TRACKING SYSTEM

VALUE ADDED COURSE REPORT

Submitted by

SREEMATHI P	**111924CS01256**
SANDHIYA G	**111922CS01227**

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

S. A. ENGINEERING COLLEGE, CHENNAI 600 077

ANNA UNIVERSITY: CHENNAI 600 025

SEPTEMBER 2025

ANNA UNIVERSITY: CHENNAI 600 025 BONAFIDE CERTIFICATE

Certified that this Mini Project Report **“TRACKING SYSTEM ”** is the Bonafide work of **“SREEMATHI P(111924CS01256), SANDHIYA G(111924CS01227)”** who carried out the mini project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.R.Geetha, M.E., Ph.D.,	Mrs.J.Sangeetha, M.E.,
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Professor	Assistant Professor
Department of	Department of
Computer Science and	Computer Science and
Engineering	Engineering
S.A. ENGINEERING COLLEGE,	S.A.ENGINEERING COLLEGE,
Avadi- Poonamallee Main Road,	Avadi-Poonamallee Main Road,
Thiruverkadu Post,	Thiruverkadu Post,
Chennai-600 077	Chennai-600 077

ACKNOWLEDGEMENT

We owe a great many thanks to a great many people who helped and support us during the completion of our project.
We take this opportunity to express our profound gratitude and deep regards to our Founder

Chairman **(Late)** Shri**. D. SUDHARSSANAM, M.L.A,** our Chairman Shri**. D. DURAISWAMY**, our humble Secretary Shri**. D. DASARATHAN** our correspondent Shri.**S. AMARNAATH**, and our Director Shri**.D. SABARINAATH**, for their exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by them time to time shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to thank our beloved

Principal **Dr.S.RAMACHANDRAN, M.E., Ph.D.,** for extending his support. Also we sincerely thank **Dr. R. GEETHA, M.E., Ph.D.,** Head of the Department of Computer Science and Engineering for her guidance and encouragement in our project work which helped us in completing this task through various stages.

We also obliged to convey our special and sincere gratitude to our project coordinators

Mrs.J.SANGEETHA, M.E.,Assistant Professor, CSE Department for their encouragement and concern during the review session along with their valuable advice, knowledge and expert guidance in our project.

Lastly, we thank almighty, our parents, all the teaching and non- teaching staff for their support and guidance throughout our project and for their constant encouragement without which this project would have not been possible.

TABLE OF CONTENTS

ABSTRACT

The Tracking System project is designed to provide an efficient solution for monitoring and managing the real-time location of objects, vehicles, or individuals. The system integrates modern technologies such as GPS (Global Positioning System), GSM (Global System for Mobile Communication), and database management to ensure accurate tracking and reporting. Through this system, users can obtain live updates of location data, route history, and movement patterns, which can be accessed via a web or mobile interface.

The primary objective of the project is to enhance security, improve resource management, and optimize operational efficiency. For instance, in transportation and logistics, the system ensures timely delivery by monitoring vehicle routes; in personal applications, it enhances safety by enabling the tracking of individuals or assets. The proposed system is cost-effective, user-friendly, and scalable, making it suitable for multiple domains such as fleet management, employee monitoring, and personal asset tracking

By combining hardware components with software integration, the Tracking System offers a reliable and automated approach to real-time monitoring, thus addressing the growing demand for smart surveillance and resource management solution

OVERVIEW

The Real-Time Product Tracking Dashboard is a web-based application that allows users to track the status of products from order to delivery. Users can add products, update their status, and remove products dynamically. All product data is stored in the browser's local Storage, ensuring that the information remains even after the page is refreshed.

METHODOLOGY

1.Project Planning and Research

Study of existing tracking systems (like courier and vehicle tracking).

Identify requirements and user interaction flow.

2.Environment Setup

Tools used: Visual Studio Code, HTML5, CSS3, JavaScript, optional local server (Node/JSON file).

3.Implementation

Creation of responsive web pages using HTML and CSS.

JavaScript functions handle data retrieval, updates, and search.

Modular development for better code management.

4.Testing and Debugging

Testing each module for correct data display and responsive design.

Fixing errors related to tracking ID input or search results.

5.Documentation and Optimization

Writing clear code documentation.

Optimizing front-end design for mobile and desktop screens

2. TOOLS DESCRIPTION

2.1 Tool overview

1. HTML5 Structure of the web page.
2. CSS3 Styling and layout of the dashboard.
3. JavaScript (ES6)
4. Dynamic functionality for adding, updating, and deleting products.
5. Local Storage API Client-side storage to save tracking data persistently.

