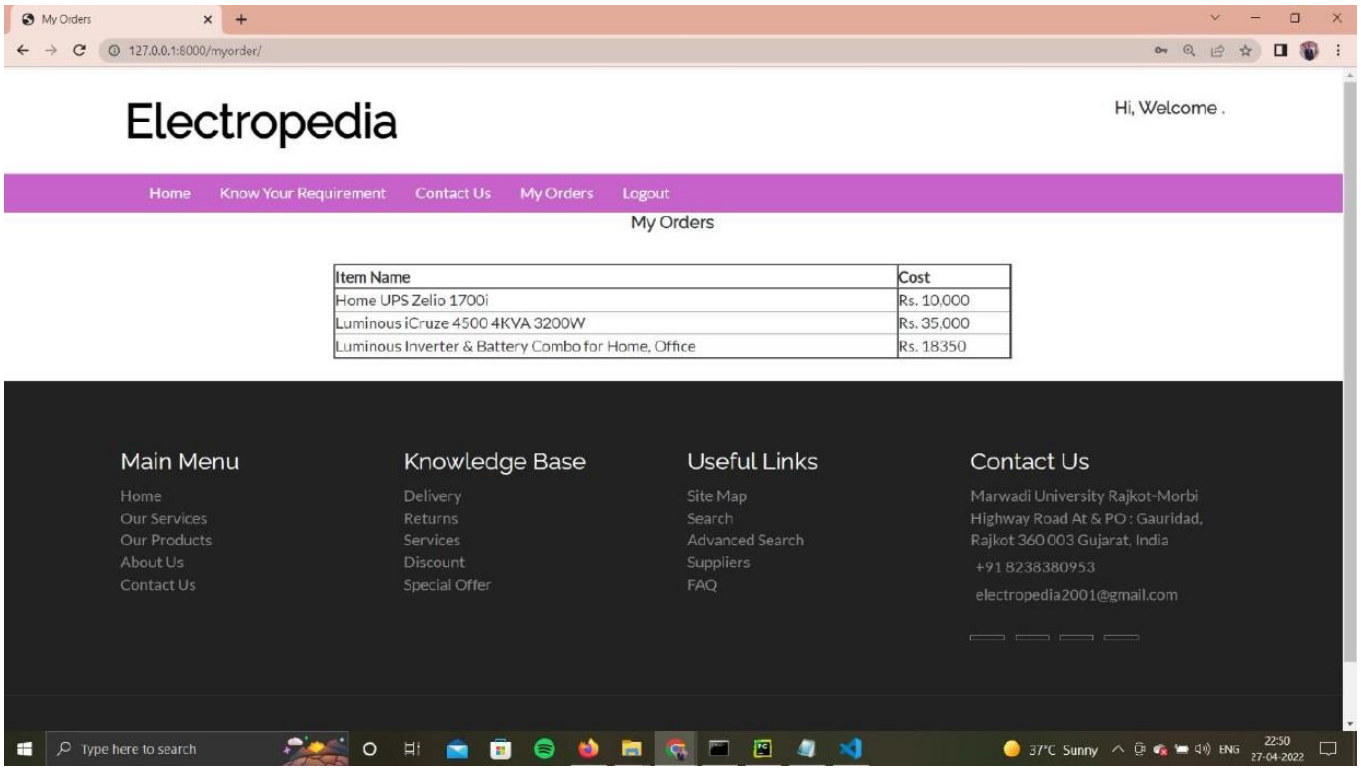


Fig.3.7.10.3 My Order Page

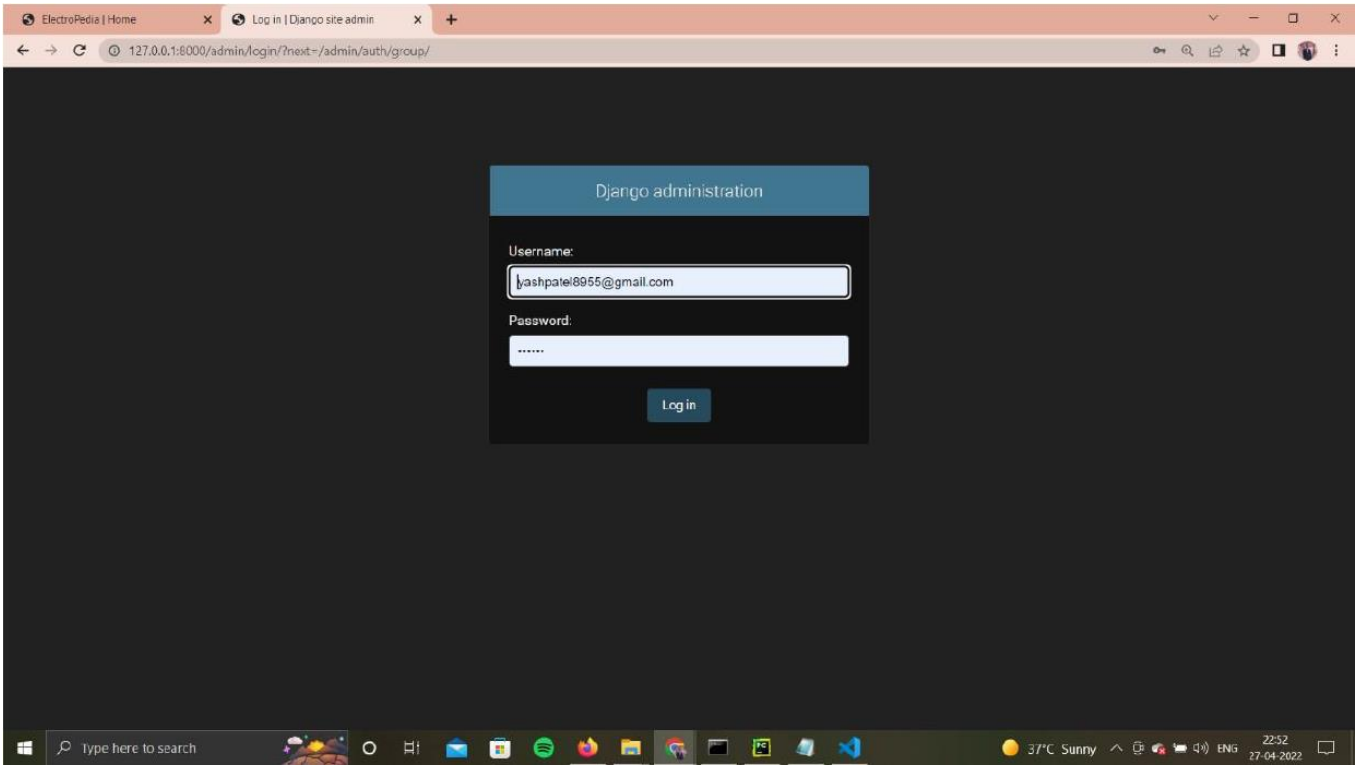


Fig.3.7.11 DJANGO ADMINISTRATION LOGIN PAGE

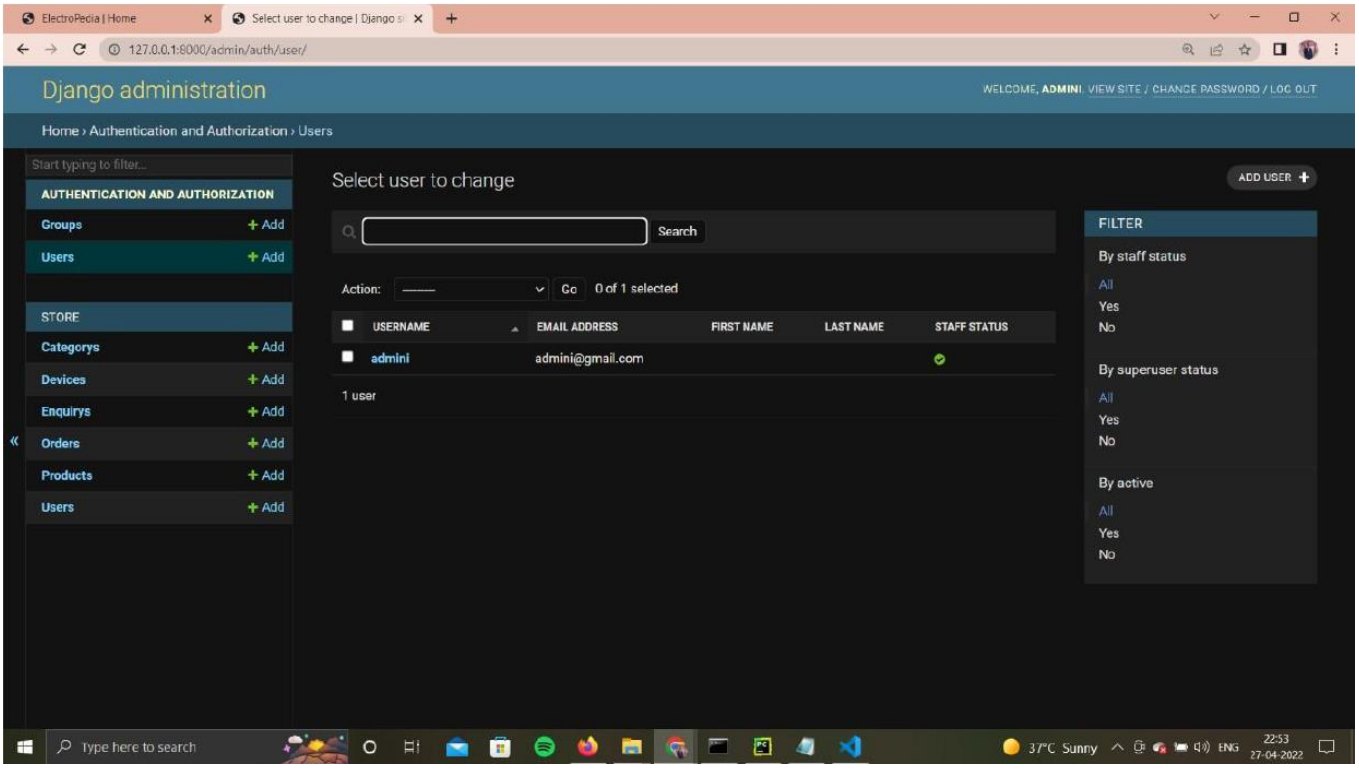


Fig.3.7.11.1 ADMIN USER LOGIN PAGE

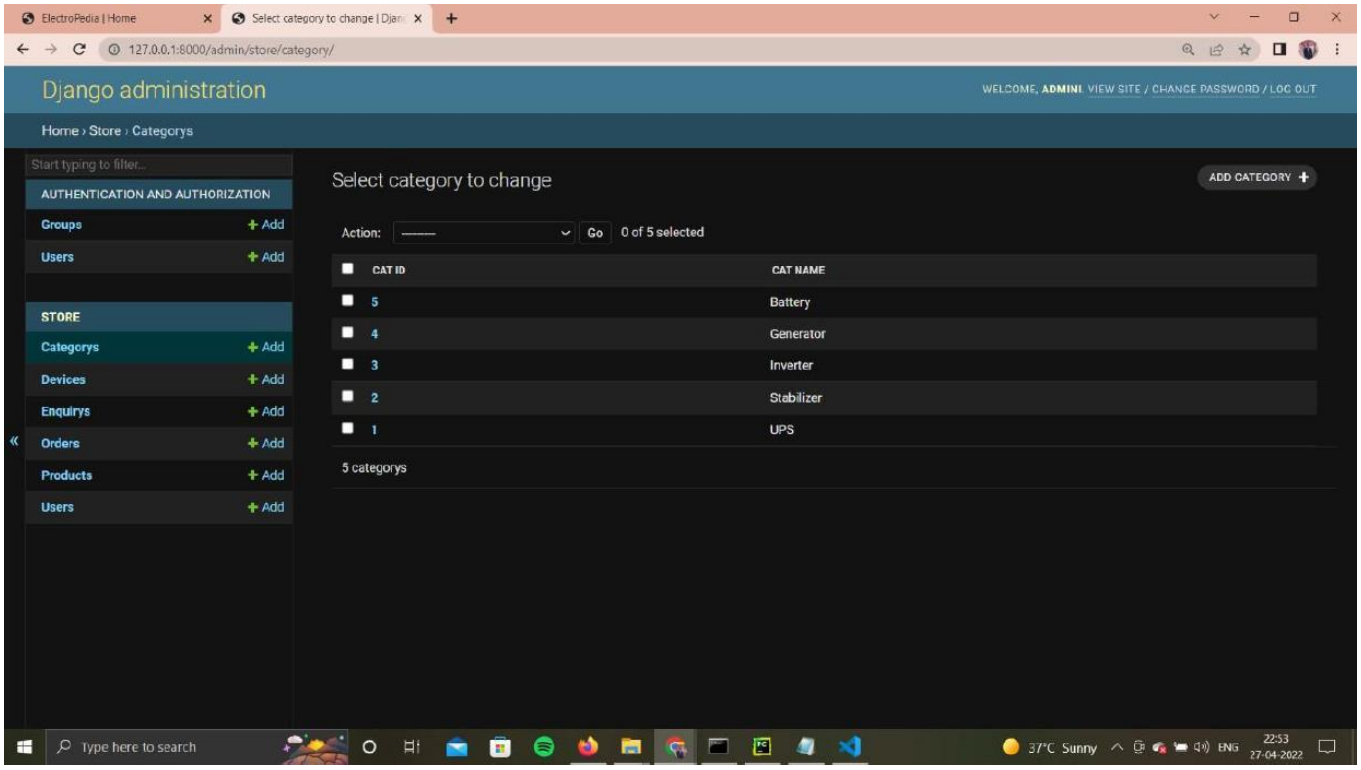


Fig.3.7.11.2 ADMIN CATEGORY PAGE

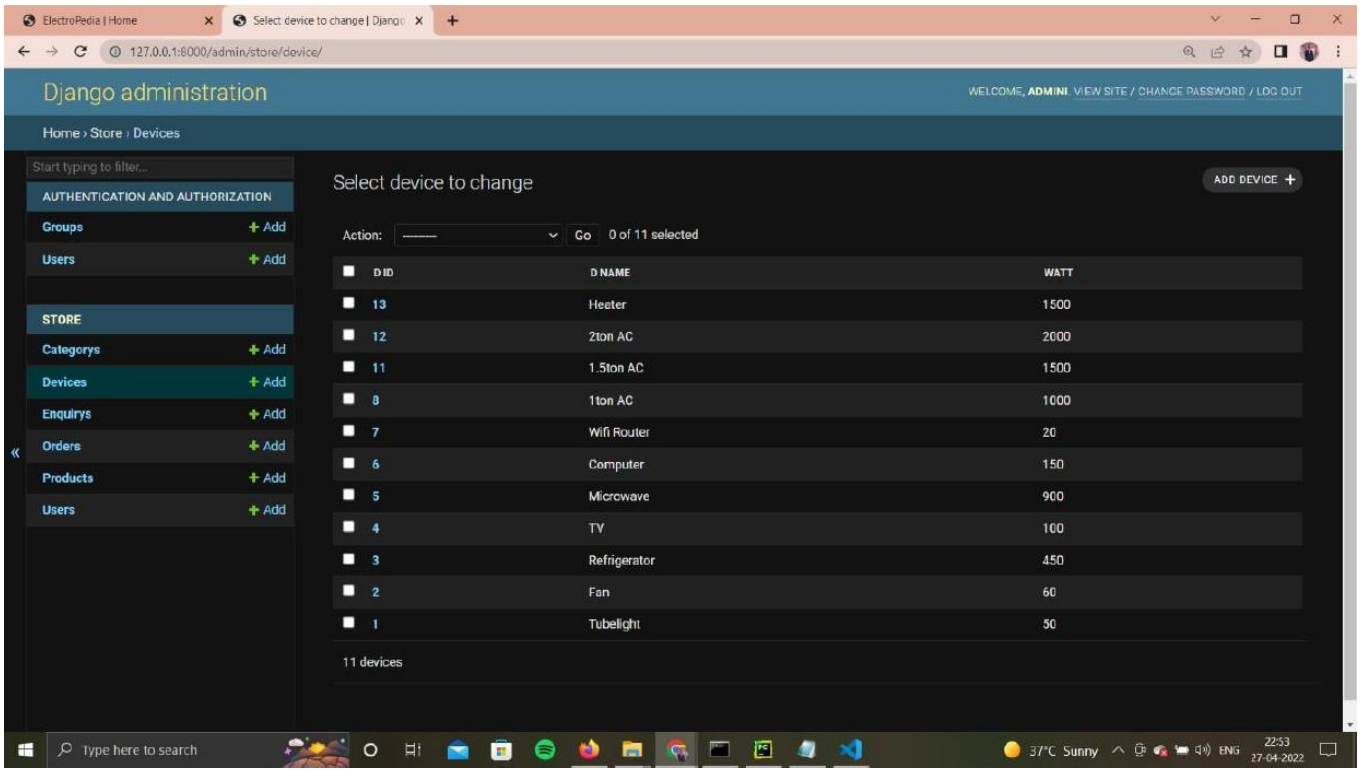


Fig.3.7.11.3 ADMIN DEVICES PAGE

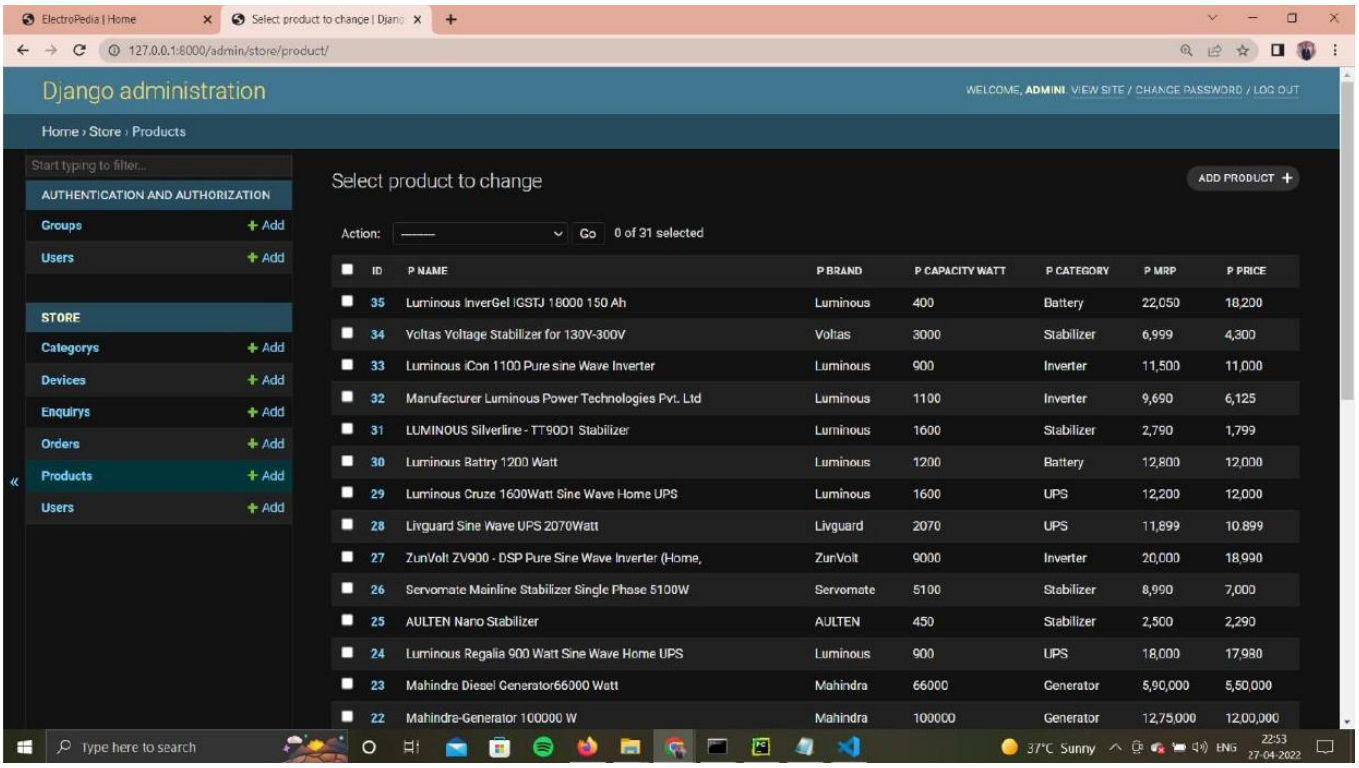


Fig.3.7.11.4 ADMIN PRODUCTS PAGE

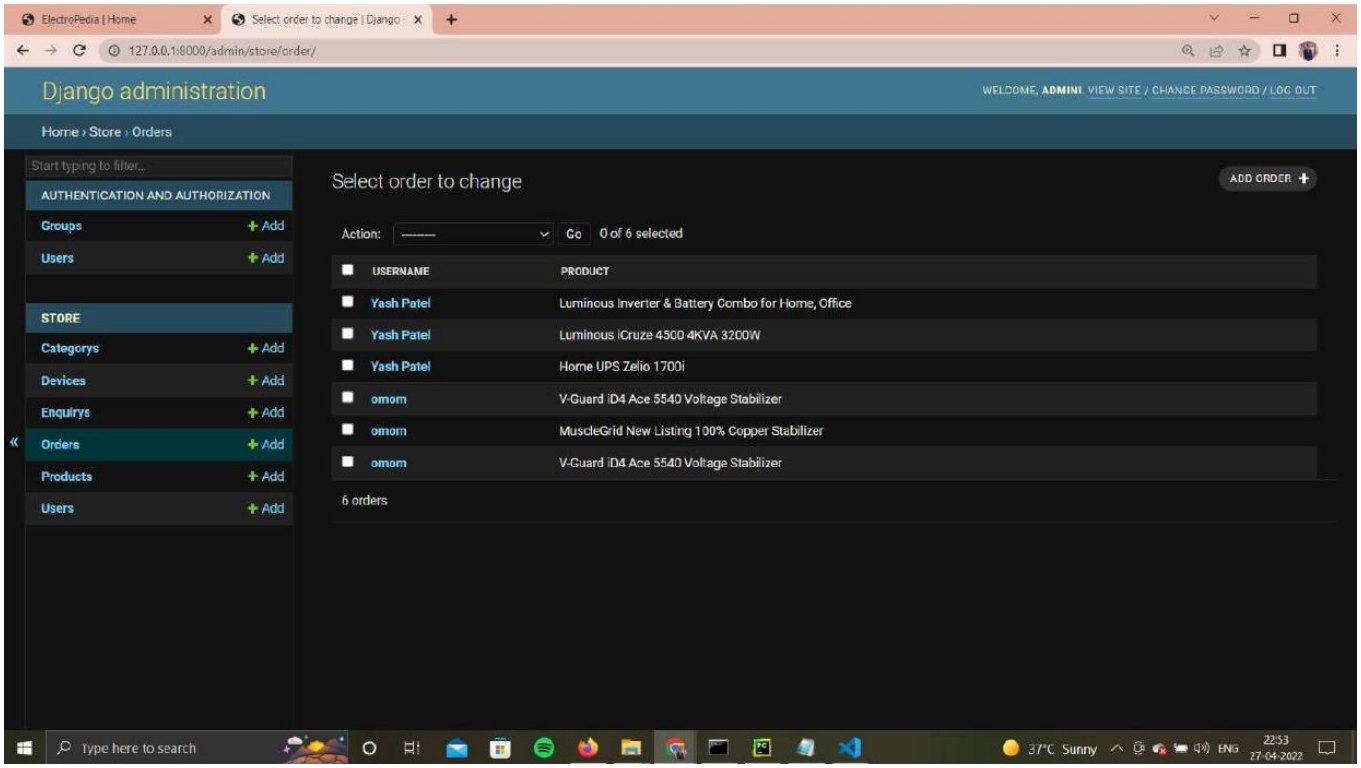


Fig.3.7.11.5 ADMIN ORDER PAGE

CHAPTER 4

IMPLEMENTATION DETAILS

In this Section we will do Analysis of Technologies to use for implementing the project.

4.1 FRONT END

4.1.1 HTML

Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

4.1.2 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML.CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

