KRISHI SANGAM

Smart Labour Recommendation System

Course Name: DevOps

Institution Name: Medicaps University – Datagami Skill Based Course

Student Name(s) & Enrolment Number(s):

Sr no	Student Name	Enrolment Number
1	SHIVRAJ SINGH PIPAWAD	EN22IT301100
2	AARTI AGRAWAL	EN22IT301002
3	SANSKAR SETHIA	EN22IT301094
4	SMRUTI PAWAR	EN24CA5030164
5	SANJARI AGRAWAL	EN22EL301046
6	HARIOM PATIDAR	EN24CA5030056

Group Name: Group 09D12

Project Number: 47

Industry Mentor Name: Mr. Vaibhav

University Mentor Name: Akshay Saxena

Academic Year: 2025-26

Problem Statement & Objectives

1. Problem Statement :

IaC Provisioning for Real Estate System

Automate the provisioning of infrastructure for a Real Estate system using Infrastructure as Code (IaC) tools like Terraform or Ansible. The project involves script-based deployment of local Docker environments or cloud instances (e.g., AWS), including the automated installation of core dependencies such as Terraform or Ansible. This approach eliminates manual configuration errors and enables repeatable, scalable environment setup.

Agriculture in India depends heavily on manual labour, yet farmers often struggle to find the right workers at the right time. Traditional labour hiring relies on personal networks, inconsistent availability, and manual searching, which leads to delays, increased costs, and reduced productivity. Farmers also lack real-time insights regarding crop planning, labour needs, and weather conditions, making agricultural decision-making more challenging. With the advancement of Artificial Intelligence, Machine Learning, and modern web technologies, it is now possible to build a system that simplifies labour discovery, provides smart recommendations, and supports farmers with data-driven insights. The Smart Labour Recommendation & Assistance System is designed to address these challenges by offering features such as skill-based labour profiles, real-time availability mapping, crop prediction, yield prediction, weather updates, and intelligent labour recommendations. This platform allows farmers to access essential information and hire suitable labour more efficiently, ultimately improving farm planning and overall agricultural productivity.

2. Project Objectives:

Based on the **Krishi Sangam** project report, the primary objectives of the system are focused on leveraging AI and modern web technologies to solve the inefficiencies of traditional agricultural labor management.

The specific project objectives include:

- **Simplifying Labour Discovery:** To provide a digital platform that replaces slow and unreliable manual searching with a fast, accurate, and farmer-friendly solution for finding skilled labor.
- **Providing Intelligent Recommendations:** To implement AI-powered modules that recommend the most suitable laborers based on specific crop types, field sizes, and seasonal conditions.
- **Enabling Data-Driven Decision Making:** To offer farmers real-time insights through crop prediction, yield prediction, and live weather updates to support more informed agricultural planning.
- **Mapping Real-Time Availability:** To create a system where farmers can view current labor availability and skill-based worker profiles, facilitating better workforce management.
- **Optimizing Farm Productivity:** To reduce delays and increased costs associated with traditional hiring methods, thereby enhancing overall efficiency and productivity in rural agricultural practices.
- **Ensuring Accessibility and Reliability:** To build a scalable and reliable system using the MERN stack (MongoDB, Express, React, Node.js) that is accessible to farmers across diverse agricultural regions.

3. Scope of the Project:

1. Modernizing Labour Logistics

Traditional farming relies heavily on word-of-mouth or local middle-men, which often leads to shortages during peak seasons (like harvesting or sowing).

- **On-Demand Access:** By bypassing manual networks, farmers gain a "digital marketplace" for labour. This reduces the **search cost** and ensures that field activities aren't stalled by a lack of manpower.
- **Smart Matching:** The system doesn't just list names; it uses **smart recommendations** to pair specific farm tasks with workers who have the relevant experience, ensuring higher quality work.

2. Data-Driven Farm Intelligence

The inclusion of AI shifts the platform from a simple utility app to a strategic planning tool.

- **Predictive Analytics:** By analyzing historical data and environmental factors, the AI provides **yield predictions**. This helps farmers manage expectations, secure buyers in advance, and plan their finances.
- **Contextual Awareness:** Integrating **real-time local weather updates** allows for tactical adjustments. For example, if heavy rain is predicted, the system can prompt the farmer to accelerate harvesting or delay fertilizer application.

3. Scalability and Accessibility

The system is designed to be inclusive, recognizing that "agriculture" looks different for everyone.

- **Micro to Macro:** The architecture supports **small-scale farmers** who may only need two workers for a day, as well as **large industrial operations** requiring complex scheduling for hundreds of staff.
- **Geographic Versatility:** Because it is designed for villages and diverse regions, it accounts for the varying agricultural cycles found across different climates.

4. Future-Proofing (The Roadmap)

The scope identifies critical "trust-building" features for future iterations:

- **Financial Security: Escrow-based payments** ensure that workers are guaranteed pay upon completion, while farmers are protected from paying for unfinished work.

- **Removing Barriers: Multi-language support** is vital for adoption in rural areas where English or tech-heavy terminology might be a barrier.
- **Security: Digital identity verification** builds a community of vetted, reliable participants, reducing the risk of fraud on both sides.

Proposed Solution

Our proposed solution is an **AI-driven Smart Labour Recommendation & Assistance System** designed to modernize agricultural resource management through a layered, data-centric approach.

1. System Architecture

The system utilizes a four-layer modular architecture to ensure scalability and real-time responsiveness:

- **Presentation Layer**: A mobile-friendly user interface built with **React.js** that provides dedicated dashboards for farmers (to post requirements and view predictions) and laborers (to update skills and availability).
- **Application Layer**: A **Node.js and Express.js** backend that handles business logic, routing, and coordination between the database, AI engine, and external APIs.
- **AI Engine Layer**: A Python-based module utilizing machine learning (Scikit-learn, TensorFlow) to perform skill matching, yield estimation, and distance-based ranking.
- **Data Layer**: A **MongoDB Atlas** database that stores labor profiles, farmer requirements, hire requests, and historical weather logs.

