MEDASSISTANCE ERP

**PROJECT REPORT**

**Group No. 67**

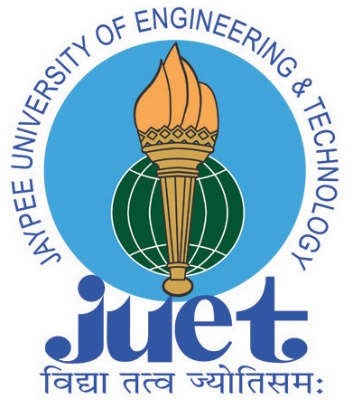
***Submitted by:***

## RICHESH GUPTA (181B165)

## ROHIT SINGH (181B172)

## SATYAM UPADHYAY (181B186)

**Under the guidance of Mr. Navaljeet Singh**

****

NOVEMBER 2019

***Submitted in partial fulfilment for the award of the degree of***

# BACHELORS OF ENGINEERING IN

**COMPUTER SCIENCE ENGINEERING**

**Department of Computer Science & Engineering JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY**

**AB ROAD, RAGHOGARH, DT. GUNA-473226 MP, INDIA**

**DECLARATION**

We hereby declare that the work reported in B. Tech. 5th semester project entitled “MedAssistance ERP”, in partial fulfilment for the award of the degree of B.Tech. submitted at Jaypee University of Engineering and Technology, Guna, as per the best of my knowledge and belief there is no infringement of intellectual property rights and copyright. In case of any violation, we will solely be responsible.

Richesh Gupta(181B165)

Rohit Singh(181B172)

Satyam Upadhyay(181B186)

November 21, 2020

Department of Computer Science and Engineering, Jaypee University of Engineering and Technology,

Raghogarh, Guna – 473226

**JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY**

**(Accredited with Grade-A by NAAC & Approved U/S 2(f) of the UGC Act, 1956)**

**A.B. ROAD, RAGHOGARH, DIST: GUNA (M.P.) INDIA Phone: 07544 267051, 267310-14, Fax: 07544 267011 Website:**

[**www.juet.ac.in**](http://www.juet.ac.in/)

# CERTIFICATE

This is to certify that the work titled “**MedAssistance ERP**” submitted by “Richesh Gupta, Rohit Singh, Satyam Upadhyay” in partial fulfilment for the award of the degree of **B.Tech** of Jaypee University of Engineering & Technology, Guna has been carried out under my supervision. As per the best of my knowledge and belief, there is no infringement of intellectual property rights and copyright. Also, this work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma. In case of any violation concern, students will solely be responsible.

Mr. Navaljeet Singh

Assistant Professor

November 21, 2020

# ACKNOWLEDGMENT

We, “Richesh Gupta (181B165)”,” Rohit Singh (181B172)”,”Satyam Upadhyay (181B186)”, would like to acknowledge the following faculties for their invaluable time and support in the development of this project:

Mr. Navaljeet Singh, mentor, without whose help and support throughout, this project would not have been possible.

Utkarsh Sharma, Project Coordinator, for providing us with all the information and material needed for the project.

Shishir Kumar, HOD(CSE), for his precious guidance and helping us with all the difficulties faced.

Richesh Gupta (181B165) Rohit Singh (181B172) Satyam Upadhyay (181B186)

**TABLE OF CONTENTS**

**Sr. No Contents Page No**

1. [Introduction 1](#_TOC_250010)
2. Objective and Goal 2
3. Advantages of ERP [4](#_TOC_250008)
4. Inspiration and Need [5](#_TOC_250007)
5. [Our Proposal 6](#_TOC_250006)
6. [Software model used 7](#_TOC_250005)
7. [Tools and Frameworks 12](#_TOC_250004)
8. [Design Diagrams 24](#_TOC_250003)
9. [Implementation and Testing 31](#_TOC_250002)
10. [Conclusion](#_TOC_250001)
11. [References 37](#_TOC_250000)

# INTRODUCTION

Enterprise Resource Planning (ERP) is a type of solution that helps businesses to manage and regulate their daily task and helps to streamline the processes which are complex to track and manage.

"MEDASSISTANCE ERP” is a website-based project. This project would provide Enterprise Resource Planning solutions to pharmacies that cannot afford costly ERPs for managing all their resources.

While going through various problem statements for our minor project we observed that medical industry had few dependencies which can be easily removed.

Which would then help benefit the consumers.

We chose to develop an open-source ERP solution such that it provides features of a modern-day ERP solutions also people have freedom to add/edit the features already present.

We decided to make this ERP free in monetary and distributive sense so that most retailers can benefit from it.

Our target audience is small cap. Retailers who feel an ERP solution is hard burden on them and feel current ERP solutions are very technical per se.

We tried developing solution which is easy to use and can run on legacy computers also.

After looking at various options available to develop the solution we chose “Django” as framework as it is robust, secure and highly integrable. Django is best for our case.

# OBJECTIVE

“To design an ERP solution which helps retailers to manage their resources.”

# 3.GOAL STATEMENT

“We do not answer questions because we want to get points or because we have nothing else to do. We’re answering questions because we want to build our reputation or we genuinely, intrinsically enjoy helping people. It’s the same reason someone might want to make a website with information. We just wish to make that a lot easier.”

By above-mentioned objective and goal, we wish to convey the message that we the developers plan to make something which would truly help people to contribute to the society and live out the true meaning of being a social animal which is to help and learn to grow to make the world a better place. Where everything is so streamlined that people in anguish could just turn to us and find relief.

