Unity Relief

ABEL MATHEW

JAISON T JACOB

LINTO LALICHAN

ROMEL K REIS

Department of Computer Applications SAINTGITS COLLEGE OF ENGINEERING (Autonomous)

Kottukulam Hills, Pathamuttom P.O., Kottayam 686532

November 2024

ABSTRACT

The UNITY Disaster Relief platform is a cutting-edge digital solution aimed at identifying and assisting individuals impacted by natural disasters. This web-based platform leverages a combination of machine learning (ML), database management, and HTML to streamline the process of tracking, locating, and providing support to affected populations. With the rise in natural disasters due to climate change and other environmental factors, there is a growing need for fast and accurate identification of disaster-affected individuals. UNITY Disaster Relief addresses this by using machine learning algorithms that process and analyze real-time data from various sources, including social media posts, satellite images, government alerts, and user-submitted information.

The backend database stores critical information about affected individuals and disaster areas, enabling rapid retrieval and data management. By applying machine learning, the platform can predict disaster spread, assess damage levels, and prioritize individuals needing urgent assistance. The HTML front end ensures a user-friendly interface where users, emergency responders, and volunteers can efficiently navigate the platform, input relevant data, and access vital information.

UNITY Disaster Relief is designed to be a scalable and flexible solution that can be adapted for different types of natural disasters such as earthquakes, hurricanes, floods, and wildfires. It not only serves as a

resource for immediate disaster response but also facilitates longer-term recovery efforts by connecting affected individuals with relief organizations. Through seamless integration of ML, database technology, and accessible web design, UNITY Disaster Relief aims to provide a comprehensive support system that reduces response time, enhances coordination, and ultimately aids in saving lives during critical disaster events

CHAPTER 1

INTRODUCTION

1.1 **Introduction to the Project**

The UNITY Disaster Relief project aims to support disaster response efforts by creating a web-based system to manage and track individuals affected by natural disasters such as landslides and floods. The primary focus is to assist relatives and responders by centralizing data on affected individuals, their locations in relief camps, and their health status, including hospitalization or death information. This project incorporates machine learning and image processing to enhance the identification and tracking of displaced individuals through facial recognition. By uploading images, users can search for missing persons, and the system can provide information on their current location or status. The platform also includes comprehensive resources for camp management, such as population tracking and resource distribution to optimize food, water, and medical supply allocation, ensuring efficiency and safety within the camps.

This innovative solution addresses a crucial need for organized information in the aftermath of disasters, helping both family members and relief agencies to locate, identify, and monitor affected individuals effectively. Through the UNITY platform, responders and loved ones are empowered with timely, accurate information, bridging communication gaps and enhancing disaster relief operations

The UNITY system not only facilitates real-time tracking and support for relief efforts but also offers a user-friendly interface designed to simplify complex tasks for both professional responders and volunteers. By enabling efficient resource management and providing an accessible platform for critical updates, UNITY ensures that essential information is available at every level of disaster response. This holistic approach enhances overall preparedness, supporting communities in crisis while building a reliable foundation for future disaster management and recovery efforts.

1.2 **Organization Profile**

SAINTGITS COLLEGE OF ENGINEERING

Saintgits College of Engineering is a self-financing technical institution located at kottayam district of Kerala. The college was established in 2002. Saintgits is approved by All India Council for Technical Education and affiliated to APJ Abdul Kalam Technological University, Kerala. The institute is accredited by NBA in 2016 for 3 years for 5 UG programs and in 2017 for 3 years for Master of Computer Application program.The college was founded by a group of well-known academicians. They are pioneering educators, having unmatched experience in the field of education with a belief that the continuous search for knowledge is the sole path to success.

The primary focus of the institution is to expose the young minds to the world of technology, instilling in them confidence and fortitude to face new challenges that enable them to excel in their chosen fields. The college inculcates the development of all facets of the mind culminating in an intellectual and balanced personality. A team of dedicated and caring faculty strives to widen the student's horizon of learning thereby achieving excellent results for every student.With a scientifically planned methodology combined with a team of handpicked faculty -the best in the teaching profession and the state-of-the-art infrastructure, the quality of the engineering education at Saintgits is unparalleled in the region.The institute has turned into a benchmark for others to emulate. With 100% seats filled from the year of inception itself, and feel confident that Saintgits can serve even better with every passing year.

Saintgits College of Engineering, right from inception, has been maintaining high levels of standards in academic and extra-curricular realms of activities. Saintgits offer a BTech Degree course in 9 engineering disciplines, and Master's Degree courses in Engineering, Computer Applications and Business Administration. In the short span of a decade of its existence and among the six batches of students that have graduated, the college bagged several universities ranks and has a remarkably high percentage of pass. The students of the first batch of MCA bagged the first two ranks in the University. The college is also the venue of national and state level seminars and symposiums and has emerged as the hub of technical education in Kerala.

1.3 Objectives of the Project

I. **Data Collection and Accessibility**: The primary objective is to create a centralized platform to collect and display data on individuals affected by natural disasters such as landslides and floods. The platform will allow users to view information on those relocated to hospitals and relief camps, helping families and officials access crucial details on the affected population's location and status.

II. **Identification and Tracking of Individuals**: The project aims to implement image processing and facial recognition technologies to identify and track people within disaster zones and relief camps. This feature will assist in locating individuals, enabling families to find missing persons more easily.

III. **Camp Management and Resource Distribution**: By integrating real-time data collection, the platform will track camp populations and support efficient resource distribution. This includes managing the supply of essentials like food, water, and medical supplies, and addressing safety and security concerns.

IV. **Machine Learning Integration**: The project includes developing a machine learning model that accurately identifies individuals from images. This model will be trained on diverse datasets to ensure high accuracy and precision, reducing the likelihood of false identifications. The model will be instrumental in recognizing people based on uploaded photos, improving search and identification processes.

V. **Ease of Access and User Interaction**: With features like a login system, contact forms, and informational pages (such as "About Us" and blogs), the platform will provide an intuitive interface for both affected individuals and their families. This approach aims to increase community engagement and provide reliable information to the public.

1.4 Purpose, Scope, and Applicability of the Project

Purpose

The primary purpose of the UNITY Disaster Relief System project is to create a comprehensive disaster relief web application that can collect and manage data on individuals affected by natural disasters, particularly landslides and floods. Through this platform, the application aims to ease the process for individuals to locate their loved ones, identifying those who have been relocated to hospitals or relief camps. The application seeks to address the urgent need for organized and accessible information during crisis times, including providing real-time information on people who have been deceased. By utilizing machine learning and image processing technologies, the project further aims to improve efficiency in tracking affected individuals. The platform is designed to enable users to upload images to help locate missing persons, thus

streamlining the search and identification process during such emergencies.

Scope

The scope of the *UNITY Disaster Relief System* project encompasses several core functionalities to support disaster relief efforts:

1. **Identification and Tracking of Individuals**:

- The system employs facial recognition technology to locate individuals within relief camps or affected areas.
- It provides a way to track the whereabouts and status of each person, helping families and friends locate their loved ones.

2. **Camp Management and Resource Allocation**:

- The application allows for tracking the population within each relief camp and manages the distribution of essential resources, such as food, water, and medical supplies.
- Security monitoring is integrated into the system to ensure safety within camps.

3. **Machine Learning and Image Processing**:

- The project incorporates machine learning to improve the precision of identifying individuals based on images.
- Image processing allows the system to handle missing person reports by identifying individuals from uploaded photos and tracking them across different camps or medical facilities.

4. **Information Accessibility and User Interaction**:

- The application's interface includes various pages, such as an index page that gives a brief description of the site, an "About Us" section, a "Relief Camps" information page, and a "Contact Us" form for direct communication.
- Users can also read blogs related to disaster awareness and preparedness on the platform.

