Project Report

on

E-Farming
(Tools & Equipment)

Submitted in partial fulfillment for the award of

Post Graduate Diploma in Advance Computing

(PG-DAC) from ORLANDO ACADEMY

INDORE

Guided By

Mr. Pratik Varma Sir

Presented by:

Mr. Mayur Raut	**Prn: 230947820026**
Mr. Tushar Chaudhari	**Prn: 230947820033**
Mr. Ujjwal Patil	**Prn: 230947820036**
Mr. Ravindra Pund	**Prn: 230947820024**
Mr. Soham Waje	**Prn: 230947820032**

Centre for Development of Advanced Computing (C-DAC),Pune

CERTIFICATE

Orlando Academy (CDAC-ACT Authorized Centre)

Indore, M.P. - 452001

This is to certify that

Mr. Mayur Raut.
Mr. Tushar Chaudhari.
Mr. Ujjwal Patil.
Mr. Ravindra Pund.
Mr. Soham Waje.

have successfully completed their project on

E-Farming

under the guidance of Mr.Pratik Verma Sir.

Project Guide
Mr. Rohit Patil

Project Supervisor
Mrs. Pratibha Rajawat

Senior Director & HOD
Mr. Kunal Kansal.

ACKNOWLEDGEMENT

This project "**E-Farming**" was a great learning experience for us and we are submitting this work to Advanced Computing Training School (CDAC).
We all are very glad to mention the name of *Mr*. ***Pratik Varma*** for hisvaluable guidance to work on this project. His / Her guidance and support helped us to overcome various obstacles and intricacies during the course of project work.

We are highly grateful to Mr. Kunal Verma (Senior Director at Orlando Academy, Indore) for his guidance and support whenever necessary while doing this course Post Graduate Diploma in *Advanced Computing (PG-DAC)* through C-DAC ACTS Indore.

Our most heartfelt thanks goes to *Mr.Pratik Verma* (Course Coordinator, Orlando Academy) who gave all the required support and kind coordination to provide allthe necessities like required hardware, internet facility and extra Lab hours to complete the project and throughout the course up to the last day here in Orlando Academy Indore.

ABSTRACT

The Business to Consumer Model has come a long way ever since it time of inception. E-Farming System is developed using HTML, CSS, JavaScript, and MYSQL. E-Farming System is an e-commerce platform where farmers can simply buy and sell out farming machinery. The project has an admin side where we can keep track of all the tools and equipment details. In the administration of this system, the administrator is crucial. The administrator in this project has total access to the network. The agricultural system known as e-farming allows users to both buy and sell machinery. It assists knowledgeable farmers in becoming even more knowledgeable. As a result, partnerships are formed with both suppliers and purchasers. To guarantee high-quality products, the E-Farming System will improve communication between farmers and retailers.

Two main technologies were used in this project: java and React Js. Java was used for backend, servicing requests using REST API architecture. React JS is used for client side rendering of the page, which offloads the load of rendering views to the client, and provides a fluid single page experience. BOOTSTRAP has been used to beautify the pages and make it appealing. MySQL has been used as database to store list of users, farmers and their products

This project has been designed and implemented in multi level architecture so as to have minimum coupling and maximum cohesion.

INTRODUCTION

The E-Farming Tool and Equipment Project aim to address the evolving needs of modern agriculture by providing an online platform for farmers to access variety of farming tools and equipment. This digital initiative leverages the power of technology to optimize resource utilization, reduce upfront costs for farmers, and promote sustainable and efficient farming practices.

E-Farming is a web-based platform that connects farmers who need agricultural machinery and equipment with those who have them available for sell. The aim of this project is to provide a convenient, cost-effective, and sustainable solution for farmers to access the latest and best tools and equipment for their farming needs.

The project aims to improve the productivity and profitability of the farmers by enabling them to access the most suitable and efficient equipment for their farming operations. Also our aims are to promote innovation and modernization in the agricultural sector by introducing new technologies and practices to the farmers.

Key Features and Objectives:

1. Online E-commerce Platform:

The project establishes an easy-to-use online platform where farmers can Buy and Sell various agricultural tools and equipment. This includes tractors, plows, harvesters, irrigation systems, and other machinery essential for different farming activities.

2. Inventory Management:

The platform incorporates a robust inventory management system that tracks the availability, condition, and maintenance history of each tool or piece of equipment. This ensures that farmers have access to well-maintained and reliable machinery.

3. User Registration and Profiles:

Farmers can create user accounts and profiles on the platform, providing them with personalized dashboards. User profiles may include information about the farmer's specific needs, preferences.

HARDWARE REQUIREMENTS

- Core i7 or greater processor
- 8 GB RAM
- Hard Disk 100GB
- 1 Gbps Internet Connection
- Backup Power Supply for 24x7 working

SOFTWARE REQUIREMENTS :

BACKEND:

- Language: Java
- Database: MySQL
- Technology: SPRING BOOT REST API
- Database Management Technology: HIBERNATE

FRONTEND:

- Technology: React JS
- Beautification: CSS/BOOTSTRAP
- Template: HTML

MODULES OF FARMERS MARKET

The project is divided into:

- Registration Module
- Sign in Module
- Equipment List Module
- Edit profile
- Sell Equipment
- Buy Equipment
- Products List module
- Admin Module
- Add products Module
- Farmers Registration Module

ARCHITECTURES USED

REST API:

REST stands for **R**epresentational **S**tate **T**ransfer.

It means when a RESTful API is called, the server will *transfer* to the client a *representation* of the *state* of the requested resource.

1. **Client** — the client is the person or software who uses the API. It can be a developer, for example you, as a developer, can use Twitter API to read and write data from Twitter, create a new tweet and do more actions in a program that you write. Your program will call Twitter's API. The client can also be a web browser. When you go to Twitter website, your browser is the client who calls Twitter API and uses the returned data to render information on the screen.
2. **Resource** — a resource can be any object the API can provide information about. In Instagram's API, for example, a resource can be a user, a photo, a hashtag. Each resource has a unique identifier. The identifier can be a name or a number.

Now let's get back to REST.

A RESTful web application exposes information about itself in the form of information about its resources. It also enables the client to take actions on those resources, such as create new resources (i.e. create a new user) or change existing resources (i.e. edit a post).