# 4.ADVANTAGES OF ERP

* Integration among different functional areas to ensure proper communication, productivity and efficiency.
* Revenue period, from invoice to receipt of cash.
* Managing the interdependencies of dynamic processes Products billing.
* Tracking of the three-way match between purchase orders (what was ordered), Stock, and Cost (what the vendor invoiced).
* The Bookkeeping for all the responsibility: Revenue monitoring Expense and revenue at a crude level.
* Solves the possibility of multiple systems synchronizing differences.
* ERP Systems integrate the data in single site.
* Our ERP product is offline so that it can be used in remote locations.
* The product has Auto Backup so that no data is lost.
* Bill is sent directly to the customer's email, so there will never be a problem losing the bill.
* Reminders for the expiry of the batch of the commodity, the replenishment of the stock and the checks shall be given to the retaliator so that the retaliator may prepare accordingly.
* GST assistance module is offered to make it easier for the retailer to pay taxes.
* The cheque module in our product is used to store all cheques.
* Limited access can be provided to an employee in a store.

# 5.INSPIRATION AND NEED

When we started studying healthcare system in our country, we analysed that there is no solution for small to small-mid cap. Retailers which was depressing and was causing major issue in some areas of India, Medicines are sometimes provided with subsidy but if operating cost is high then automatically it affects the nett cost of the medicine.

Therefore, we thought of creating a solution specially targeting small to small- mid cap retailers which would concentrate on reducing their operating cost and would help the retailers.

# 6. MAJOR ERP ANALYSIS

# Major ERP solutions have steep learning curve.

# Not getting update with time.

# Very expensive in operating cost and getting support.

# Some operations like managing accounts and cheques are still manual.

# Major ERP solutions still don’t have support for GSTR.

# No module to analyze data.

# Some of ERP solutions are online which require internet connection at all times.

# With retailers, raw bills are in trend. Which results in Tax evasion.

# No support for managing staff.

# ERPs Cannot be modified as per the customer.

# Expensive plan if you want to scale up.

# Some ERP solutions requires powerful hardware.

# No community to help customers free of cost.

# Due to solutions being proprietary, no distribution is allowed.

# 7.OUR PROPOSAL

* Our proposal is to create an inexpensive ERP solution which also saves money of medicine retailers.
* The basic structure for the start would be similar to regular ERP solutions
* The project will have multiple modules which can be edited as per the requirements of individual retailers.
* This project will be free in both monetary and distributive sense.
* Scaling of project will be easy.
* The project will be offline.
* We also will add Mechanism to add, edit and delete users in the project and would also limit their access according to their credentials.
* Code will be very modular which would make customization very easy.
* Automation will be provided to help in tedious and technical tasks can be done easily. For example - GSTR generation and taking backup.
* Instant E-bills which still is not included in many ERP solutions.
* Providing insights about sale and profits.
* Fast processing of data.
* Security and Ease of use is our priority.

# 8.SOFTWARE MODEL USED

# After analysing the needs of our project and the deadline to complete the project we decided to go with the Agile development model which gives us the flexibility to develop and test our project without being highly dependent on the modules which are part of other developers’ code cycle.

# We figured that this will not only give us flexibility to write codes, test and manage but also will help us in integration of the project too.

# Agile is efficient and has approach called continuous development and testing throughout the project cycle which supports even late execution of ideas and its integration.

# As we were also loose on our requirement side of the project because of inexperience we thought that this method would yield maximum result rather than other methodologies within given time frame.

# 

# [11] Agile Model Diagram

# Agile development is used wherever quick development is required, in our case it was.

# In Agile there are several stages, namely –

# Scope of the project

# Requirement Analysis

# Iteration

# Support

# Retire

# 8.1 Scope of the project

# In scope of project stage, we try to analyse the project’s tentative timeline and it’s cost

# Then we try to we analyse the economic and technical feasibility.

# 8.2Requirement Analysis

# During requirement analysis we ask or judge what the end-user needs and how to make it reality.

# We perform various analysis in sense that what resources does this project needs, what tools we are going to use, which platform to use, what features should be added, how to add those features, how are we planning to make the project efficient and cost friendly.

# Risk assessment is one of the analysis too, whether the project respects the ROI or not. If the market is ready for the product or not are some of the criterions that we brainstorm on.

# 8.3 Iteration

# In this stage we actually start implementation of the drafted idea.

# In every iteration we concentrate on one feature and complete it entirely,

# We perform its unit testing, integration testing and then try to remove bug if any,

# 

# [12] Iteration diagram

# In every iteration we increment features and deploy changes of previous iteration if any, and we iterate until project is complete or customer is satisfied.

# Support

# At this stage we maintain already written modules, we try to rectify bug reports and

# Listen to customer feedbacks and suggestions.

# Then if required we can reiterate for some features to be added and again start the SDLC cycle.

# 8.5. Retire

# In retire phase we terminate all development operations of the project and we make plan to terminate the project.

# First, we analyse the system interactions and address each system to convert them legacy.

# Now we determine the retirement strategy and update the documentation for the same.

# After documentation we migrate users (if we want) and Archive the project.

# Remember archiving can be both public and private, whichever the leadership decides.

# Then we remove the system and revoke all access from it/ to it.

# TOOLS AND FRAMEWORKS

# Python -

# Python with dynamic semantics is an interpreted, object-oriented, high-level programming language. Combined with dynamic typing and dynamic linking, its high-level built in data structures make it very attractive for Rapid Application Creation, as well as for use as a scripting or glue language to link existing components together Python's quick, easy to learn syntax emphasizes readability and therefore reduces the cost of software maintenance. Modules and packages are provided by Python, which facilitates software modularity and reuse of code. For all major platforms, the Python interpreter and the comprehensive standard library are accessible in source or binary form without charge and can be freely distributed.