2. Core Functional Modules

The solution is comprised of several integrated modules that work together to provide comprehensive agricultural support:

- **Labour Manager**: Manages skill-based worker profiles, including experience levels and real-time availability status.
- **Smart Recommendation Engine**: Calculates suitability scores for laborers based on the specific crop type, field size, and task requirements provided by the farmer.
- **Agricultural Prediction Models**: Includes a **Crop Prediction Model (CPM)** and **Yield Prediction Model (YPM)** to help farmers estimate performance and plan workforce needs.
- **Weather Advisory Module**: Integrates with external weather APIs to provide real-time updates on temperature and rainfall, helping farmers mitigate risks.

3. Operational Workflow

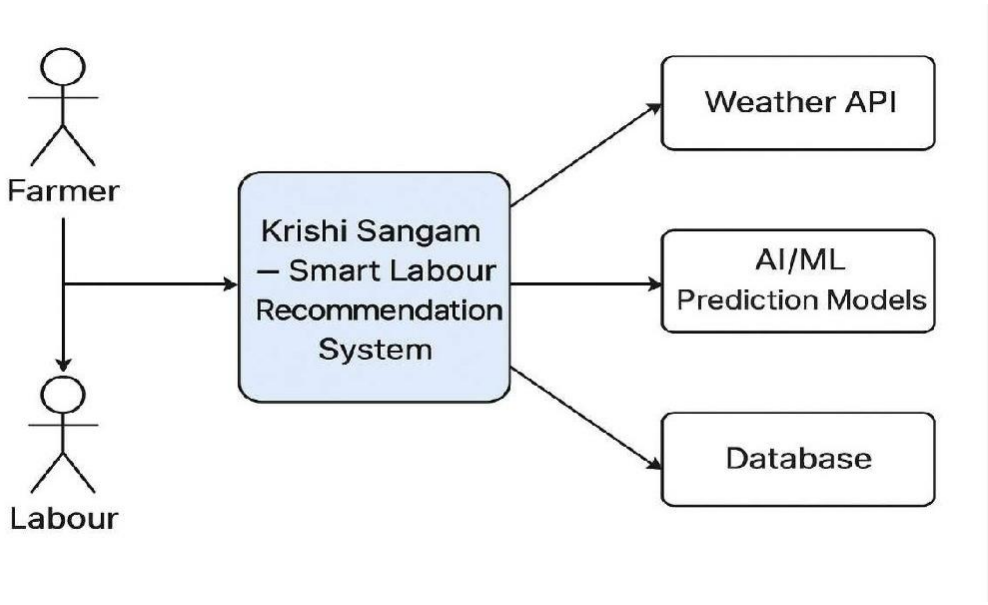
The system follows a structured request-processing-result pattern:

1. **Registration**: Laborers register their skills and location.
2. **Input**: Farmers enter farm requirements (e.g., harvesting task in a 5-acre field).
3. **Processing**: The system fetches weather data and executes AI models to predict crop needs and identify the best labor matches.
4. **Action**: The farmer reviews a ranked list of nearby laborers, checks their suitability scores, and sends hire requests directly through the platform.

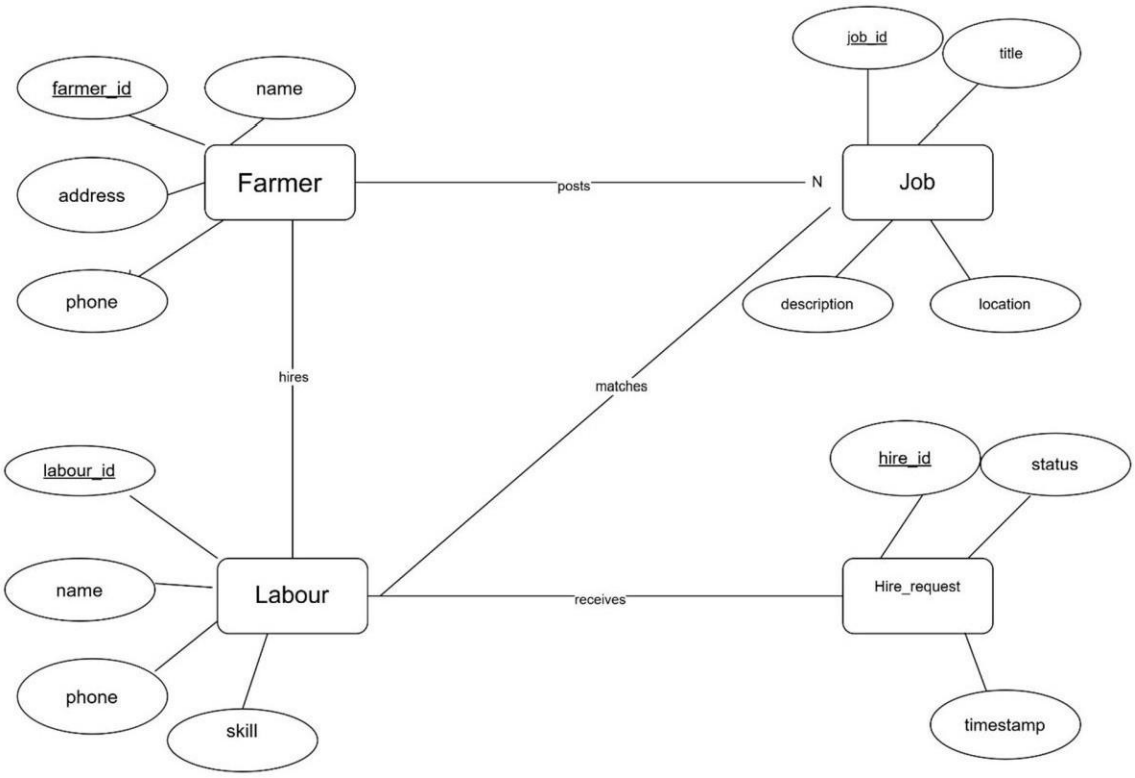
4. Technical Feasibility & Impact

- **Technologies**: The solution is built on the **MERN stack** (MongoDB, Express, React, Node.js) with Python for AI components, ensuring high-speed data access and cross-platform compatibility.
- **Outcome**: By replacing manual networks with a digital marketplace, the system reduces "labor mismatch," minimizes delays in field activities, and improves overall farm productivity.