5. **Login and Registration**:

- Users are provided with a login page to securely access information, and an option for

registration is available to enter the details of affected people, further enabling information management and accessibility for users.

Applicability

The *UNITY Disaster Relief System* project has broad applicability in emergency management and disaster response scenarios. It serves as a vital resource during natural disasters by offering a systematic approach to locate and manage data on affected individuals. The platform can be particularly valuable for:

1. **Government and Relief Organizations**:

- The system can assist government agencies and humanitarian organizations in efficiently managing resources and locating individuals during large-scale emergencies.
- The tracking capabilities enable agencies to have real-time data on the population in various relief camps, helping them deploy resources more effectively.

2. **Families and Relatives of Affected Individuals**:

- Families looking for their loved ones can utilize the platform to check whether their relatives have been transferred to specific hospitals or camps.
- The facial recognition and image processing functionalities allow families to upload images of missing persons and receive updates on their status if they are located in any connected facility.

3. **Disaster Preparedness and Community Awareness**:

- By offering resources such as blogs on disaster preparedness, the platform also educates users on how to respond to emergencies.
- The knowledge-sharing aspect helps create a more resilient community, informed about steps to take during disasters.

This disaster relief system provides a streamlined way to connect, inform, and manage information during crisis situations, thereby making it an invaluable tool for affected individuals, response teams, and authorities. The system's integration of advanced technologies like machine learning and facial recognition further expands its utility, paving the way for more organized and effective disaster response solutions.

CHAPTER 2

REQUIREMENTS AND ANALYSIS

2.1 Existing System

ReliefWeb is an essential platform that provides real-time information and resources related to humanitarian crises, helping organizations and individuals respond effectively to emergencies. While it offers a wealth of data, situation reports, and job opportunities, it also has some disadvantages. One limitation is the potential for information overload, as users may find it challenging to sift through the vast amount of data to find relevant insights. Additionally, the site relies on contributions from various organizations, which can lead to inconsistencies in reporting quality and timeliness. Furthermore, while ReliefWeb aims to be comprehensive, it may not cover all local or less-publicized crises, potentially leaving gaps in the information available to responders. These factors can hinder the platform's overall effectiveness in guiding timely humanitarian action.

2.2 Proposed System - Gap Identification

The UNITY Disaster Relief System is a comprehensive solution designed to assist families and relief organizations in managing disaster scenarios, especially in situations like floods and landslides. The platform offers tools for identifying affected individuals, tracking resources, and managing relief camps, making it an invaluable tool for disaster response. This system addresses existing gaps in disaster relief technology by offering an accessible, customizable, and highly efficient solution.

Key Features of the Proposed System

- **Identification and Tracking of Individuals**

Facial Recognition: The UNITY system utilizes advanced facial recognition technology to identify individuals within disaster zones or relief camps.

Image-Based Search: Families can upload photos of missing individuals, allowing the system to match these with its database, providing real-time information on the location and status of affected people.

- Camp Management and Resource Distribution

Population Tracking: The system tracks the number of individuals in various relief areas, ensuring accurate data on camp population.

Resource Management: Facilitates effective distribution of critical resources such as food, water, and medical supplies based on current camp needs.

Security Monitoring: Integrates surveillance and real-time monitoring to enhance the safety and security of camp residents.

- User-Friendly Interface

The system includes an intuitive interface with various options for users to navigate easily. It provides an index page, detailed camp listings, registration options, and blogs about disaster management for family members and relief workers alike.

- Camera

In the UNITY Disaster Relief platform, the camera plays a crucial role in identifying individuals affected by natural disasters. By allowing the platform to capture real-time images of people, the camera helps streamline the identification process for those needing assistance. Using the captured image, the platform's machine learning algorithms can compare the photo against existing records in the database, helping to verify the person's identity or find missing persons in affected areas. This camera-based identification system ensures that aid reaches those in need more efficiently, supporting faster and more accurate disaster response and relief efforts.

- Cost-Effective Solution

The UNITY system is designed to minimize initial and ongoing costs by eliminating the need for extensive cloud-based storage through local storage options and by reducing dependency on third-party services. This cost-effectiveness makes it accessible to a wider range of communities and relief organizations.

2.3 Feasibility Study

During the system analysis the feasibility study of the proposed system is to be carried out to ensure that the proposed system is not a burned to a company. A feasibility analysis evaluates the project's potential for the success; therefore,perceived objectivity is an essential factor in the credibility of the study for potential investors and lending institutions.

When feasibility study is carried out there are four main areas of consideration

- Technical Feasibility
- Financial Feasibility
- Ethical Feasibility

widespread acceptance, behavior studies and user feedback will be conducted pre- deployment. This approach confirms the system's viability, as it meets functional and economic feasibility criteria, creating a supportive user experience that aligns with their needs in disaster scenarios.

Project Overview: UNITY Disaster Relief is a web-based platform designed to assist in the identification and support of individuals affected by natural disasters. The platform integrates machine learning (ML), database management, and web technologies, including HTML, to provide an effective and user-friendly solution for disaster relief agencies and affected individuals. The primary functions include identifying affected persons, connecting them with resources, and streamlining communication for emergency response teams.

CHAPTER 3

SYSTEM SPECIFICATION

3.1.1 **Software Specifications**

Operating System: Windows

Language: HTML, CSS, Python, Django, Machine Learning

3.1.2 Hardware Specification

Memory : 8GB RAM and above
Processor : Intel Core i5 and above

3.2 **Functional Specification and User Characteristics**

Functional specifications:

1. Identification and Monitoring

Facial Recognition and Image Processing: The UNITY system integrates facial recognition and image processing technology to identify individuals in disaster zones or camps. This feature enables families to locate missing members by uploading photos, which the system uses to match with the database, offering real-time information on the individuals' status and location.

2. Camp Monitoring:

In UNITY Disaster Relief, camp updating refers to the process of recording and maintaining up-to-date information on individuals newly admitted to relief camps. This is crucial for managing resources, tracking occupancy, and providing effective assistance. When people arrive at relief camps, their information—such as name, age, medical needs, and contact information—is documented and stored in the system's database. The following steps outline how camp updating can be efficiently managed in UNITY Disaster Relief:.

3. Real-Time Alerts and Notifications

The system generates real-time alerts for families and relief organizations, informing them of updates regarding individual identification, resource needs, and changes in camp status. This feature includes visual and audio notifications, ensuring stakeholders are immediately informed.

4. Remote Access and Control

UNITY allows users to access and control the system remotely. Families can monitor the status of loved ones from any location, while relief workers can manage camp resources and receive real-time data on camp population, resource allocation, and safety.

5. AI and Machine Learning

Leveraging machine learning algorithms, UNITY can analyze visual data to recognize faces, predict resource

needs, and automatically detect irregularities or emergencies within the camps. This feature enhances accuracy and reduces response time, supporting proactive disaster management.

6. User characteristics

Concerned Families and Relatives: The system is ideal for families and relatives seeking real- time updates and identification of loved ones affected by disasters, providing them with a reliable source of information and location tracking.

7. Tech-Savvy Relief Workers and Volunteers

UNITY is intended for users comfortable with digital tools, allowing relief workers to access data, monitor camp resources, and make informed decisions through mobile and desktop interfaces.

8. Budget-Conscious NGOs and Government Agencies:
With a cost-effective design, UNITY appeals to organizations that require affordable yet comprehensive disaster management solutions, reducing dependence on third-party services while optimizing on-the-ground resource distribution.

9. Customizable for Organizational Needs:

The system is suitable for users who require a flexible solution, enabling easy modification of system components to fit specific disaster scenarios, be it urban, rural, or remote environments.