# Python was created in the late 1980s, and first released in 1991, by Guido van Rossum as a successor to the ABC programming language. Python 2.0, released in 2000, introduced new features, such as list comprehensions, and a garbage collection system with reference counting, and was discontinued with version 2.7 in 2020. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible and much Python 2 code does not run unmodified on Python 3.

# Python is typed and garbage-collected dynamically. It supports different paradigms of programming, including structured (specifically, procedural), object-oriented, and functional programming. Because of its comprehensive standard library, Python is sometimes defined as a language that includes batteries.

# The broad standard library of Python, widely cited as one of its greatest strengths, offers instruments that are suitable for several tasks. Many common formats and protocols like MIME and HTTP are supported for Internet-facing applications. It contains modules for the creation of graphical user interfaces Connecting to relational databases, generating pseudorandom numbers, arithmetic, manipulating regular expressions, and checking units with arbitrary-precision decimals. Specifications cover several parts of the standard library (for instance, the implementation of the Web Server Gateway Interface (WSGI) meets PEP 333), but most modules are not. They are specified by their code, internal documentation, and test suites. However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

# the Python Package Index (PyPI), the official repository for third-party Python software, contains over 200,000packages with a wide range of functionality, including:

# • Automation

# • Data analytics

# • Databases

# • Documentation

# • Graphical user interfaces

# • Image processing

# • Machine learning

# • Mobile App

# • Multimedia

# • Networking

# • Scientific computing

# • System administration

# • Test frameworks

# • Text processing

# • Web frameworks

# • Web scraping

# Django (Framework)

Django is a free and open-source web application framework, written in Python. A web framework is a set of components that helps you to develop websites faster and easier.

You always need a similar collection of components when you create a website: a way to handle user authentication (signing up, signing in, signing out a website management panel, forms, a way to upload files, etc.

Several well-known websites which use Django are PBS, Instagram, Disqus, Washington Times, Bitbucket and Mozilla.

Frameworks exist to save you from having to reinvent the wheel and to help alleviate some of the overhead when you’re building a new site.

The core Django framework can be seen as an MVC architecture, despite having its own nomenclature, such as naming the callable objects generating the "views" of HTTP responses.[10] It consists of an object-relational mapper (ORM) Mediating between data models (defined as Python classes) and a relational database ('Model'), a Web Templating System ('View') system for processing HTTP requests, and a regular-expression 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 signals
  + 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

The main Django distribution also bundles a number of applications in its "contrib" package, including:

* an extensible authentication system
* the dynamic administrative interface
* tools for generating RSS and Atom syndication feeds
* a "Sites" framework that allows one Django installation to run multiple websites, each with their own content and applications
* tools for generating Google Sitemaps
* built-in mitigation for cross-site request forgery, cross-site scripting, SQL injection, password cracking and other typical web attacks, most of them turned on by default
* a framework for creating GIS applications
* **Semantic UI**

Semantic UI is a thematically-designed front-end development platform. It contains pre-built semantic components that use Hu-man-friendly HTML to help build beautiful and responsive layouts.

The first pre-release appears on GitHub on September 2013, created by Jack Lukic.

* Semantic UI 1.x was first released in November 2014 with breaking changes to previous pre-releases.
* Semantic UI 2.x was first released in June 2015 and introduced new ui , several bug fixes, enhancements, and default theme improvements.

A modern front-end development platform, motivated by LESS and jQuery, is Semantic UI. It has a sleek, subtle, and flat interface look that offers a user experience that is lightweight. According to the Semantic UI website, the purpose of the framework is to empower de-signers and developers "by creating a language for sharing UI".

The framework utilizes concise HTML, intuitive JavaScript, and simplified debugging to make a front-end development a fun and delightful experience

And it integrates with React, Angular, Meteor, Ember, Django and many other frameworks to help organize UI layer alongside application logic.

Semantic UI considers terms and groups as concepts which are exchangeable. Classes use syntax from natural languages to intuitively connect concepts, such as noun/modifier relationships, word order, and plurality.

Semantic uses simple phrases called behaviours that trigger functionality.

* **JavaScript**

JavaScript is a programming language that conforms to the ECMAScript specification, also abbreviated as JS. High-level, frequently just-in-time compiled, and multi-paradigm JavaScript. It has curly-bracket syntax, dynamic typing, object-orientation based on a template, and first-class functions.

JavaScript is the Web's main client-side scripting language, with 95 percent of it used for this purpose by web pages. Scripts from HTML documents are inserted in or included and communicate with the DOM. There is a built-in JavaScript engine in all major web browsers that executes the code on the user's computer.

JavaScript is one of the World Wide Web's main technologies, alongside HTML and CSS. JavaScript allows interactive web pages and is an integral part of web applications. For client-side page actions, the vast majority of websites use it and all major web browsers have a dedicated JavaScript engine to execute it.

JavaScript embraces event-driven, functional and im-perative programming types as a multi-paradigm language. For dealing with text, dates, regular expressions, standard data structures, and the Document Object Model, it has application programming interfaces (APIs) (DOM). However the language itself does not provide any in-put/output (I/O), such as networking, storage, or graphics facilities, as the host envi-ronment (usually a web browser) provides those APIs.