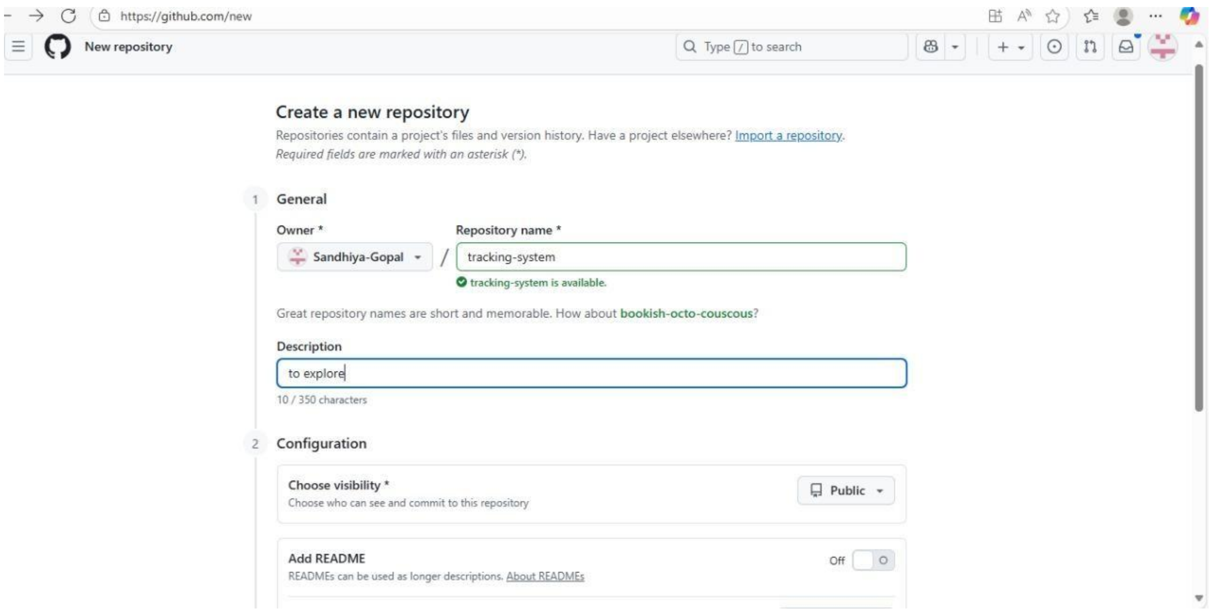
2.2 FEATURES IMPLEMENTED

1. Add Product – Enter a product name and initial status to add it to the dashboard.
2. Update Status – Change the product's tracking stage dynamically using a dropdown.
3. Delete Product – Remove products from the list when delivered or no longer needed.
4. Persistent Storage – Data is saved using local Storage, so it stays even after refreshing the browser.
5. Responsive UI – Simple and clean layout with instant updates.

2.3 PROJECT SETUP

The **Tracking System** project was completely set up and automated using **GitHub** and **Jenkins**. GitHub was used for version control and collaboration, while Jenkins handled continuous integration and deployment (CI/CD).
The following steps explain the entire setup process.

Step 1: Create GitHub Repository



Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

1 General

Owner *: Sandhiya-Gopal / Repository name *: tracking-system

tracking-system is available.

Great repository names are short and memorable. How about **bookish-octo-couscous**?