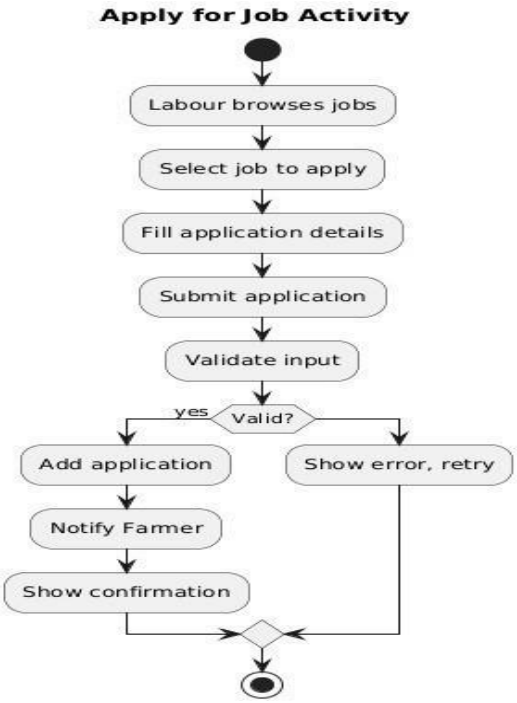
Architectural Context Diagram

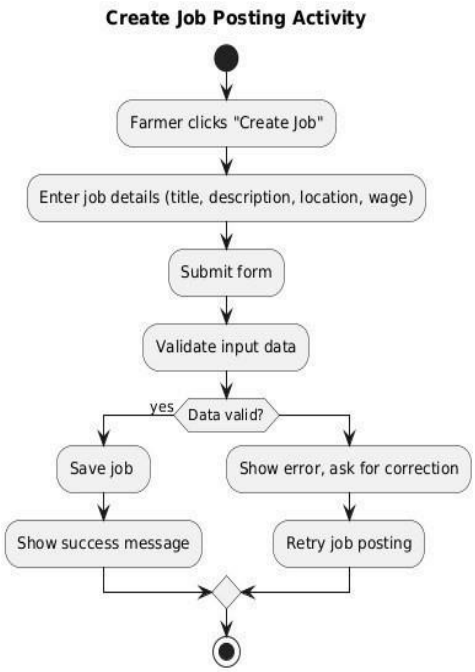


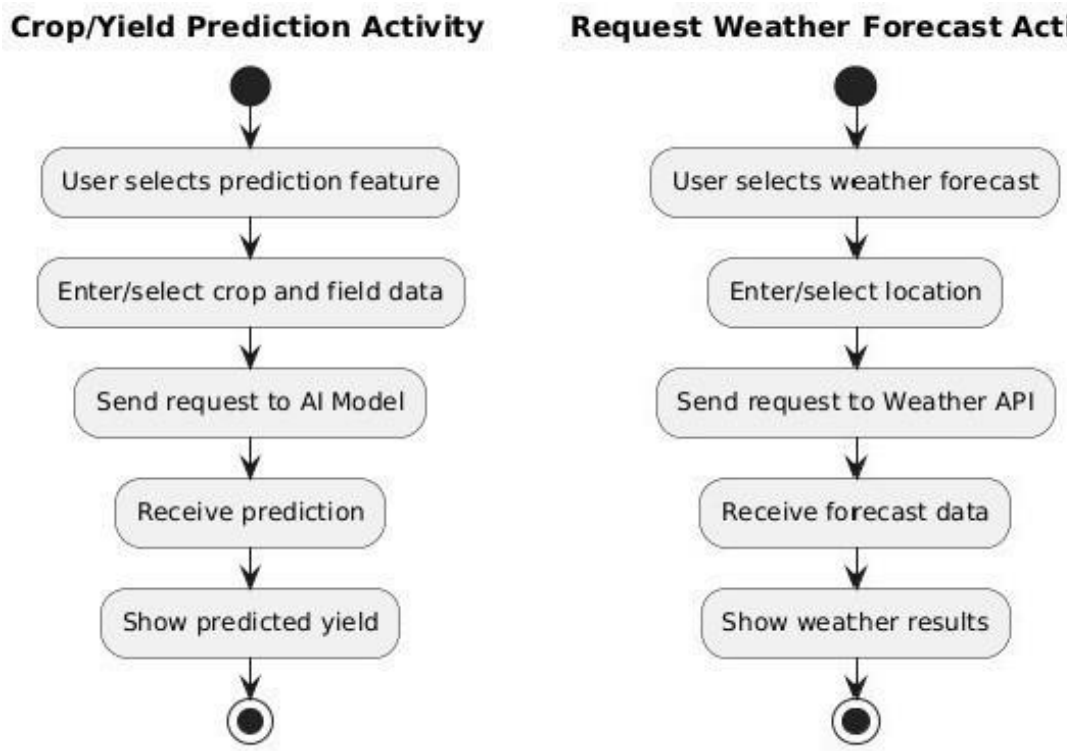
Entity-Relation Diagram:

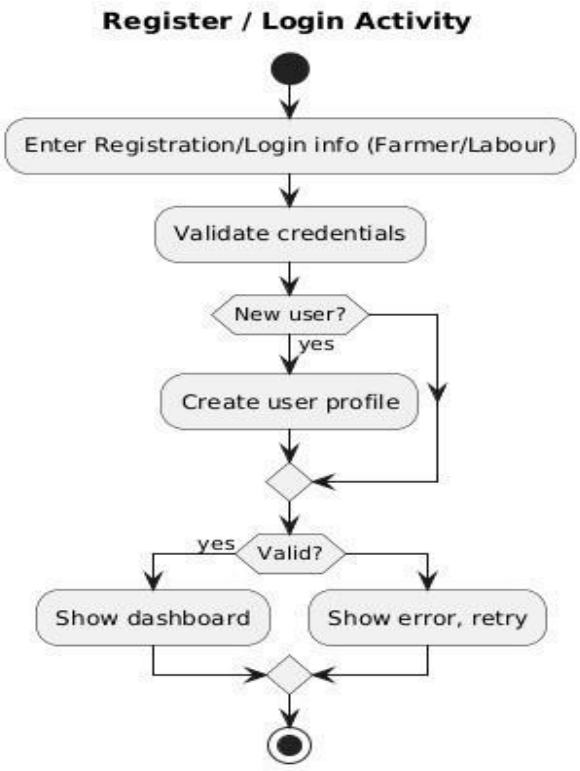


Activity Diagrams:

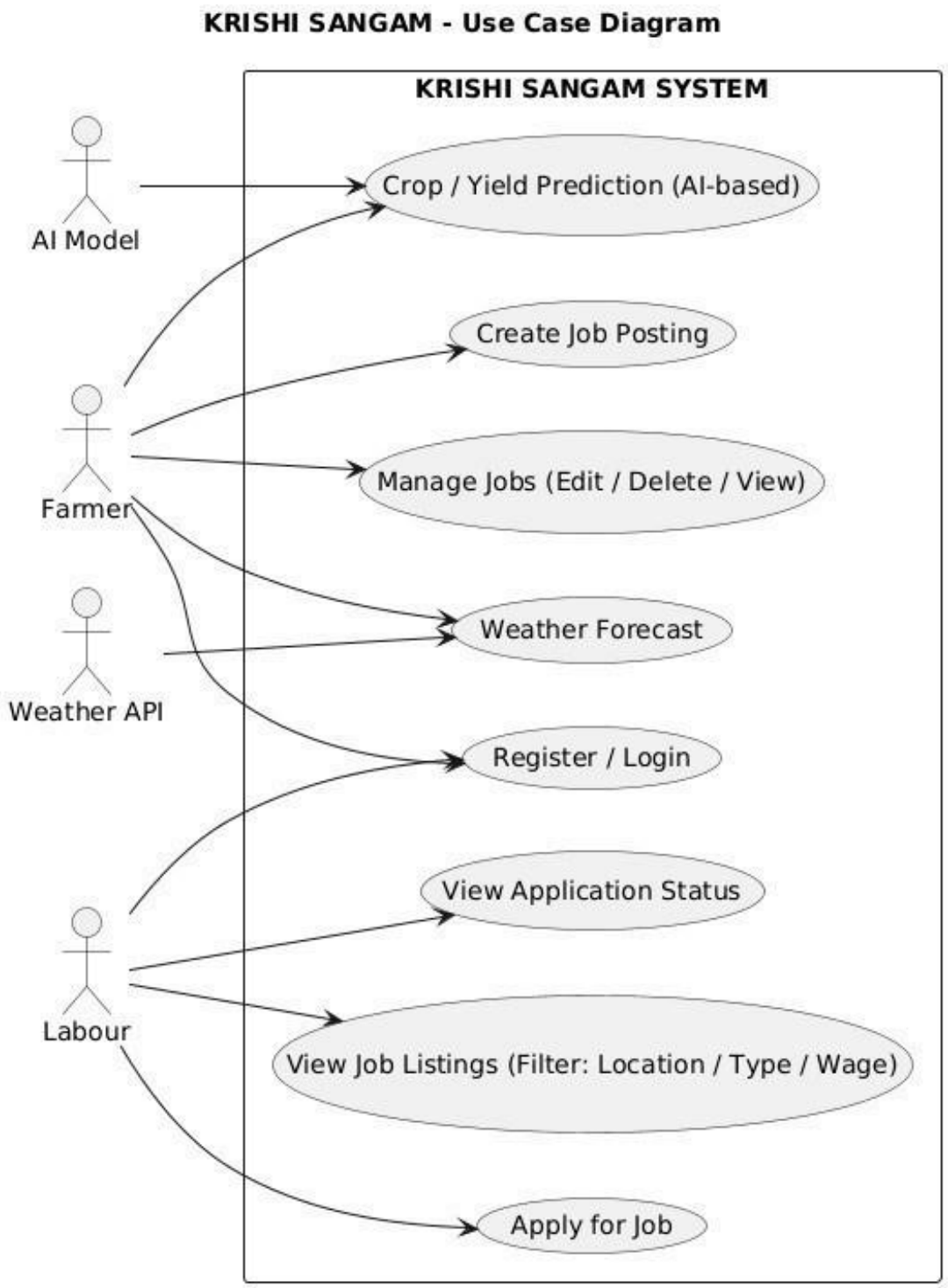





Crop/Yield Prediction Activity
User selects prediction feature
Enter/select crop and field data
Send request to AI Model
Receive prediction
Show predicted yield
Request Weather Forecast Activity
User selects weather forecast
Enter/select location
Send request to Weather API
Receive forecast data
Show weather results


Register / Login Activity
Enter Registration/Login info (Farmer/Labour)
Validate credentials
New user?
yes
Create user profile
yes
Valid?
Show dashboard
Show error, retry

Use Case Diagram:



1. Tools & Technologies Used :

The project utilizes a well-defined set of tools and technologies, each selected to support different layers of application development and DevOps automation. These tools collectively enable the creation of a reliable, scalable, and maintainable retail web application.