CSS information can be provided from various sources. These sources can be the web browser, the user and the author. The information from the author can be further classified into inline, media type, importance, selector specificity, rule order, inheritance and property definition. CSS style information can be in a separate document or it can be embedded into an HTML document. Multiple style sheets can be imported. Different styles can be applied depending on the output device being used; for example, the screen version can be quite different from the printed version, so that authors can tailor the presentation appropriately foreach medium.The style sheet with the highest priority controls the content display. Declarations not set in the highest priority source are passed on to a source of lower priority, such as the user agent style. The process is called cascading.

One of the goals of CSS is to allow users greater control over presentation. Someone who finds red italic headings difficult to read may apply a different style sheet. Depending on the browser and the web site, a user may choose from various style sheets provided by the designers, or may remove all added styles and view the site using the browser's default styling, or may override just the red italic heading style without altering other attributes.

4.2 BACK END

4.2.1 PYTHON DJANGO

Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and "pluggability" 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.

Tool Used: PYCHARM, VS CODE, GOOGLE CHROME, MOZILLA FIREFOX.

Database: A database is a collection of tables, with related data.

Table: A table is a matrix with data. A table in a database looks like a simple spadsheet.

Column: One column (data element) contains data of one and the same kind, for example the column postcode.

Row: A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.

Redundancy: Storing data twice, redundantly to make the system faster.

Primary Key: A primary key is unique. A key value cannot occur twice in one table. With a key, you can find at most one row.

Foreign Key: A foreign key is the linking pin between two tables.

Compound Key: A compound key (composite key) is a key that consists of multiple columns because one column is not sufficiently unique.

Index: An index in a database resembles an index at the back of a book.

Referential Integrity: Referential Integrity makes sure that a foreign key value always points to an existing row.

4.3 Coding Standards

Normally, good software development organization requires their programmers to adhere to some well defined and standard style of coding called coding standard.

4.3.1 Variable Standards:

We have used meaningful variables name.

4.3.2 Comment Standards:

The comment should describe what is happening, how it is being done, what parameters mean, which global are used and which are modified, and any registration or bugs. The standards I have followed are:

- Every script should begin with a comment block, which describes the scripts purpose; any argument used (if applicable), and return values (if applicable), inputs-outputs, and name of script.
- Comment may also be used in the body of the script to explain individual sections or lines of codes.
- It is also used to describe variable definition or declarations.
- Inline comments should be made with the. //. Comment style and should be indented at the same level as the code described.
- For multiple line comments we write between /* ….*/.

4.4 Program/Module Specification

- System GUI must be as simple and user friendly as anyone can use it. At front side we implemented registration form for access the system.
- Authentication is necessary to enter into the system only if one requires to start his/her own project. This is required to prevent unauthorized access to the system.
- If someone steals the password of the administrator or any regular user then he can able to change the database or misuse the system and can enter in restricted area so for this purpose system will provide encrypted password storage format in the database. Option to change the Password.
- A Session is maintained throughout the system when a particular user enters into the system. The Session is regularly checked whenever it is required.