Originally, JavaScript engines were only used in web browsers, but now some servers typically via Node.js, are em-bedded. They are also embedded in a range of frameworks such as Electron and Cordova generated applications.

While JavaScript and Java are similar, including the name of the language, syntax, and the respective standard libraries, the two languages are distinct and vary greatly in nature.

JavaScript and the DOM offer malicious writers the ability to deliver scripts to run through the Web on a client device. Using two limitations, browser writers mitigate this risk. Second, scripts run in a sandbox in which only Web-related activities can be done, not programming tasks like file development for general purposes. Second, scripts are limited by the same-origin policy: scripts from one site do not have access to information sent to another site, such as usernames, passwords, or cookies. Most security vulnerabilities related to JavaScript are violations of either the same policy of origin or the sandbox. There are subsets of general JavaScript-ADsafe, Secure ECMAScript (SES)-that provide higher security levels, particularly on third-party code (such as ads). Caja is another JavaScript and HTMLL third-party secure embedding and isolation project..

The majority of websites use a third-party JavaScript library or web application framework as part of their client-side page scripting.

jQuery is the most popular library, used by over 70% of websites.

There are subsets of general JavaScript-ADsafe, Secure ECMAScript (SES)-that provide higher security levels, particularly on third-party code (such as ads). Caja is another JavaScript and HTMLL third-party secure embedding and isolation project.

In contrast, the term "Vanilla JS" has been coined for websites not using any libraries or frameworks, instead relying entirely on standard JavaScript functionality.

* **HTML**

The Hypertext Markup Language (HTML) is the default document markup language that is meant to be displayed in a web browser. Technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript can assist with this.

Web browsers use a web server or local storage to accept HTML documents and turn the documents into multimedia web pages. HTML defines the layout of a web page semantically and indications for the presentation of the document were originally included.

The building blocks of HTML pages are HTML elements. With HTML constructs, the rendered page can embed images and other artifacts, such as interactive forms. By denoting structural semantics for text, such as headings, paragraphs, lists, links, quotes, and other objects, HTML provides a means to construct organized documents. HTML components, written using angle brackets, are delineated by tags.

Tags like <img /> and <input /> add information directly to the list. Other tags, such as <p>, surround and include document text information and can include other tags in the form of sub-elements. HTML tags are not viewed by browsers, but are used to interpret the content of the document.

TML can embed programs written in a scripting language such as JavaScript, which has an effect on web page behaviour and content. The look and layout of the content are specified by CSS inclusion. Since 1997, the World Wide Web Consortium (W3C), former HTML maintainer and current CSS standards maintainer, has promoted the use of CSS over explicit presentational HTML.

The World Wide Web consists primarily of HTML documents distributed via the Hypertext Transfer Protocol from web servers to web browsers (HTTP). However in addition to HTML, HTTP is used for serving images, sound, and other content. Other information along with the document is transmitted to allow the web browser to know how to manage each document it receives. The MIME type (e.g., text/html or application/ xhtml +xml) and character encoding (see Character e) are typically included in this meta data.

* **CSS**

Cascading Style Sheets (CSS) is a language for style sheets used to describe the presentation of a document written in a markup language such as HTML. In addition to HTML and JavaScript, CSS is a key technology of the World Wide Web.

CSS is designed to allow presentation and material, including layout, colors, and fonts, to be separated. This separation will enhance usability of content, provide greater flexibility and control in the presentation characteristics specification. By defining the required CSS in a separate .css file, allowing multiple web pages to share formatting, minimizing complexity and duplication in the structural text, and allowing the .css file to be cached to increase the page load speed between the pages that share the file and its formatting.

Formatting and content separation also allows it possible to display the same markup page for various rendering methods in different styles, such as on-screen, print, voice (via speech-based browser or screen reader), and on Braille-based touch devices. CSS also has alternative formatting guidelines when viewing content on a mobile device.

Cascading the name comes from the defined priority scheme to decide which style rule applies if a particular element fits more than one rule. This cascading priority scheme is predictable

The World Wide Web Consortium holds the CSS specifications (W3C). Internet Media Type (MIME type) text/css is registered under RFC 23188 for use with CSS (March 1998). A free CSS validation service for CSS documents is run by W3C.

CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties.

A style sheet consists of a list of *rules*. Each rule or rule-set consists of one or more *selectors*, and a *declaration block*.

Before CSS, nearly all presentational attributes of HTML documents were contained within the HTML markup. All font colors, background styles, element alignments, borders and sizes had to be explicitly described, often repeatedly, within the HTML. CSS lets authors move much of that information to another file, the style sheet, resulting in considerably simpler HTML.

* **Ajax**

# Ajax is a series of web development techniques that build asynchronous web applications utilizing several web technologies on the client side.

# With Ajax, web applications can send and receive data from a server asynchronously (in the background) without interfering with the display and behaviour of the current page. Ajax enables web pages and by extension, web applications to dynamically modify content without the need to refresh the entire website by decoupling the data exchange layer from the presentation layer. n practice, modern implementations commonly utilize JSON instead of XML.

# Ajax is a community of technologies, rather than a single technology. In combination, HTML and CSS can be used to mark up and style content. JavaScript will then change the webpage to dynamically view the new data and allow the user to interact with it.

# The XMLHttpRequest built-in object, or the latest "fetch()" functionality in JavaScript since 2017, is widely used to run Ajax on web pages, allowing websites to load content onto the screen without refreshing the page. Ajax isn't a modern technology, or a foreign language, but a new way of using current technologies.