For the frontend layer, HTML, CSS, and JavaScript are used to design and implement the user interface. HTML provides the structural layout of the web pages, CSS handles styling and visual presentation, and JavaScript enables dynamic behavior and interaction with backend APIs. This combination ensures a lightweight and responsive frontend.

The backend layer is developed using Node.js and Express.js. Node.js offers a fast, event-driven runtime environment suitable for scalable web applications, while Express.js simplifies request handling, routing, and API development. Together, they form the core application logic and service layer.

For containerization, Docker and Docker Compose are employed. Docker packages the application and its dependencies into a portable container, ensuring consistency across environments. Docker Compose further simplifies application execution by managing container configurations and enabling easy startup and shutdown.

The project implements Continuous Integration (CI) using GitHub Actions. This tool automates dependency installation, build processes, and testing whenever code changes are pushed to the repository, improving code reliability and reducing manual effort.

A Jenkins-ready CI/CD architecture is included to support future deployment automation. Jenkins can be integrated to extend CI into full continuous deployment pipelines.

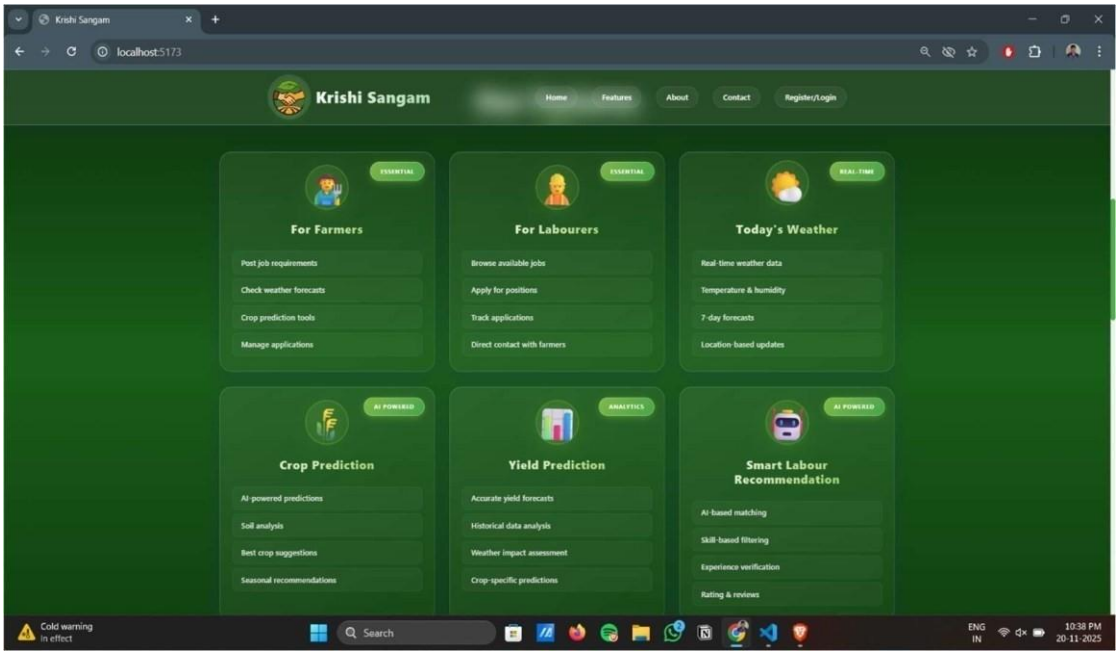
For version control, Git and GitHub are used to track changes and manage collaboration. The project is developed and tested on a Linux operating system, which is widely used in DevOps and production environments.

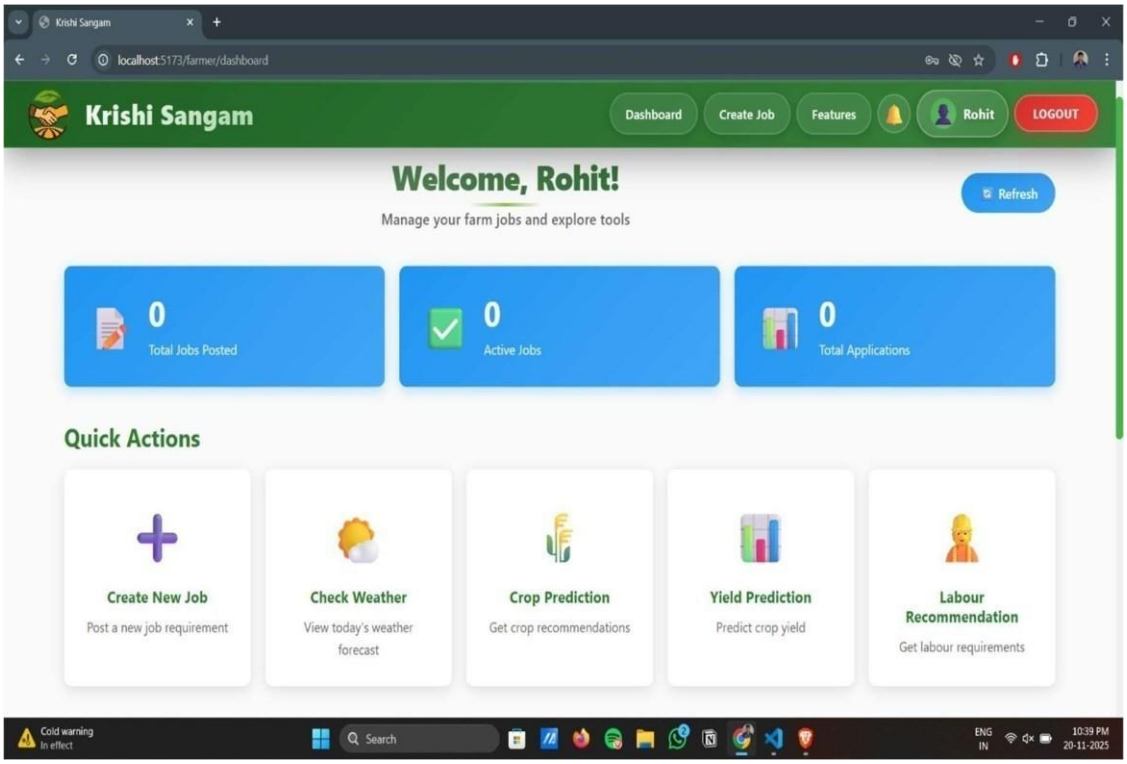
The following technologies, frameworks, libraries, and deployment platforms are used for the development of the Smart Labour Recommendation & Assistance System:

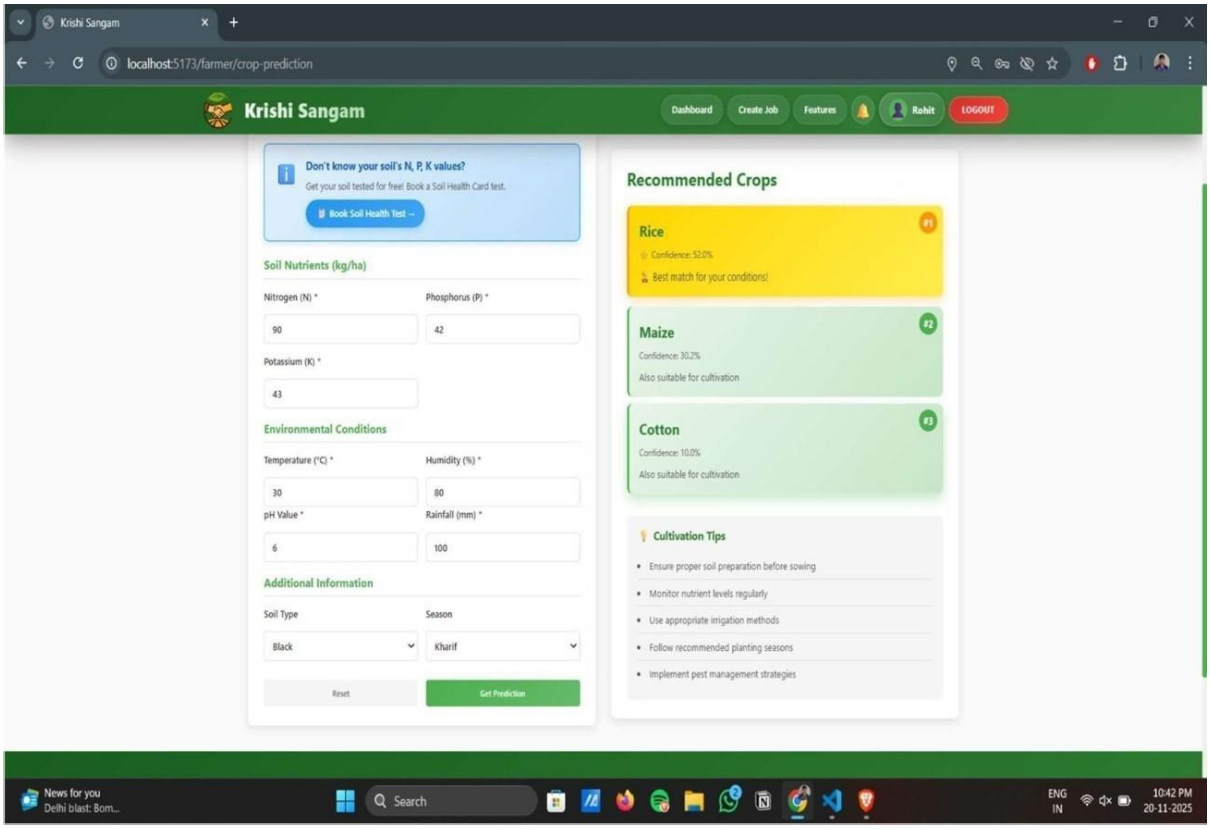
Category	Technology Used
Frontend Languages	React.js, HTML5, CSS3, JavaScript
Backend Languages	Node.js, Express.js
Database Technology	MongoDB, MongoDB Atlas
AI / ML Libraries	Python, Scikit-learn, TensorFlow, Pandas, NumPy
Weather Integration	OpenWeatherMap API or similar
Cloud Hosting	Render/ Vercel / Firebase / AWS EC2
Programming Languages Used	JavaScript, Python
API Style	REST API
Version Control	Git, GitHub
Data Visualization (Optional)	Chart.js / Recharts
Authentication	JWT (JSON Web Token) / OAuth
Deployment Tools	Docker (optional), GitHubActions (optional CI/CD)