In order for your APIs to be RESTful, you have to follow a set of constraints when you write them. The REST set of constraints will make your APIs easier to use and also easier to discover, meaning a developer who is just starting to use your APIs will have an easier time learning how to do so.

It means when a RESTful API is called, the server will *transfer* to the client a *representation* of the *state* of the requested resource.

For example, when a developer calls Instagram API to fetch a specific user (the resource), the API will return the state of that user, including their name, the

number of posts that user posted on Instagram so far, how many followers they have, and more.

The representation of the state can be in a JSON format, and probably for most APIs this is indeed the case. It can also be in XML or HTML format.

What the server does when you, the client, call one of its APIs depends on 2 things that you need to provide to the server:

1. An identifier for the resource you are interested in. This is the URL for the resource, also known as the **endpoint**. In fact, URL stands for Uniform Resource Locator.
2. The operation you want the server to perform on that resource, in the form of an **HTTP method**, or **verb**. The common HTTP methods are GET, POST, PUT, and DELETE.

Client-side renders refers to the rendering of browser content via JavaScript. Therefore, in place of receiving content directly from the HTML document, users receive a bare HTML document along with a JavaScript that assists with rendering the site using the browser.

This is a more recent method of rendering websites and has gained popularity after JavaScript libraries began incorporating into its development style.

This is different from server-side rendering as in this case the server is responsible only for loading a minimal version of the website. The rest of the rendering is taken care of by the JavaScript library at the client-side.

The initial request returns an HTML file. On Subsequent requests, the client side calls the corresponding API endpoints. Furthermore, you can store the data in a state and cache the data. By not fetching the data we are being friendly to mobile data users, limiting unnecessary API calls, and minimizing re-renders of applications due to state updates.

If you've used a frontend library before, you'd already know about the bundle.js file that's created when you build your application. Here's what actually happens under the hood.

1. The library generates a static bundle when you build the application.
2. The bundle.js and the index.html (along with other static assets) are downloaded to the client-side. The source code returned will have very little indexable HTML.
3. Once downloaded, the browser will render the application to you.
4. Another round of HTTP communication is triggered to fetch the actual data from an API endpoint.
5. Subsequent requests result in new data being fetched without reloading the entire page. This makes it feel like the page is blazing fast.

CONCLUSION

The package was designed in such a way that future modifications can be done easily. The following conclusions can be deduced from the development of the project:

- Automation of the entire system improves the efficiency
- It provides a friendly graphical user interface which proves to be better when compared to the existing system.
- It gives appropriate access to the authorized users depending on their permissions.
- It effectively overcomes the delay in communications.
- Updating of information becomes so easier.
- System security, data security and reliability are the striking features.
- The System has adequate scope for modification in future if it is necessary.

FUTURE SCOPE

The future scope for an E-Farming Tool and Equipment e-commerce website project is also very promising, as it can provide a platform for farmers to sell or rent their tools and equipment to other farmers or customers who need them. Some of the possible benefits and opportunities of this project are:

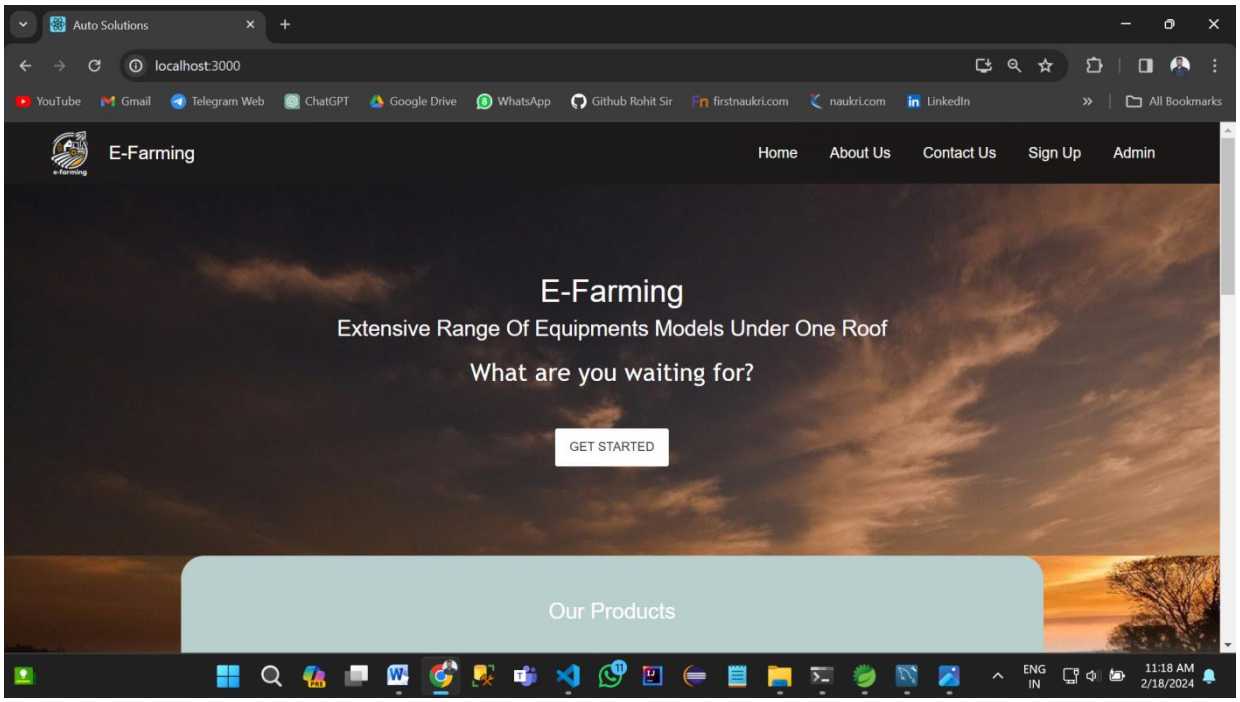
- It can help farmers generate additional income by renting or selling their idle or surplus tools and equipment, which can otherwise be costly to maintain or store.
- It can help farmers save money by renting or buying the tools and equipment they need from other farmers, instead of purchasing them from the market or dealers, which can be expensive or unavailable.
- It can help farmers access a wider range of tools and equipment, which can improve their farming efficiency and quality, as well as enable them to adopt new technologies and practices.