Description: to explore

10 / 350 characters

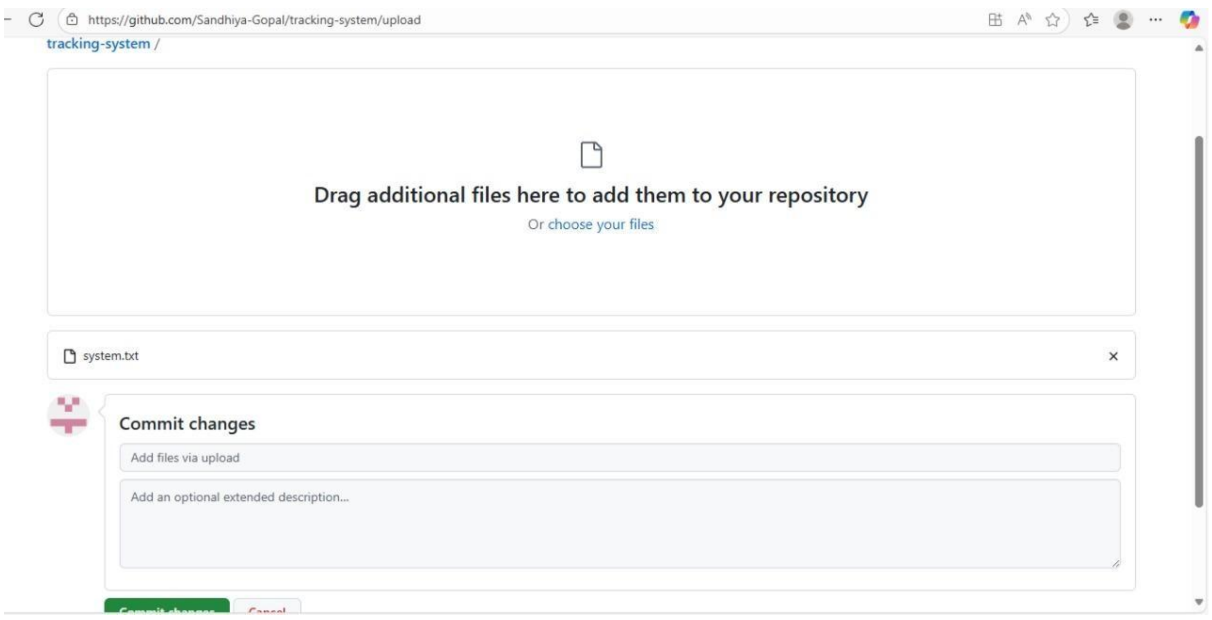
2 Configuration

Choose visibility *: Public
Choose who can see and commit to this repository

Add README: Off
READMEs can be used as longer descriptions. About READMEs

1. Login to your **GitHub** account.
2. Click on **"New Repository"**.
3. Enter repository name: tracking-system.
4. Add a short description like "A web-based Tracking System project."
5. Choose visibility as **Public** or **Private** as required.
6. Click **"Create Repository."**