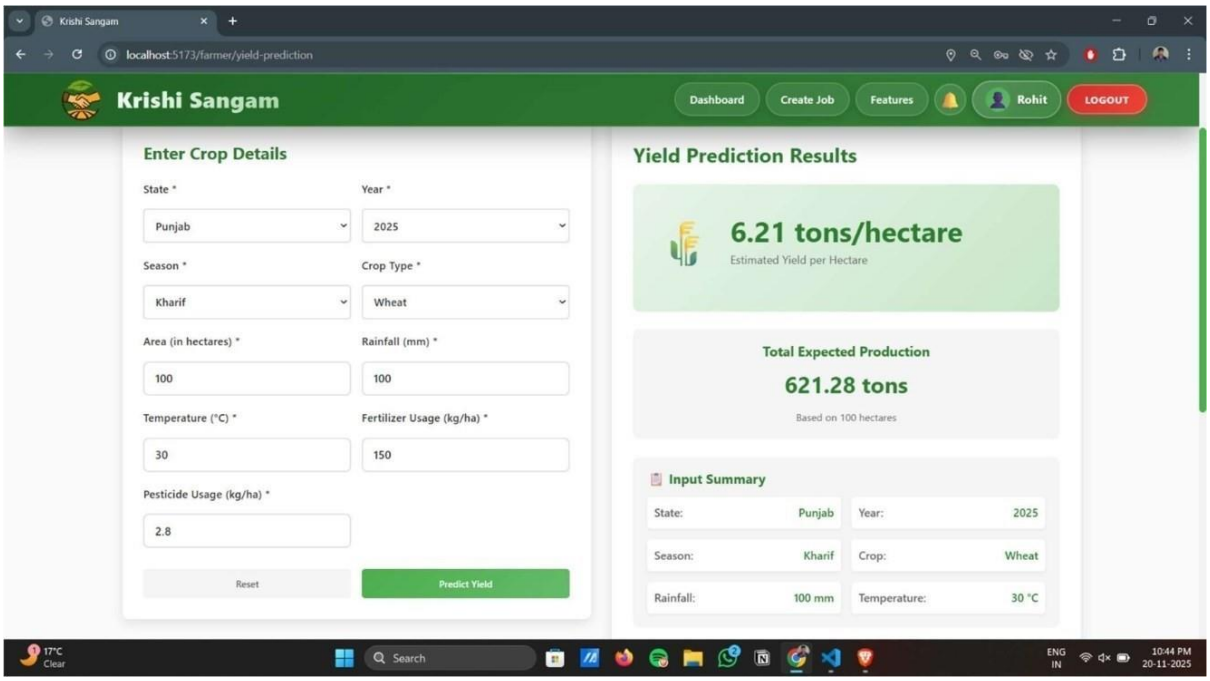
Results & Output

1. FrontEnd Screenshots

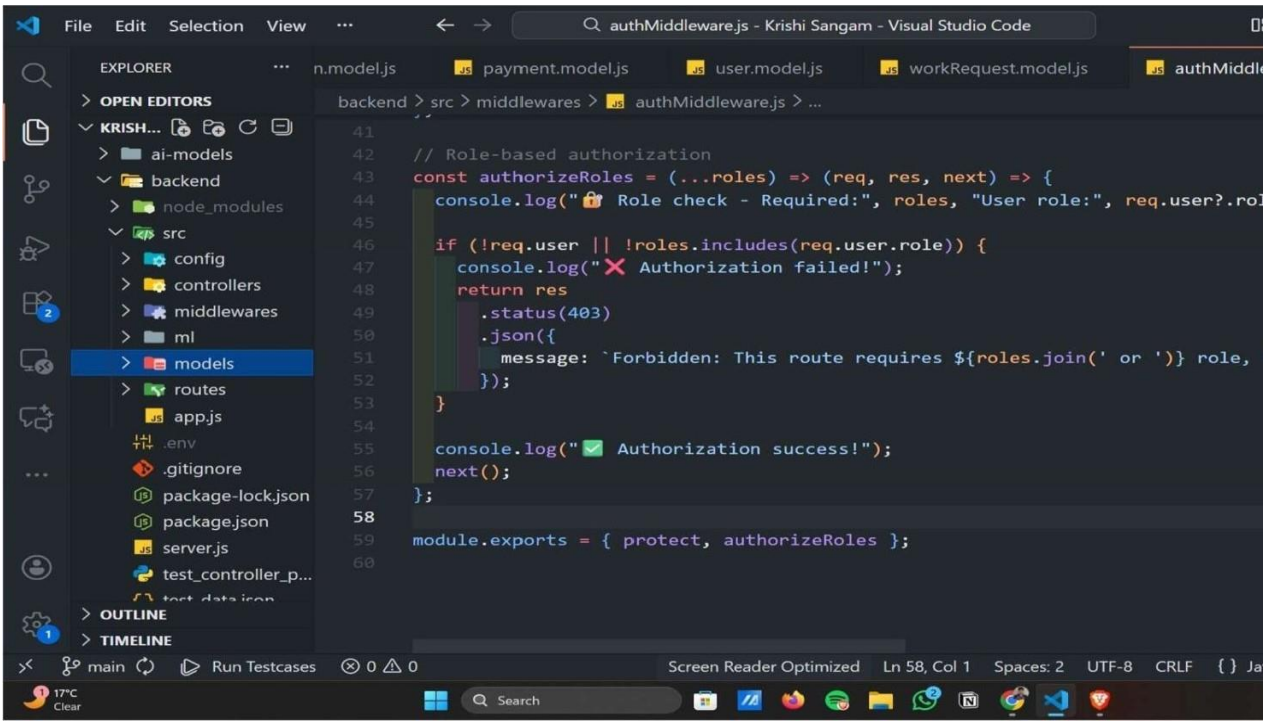





Krishi Sangam
localhost:5173/farmer/crop-prediction
Krishi Sangam
Dashboard
Create Job
Features
Rohit
LOGOUT
Don't know your soil's N, P, K values?
Get your soil tested for free! Book a Soil Health Card test.
Book Soil Health Test →
Soil Nutrients (kg/ha)
Nitrogen (N) *
90
Phosphorus (P) *
42
Potassium (K) *
43
Environmental Conditions
Temperature (°C) *
30
Humidity (%) *
80
pH Value *
6
Rainfall (mm) *
100
Additional Information
Soil Type
Black
Season
Kharif
Reset
Get Prediction
Recommended Crops
Rice
Confidence: 52.0%
Best match for your conditions!
Maize
Confidence: 30.2%
Also suitable for cultivation
Cotton
Confidence: 10.0%
Also suitable for cultivation
Cultivation Tips
Ensure proper soil preparation before sowing
Monitor nutrient levels regularly
Use appropriate irrigation methods
Follow recommended planting seasons
Implement pest management strategies
News for you
Delhi blast: Bom...
Search
ENG
IN
10:42 PM
20-11-2025