# Ajax has become a large category of web technologies that can be used to incorporate a web application that interacts in the background with a server without interacting with the page's current state.  In the article that coined the term Ajax, Jesse James Garrett explained that the following technologies are incorporated:

* HTML (or XHTML) and CSS for presentation
* The Document Object Model (DOM) for dynamic display of and interaction with data
* JSON or XML for the interchange of data, and XSLT for XML manipulation
* The XMLHttpRequest object for asynchronous communication
* JavaScript to bring these technologies together
* **Selenium**

Selenium is a lightweight platform used for web application testing. Selenium offers a replay method for developing functional experiments without needing to learn the language of test scripting (Selenium IDE) A domain-specific language (Selenese) test is also provided to write tests in a variety of common programming languages, including C#, Groovy, Java, Perl, PHP, Python, Ruby, and Scala. It is then possible to run the tests against most current web browsers. Selenium runs on Windows, MacOS and Linux. The software is open source, published under the Apache License 2.0.0.

Selenium was originally created as an internal method at ThoughtWorks by Jason Huggins in 2004. Before Paul Hammant joined the team and led the development of the second mode of operation that would later become Selenium Remote Control, Huggins was later joined by other programmers and testers at ThoughtWorks (RC). That year the tool was open source.

Selenium is composed of several components with each taking on a specific role in aiding the development of web application test automation

* + Selenium IDE
  + Selenium client API
  + Selenium Remote Control
* **PostgreSQL**

We chose PostgreSQL as our database, because of its ability to manage quite many concurrent users and still manage to be fast.

PostgreSQL also supports various data-structure without which our project would have been very difficult to make.

Data-structure such as array of various datatype also manipulation of the same is quite easy to implement in PostgreSQL so we didn’t used JSON or XML which is open data-structure, quite insecure.

If we talk about PostgreSQL, it was first released in1996. It’s an opensource project which is written in C.

It’s a relational database management system, so it supports all general RDBMS features plus some exciting features too.

* **Browsers**

Browsers are needed in our project to access our portal through UI we developed.

You still can access our project through some terminal tools but with our developed you can easily do all of the task and we handle all kinds of Error efficiently so it would never crash.

Browsers are software which are used to explore the internet if we sum it up in one sentence,

They are essentially tools that execute the webpages and technologies behind it.

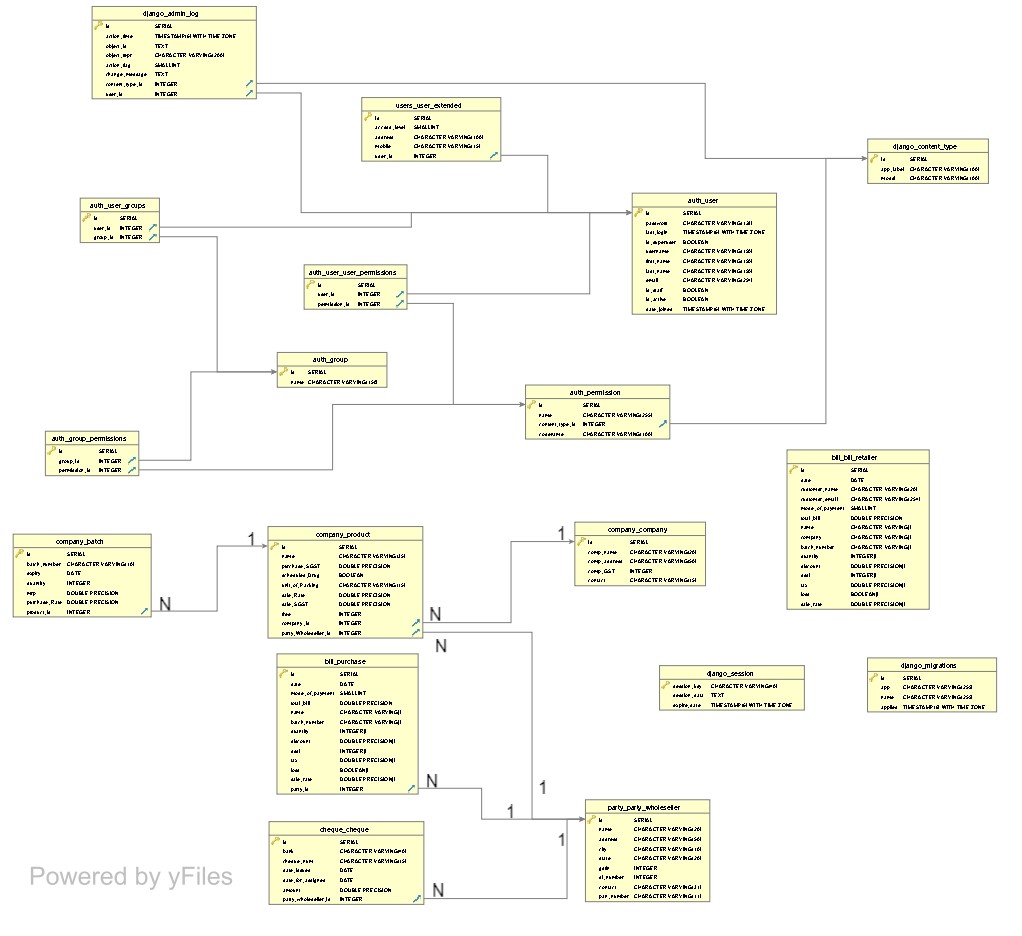
Examples of Browsers – Mozilla Firefox, Chromium,Netscape.