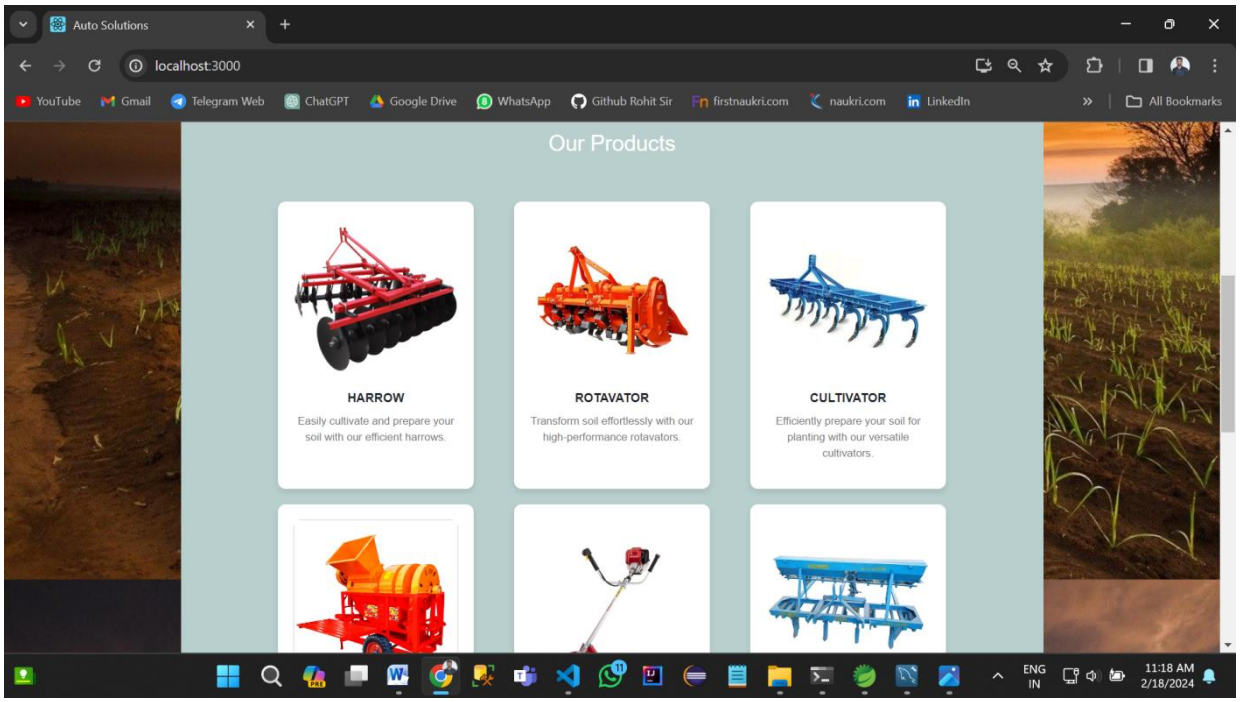
- It can help farmers reduce the environmental impact of their tools and equipment, by promoting their optimal utilization and sharing, as well as reducing the need for transportation and storage .
- It can help farmers build a network and community with other farmers, who can share their knowledge, experience, and feedback on the tools and equipment, as well as provide support and assistance.

According to some sources, e-commerce is a growing trend in the agricultural sector, and can offer various benefits for the farmers and the consumers. Therefore, an E-Farming Tool and Equipment e-commerce website project can tap into these opportunities and create a positive impact for the farmers and the society.