Krishi Sangam
localhost:5173/farmer/yield-prediction
Krishi Sangam
Dashboard
Create Job
Features
Rohit
LOGOUT
Enter Crop Details
State *
Punjab
Year *
2025
Season *
Kharif
Crop Type *
Wheat
Area (in hectares) *
100
Rainfall (mm) *
100
Temperature (°C) *
30
Fertilizer Usage (kg/ha) *
150
Pesticide Usage (kg/ha) *
2.8
Reset
Predict Yield
Yield Prediction Results
6.21 tons/hectare
Estimated Yield per Hectare
Total Expected Production
621.28 tons
Based on 100 hectares
Input Summary
State: Punjab
Year: 2025
Season: Kharif
Crop: Wheat
Rainfall: 100 mm
Temperature: 30 °C
17°C
Clear
Search
ENG
IN
10:44 PM
20-11-2025

2. Backend:


File Edit Selection View
authMiddleware.js - Krishi Sangam - Visual Studio Code
EXPLORER
OPEN EDITORS
KRISH...
ai-models
backend
node_modules
src
config
controllers
middlewares
ml
models
routes
app.js
.env
.gitignore
package-lock.json
package.json
server.js
test_controller_p...
OUTLINE
TIMELINE
n.model.js
payment.model.js
user.model.js
workRequest.model.js
backend > src > middlewares > authMiddleware.js > ...
// Role-based authorization
const authorizeRoles = (...roles) => (req, res, next) => {
console.log(" Role check - Required:", roles, "User role:", req.user?.rol
if (!req.user || !roles.includes(req.user.role)) {
console.log(" Authorization failed!");
return res
.status(403)
.json({
message: `Forbidden: This route requires ${roles.join(' or ')} role,
});
}
console.log(" Authorization success!");
next();
};
module.exports = { protect, authorizeRoles };
main
Run Testcases
0 0
Screen Reader Optimized
Ln 58, Col 1
Spaces: 2
UTF-8
CRLF
17°C
Clear
Search

3. **Key outcomes:**

The implementation of **Krishi Sangam** yields several transformative outcomes that bridge the gap between traditional farming and modern digital efficiency. By centralizing labour management, the system significantly **reduces search costs and operational downtime**, ensuring that critical agricultural activities like sowing and harvesting are completed within optimal seasonal windows. The integration of AI-driven crop and yield predictions empowers farmers with **data-backed foresight**, transitioning them from reactive to proactive planning.

Furthermore, the platform fosters a **transparent and verified agricultural marketplace**, where labourers gain better visibility and consistent employment opportunities while farmers benefit from a reliable, skill-matched workforce. Ultimately, these advancements lead to **enhanced resource optimization, increased farm productivity, and improved socio-economic stability** for rural farming communities.

Another important achievement was the **containerized deployment of the application** using Docker and Docker Compose. Containerization ensured consistent application behaviour across different environments and simplified execution and testing. Docker Compose further enabled easy management of application runtime in a production-like setup.

The project also adhered to **industry-standard DevOps practices**, including clean project structure, version control using Git and GitHub, automation-driven workflows, and CI/CD readiness. These outcomes collectively demonstrate practical knowledge of DevOps tools and methodologies and provide a strong foundation for building scalable and maintainable software systems.

Conclusion:

The Smart Labour Recommendation & Assistance System effectively solves a major challenge in Indian agriculture by simplifying the process of finding skilled and available labour. Traditional labour hiring depends on manual searching and personal networks, which often leads to delays, higher costs, and reduced efficiency. By using AI, Machine Learning, and real-time data, the system provides farmers with accurate information on labour availability, weather updates, crop requirements, and yield predictions, enabling them to plan their work more efficiently.

Overall, the project successfully demonstrates how technology can modernize agricultural labour management. By offering intelligent labour recommendations, skill-based worker profiles, and data-driven insights, the system helps farmers make smarter decisions and improve productivity. This solution has strong potential to enhance rural agricultural practices and contribute to a more organized and efficient farming ecosystem.

Future Scope & Enhancements:

Future enhancements for the Smart Labour Recommendation & Assistance System can significantly improve usability and efficiency for both farmers and labourers. A dedicated Android/iOS mobile app can provide faster performance, offline access, regional language support, and real-time notifications. Geo-tracking and smart route optimization enable farmers to locate nearby available labour while suggesting efficient routes to reduce travel time. Voice-enabled assistance in regional languages like Hindi can make the platform more accessible for users with low digital literacy. Automated weather impact alerts can help farmers plan labour activities in response to rainfall, storms, or temperature changes. Verified digital worker profiles allow labourers to showcase their skills, improving trust and enabling farmers to hire suitable workers quickly. Seasonal labour pooling and group hiring features support large agricultural tasks such as harvesting or transplanting. Additionally, integration of secure payment gateways and digital contracts can ensure transparent and hassle-free transactions between farmers and labourers.

Future improvements for the Smart Labour Recommendation & Assistance System include a dedicated mobile app for faster performance, offline access, regional language support, and real-time notifications. Geo-tracking and smart route optimization can help farmers find nearby labour quickly, while voice-enabled assistance in local languages makes the platform accessible for users with low digital literacy. Automated weather alerts, verified digital worker profiles, and seasonal labour pooling can improve planning, trust, and efficiency. Secure payment gateways and digital contracts can ensure transparent and smooth transactions between farmers and labourers.