Step 2: Upload Project Files to GitHub



Drag additional files here to add them to your repository

Or choose your files

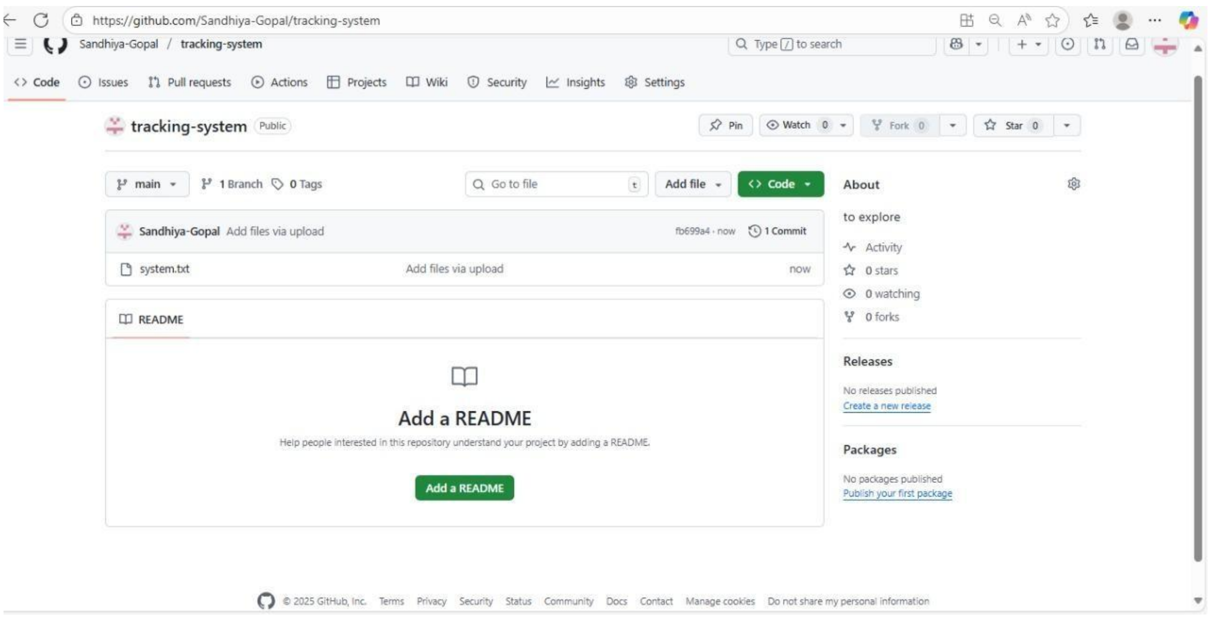
system.txt

Commit changes

Add files via upload

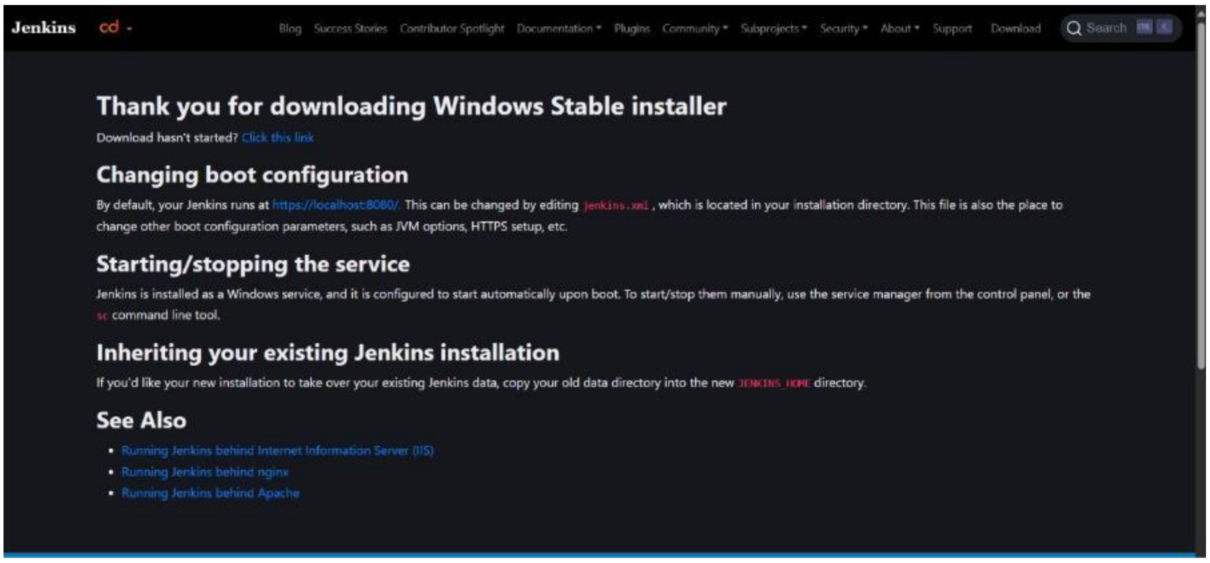
Add an optional extended description...

1. In your repository page, click “Add file → Upload files.”
2. Upload the HTML, CSS, JavaScript, and image files of the Tracking System.
3. Commit the files with a message
4. Initial project upload - Tracking System
5. Verify that all files appear correctly in your repository.


https://github.com/Sandhiya-Gopal/tracking-system
Sandhiya-Gopal / tracking-system
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
tracking-system Public
main 1 Branch 0 Tags Go to file Add file Code
Sandhiya-Gopal Add files via upload fb699a4 · now 1 Commit
system.txt Add files via upload now
README
Add a README
Help people interested in this repository understand your project by adding a README.
Add a README
About
to explore
Activity
0 stars
0 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package

Step 3: Install Jenkins

1. Download and install **Jenkins** from the official website: https://www.jenkins.io.


Jenkins
Blog Success Stories Contributor Spotlight Documentation Plugins Community Subprojects Security About Support Download Search

Thank you for downloading Windows Stable installer

Download hasn't started? Click this link

Changing boot configuration

By default, your Jenkins runs at https://localhost:8080/. This can be changed by editing jenkins.xml, which is located in your installation directory. This file is also the place to change other boot configuration parameters, such as JVM options, HTTPS setup, etc.

Starting/stopping the service

Jenkins is installed as a Windows service, and it is configured to start automatically upon boot. To start/stop them manually, use the service manager from the control panel, or the sc command line tool.