**10.** **DESIGN DIAGRAMS**

**1. E-R Diagram:**

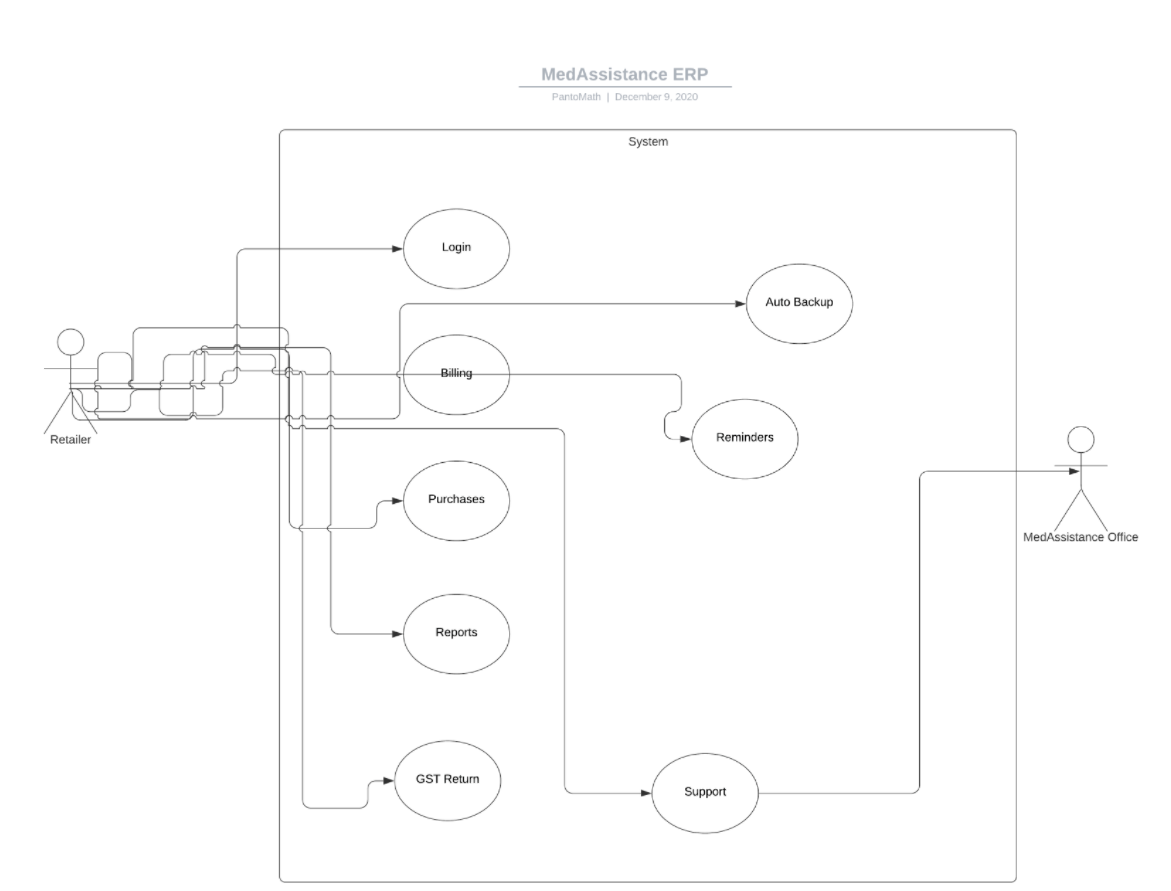
The Entity Relationship Diagram, also known as ERD, ER diagram is a type of structural diagram for use in database design. An ER Diagram contains different symbols and connectors that visualize two

important Information. The major entities within the system scope, and the interrelationships among the entities.



**2. Use Case Diagram:**

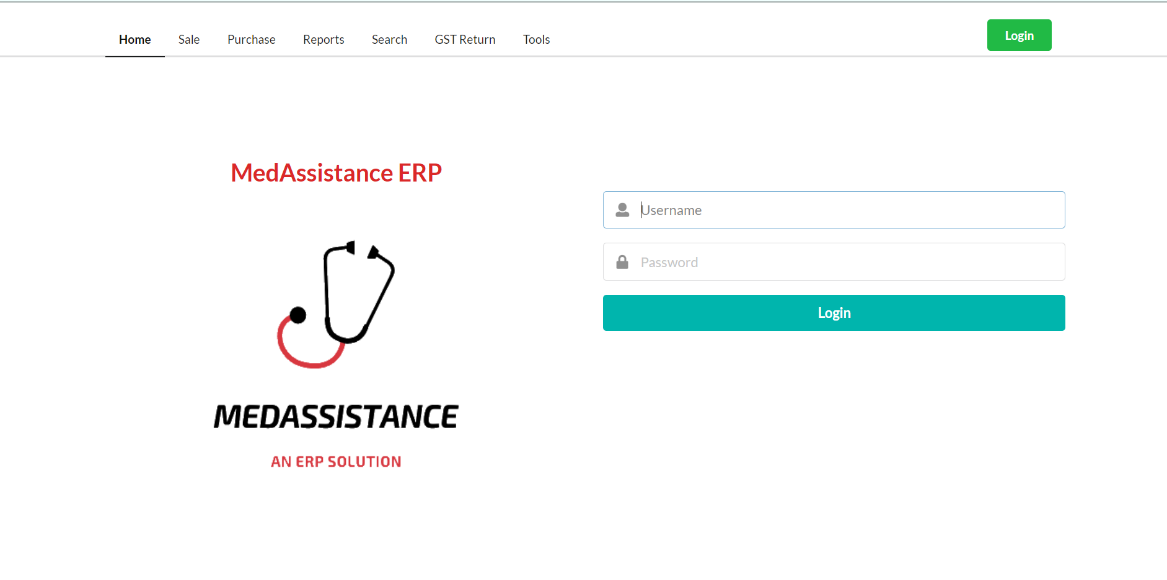
A Use Case Diagram is a graphic depiction of the interactions among the elements of a system. A Use case is a methodology used in system analysis to identify clarify and organize system requirements.