CHAPTER 4

INTEGRATED DEVELOPMENT ENVIRONMENT

4.1 **Integrated Development Environment**

In developing UNITY DISASTER, an integrated development environment (IDE) is essential for building and testing various functionalities, such as image processing and machine learning, to locate and manage affected individuals. The IDE simplifies code development and debugging, streamlining efforts to program key modules like facial recognition, data processing, and real-time tracking. Here, we'll explore how an IDE aids in developing each feature of UNITY DISASTER.

Installation and Setup

To begin working on UNITY DISASTER, the IDE must first be installed. This process may differ based on the operating system but generally involves selecting the right modules and libraries for facial recognition, data management, and machine learning. After installation, setup involves linking the IDE to necessary datasets and configuring tools for effective code integration.

Code Editor

The IDE's code editor supports writing scripts to perform tasks like tracking affected individuals and managing data on relief camps. It includes features like syntax highlighting, auto-indentation, and error highlighting, making it easier to develop complex functionalities such as data visualization and real-time tracking. These features are crucial for maintaining the accuracy and readability of the project's codebase.

Serial Monitor and Data Visualization

For UNITY DISASTER, data communication is key. The Serial Monitor or similar debugging tool allows developers to view real-time data transferred between the IDE and

data storage, essential for monitoring live updates. It displays outputs such as camp population and supply statuses and helps developers test the accuracy of image recognition data transmitted from affected areas.

Data Upload and Verification

The IDE's uploader feature is used to deploy code efficiently, connecting to hardware (e.g., IoT modules) that communicates with UNITY DISASTER's database. Developers can verify code before deployment to catch errors early, which is critical for ensuring reliable data transmission during real-time disaster response.

Customization

The IDE's customizable interface allows developers to personalize the editor settings to improve productivity. It enables adjustments to visual elements, keyboard shortcuts, and default libraries, all of which help streamline workflow when handling a large-scale project like UNITY DISASTER.

Advanced Features and Debugging

Advanced users can leverage the IDE's compatibility with other code editors (e.g., Visual Studio Code) for developing machine learning models outside the standard IDE environment. Debugging tools, like breakpoints, aid in troubleshooting algorithms, especially for critical tasks like image processing accuracy. The IDE's command-line interface (CLI) enables fast deployment and remote control, beneficial in high-stakes, real-time scenarios.

4.2 **Components of UNITY DISASTER IDE**

1. Code Editor

The Code Editor is used to write scripts that facilitate individual identification and tracking through image recognition and data processing. It supports auto-completion, indentation, and formatting, ensuring clean code for essential tasks like database updates and real-time data analysis.

2. Library Manager

The Library Manager is used to manage machine learning and computer vision libraries that power facial recognition in UNITY DISASTER. It allows for the seamless integration of libraries, such as TensorFlow and OpenCV, which enhance the accuracy of individual identification within disaster zones.

3. Sketchbook (Project Repository)

The Sketchbook serves as a project repository for all UNITY DISASTER modules. It provides easy access to different functionalities, such as camp management and image processing scripts, ensuring streamlined organization of each project component.

4. Examples and Templates

The IDE includes pre-written templates that developers can use as a starting point for UNITY DISASTER's

unique functionalities. These examples help streamline development for complex tasks, such as image processing and data tracking, by providing a foundation to build upon.

5. Preferences and Customization

The Preferences menu allows developers to configure IDE settings, such as adjusting the font size and setting project-specific defaults for the UNITY DISASTER system. Customizations improve workflow efficiency and help personalize the IDE for better usability during disaster response projects.

4.3 Library Used

NumPy (numpy): NumPy is a library used for numerical and array operations. In this code, it is mainly used to handle arrays, perform mathematical operations, and manage the cosine similarity calculations on embeddings.

1. TensorFlow (tensorflow): TensorFlow is an open-source deep learning library developed by Google. In this script:
2. TensorFlow provides the pre-trained ResNet50 model, which extracts feature embeddings from images.
3. Keras, which is part of TensorFlow, offers utilities such as preprocess_input to preprocess images for compatibility with ResNet50 and layers for pooling.
4. The ResNet50 model is configured to exclude the top layer, meaning it is used purely for feature extraction, not classification.

scikit-learn (sklearn):

1. Cosine Similarity (pairwise.cosine_similarity): This function calculates the cosine similarity between embeddings, which is used to identify images with similar content.
2. PCA (decomposition.PCA): Principal Component Analysis (PCA) is a dimensionality reduction technique that reduces the noise and complexity in high-dimensional data, making similarity calculations more efficient and reducing storage requirements.

Pillow (PIL): The Pillow library is a powerful image-processing library in Python. Here, it is used to:

1. Load and manipulate images from the dataset.
2. Convert images to RGB format and resize them for compatibility with the ResNet50 model.

3. Display images for visual confirmation of similarity results.

Matplotlib (matplotlib.pyplot): Matplotlib is a plotting library used to create visualizations. In this code, it is used to display the input image and the most similar image found in the dataset side by side.

OS Library (os): The OS module provides functions for interacting with the file system. Here, it is used to:

1. Access the dataset directory and list image file paths.

2. Construct full file paths for each image.program

CHAPTER 5

Code

5.1 code for Unity Disaster System

➢ Code for image processing import numpy as np
import tensorflow as tf
from tensorflow.keras.applications.resnet50 import preprocess_input from sklearn.metrics.pairwise import cosine_similarity
from sklearn.decomposition import PCA from PIL import Image
import os
import matplotlib.pyplot as plt

# Function to load and preprocess images using TensorFlow's preprocess_input for ResNet50 def load_and_preprocess_image(image_path, target_size=(224, 224)):
image = Image.open(image_path).convert('RGB') # Ensure RGB format image = image.resize(target_size)
image_array = np.array(image)
image_array = preprocess_input(image_array) # Preprocess input for ResNet return np.expand_dims(image_array, axis=0) # Add batch dimension

# Load a pre-trained ResNet50 model with added max pooling to enrich the feature representation
base_model = tf.keras.applications.ResNet50(weights='imagenet', include_top=False) global_avg_pooling =

```
tf.keras.layers.GlobalAveragePooling2D() global_max_pooling = tf.keras.layers.GlobalMaxPooling2D()

def get_image_embedding(image_path):
preprocessed_image = load_and_preprocess_image(image_path) features =
base_model.predict(preprocessed_image, verbose=0)
# Combine average and max pool embeddings
avg_pool_embedding = global_avg_pooling(features).numpy().flatten() max_pool_embedding =
global_max_pooling(features).numpy().flatten()

# Concatenate both embeddings
combined_embedding = np.concatenate([avg_pool_embedding, max_pool_embedding])

# Normalize embedding
combined_embedding /= np.linalg.norm(combined_embedding) return combined_embedding

# Set the dataset path
dataset_path = r'E:\RELIEF_PROJECT_VSCODE\unityrelief\templates\gender_dataset\men'

# Gather image paths
image_paths = [os.path.join(dataset_path, fname) for fname in os.listdir(dataset_path)
if fname.lower().endswith(('.png', '.jpg', '.jpeg'))]

# Load embeddings embeddings = [] image_data = {}

for image_path in image_paths:
embedding = get_image_embedding(image_path) embeddings.append(embedding)
image_data[image_path] = {"metadata": "sample metadata"} embeddings = np.array(embeddings)

# Apply PCA to reduce noise in embeddings pca = PCA(n_components=100)
reduced_embeddings = pca.fit_transform(embeddings)
# Function to find similar images with cosine and percentile-based threshold def
find_similar_image(input_image_path, reduced_embeddings, image_paths, percentile=90):
input_embedding = get_image_embedding(input_image_path) input_embedding_reduced =
pca.transform([input_embedding])
```

```
# Calculate cosine similarity
cosine_similarities = cosine_similarity(input_embedding_reduced, reduced_embeddings)[0]

# Find the best match by cosine similarity best_match_index = np.argmax(cosine_similarities)
best_match_image_path = image_paths[best_match_index] best_match_score =
cosine_similarities[best_match_index]

# Calculate percentile-based threshold
similarity_threshold = np.percentile(cosine_similarities, percentile)

if best_match_score > similarity_threshold:
return best_match_image_path, best_match_score else:
return None, None

# Define the input image path
input_image_path = 'E:\\RELIEF_PROJECT_VSCODE\\unityrelief\\templates\\face.jpg'

# Run similarity search try:

match_image_path, similarity_score = find_similar_image(input_image_path, reduced_embeddings,
image_paths)
if match_image_path:
print(f"Found similar image: {match_image_path} with score {similarity_score}") print("Associated data:",
image_data[match_image_path])

# Display input and matched images input_image = Image.open(input_image_path)
matched_image = Image.open(match_image_path) plt.figure(figsize=(10, 5))

# Plot input image plt.subplot(1, 2, 1) plt.title("Input Image") plt.imshow(input_image) plt.axis('off')

# Plot matched image plt.subplot(1, 2, 2)
plt.title("Most Similar Image in Dataset") plt.imshow(matched_image)  plt.axis('off')

plt.show() else:
print("No similar image found.") except Exception as e:
print("Error:", e)
```