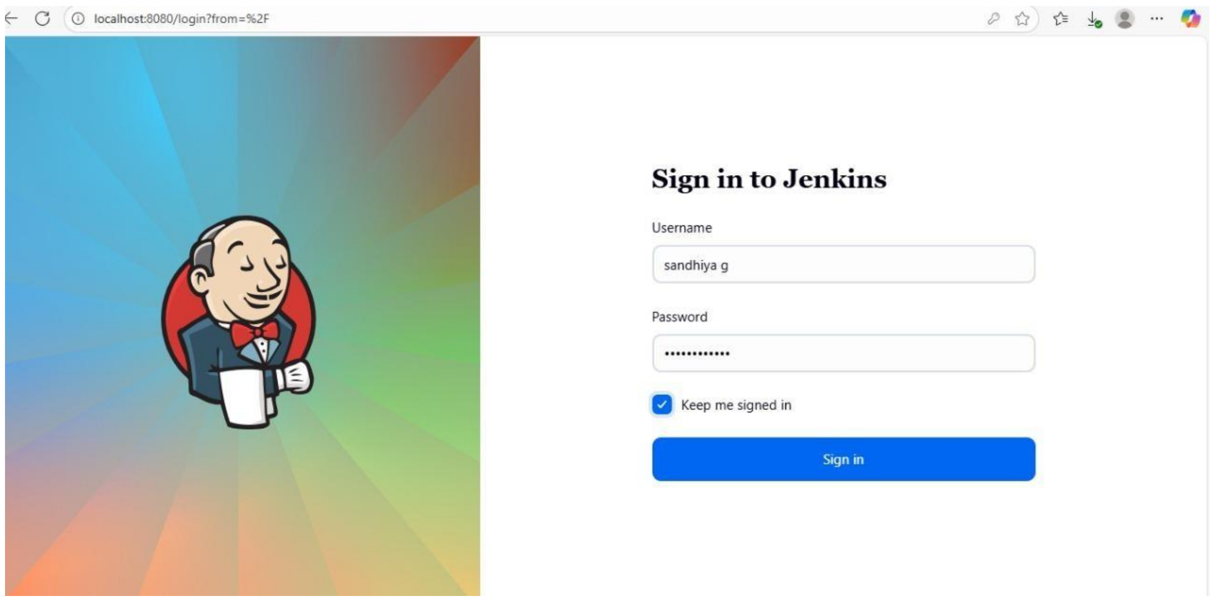
Inheriting your existing Jenkins installation

If you'd like your new installation to take over your existing Jenkins data, copy your old data directory into the new JENKINS_HOME directory.

See Also

- Running Jenkins behind Internet Information Server (IIS)
- Running Jenkins behind nginx
- Running Jenkins behind Apache