****

**11. IMPLEMENTATION AND TESTING**

**Login Page:**

This is the home page or landing page of our website. In order that only that authentic users can look for the features provided by the ERP we have implemented a compulsory login page. A login form is used to enter authentication credentials to access a restricted page or form. The page contains a field for the username and another for the password. When the login form is submitted its underlying code checks that the credentials are authentic, giving the user can access the restricted page. If a user is not able to provide authentic credentials they will not be able to proceed past the login form.

****

# 

# Manage Staff:

# After logging in the admin can manage the staff by adding, editing or deleting the staff.

# 

# Adding the Staff:

# For adding the staff, the admin needs to fill up a form where he needs to fill the basic details of the staff like First Name, Last Name, Phone Number, Username for the staff. The admin also needs to specify the access level of the staff. He can choose from basic, intermediate and admin level, each level having its own functionalities and powers.

# 

# Deleting Staff:

# 

# All the staff are listed on the delete page. The admin can delete any staff by just simply clicking the delete button under the username of the staff.

# 

# 

# Sales:

# To generate a sale’s bill the ERP provides a sale form where the user can fill all the essential details about the sale and generate a bill for both himself and the customer. The form fields include basic details of the customer and the details about the medicine that the customer is buying.

# 

# After the submission of the form a copy of the bill is downloaded on the system and one copy is send to the customer via mail which has been input in the form. The sale page also provides an option to view all the bills that have produced till now.

# Purchase:

# Similar to the sale page, there is a purchase page where the retailer can fill up the form regarding the purchases they have made from the different wholesalers. This form fields includes name of the wholesaler and the details about the products that have been purchased from them.

# 

# Reports:

# This page gives an overview for all the sales that the retailer has made over the period of time.

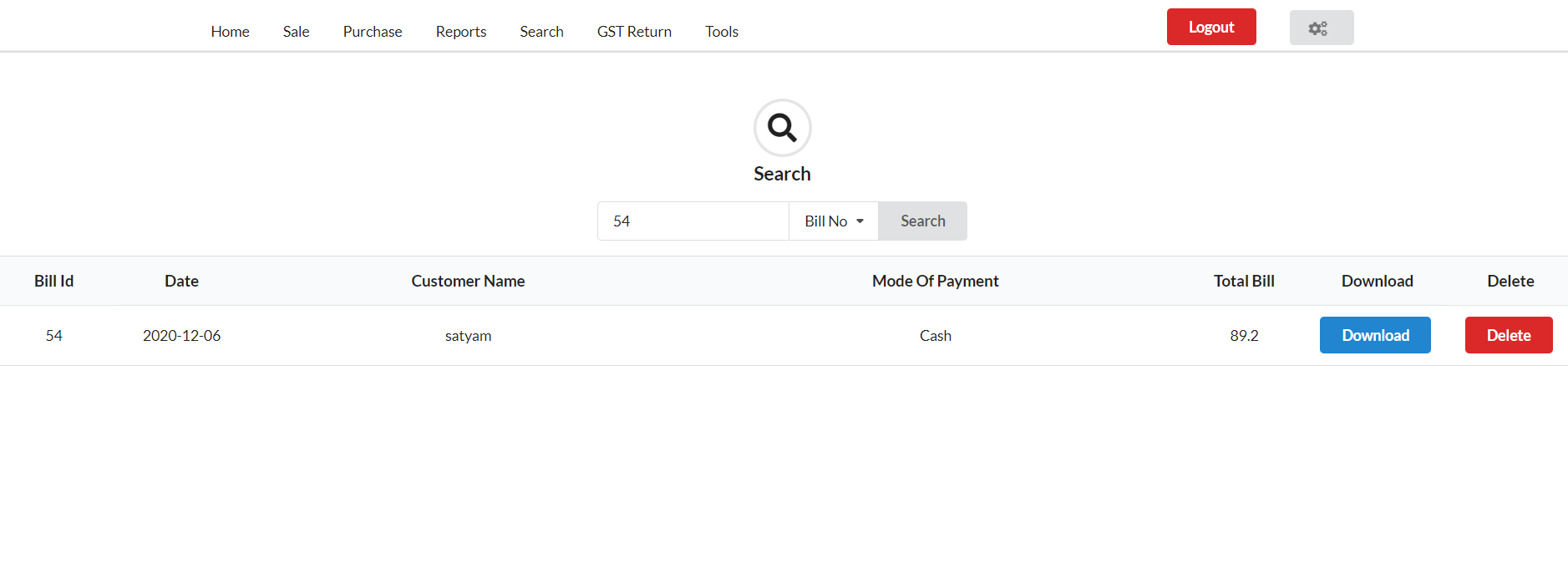
# The top section contains information about items sold and purchased, items in stock and the number of employees. The mid or chart section contains the graphical representation of the stocks sold in the past. The footer or analysis section contains facts about the sale made last year, last month and last 24 hours.