CHAPTER 5

TESTING AND IMPLEMENTATION

The term implementation has different meanings ranging from the conversation of a basic application to a complete replacement of a computer system. The procedures however, are virtually the same. Implementation includes all those activities that take place to convert from old system to new. The new system may be totally new replacing an existing manual or automated system or it may be major modification to an existing system. The method of implementation and time scale to be adopted is found out initially. Proper implementation is essential to provide a reliable system to meet organization requirement.

5.1 UNIT TESTING

5.1.1 Introduction

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for component testing. Ideally, each test case is independent from the others. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended.

5.1.2 Benefits

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits.

1) Find problems early : Unit testing finds problems early in the development cycle. In test-driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build. If the unit tests fail, it is considered to be a bug either in the changed code or the tests themselves. The unit tests then allow the location of the fault or failure to be easily traced. Since the unit tests alert the development team of the problem before handing the code off to testers or clients, it is still early in the development process.

2) Facilitates Change : Unit testing allows the programmer to refactor code or upgrade system libraries at a later date, and make sure the module still works correctly (e.g., in regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can be quickly identified. Unit tests detect changes which may break a design contract.

3) Simplifies Integration : Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier.

4) Documentation : Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit, and how to use it, can look at the unit tests to gain a basic understanding of the unit's interface (API).Unit test cases embody characteristics that are critical to the success of the unit. These characteristics can indicate appropriate/inappropriate use of a unit as well as negative behaviors that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development.

5.2 INTEGRATION TESTING

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing**.**

5.2.1 Purpose

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e., assemblages (or groups of units), are exercised through their interfaces using black-box testing, success and error casesbeing simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e., unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.Software integration testing is performed according to the software development life cycle (SDLC) after module and functional tests.

5.2.1.1 Big Bang

In the big-bang approach, most of the developed modules are coupled together to form a complete software system or major part of the system and then used for integration testing. This method is very effective for saving time in the integration testing process. However, if the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing.A type of big-bang integration testing is called "usage model testing" which can be used in both software and hardware integration testing. The basis behind this type of integration testing is to run user-like workloads in integrated user-like environments. In doing the testing in this manner, the environment is proofed, while the individual components are proofed indirectly through their use. Usage Model testing takes an optimistic approach to testing, because it expects to have few problems with the individual components. The strategy relies heavily on the component developers to do the isolated unit testing for their product. The goal of the strategy is to avoid redoing the testing done by the developers, and instead flesh-out problems caused by the interaction of the components in the environment. For integration testing, Usage Model testing can be more efficient and provides better test coverage than traditional focused functional integration testing. To be more efficient and accurate, care must be used in defining the user-like workloads for creating realistic scenarios in exercising the environment. This gives confidence that the integrated environment will work as expected for the target customers.

5.2.1.2 Top-down And Bottom-up

Bottom-up testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested.All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage.Top-down testing is an approach to integrated testing where the top integrated modules are tested and the branch of the module is tested step by step until the end of the related module.Sandwich testing is an approach to combine top down testing with bottom up testing.

5.3 SOFTWARE VERIFICATION AND VALIDATION

5.3.1 Introduction

In software project management, software testing, and software engineering, verification and validation (V&V) is the process of checking that a software system meets specifications and that it fulfills its intended purpose. It may also be referred to as software quality control. It is normally the responsibility of software testers as part of the software development lifecycle. Validation checks that the product design satisfies or fits the intended use (high-level checking), i.e., the software meets the user requirements.This is done through dynamic testing and other forms of review.Verification and validation are not the same thing, although they are often confused. Boehm succinctly expressed the difference between

Validation : Are we building the right product?
Verification : Are we building the product right?
According to the Capability Maturity Model (CMMI-SW v1.1)

Software Verification: The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.

Software Validation: The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements.

M&S Verification is the process of determining that a • computer model, simulation, or federation of models and simulations implementations and their associated data accurately represent the developer's conceptual description and specifications.
M&S Validation is the process of determining the degree to which a model, simulation, or federation of models and simulations, and their associated data are accurate representations of the real world from the perspective of the intended use(s).

5.3.2 Classification of Methods

In mission-critical software systems, where flawless performance is absolutely necessary, formal methods may be used to ensure the correct operation of a system. However, often for non-mission-critical software systems, formal methods prove to be very costly and an alternative method of software V&V must be sought out. In such cases, syntactic methods are often used.