2. Launch Jenkins on your system using the default port:
3. http://localhost:8080


localhost:8080/login?from=%2F

Sign in to Jenkins

Username: sandhiya g

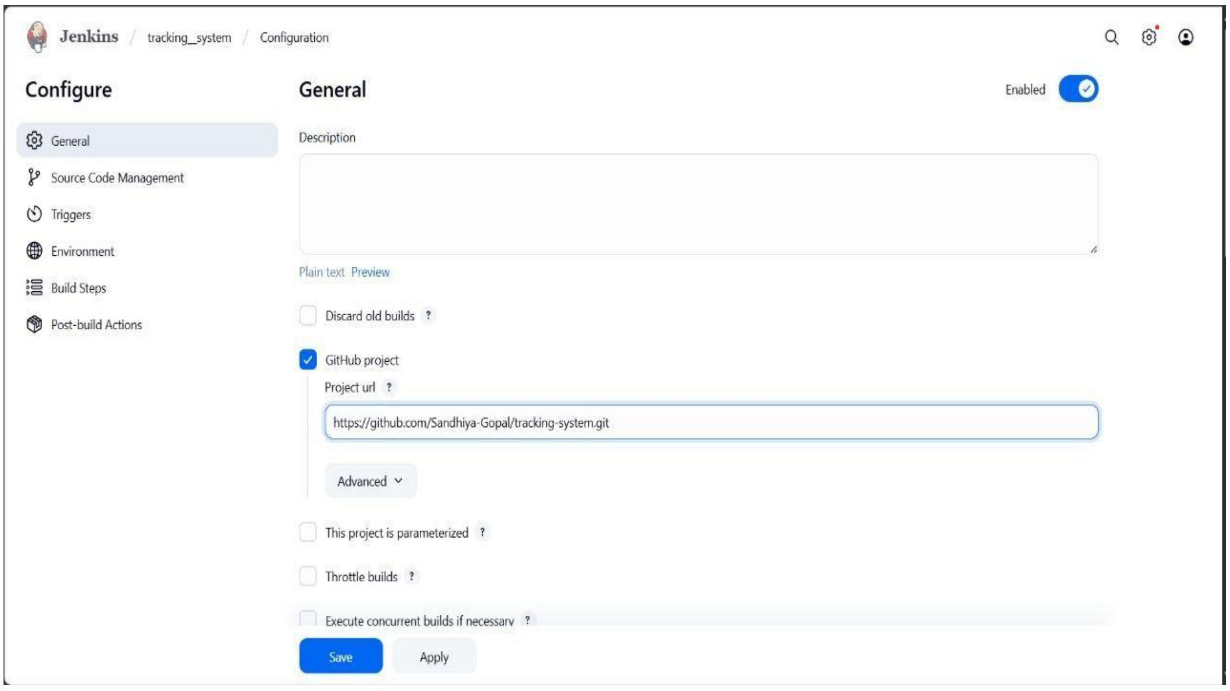
Password: ••••••••••••

✓ Keep me signed in

Sign in

4. Complete the initial setup by unlocking Jenkins with the administrator password.
5. Install the **Suggested Plugins**, which include:
   - **Git Plugin** (for connecting with GitHub)
   - **HTML Publisher Plugin** (for web reports)

Step 4: Connect Jenkins with GitHub


Jenkins / tracking_system / Configuration

Configure: General, Source Code Management, Triggers, Environment, Build Steps, Post-build Actions

General — Enabled ✓

Description: (empty)

Plain text Preview

☐ Discard old builds

✓ GitHub project

Project url: https://github.com/Sandhiya-Gopal/tracking-system.git

Advanced

☐ This project is parameterized

☐ Throttle builds

☐ Execute concurrent builds if necessary

Save Apply

1. In Jenkins, go to **“Manage Jenkins → Plugins → Available Plugins.”**
2. Ensure that **Git** and **GitHub Integration** plugins are installed.
3. Generate a **Personal Access Token** from GitHub for authentication.
   - Go to GitHub → **Settings → Developer Settings → Personal Access Token.**
   - Copy the token and save it securely.
4. In Jenkins, add your GitHub credentials:
   - Go to **“Manage Jenkins → Credentials → Global → Add Credentials.”**
   - Choose “Secret Text” and paste the GitHub token.

Step 5: Create a Jenkins Job

1. In Jenkins, click **“New Item.”**
2. Enter project name: Tracking-System-Automation.
3. Choose **Freestyle Project** and click **OK.**
4. Under **Source Code Management**, select **Git.**
5. Paste your GitHub repository URL,
6. https://github.com/username/tracking-system.git
7. Under **Build Triggers**, enable:
   - **GitHub hook trigger for GITScm polling**, or
   - **Poll SCM** (for automatic builds).

Step 6: Configure Build Section

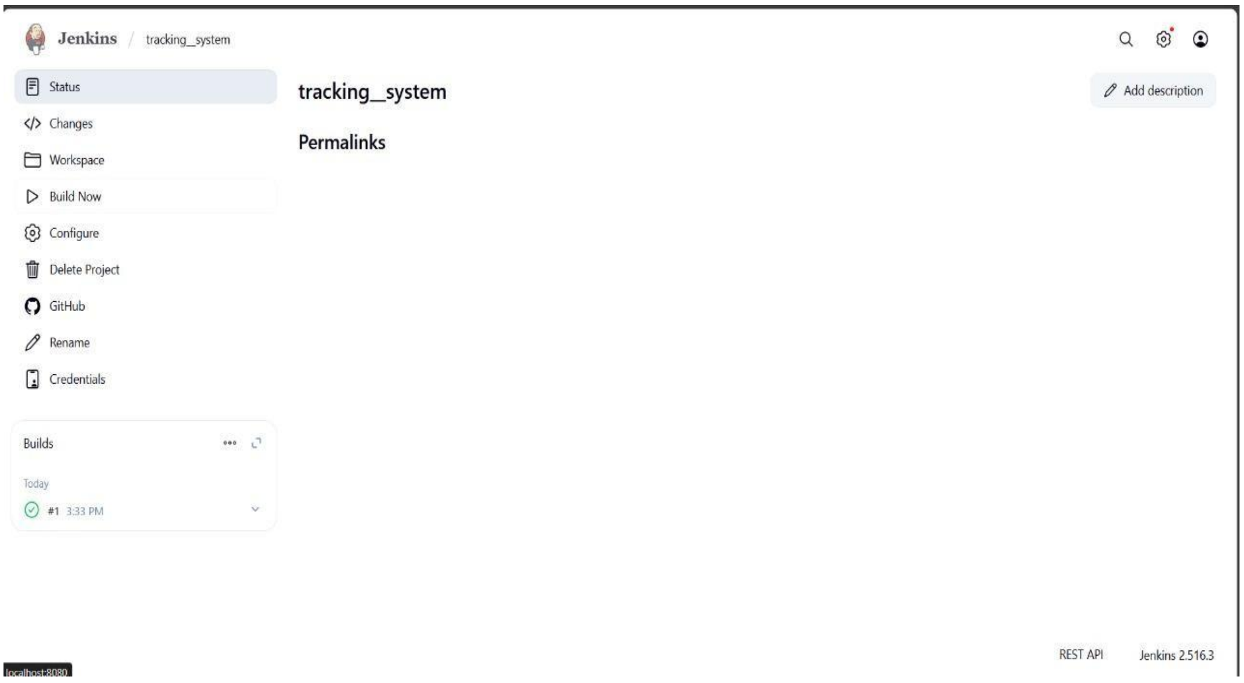
1. Scroll to the **Build** section.
2. Click **“Add build step → Execute Windows batch command”** (or shell script in Linux).
3. Enter the commands:
4. echo "Starting Tracking System Build..."
5. echo "Checking HTML, CSS, and JavaScript files..."
6. echo "Build completed successfully!"
7. Save the project configuration.