# 

# 

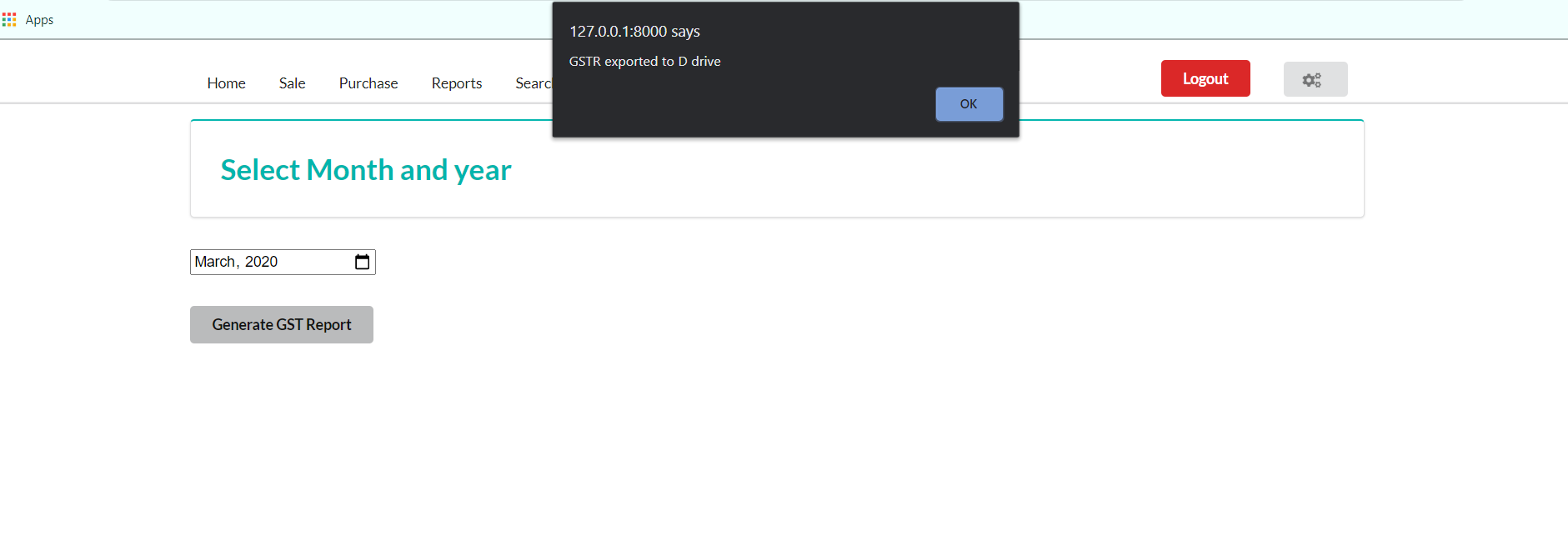
**Search:**

The search page gives the retailer an interface to search all the bills that have been produced till now. The bill can be searched from 4 entities Customer Name, Bill Number, Date of Purchase and the Amount of the bill. After displaying the details of the bill, the retailer is given the option to download the bill or to delete it from the database.



**GST Return:**

Today**,** retailers face many problems in the GST management of their shop. For this they have to high paying CA for filing their GST. This problem is solved by our ERP by providing a GST return interface where the retailer can export the information of the GST of sale and purchase made over the period of time directly to its system.

****

# Backup:

# This page appears when you try to auto backup and enter correct credentials for the same.

# Backup is generated.

# 

# Instant E-bills:

# This is the screenshot of the email received at the consumer end when a valid email is entered.

# Instant email bill reduces the paper consumption.

# 

# 12.Conclusion

# In today’s world, internet’s boon made it possible to organise even the most disorganised markets,

# Be it taxi vendors or restaurant. Even travellers are organised by some means.

# But no organisation should be able to earn a fortune just to provide the portal to organise, they should provide some concurrent value to the users.

# With this idea on our mind, we developed this project and tried to cover all the features which would provide the value to our users concurrently.

# We build a solution which scales well up to 1000 concurrent system and would still be quick like a blizzard.

# 1000 concurrent system is very high for even large cap retailers and also analyses data in real-time and generates reports automatically, with automation included our solution generates whole database backup in a blink while juggling data for analysis.

# And all of these with price Rs.0.

# We build this on open-source ecosystem with very modular codes so that even a student can customize our solution for their specific needs.

# We can further develop this project to support Wholesaler with little tweaks and we can offer our services like “Data Analysis” and “Support” for monetary gains.

# 12.REFERENCES

1. https://docs.python.org/3.7/
2. https[://ww](http://www.quora.com/)w.q[uora.com](http://www.quora.com/)
3. https://en.wikipedia.org/wiki/Web\_development
4. https[://ww](http://www.w3schools.com/)w.[w3s](http://www.w3schools.com/)c[hools.com/](http://www.w3schools.com/)
5. https[://ww](http://www.udemy.com/course/python-and-django-full-stack-web-)w.u[demy.com/course/python-and-django-full-stack-web-](http://www.udemy.com/course/python-and-django-full-stack-web-)

developer-bootcamp/learn/lecture/6598198?start=0#overview

1. https[://ww](http://www.youtube.com/channel/UCZc238wXqeN2M2uXQaY9MEw)w.y[outube.com/channel/UCZc238wXqeN2M2uXQaY9MEw](http://www.youtube.com/channel/UCZc238wXqeN2M2uXQaY9MEw)
2. https://docs.djangoproject.com/en/2.2/
3. https://getbootstrap.com
4. https://developer.mozilla.org/en-US/
5. https://stackoverflow.com

11. http://www.enterpriseunifiedprocess.com/essays/retirementPhase.html