**VRIKSHNETRA**

A Synopsis

for

Project-Work-II

**BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING**

BY

**Pradyumna Rajnekar EN21CS301561**

**Prateek Patidar EN21CS301574**

Under the Guidance of

**Prof. Kailash Kumar Baraskar**

**&**

**Prof. Garima Silakari**

****

**Department of Computer Science & Engineering**

**Faculty of Engineering**

**MEDI-CAPS UNIVERSITY,INDORE- 453331**

**AUG - DEC 2024**

**Introduction:**

In the digital age, plant identification platforms have become an integral tool for nature enthusiasts, allowing users to discover, learn, and share knowledge about various plant species. However, existing platforms often lack the structure necessary for accurate identification and user engagement. Users frequently encounter misidentifications or lack of detailed plant information, which can significantly hinder their learning experience. Furthermore, the absence of category-based filtering makes it difficult for users to find relevant plant species or similar plants of interest.

**Problem Statement**:

The primary problem addressed by this project is the lack of a structured platform for accurate plant species identification, where users can upload images of plants and receive precise results. Current platforms do not provide an organized way for users to identify plants based on reliable visual input, leading to frequent misidentifications and a frustrating user experience. This lack of structured identification results in users either receiving incorrect information or struggling to find relevant plant data. Additionally, there is no existing platform that offers category-based filtering, which is essential for users to explore specific types of plants or similar species of interest.

**Literature Review**:

Several popular platforms offer plant identification and related content, but none provide the highly accurate, real-time, and genre-specific feedback that Vrikshnetra aims to deliver.

● **PlantSnap**: While PlantSnap offers plant identification through image uploads, it lacks a clear categorization system for users to explore plants based on their species or region. Additionally, it does not focus on real-time feedback, making the identification process less interactive for users.

● **LeafSnap** :This platform allows users to identify plant species, but its primary limitation is in providing accurate results for lesser-known or rare species. Vrikshnetra fills this gap by utilizing a CNN-based model to provide highly accurate plant species identification, even for uncommon plants.

● **PictureThis**: Known for its plant identification abilities, PictureThis offers basic image recognition but does not include a structured way for users to filter results based on plant families or types. Vrikshnetra addresses this by offering detailed filtering and categorization, ensuring a more structured exploration of plant species.

● **Category-Specific Organization**: Existing platforms such as Flora Incognita offer plant identification but do not provide category-based filtering, making it difficult for users to search for specific types of plants. Vrikshnetra adapts the concept of category-based organization.

● **Localized Plant Databases**: While platforms like iNaturalist provide global databases for plant and wildlife identification, they often lack focus on regional flora. Vrikshnetra sets itself apart by emphasizing plants native to India, such as neem, peepal, and banyan, offering a more localized approach to plant identification. This tailored solution improves accuracy and relevance for users in specific regions, filling the gap left by global platforms.

Vrikshnetra differentiates itself by providing real-time identification, enhanced accuracy through CNN technology, and an organized, category-based filtering system to make plant exploration more seamless and informative for users.

**Objectives**:

1. To Develop a Real-Time Plant Identification Platform:

● Provide users with a platform where they can upload images of plants and receive real-time results for species identification using a CNN-based model.

2. To Ensure Accurate and Reliable Identification:

● Implement an advanced machine learning model focused on improving plant identification accuracy, especially for species native to India.

3. Species by Categories:

● Enable users to browse and identify plants through a category-based filtering system (e.g., medicinal plants, flowering plants), allowing for more focused and relevant identification.

4. To Facilitate a Scalable and Secure Platform:

● Build a highly scalable and secure system using HTML/CSS for the frontend and CNN for the backend, capable of handling large numbers of concurrent users and identification requests efficiently.

5. To Offer Social Login and Seamless User Authentication:

● Ensure users can easily upload plant images, interact with the identification system, and explore related plant species for a smooth, engaging experience.

**Methodology**:

The ‘Vrikshnetra’ project will be developed using the following methodology:

**1. Requirements Gathering**:

● Gather the primary requirements such as plant image uploads, plant species detection using a CNN model, and a seamless user experience.

● Conduct surveys and interviews with potential users to understand their expectations for a plant identification platform, including speed, accuracy, and ease of use.

**2. UI/UX Design (Frontend)**

* HTML Structure: The layout needs to be simple, allowing users to upload an image and view the results easily. We can use basic HTML5, CSS, and a few JavaScript enhancements (if needed).
* CSS Design: For responsiveness and styling, we can use Flexbox or CSS Grid for a clean, mobile-friendly layout. We can use some pre-built UI components to keep things consistent and modern, like buttons and input fields, styled with CSS.
* Image Upload Interface: Provide a simple file input for users to upload their plant image. A button can trigger the upload process, which is handled by JavaScript and passed to your Python-based backend.
* Result Display: After processing the image with your CNN model, show the results in a card-like or modal layout to display plant information, such as species, care tips, or other relevant details.

**3. Development**:

* Frontend Development (HTML/CSS + JavaScript):
  + Develop a basic HTML/CSS front end where users can upload plant images and receive results in a user-friendly layout.
  + JavaScript will send the uploaded image to the Python backend for processing and handle the file selection and interaction with the backend.
* Backend Development (Python):
  + The plant species identification will be powered by a **Convolutional Neural Network (CNN)** model implemented in **Python**. The model will be trained on a dataset of plant images and their corresponding species labels.
  + **Python Libraries:** Use popular Python libraries like **TensorFlow**, **Keras**, or **PyTorch** to build, train, and run the CNN model.

**4. Testing**:

* Validate the CNN model using various plant species images to ensure high accuracy and reliable results.
* Test the entire system for image uploads, identification accuracy, and overall performance.

**5. Deployment and Maintenance**:

* Deploy the application on a cloud platform such as AWS or Heroku.
* Use CI/CD pipelines for continuous updates and bug fixes.
* Monitor performance and user engagement to optimize server resources and improve user experience.

**References:**

**1. HTML and CSS Documentation**

Mozilla Foundation. (2024). *HTML and CSS Documentation*. Available at: **https://developer.mozilla.org/en-US/docs/Web**

**2. Convolutional Neural Networks (CNN) Overview**

LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep Learning: Convolutional Networks for Image Recognition*. Nature, 521(7553), 436- 444. Available at: **https://www.nature.com/articles/nature14539**

**3. Plant Disease Detection Using CNNs**

Mohanty, S.P., Hughes, D.P., & Salathé, M. (2016). *Using Deep Learning for Image-Based Plant Disease Detection*. Frontiers in Plant Science, 7, 1419. Available at:

**https://www.frontiersin.org/articles/10.3389/fpls.2016.01419/full**

**4. WebSocket Protocol Documentation**

IETF. (2024). *RFC 6455: The WebSocket Protocol*. Available at: **https://datatracker.ietf.org/doc/html/rfc6455**

**5. TensorFlow Documentation**

Google Research. (2024). *TensorFlow: An Open-Source Machine Learning Framework*. Available at: **https://www.tensorflow.org/**

**6. VGGNet and Deep Learning Applications**

Simonyan, K. & Zisserman, A. (2015). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. Available at: **https://arxiv.org/abs/1409.1556**

**7. Keras API Documentation**

Chollet, F. (2024). *Keras: The Python Deep Learning Library*. Available at: **https://keras.io/**

**8. OpenCV Library Documentation**

OpenCV Team. (2024). *OpenCV: Open Source Computer Vision Library*. Available at: **https://opencv.org/**