➢ index.html

```
{% load static %}

<!DOCTYPE html>
<html>
<head>
<title> Unity Relief | Home </title>
<!-- Web Fonts -->
<link rel="stylesheet"
href="//fonts.googleapis.com/css?family=Open+Sans:400,300,600&amp;subset=cyrillic,latin ">

<!-- CSS Global Compulsory -->
<link rel="stylesheet" href="{% static 'plugins/bootstrap/css/bootstrap.min.css' %}">
<link rel="stylesheet" href="{% static 'css/style.css' %}">

<!-- CSS Header and Footer -->
<link rel="stylesheet" href="{% static 'css/header.css' %}">
<link rel="stylesheet" href="{% static 'css/footer.css' %}">

                <!-- CSS Implementing Plugins -->
                <link rel="stylesheet" href="{% static 'plugins/line-icons-pro/styles.css'
%}">
                <link rel="stylesheet" href="{% static 'plugins/line-icons/line-icons.css'
%}">
                <link rel="stylesheet" href="{% static 'plugins/font-awesome/css/font-
```

```
awesome.min.css' %}">

<!-- CSS Customization -->
<link rel="stylesheet" href="{% static 'css/custom.css' %}">




</head>

<body>
<div class="wrapper">
<!--=== Header v1 ===-->
<div class="header-v1">
<!-- Topbar -->
<div class="topbar-v1">
<div class="container">
<div class="row">
<div class="col-md-6">
<ul class="list-inline top-v1-contacts">
<li>
<i class="fa fa-envelope"></i> Email: unityrelief@gmail.com
</li>
<li>
<i class="fa fa-phone"></i> Contact no : 6282645889
</li>
</ul>
</div>
</div>
</div>
</div>

<!-- End Topbar -->

<!-- Navbar -->
<div class="navbar mega-menu" role="navigation">
<div class="container">
```

```
<!-- Brand and toggle get grouped for better mobile display -->
<div class="res-container">
<button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar- responsive-collapse">
<span class="sr-only">Toggle navigation</span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
</button>

<div class="navbar-brand">
<a href="index.html">
<img src="{% static 'img/logo/unity_white.jpg'%}" alt="Logo">
</a>
</div>
</div><!--/end responsive container-->

<!-- Collect the nav links, forms, and other content for toggling -->
<div class="collapse navbar-collapse navbar-responsive-collapse">
<div class="res-container">
<ul class="nav navbar-nav">

<li class="mega-menu-fullwidth">
<a href="/register" >

➢ REGISTRATION
</a>
</li>
<li class="mega-menu-fullwidth">
<a href="/hospitaladmitted" >

➢ HOSPITAL
</a>
</li>
<li class="mega-menu-fullwidth">
<a href="/similarity" >
```

➢ SEARCH

<li class="mega-menu-fullwidth">

➢ RELIEF CAMPS

<li class="mega-menu-fullwidth">

➢ BLOGS

<li class="mega-menu-fullwidth">

➢ CONTACT US

<li class="mega-menu-fullwidth">

 ABOUT US

<li class="mega-menu-fullwidth">

 LOGIN


```
</div>
</div>
</div>
</div>
<!-- End Navbar -->



<!-- Slider -->
<div id="slide">
<div class="slideshow-container">
<div class="mySlides fade"> <img src="{% static 'img/landslide_1.jpg' %}" alt="Slider1"
style="width:100%"> </div>
<div class="mySlides fade"> <img src="{% static 'img/index_2.webp' %}" alt="Slider2"
style="width:100%"> </div>
<div class="mySlides fade"> <img src="{% static 'img/index_3.jpg' %}" alt="Slider3" style="width:100%">
</div>
<a class="prev" onclick="plusSlides(-1)">&#10094;</a> <a class="next"
onclick="plusSlides(1)">&#10095;</a>
</div>
<br>
<div style="text-align:center">
<span class="dot" onclick="currentSlide(1)"></span>
<span class="dot" onclick="currentSlide(2)"></span>
<span class="dot" onclick="currentSlide(3)"></span>
<span class="dot" onclick="currentSlide(4)"></span>
</div>
</div>



<!-- End of Slider -->

<!--=== Welcome to Unity===-->
<div class="container content-md welcomeSection">
<div class="row section1">
```

```
<div class="col-md-6" style="margin-bottom: 40px;" style="border:2px solid black background: green;">
<h2 class="title-v2">WELCOME TO <span style="color: #72c02c;"> UNITY </span></h2>
<p style="text-align: justify; font-size: 14px;">Welcome to our webpage, "Relief and Recovery: Flood and Landslide Survivors," dedicated to storing and sharing data about individuals displaced to relief camps and hospitals due to these devastating events.  </p>

<a href="about.html" class="btn-u btn-brd btn-brd-hover">Read More</a>
</div>
<div class="col-md-6 overflow-h">
<img style="border-radius: 50px; margin-left: 30px;" src="{% static 'img/index_landslide.jpg' %}" alt="">
</div>
</div>
</div>
<!--=== End About Us ===--


<!-- End Content Part -->
<div class="footer-v1">
<div class="footer">
<div class="container">
<div class="row">
<!-- About -->
<div class="col-md-3 " style="margin-bottom: 40px;">
<a href="/index"><img id="logo-footer" class="footer-logo" src="{% static 'img/logo/unity_white.jpg' %}" alt=""></a>
<p> "Relief and Recovery: Flood and Landslide Survivors," dedicated to storing and sharing data about individuals displaced to relief camps and hospitals due to these devastating events.
</p>

</div><!--/col-md-3-->
<!-- End About -->

<!-- Latest -->
<div class="col-md-3 " style="margin-bottom: 40px;">
<div class="posts">
<div class="headline"><h2>Latest Posts</h2></div>
```

```
<ul class="list-unstyled latest-list">

<li>
<a href="/blog">Latest Images</a>

</li>
</ul>
</div>
</div><!--/col-md-3-->
<!-- End Latest -->

<!-- Link List -->
<div class="col-md-3 " style="margin-bottom: 40px;">
<div class="headline"><h2>Useful Links</h2></div>
<ul class="list-unstyled link-list">
<li><a href="/about">About us</a><i class="fa fa-angle-right"></i></li>
<li><a href="/contact">Contact us</a><i class="fa fa-angle-right"></i></li>

</ul>
</div><!--/col-md-3-->
<!-- End Link List -->

<!-- Address -->
<div class="col-md-3 map-img " style="margin-bottom: 40px;">
<div class="headline"><h2>Contact Us</h2></div>
<address class="" style="margin-bottom: 40px;"> Unity Relief <br />
Kottayam, IN <br /> Phone: 6282645889 <br />
Email: unityrelief@gmail.com
</address>
</div><!--/col-md-3-->
<!-- End Address -->
</div>
</div>
</div>
```

```
<!--=== End Footer ===-->
</div><!--/wrapper-->
</body>
</html>
```