5.3.3 Test Cases

A test case is a tool used in the process. Test cases may be prepared for software verification and software validation to determine if the product was built according to the requirements of the user. Other methods, such as reviews, may be used early in the life cycle to provide for software validation.

5.3.3.1 Test Suit

Admin login test:

Test Case	Test Data	Test Result	Test Report
Blank Username	Username	Invalid	Fill required detail
Invalid Username	Username: ADMIN	Invalid	Username Incorrect
Invalid Password	Password: user	Invalid	Password Incorrect
Valid Username and Password	Username: admin Password: admin	Valid	Login

Table 5.3.3.1.1 Admin Login Test

5.4 Black-Box Testing

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher level testing, but can also dominate unit testing as well

5.4.1 Test Procedures

Specific knowledge of the application's code/internal structure and programming knowledge in general is not required. The tester is aware of what the software is supposed to do but is not aware of how it does it. For instance, the tester is aware that a particular input returns a certain, invariable output but is not aware of how the software produces the output in the first place.

5.4.2 Test Cases

Test cases are built around specifications and requirements, i.e., what the application is supposed to do. Test cases are generally derived from external descriptions of the software, including specifications, requirements and design parameters. Although the tests used are primarily functional in nature, non-functional tests may also be used. The test designer selects both valid and invalid inputs and determines the correct output, often with the help of an oracle or a previous result that is known to be good, without any knowledge of the test object's internal structure.

5.5 White-Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT). White-box testing can be applied at the unit, integration and system levels of the software testing process. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

5.5.1 Levels

1) Unit testing : White-box testing is done during unit testing to ensure that the code is working as intended, before any integration happens with previously tested code. White-box testing during unit testing catches any defects early on and aids in any defects that happen later on after the code is integrated with the rest of the application and therefore prevents any type of errors later on.

2) Integration testing : White-box testing at this level are written to test the interactions of each interface with each other. The Unit level testing made sure that each code was tested and working accordingly in an isolated environment and integration examines the correctness of the behaviorin an open environment through the use of white-box testing for any interactions of interfaces that are known to the programmer.

3) Regression testing : White-box testing during regression testing is the use of recycled white-box test cases at the unit and integration testing levels.

5.5.2 Procedures

White-box testing's basic procedures involves the tester having a deep level of understanding of the source code being tested. The programmer must have a deep understanding of the application to know what kinds of test cases to create so that every visible path is exercised for testing. Once the source code

is understood then the source code can be analyzed for test cases to be created. These are the three basic steps that white-box testing takes in order to create test cases:

Processing involves performing risk analysis to guide whole testing process, proper test plan, execute test cases and communicate results. This is the phase of building test cases to make sure they thoroughly test the application the given results are recorded accordingly.

Output involves preparing final report that encompasses all of the above preparations and results.

5.5.3 Advantages

White-box testing is one of the two biggest testing methodologies used today. It has several major advantages:

Side effects of having the knowledge of the source code is beneficial to thorough testing.

Optimization f code by revealing hidden errors and being able to remove these possible defects. Gives the programmer introspection because developers carefully describe any new implementation.

5.5.4 Disadvantages

Although white-box testing has great advantages, it is not perfect and contains some disadvantages:

White-box testing brings complexity to testing because the tester must have knowledge of the program, including being a programmer. White-box testing requires a programmer with a high level of knowledge due to the complexity of the level of testing that needs to be done.

5.6 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

CHAPTER 6

LIMITATIONS

Software development is never an ending process and continues the life of the software as per the changing needs of the user from time to time. The project is no doubt has been developed keeping in mind easy modification and enhancement that may be required from time to time.

However, there are many scopes to modify this software. As because due to a shortage of time, we here become unable to include many things. We are trying to cover all their existing system for sales return records of the items but due to a shortage of time, we become unable to include many things. Due to the lack of time, I here include none of them and a future scope one can develop these returns which are so much essential. Only with a little more doing it is possible to design the formats for those returns. Moreover, an online system will be more helpful to the organization. . With almost the same data with only a little modification, an online system can be designed to fulfill their demands. All these can be considered to be future scope for this project.

CHAPTER 7

CONCLUSION

ELECTROPEDIA – Electronic product Website which includes home page, log in, sign up, Know your requirement page, contact us, etc. It is helpful for people having a busy lifestyle and users not working without electricity. The website helps buy a product as per customer requirement. Customer does not require to make any physical activity, by just using Electropedia website he/she can know his/her requirement by just sitting at home and make life easy.

CHAPTER 8

REFERENCES

https://docs.soliditylang.org/en/v0.8.12/

https://vitejs.dev/