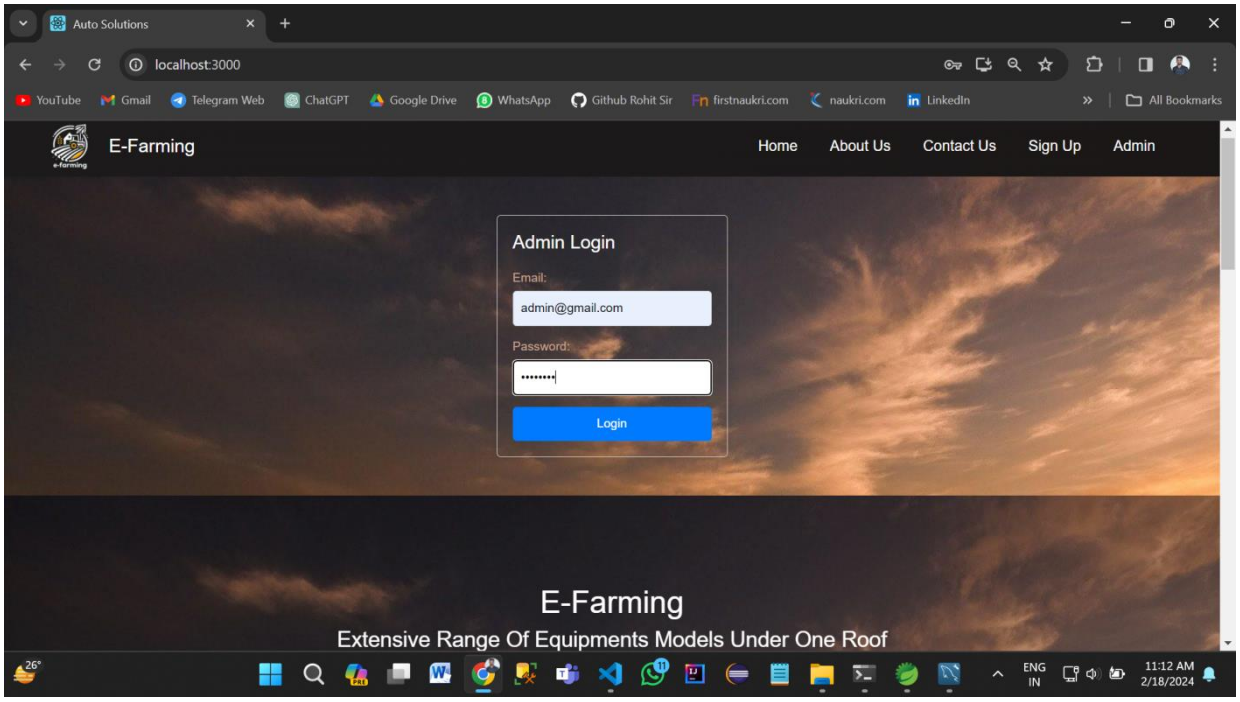
SCREENSHOTS

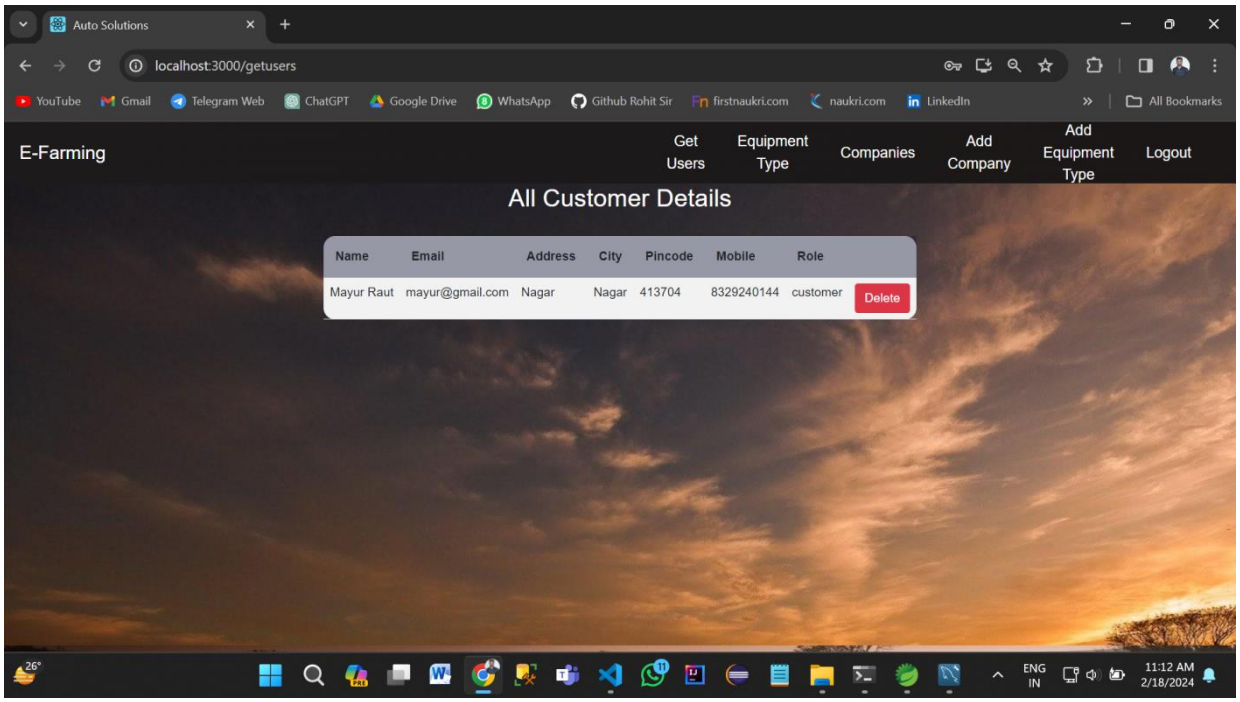
1. Home Page -

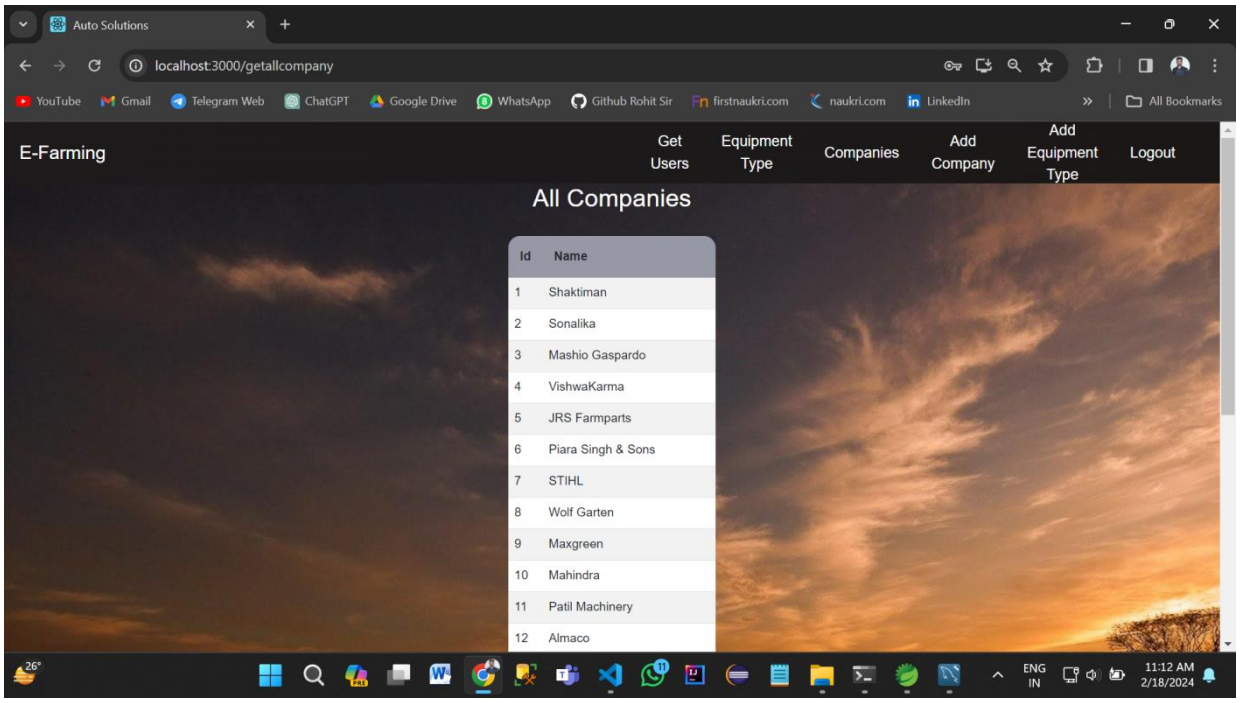


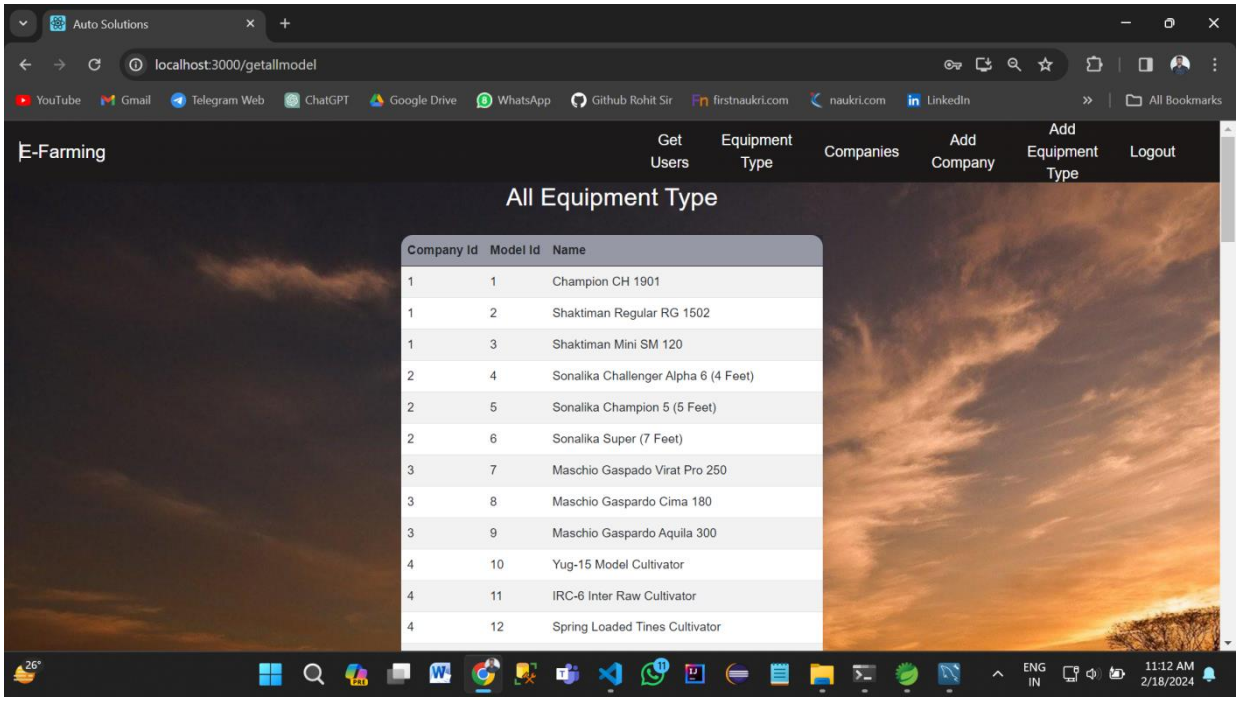


2. Admin Login -

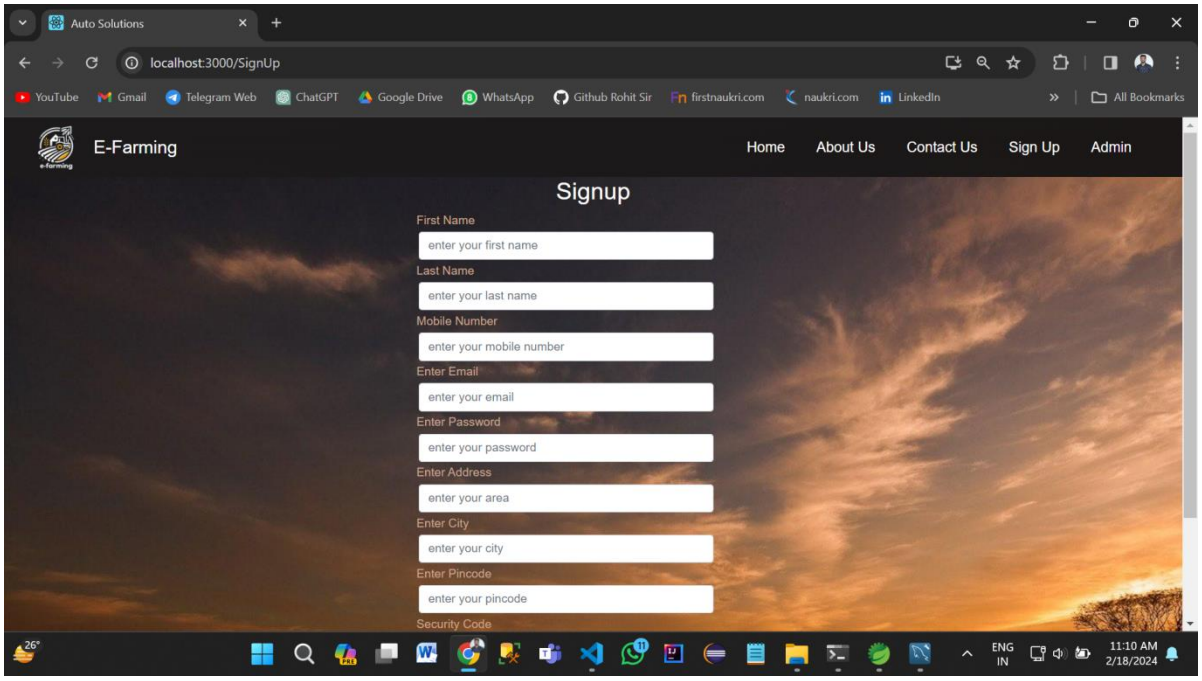




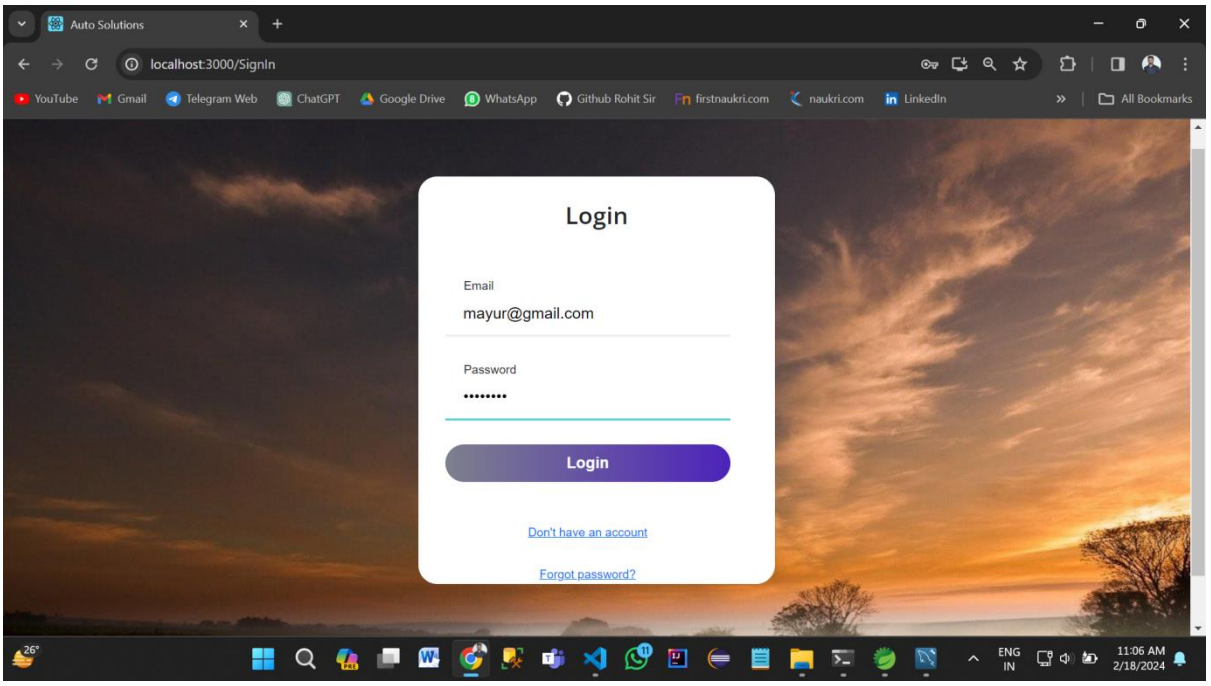

Auto Solutions
localhost:3000/getallcompany
YouTube
Gmail
Telegram Web
ChatGPT
Google Drive
WhatsApp
Github Rohit Sir
firstnaukri.com
naukri.com
LinkedIn
All Bookmarks
E-Farming
Get Users