Step 7: Set Up Continuous Integration

1. Each time a change (commit or push) is made to the GitHub repository, Jenkins automatically:
   - Pulls the latest code.
   - Executes the build script.
   - Publishes the updated version for testing or deployment.
2. This ensures **automatic validation and deployment** without manual intervention.

Step 8: Verify Build and Deployment

1. Open Jenkins Dashboard.
2. Click on the job name → **Build Now.**
3. Check the console output for messages:
4. Build Successful
5. Confirm that Jenkins is successfully pulling and building the latest version from GitHub.



2.4 SYNTHESIS

2.4.1 Project Synthesis

The project successfully implements a responsive tracking interface that can search and display tracking information dynamically. The system provides a clear visual representation of item status updates.

2.4.2 Methodology Synthesis

The development process followed an **agile approach**, allowing incremental improvements in functionality, such as layout, validation, and interactivity.

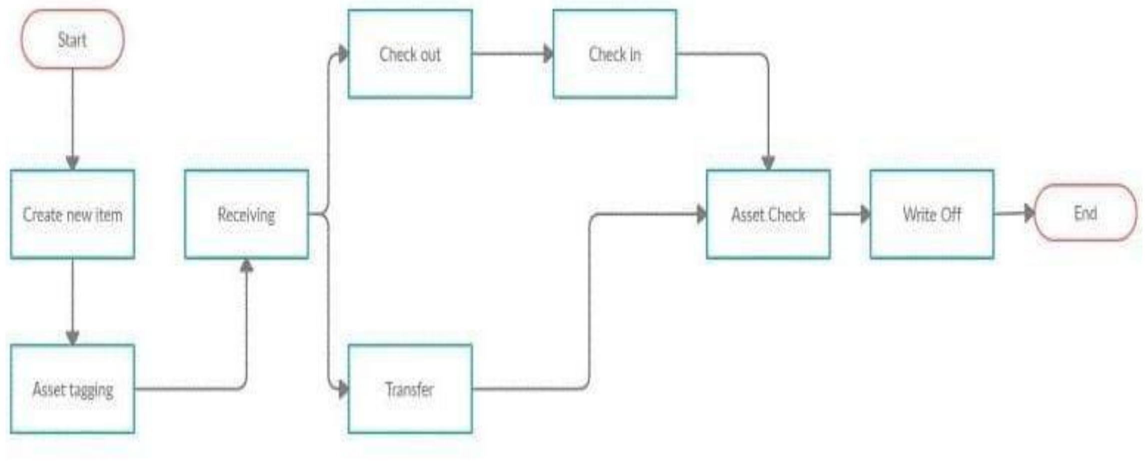
2.4.3 Outcomes and Achievements

- Fully functional tracking page with search box and live updates.
- Responsive design compatible with all devices.
- Successful integration of JavaScript for dynamic behaviour.