➢ Style.css

```
/*Import Global Compulsory CSS Files*/ @import url(custom.css);

/*Import CSS Plugins*/

@import url(plugins/style-switcher.css);

* {
border-radius: 0;
}

body { color: #333;
font-size: 13px; line-height: 1.6;
}

body.header-fixed-space { padding-top: 94px;
}
body.header-fixed-space-v2 { padding-top: 135px;
}

@media (max-width: 991px) { body.header-fixed-space, body.header-fixed-space-v2 { padding-top: 0;
}
}

p,
li, li a,
label { color: #555;
}

a {
color: #72c02c;
text-decoration: none;
}

a, a:focus, a:hover, a:active, button,
button:hover { outline: 0 !important;
}

a:focus {
text-decoration: none;
```

```
}
a:hover {  color: #72c02c;
text-decoration: underline;
}

/*Boxed Layout
*/      ----------.
.wrapper { background: #fff;
}

.boxed-layout { padding: 0;
box-shadow: 0 0 5px #ccc; margin-left: auto !important; margin-right: auto !important;
background: url(../img/patterns/15.png) repeat;
}

/*Hiding arrows for select field in IE*/ select::-ms-expand {
display: none;
}
```

5.2 code detailed

Step-by-Step Explanation:

1. Libraries and Model Setup

Import necessary libraries for numerical operations (numpy), machine learning models (tensorflow for ResNet50), image processing (PIL), and visualization (matplotlib).

Load a pre-trained ResNet50 model (without the top layer) to use as a feature extractor. This model processes images and produces embeddings based on the content of the image, which we later use for comparison.

2. Image Embedding Extraction with ResNet50 Function: get_image_embedding:

Load and preprocess the image: load_and_preprocess_image() resizes images to a standard size and preprocesses them using ResNet50's requirements.

Feature Extraction: The base_model (ResNet50 without the final classification layer) is used to extract image features.

Pooling Layers:

Global Average Pooling and Global Max Pooling layers are applied to the features, capturing both average and maximum feature values for a richer representation.

Embedding Normalization: Concatenate average and max-pooled embeddings, then normalize to create a final embedding vector for each image.

3. Dataset Loading and Embedding Calculation

Dataset Path: Specifies the folder path where all images are stored.

Gathering Image Paths: Collects paths of images in the dataset folder that have common image extensions (.png, .jpg, .jpeg).

Embeddings and Metadata Storage:

Loop through each image, calculate its embedding, and store it in a list (embeddings). Additionally, store metadata for each image path in image_data.

4. Dimensionality Reduction with PCA

PCA Initialization: Apply PCA to reduce the embedding vectors to 100 dimensions (n_components=100), which helps in removing noise and reducing computation costs.

Embedding Transformation: Transforms the original embeddings to their reduced form (reduced_embeddings).

5. Similarity Search Function Function: find_similar_image:

Calculate the Embedding of Input Image: Computes the embedding for the input image and reduces its dimensionality with PCA.

Cosine Similarity: Computes cosine similarity between the input image embedding and each embedding in the dataset.

Best Match and Threshold:

Finds the image with the highest cosine similarity score (best_match_score) and sets a threshold based on the percentile parameter (e.g., the 90th percentile).

If the highest score is above this threshold, the function returns the best match image path and similarity score; otherwise, it returns None.

6. Running the Similarity Search

The code defines input_image_path, and then attempts to find and display the most similar image:

If a similar image is found, it prints the path and similarity score and displays both the input and matched images.

If no image meets the similarity threshold, it notifies the user.

5.3 **Testing Approaches**

Testing is a crucial aspect of developing a disaster management system, as it ensures that the system

functions correctly, meets the required specifications, and operates reliably in various scenarios. This section explores various testing approaches that can be employed to validate and verify the effectiveness of the Unity Disaster Relief System.

Unit Testing:

Unit testing focuses on verifying individual components of the system, such as different modules or functions. For a disaster management system, this involves testing individual modules, such as the facial recognition model, database access functions, or specific frontend

components. Test cases can be created to check the accuracy of image processing, the integrity of data storage, and the responsiveness of individual components. Unit testing helps identify and fix issues within specific parts of the system.

Integration Testing:

Integration testing involves testing the interaction and communication between different modules of the system. This ensures that each unit, when combined, works seamlessly together. For the Unity Disaster Relief System, integration testing could include verifying how the facial recognition model integrates with the Django backend, how data flows from the frontend to the database, and how images are processed and returned with relevant information. This testing helps identify any inconsistencies or problems when components interact.

Functional Testing:

Functional testing verifies if the overall system functionality meets the desired requirements. For the Unity Disaster Relief System, this involves testing each major feature, such as image upload, recognition, and matching, as well as reporting and locating individuals. Testing ensures the system provides accurate information about individuals, including whether they are in a camp, hospital, or missing. This approach ensures that the system performs as expected and delivers the intended functionality.

Performance Testing :

Performance testing assesses how well the Unity Disaster Relief System operates under different conditions. This includes testing response time, processing speed, and resource usage. For instance, performance testing could measure how quickly the system identifies individuals and retrieves relevant data. Additionally, it could assess how the system handles a large number of simultaneous requests from multiple users. This helps identify potential bottlenecks or inefficiencies in the system.

Security Testing :

Security testing evaluates the system's resilience against security threats. This includes testing for vulnerabilities such as unauthorized data access, data breaches, or tampering attempts. For the Unity Disaster Relief System, you can test if sensitive data is encrypted, if user data is secured, and if the system resists unauthorized access attempts. Security testing ensures that the system safeguards user privacy and sensitive information.

Usability Testing :

Usability testing evaluates the system's user experience and interface. It aims to ensure that the system is intuitive, accessible, and easy to use, especially in high-stress situations.
Usability testing could involve assessing how easily users can search for missing individuals, understand results, and interact with features like registration and report generation. This helps identify usability issues and improve areas to enhance user satisfaction and system accessibility.

Compatibility Testing :

Compatibility testing verifies the system's compatibility with different devices, platforms, or configurations. This includes testing the system's web interface on various web browsers, operating systems, and devices. For example, you can test if the Unity Disaster Relief System works seamlessly across desktop and mobile platforms. Compatibility testing ensures the system interacts effectively with the intended target environments.

Real-World Testing:

Real-world testing involves testing the system in an environment similar to its intended deployment scenario. For the Unity Disaster Relief System, this could involve testing in conditions that mimic actual disaster settings, such as different lighting conditions for image recognition, varying internet connectivity, and real user interactions. Real-world testing helps uncover issues that may arise during practical deployment and ensures reliable performance.

One key component of real-world testing is simulating natural disasters, such as earthquakes, floods, and hurricanes, in collaboration with local authorities and relief agencies. These simulations provide insights into the platform's ability to deliver accurate situational awareness and help decision-makers prioritize critical interventions. Furthermore, real-world testing helps identify challenges related to system scalability, user interface design, and

interoperability with other disaster response tools, allowing for continuous refinement and optimization of the technology.