Equipment Type
Companies
Add Company
Add Equipment Type
Logout
All Companies
Id
Name
1 Shaktiman
2 Sonalika
3 Mashio Gaspardo
4 VishwaKarma
5 JRS Farmparts
6 Piara Singh & Sons
7 STIHL
8 Wolf Garten
9 Maxgreen
10 Mahindra
11 Patil Machinery
12 Almaco
26°
ENG IN
11:12 AM
2/18/2024


Auto Solutions
localhost:3000/getallmodel
YouTube
Gmail
Telegram Web
ChatGPT
Google Drive
WhatsApp
Github Rohit Sir
firstnaukri.com
naukri.com
LinkedIn
All Bookmarks
E-Farming
Get Users
Equipment Type
Companies
Add Company
Add Equipment Type
Logout
All Equipment Type
Company Id
Model Id
Name
1 1 Champion CH 1901
1 2 Shaktiman Regular RG 1502
1 3 Shaktiman Mini SM 120
2 4 Sonalika Challenger Alpha 6 (4 Feet)
2 5 Sonalika Champion 5 (5 Feet)
2 6 Sonalika Super (7 Feet)
3 7 Maschio Gaspado Virat Pro 250
3 8 Maschio Gaspardo Cima 180
3 9 Maschio Gaspardo Aquila 300
4 10 Yug-15 Model Cultivator
4 11 IRC-6 Inter Raw Cultivator
4 12 Spring Loaded Tines Cultivator
26°
ENG IN
11:12 AM
2/18/2024