2.4.4 Conclusion

The Real-Time Product Tracking Dashboard is a simple yet effective web-based solution for tracking product statuses without requiring a server or database. It showcases how HTML, CSS, and JavaScript can work together to build an interactive and user-friendly application. By leveraging local Storage, the system ensures data persistence, making it ideal for small businesses, personal use, or as a learning project for beginners in web development. This project successfully demonstrates real-time updates, dynamic data handling, and efficient workflow management, providing a solid foundation for scaling into more advanced tracking systems in the future.

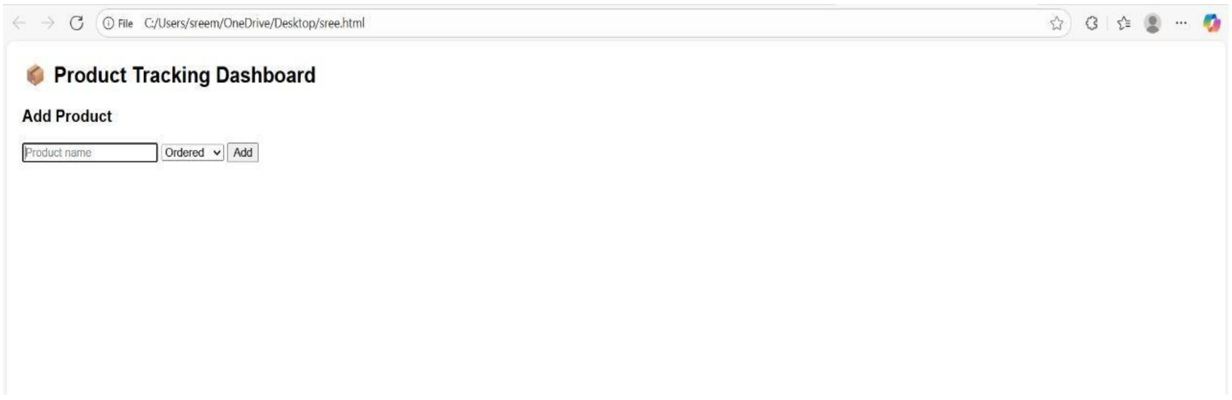
WORKFLOW CHART



SOURCE CODE

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Real-Time Product Tracker</title>

  <style>

   Body

 {

 font-family: Arial; margin: 20px;

 }

  .Product

 {

 margin-bottom: 10px; padding: 10px; border: 1px solid #ccc;

 }

  .status

 {

font-weight: bold;

}

  </style>

</head>

<body>

  <h2>  Product Tracking Dashboard</h2>

  <div id="productList"></div>


  <h3>Add Product</h3>

  <input type="text" id="productName" placeholder="Product name">
 <select id="productStatus">
  <option value="Ordered">Ordered</option>
  <option value="Packed">Packed</option>
  <option value="Shipped">Shipped</option>
  <option value="Delivered">Delivered</option>
 </select>
 <button onclick="addProduct()">Add</button>

 <script>                    let      products      =
JSON.parse(localStorage.getItem('products')) || [];

  function renderProducts()
 {
   const list = document.getElementById('productList');    list.innerHTML = '';    products.forEach((p,
index) =>
{
     list.innerHTML += `
      <div class="product">
       <span>${p.name}</span> -
       <span class="status">${p.status}</span>
       <select onchange="updateStatus(${index}, this.value)">
        <option value="Ordered">Ordered</option>
        <option value="Packed">Packed</option>
        <option value="Shipped">Shipped</option>
        <option value="Delivered">Delivered</option>
       </select>
       <button onclick="delete Product(${index})"> </button>
      </diV>   });
   }
   function add Product()
 {
     const                    name                    =                    document.
getElementById('productName').value;                  const   status   =
document.getElementById('productStatus').value;         if (name) {
products.  push({  name,  status  });                 local    Storage.  set
Item('products', JSON. stringify(products));        render Products();
document. Get Element By Id('product Name').value = '';
     }
   }

   function update Status(index, new Status)
{
     products[index].status = new Status;       local Storage. Set
Item('products',   JSON.   stringify(products));                render
Products();
   }

   function delete Product(index)
{
     products.  splice(index, 1);
     local Storage. Set  Item('products', JSON. stringify(products))   render Products()    }  render  Products();
</script>
</body>
</html>
```

THE OUTPUT



File C:/Users/sreem/OneDrive/Desktop/sree.html

Product Tracking Dashboard

Add Product