5.4 **Implementation Approaches**

Implementing the Unity Disaster Relief System involves several steps and considerations. Key aspects include software setup, image processing and machine learning integration, backend development, frontend design, connectivity, and user interface design.

Software Setup:
The first step is to set up a development environment that includes all necessary tools and libraries. Since this project is developed in VS Code, configure it with the Django framework for backend functionality and the necessary libraries for image recognition, such as TensorFlow or OpenCV. Ensure that all dependencies are installed, and test your setup to confirm compatibility.

Image Recognition and Processing:

Machine Learning Model: Integrate a pre-trained model (like ResNet50) or a custom-trained model for facial recognition and image analysis. This model will be used to detect and match faces to help locate individuals affected by disasters.

Embedding Generation and Comparison: Write code to process each uploaded image, generating an embedding (a unique feature representation of the face) and comparing it with a database of stored embeddings to find matches. Use cosine similarity and PCA for accurate matching and efficient comparison.

Model Training and Optimization: Continuously refine the model by training on updated datasets and optimizing for better accuracy. This ensures that the system reliably identifies individuals in different lighting and environmental conditions.

Applications of Image Recognition in UNITY Disaster Relief

- Damage Assessment: After a disaster, the platform can automatically analyze satellite or aerial images to assess the extent of the damage. By identifying destroyed buildings, blocked roads, or infrastructure failures, the system provides responders with vital information to help prioritize rescue and repair efforts.

- Infrastructure Monitoring: By continuously processing images of critical infrastructure, the system can detect early signs of damage or wear, helping to prevent further breakdowns. For example, images of bridges and roads can be analyzed for structural cracks or flooding, which would be crucial for diverting traffic or sending repair teams.

- Search and Rescue Operations: Image recognition can be used to analyze imagery from drones or high-altitude satellites to locate survivors or identify safe routes for search teams. The system can spot signs of human activity, such as large gatherings or movement patterns, which could indicate the location of displaced people in need of aid.

Backend Development:

Develop the backend using Django, which will manage the database, handle API requests, and control application logic.

Database Management: Set up a database (e.g., PostgreSQL or SQLite) to store data related to individuals, relief camps, and image embeddings. Ensure that the database is optimized for quick access to individual records and efficient handling of queries.

API Creation: Create RESTful APIs to allow data to be securely accessed, updated, and processed by both the frontend and image recognition components. APIs should handle user authentication, image uploads, and data retrieval requests.

Data Processing and Decision Making:
Implement logic in the backend to handle image uploads, process recognition requests, and manage individual identification data. For instance, when an image is uploaded, the backend initiates the image recognition process, searches for matches, and returns results indicating if the individual is located in a relief camp or hospital.

Decision Rules: Define decision-making rules that evaluate if an individual is matched based on similarity scores and thresholds. These rules can help determine whether the system should confirm a match or prompt the user to verify additional details.

Connectivity and Remote Access:

Since the system is web-based, connectivity is essential for remote access. Host the Django server on a cloud platform (such as AWS or Heroku) to ensure reliable access for users and disaster relief personnel.

Notifications: Integrate notifications or alerts (e.g., via email or SMS APIs) that can notify users when there is a status update about an individual. This feature could be useful in keeping families informed.

User Interface (UI) Design:

Design the UI using HTML, CSS, and JavaScript to create an accessible, user-friendly web interface. Key components may include:

Homepage: Provide an overview of the system's functionality and instructions.

Image Upload Feature: Enable users to upload images for identification, and ensure that the interface guides users through the process.

Relief Camp Information: Display details about available relief camps, locations, and resources.

Search and Notification: Allow users to search for individuals and receive notifications if a match is found.

Use responsive design principles to ensure the interface is accessible on different devices, including mobile.

Scalability and Optimization:

Design the system with scalability in mind to handle large numbers of images and requests. Use PCA to reduce image embedding dimensions, helping the system manage large datasets and process images quickly.

Optimize server configurations and database queries to maintain quick response times even under high user loads.

Testing and Iteration:

Conduct thorough testing at each stage to ensure system reliability and accuracy:

Unit Testing: Test individual modules, such as image upload and recognition functions, to verify correctness.

Integration Testing: Test interactions between the frontend, backend, and machine learning components to ensure smooth data flow.

Functional Testing: Verify that core features, like facial recognition and notification systems, operate as intended.

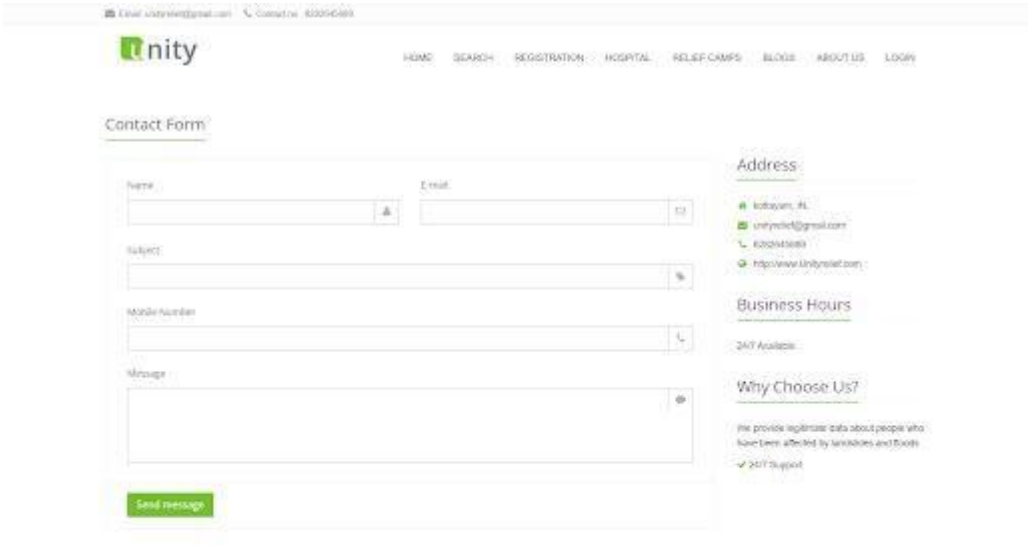
Usability Testing: Evaluate the user interface to ensure it is intuitive and accessible, especially for users in high-stress situations.

Document test results and iterate on feedback to continuously improve the system’s performance, security, and usability.

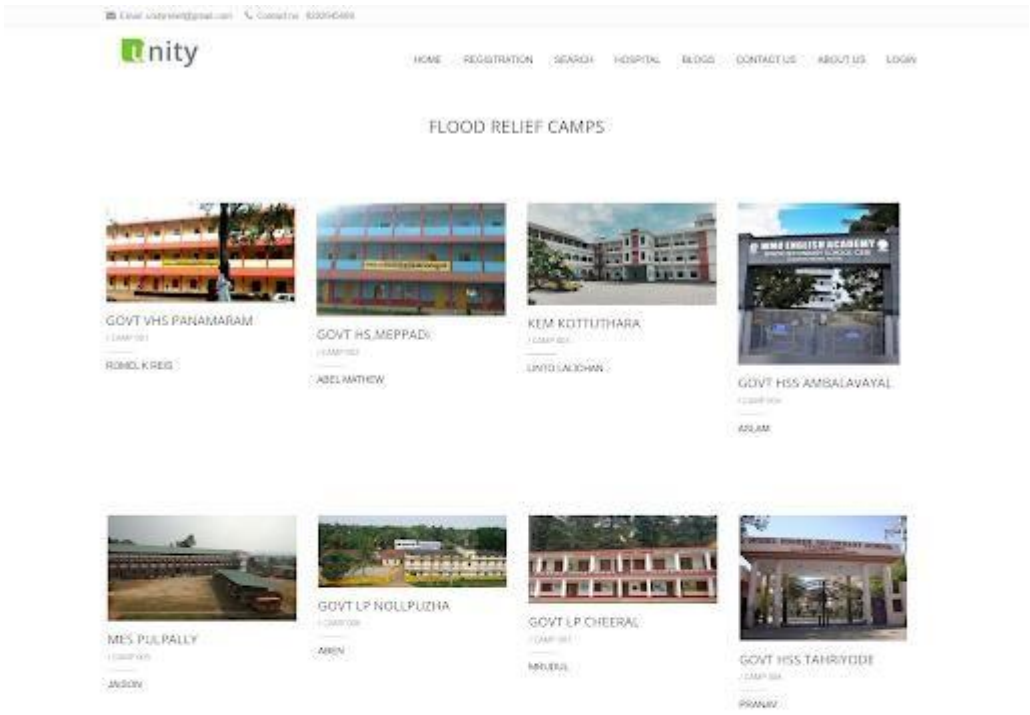
OUTPUT


nity
REGISTRATION
SEARCH
LOGIN
HOSPITAL
HOME
RELIEF CAMPS
BLOGS
CONTACT US
ABOUT UNITY RELIEF
FLOOD

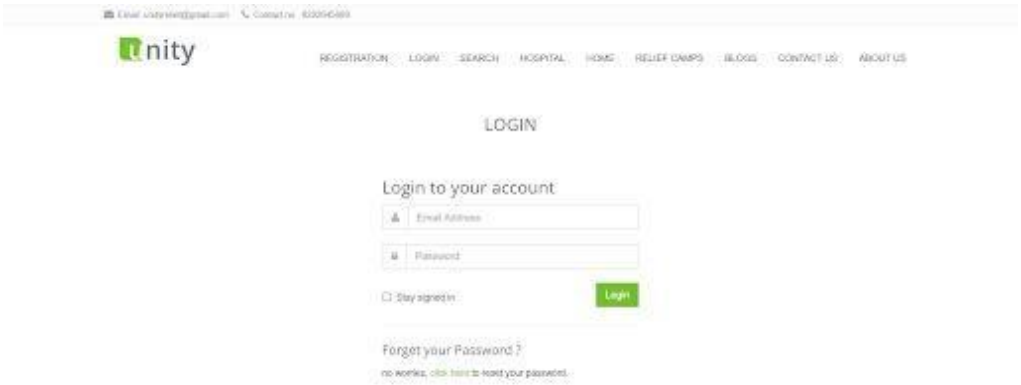

nity
Latest Posts
Useful Links
Contact Us
Latest Images
Terms and Conditions
ABOUT US
CONTACT US


nity
HOME
SEARCH
REGISTRATION
HOSPITAL
RELIEF CAMPS
BLOGS
ABOUT US
LOGIN
Contact Form
Name
E-mail
Subject
Mobile Number
Message
Send message
Address
Business Hours
Why Choose Us?

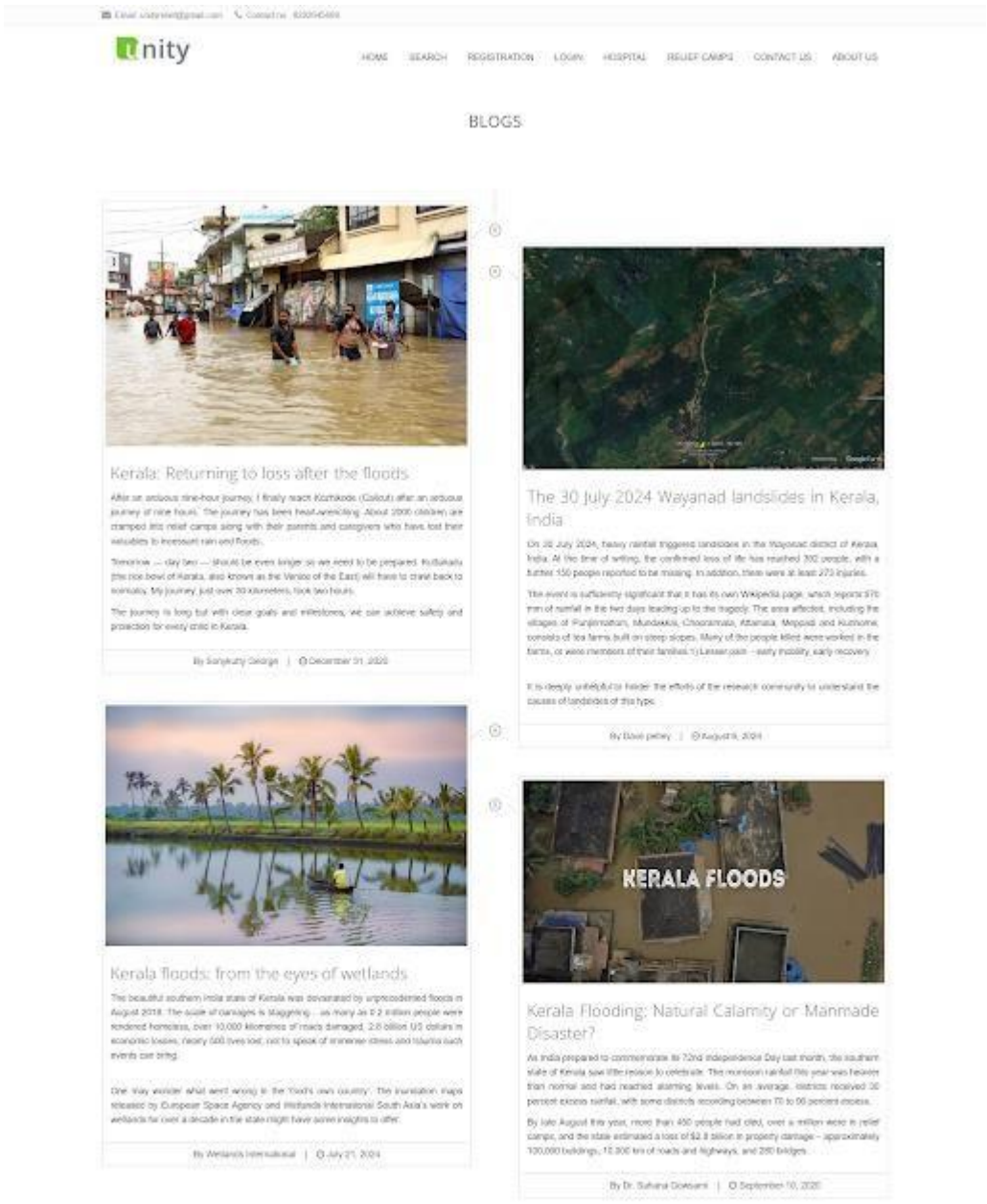

nity
Latest Posts
Useful Links
Contact Us
Latest Images
Terms and Conditions
ABOUT US
CONTACT US


nity
HOME
REGISTRATION
SEARCH
HOSPITAL
BLOGS
CONTACT US
ABOUT US
LOGIN
FLOOD RELIEF CAMPS
GOVT VHS PANAMARAM
GOVT HS,MEPPADI
KEM KOTTUTHARA
GOVT HSS AMBALAVAYAL
MES PULPALLY
GOVT LP NOLLPUZHA
GOVT LP CHEERAL
GOVT HSS TAMIYODE


nity
Latest Posts
Useful Links
Contact Us
Latest Images
ABOUT US
CONTACT US
Unity Relief
Kottayam, IN


nity
REGISTRATION
LOGIN
SEARCH
HOSPITAL
HOME
RELIEF CAMPS
BLOGS
CONTACT US
ABOUT US
LOGIN
Login to your account
Login
Forget your Password ?


nity
Latest Posts
Useful Links
Contact Us
Latest Images
Terms and Conditions
ABOUT US
CONTACT US
Unity Relief
Kottayam, IN


nity
HOME SEARCH REGISTRATION LOGIN HOSPITAL RELIEF CAMPS CONTACT US ABOUT US
BLOGS
Kerala: Returning to loss after the floods
The 30 July 2024 Wayanad landslides in Kerala, India
Kerala floods: from the eyes of wetlands
KERALA FLOODS
Kerala Flooding: Natural Calamity or Manmade Disaster?
nity
Latest Posts
Useful Links
Contact Us
ABOUT US
CONTACT US

CHAPTER 6

CONCLUSIONS

6.1 Introduction

A disaster management system is essential for effective and timely response during natural disasters, providing critical support to affected individuals and families. With advancements in web technologies, it has become increasingly feasible to implement sophisticated systems that manage and disseminate crucial information in real-time. This conclusion summarizes the key aspects of the Unity Disaster Relief System, discussing its benefits, potential applications, and future possibilities for enhancing disaster relief efforts.

The Unity Disaster Relief System combines image recognition technology, data management, and real-time reporting to support efficient disaster response. Through machine learning-based image processing, the system can identify and track individuals across relief camps and hospitals, offering families a way to locate missing persons. By integrating machine learning and web technologies, the Unity Disaster Relief System can help identify and reunite affected people with their families. One of the primary advantages of this system is its ability to provide a centralized platform for collecting, processing, and displaying real-time information about individuals' statuses and locations.

Using web-based technologies like Python (Django), HTML/CSS, and machine learning offers flexibility in

developing this disaster management solution. The Django framework supports robust data handling and API development, while the machine learning model enables accurate facial recognition for identifying individuals. The data management functionality allows the system to store and organize detailed records of each affected individual's location and status. By integrating these components, the Unity Disaster Relief System aims to provide timely information to families, facilitating communication and support for affected individuals.

The benefits of this system extend beyond identification and location tracking. The Unity Disaster Relief System also offers features like remote access, allowing disaster management personnel and families to access the system from any device, whether they are at the scene or remotely located. This accessibility provides a crucial link between relief organizations,

medical staff, and families, creating a streamlined approach to information-sharing and support in times of crisis.

Furthermore, this disaster management system can be expanded to integrate with broader emergency response frameworks, such as relief supply tracking, medical reporting, and public safety updates. For example, the system could be linked with other disaster response technologies to improve coordination among relief agencies. This integration enables a more comprehensive approach to disaster response, providing essential support to those affected and enhancing the overall effectiveness of relief efforts.

The Unity Disaster Relief System serves as a starting point for creating technology-driven disaster relief solutions. Future enhancements could include advanced machine learning algorithms for improved accuracy in individual recognition, expanded data storage capabilities for large-scale disaster response, and the integration of cloud platforms to improve scalability. These improvements could allow for real-time data analytics, advanced notifications, and seamless interaction with other disaster management technologies. As technology continues to advance, systems like Unity Disaster Relief will play an increasingly vital role in safeguarding communities and assisting families in times of need.

6.2 **Limitations of the System**

While web-based disaster management systems offer significant advantages in terms of accessibility and scalability, the Unity Disaster Relief System has its limitations, especially in the context of large-scale deployments and advanced functionality. This section explores some limitations of the system and their potential implications for disaster management.

Machine Learning Limitations: The accuracy of facial recognition depends heavily on the quality of the dataset and the model's training. In situations where lighting, image resolution, or angles vary, the system may face challenges in reliably identifying individuals.

Data Privacy and Security: Handling sensitive data, especially personal information and images, introduces privacy concerns. Although encryption and secure access measures are implemented, data breaches or unauthorized access could compromise user trust and data security.

Scalability and Resource Demands: High-volume traffic or numerous simultaneous image uploads during a large-scale disaster could strain the system's resources, potentially causing delays in processing. Expanding the system's capacity requires additional server resources, which may increase operational costs.

Real-World Conditions and Reliability: Variations in lighting, environmental conditions, and image quality may impact the effectiveness of facial recognition. In real disaster scenarios, photos may not always meet ideal quality standards, which could limit the system's reliability.

Connectivity Constraints: The system relies on internet connectivity to provide remote access to users. In disaster-stricken areas with limited or disrupted connectivity, access to the system may be restricted, affecting families' ability to receive timely updates.

Maintenance and Upgrades: As machine learning models and software libraries evolve, regular maintenance is necessary to ensure compatibility and reliability. However, this may require periodic re-training of the model and updating the software environment.

By understanding and addressing these limitations, the Unity Disaster Relief System can be optimized to provide effective and reliable support in various disaster situations. Implementing safeguards and optimizing system efficiency will further enhance the system's usability and resilience.

6.3 **Future Scope of the Project**

The future scope of the Unity Disaster Relief System holds immense potential for advancements and innovation, offering possibilities for expanded functionalities and improved

support in disaster management efforts. This section discusses potential advancements and applications that could revolutionize the field of disaster response and support.

Integration of Artificial Intelligence (AI): The integration of advanced AI techniques can enhance the system's capabilities. AI-driven algorithms could enable more sophisticated image analysis, individual tracking, and prediction of rescue needs. AI could also help identify patterns in disaster response data, providing insights for future planning.

Cloud Integration and Scalability: By leveraging cloud platforms, the Unity Disaster Relief System could scale to handle large volumes of data, providing support for larger-scale disasters. Cloud integration also enables remote access, data backup, and synchronization across multiple devices, increasing the system's reliability and efficiency.

Enhanced Data Analytics: Advanced analytics tools can be used to gain insights into the data collected, identifying trends and improving resource allocation. Analytics could aid in understanding the distribution of affected individuals and the demand for resources at different relief camps.

Improved User Interfaces: Future iterations could incorporate more intuitive and user-friendly interfaces. For example, a mobile application could provide real-time alerts and easy access to individual data. Interactive dashboards could provide disaster relief personnel with quick, data- driven insights.

Interoperability with Emergency Management Systems: Integrating the system with broader emergency management platforms would allow for more seamless coordination with other disaster response agencies. This interoperability could enable faster response times, better data sharing, and more comprehensive support for relief operations.

Mobile App and Remote Monitoring: Developing a dedicated mobile app would enhance remote access capabilities, allowing families and relief workers to monitor the status of affected individuals in real-time. Features like push notifications, GPS location, and live updates could further improve accessibility and awareness.

CHAPTER 7
REFERENCES

7.1 List of Reference Academic Papers & Articles

Gazi, T., & Rahman, A. (2019). A survey on disaster management using machine learning and data mining techniques. International Journal of Disaster Risk Reduction.

Srivastava, A., & Varshney, S. (2018). Human identification using face recognition - A review. Procedia Computer Science.

Books

Goodfellow, I., et al. (2016). Deep Learning. MIT Press.

Brownlee, J. (2019). Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python.

Web Resources

TensorFlow Documentation: https://www.tensorflow.org/ Django Framework:

https://docs.djangoproject.com/

Kaggle Datasets for Image Recognition: https://www.kaggle.com/

Tutorials & Courses

Coursera - Introduction to TensorFlow for AI, ML, and DL by Andrew Ng.

Reports & Guidelines

UNDRR - Using Machine Learning for Disaster Risk Reduction https://www.undrr.org/ FEMA - National

Incident Management System (NIMS) https://www.fema.gov/