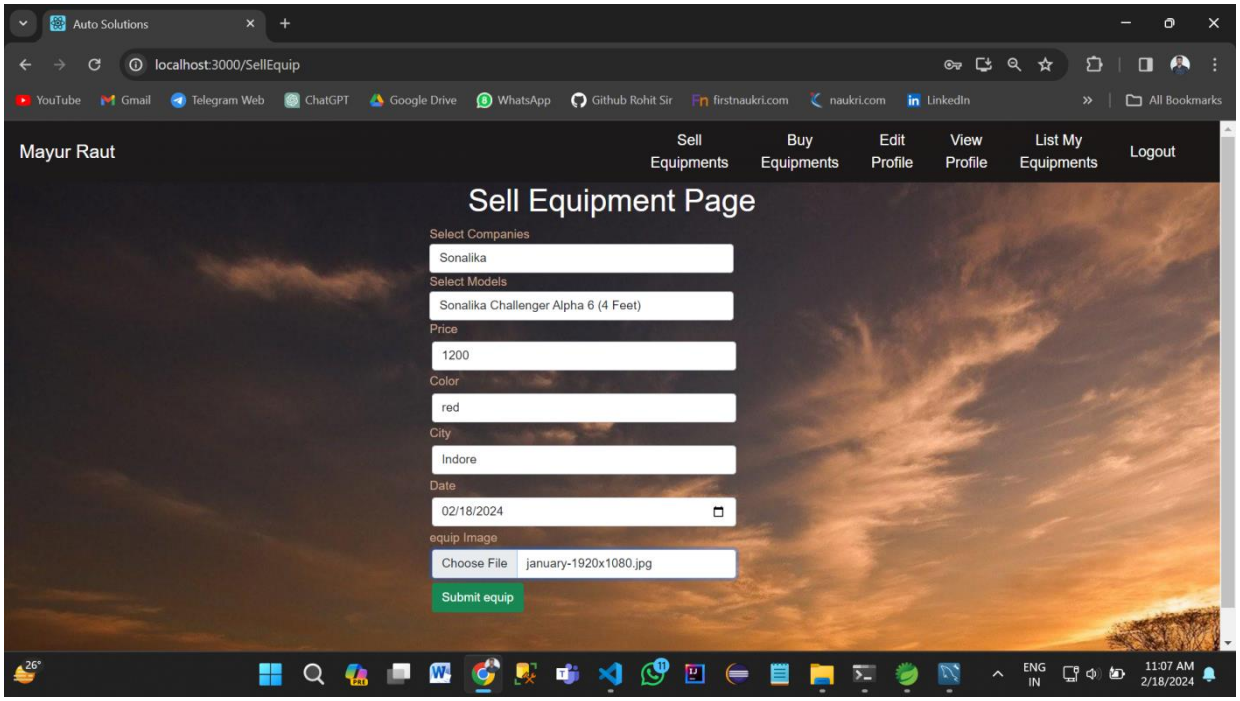
3. Sign Up Page



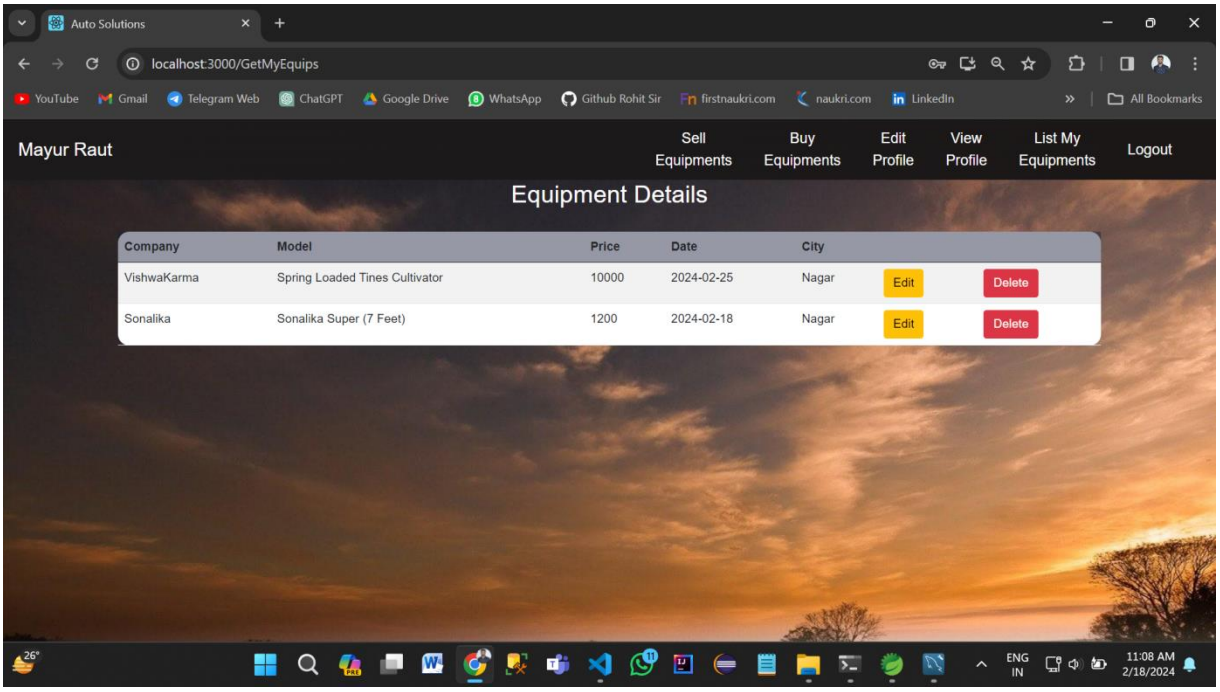
4. Login Page



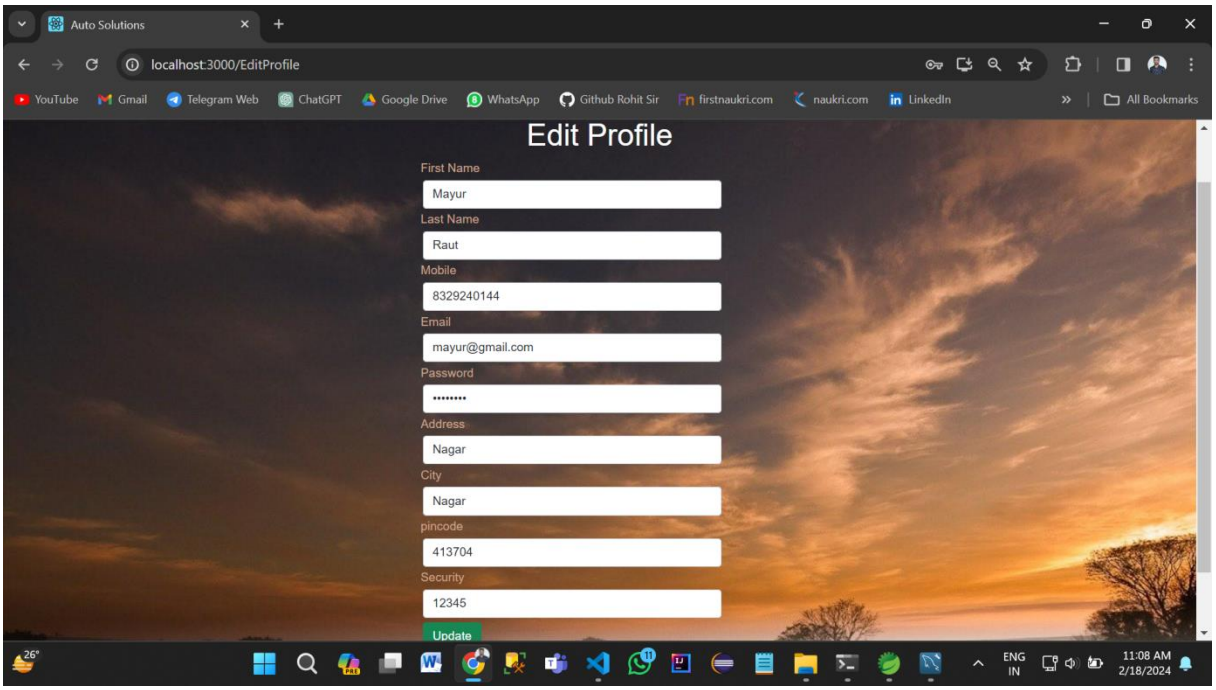
5. Buy Equipment -



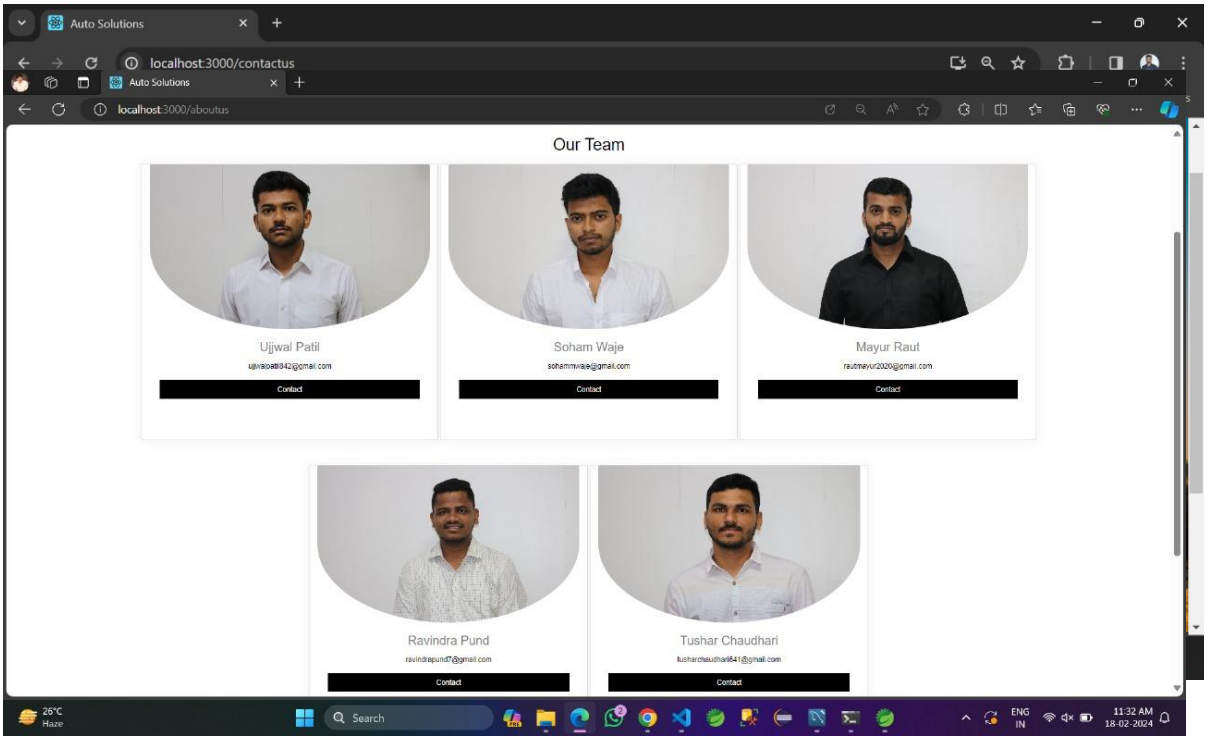
6. Product List Page



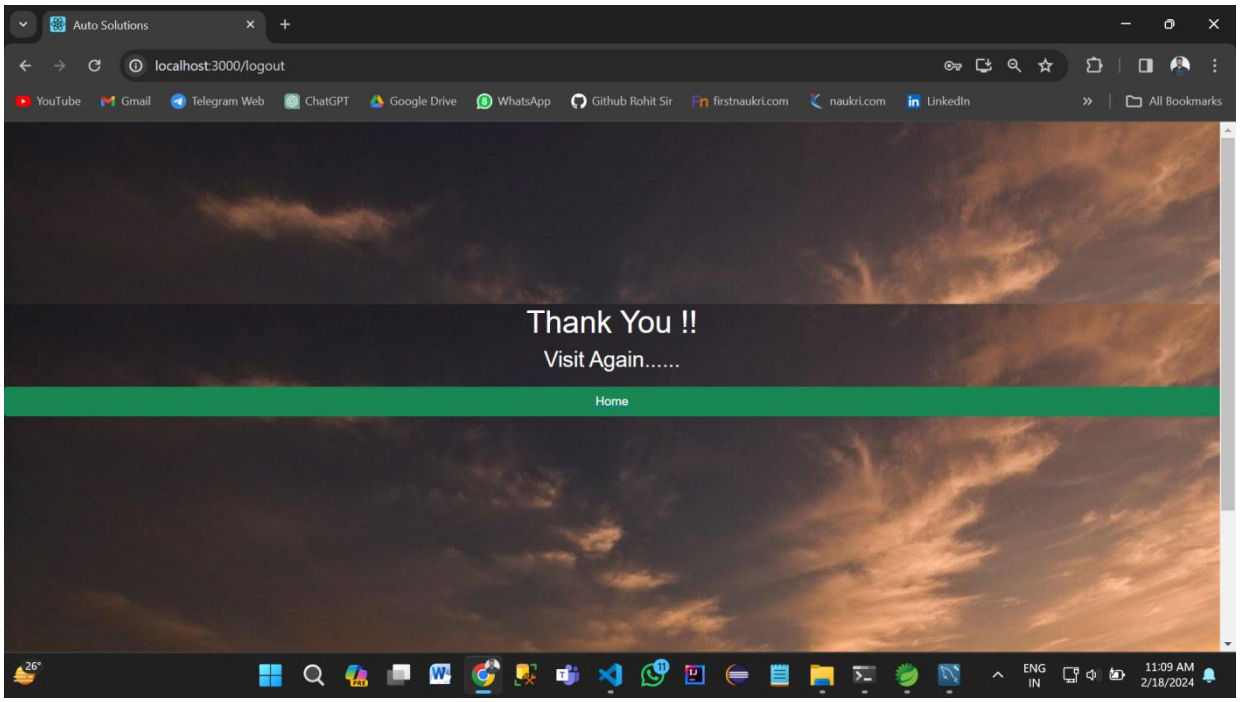
7. Edit Farmer



8. Contact Us Page



9.Log out page -



REFERENCES

- https://www.epramaan.gov.in/
- https://spring.io/
- https://react.dev/reference/react/
- https://www.javascript.com/
- https://developer.mozilla.org/en-US/docs/Web/JavaScript
- https://jquery.com/
- https://www.ieee-security.org/TC/SP2017/papers/226.pdf
- https://www.ndsssymposium.org/wpcontent/uploads/2018/02/ndss2018posters_paper_8.pdf
- https://hibernate.org/
- https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller