SHIFT FLOW- EMPLOYEE PERFORMANCE APP

A MINI PROJECT REPORT

Submitted by

PRATHIBA.D [211422104347]

PRATHIKSHA.J [211422104348]

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University,Chennai)

OCTOBER 2024

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project **"SHIFT FLOW-EMPLOYEE PERFORMANCE APP"** is the bonafide work of **"PRATHIBA.D [211422104347]** and **PRATHIKSHA.J [211422104348]"**who carried out the project work under my supervision.

SIGNATURE

Dr. L. JABASHEELA, M.E., Ph.D.,
PROFESSOR,
HEAD OF THE DEPARTMENT.

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NAZARETHPET,
POONAMALLE,
CHENNAI, 600123.

SIGNATURE

Dr. M. SHYAMALA DEVI, M.E., Ph.D.
PROFESSOR,

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NAZARETHPET,
POONAMALLE,
CHENNAI, 600123.

Certified that the above candidates were examined in the End Semester Project Viva-Voce Examination held on _____.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent Dr.P.CHINNADURAI, M.A., Ph.D for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors Tmt.C. VIJAYARAJESWARI, Dr.C.SAKTHIKUMAR ,M.E.,Ph.D and

Tmt. SARANYASREE SAKTHIKUMAR, B.E., M.B.A.,Ph.D for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal Dr. K. Mani, M.E., Ph.D for his timely concern and encouragement provided to us through out the course.

We thank the HOD of CSE Department, Dr. L. JABASHEELA,M.E.,Ph.D for the support text ended through out the project.

We would like to thank my Project Guide Dr.M.SHYAMALA DEVI,M.E, P.hd and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

DECLARATION BY THE STUDENTS

We **PRATHIBA.D (211422104347)** and **PRATHIKSHA.J (211422104348)** hereby declare that this project report titled **"SHIFT FLOW-EMPLOYEE PERFORMANCE APP"**, under the guidance of **Dr. SHYAMALA DEVI**is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

1.PRATHIBA.D (211422104347)

2.PRATHIKSHA.J (211422104348)

ABSTRACT

The Employee Performance Digital Dashboard is a powerful tool that can help organizations improve employee performance and achieve their strategic goals.It provides real-time insights into key performance metrics, enabling organizations to identify areas for improvement, track progress towards goals, and make data-driven decisions. The dashboard integrates multiple data sources, offering a centralized platform for tracking KPIs such as productivity, quality, engagement, and attendance.It features customizable views tailored to different roles within the organization, allowing managers and HR professionals to monitor individual and team performance effectively.By presenting critical data in a clear and actionable format, the dashboard helps identify high performers, pinpoint areas for improvement, and support data-driven performance reviews.The dashboard promotes transparency and accountability, fostering a culture of continuous improvement and employee engagement.Implementing this tool can enhance decision-making processes, increase productivity, and drive strategic growth through more effective performance management.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The Shift Flow application is a robust, digital task management and employee performance monitoring tool, designed to address the challenges of traditional workforce management systems. It offers an intuitive platform where managers and employees can collaborate seamlessly to ensure operational efficiency, alignment with organizational goals, and continuous performance tracking.

Through features such as task delegation, real-time notifications, and progress tracking with visual aids like pie and bar charts, Shift Flow streamlines communication between managers and their teams. Employees benefit from clear, actionable instructions while maintaining control over their daily responsibilities, ensuring tasks are completed on time. Managers, in turn, gain valuable insights into team productivity, empowering them to make data-driven decisions and implement continuous performance evaluations.

The proposed Employee Performance Digital Dashboard, a core component of Shift Flow, leverages real-time data and visual analytics to provide a comprehensive view of individual and team performance. By addressing the limitations of outdated performance management systems, this dashboard enhances employee engagement and improves decision-making processes.

In a competitive business environment where digital solutions drive success, Shift Flow emerges as a strategic tool. It fosters transparency, accountability, and long-term loyalty among employees while supporting organizations on their path to sustainable growth. With its focus on user experience and operational efficiency,

Shift Flow serves as a valuable asset for building a culture of productivity and excellence.

1.2 PROBLEM DEFINITION

Task management and communication between employees and managers assume a critical role in augmenting productivity and accountability across diverse industries. In contemporary dynamic work environments, it is imperative for managers to effectively delegate tasks and monitor employee performance in order to ensure streamlined operations and alignment with organizational goals.

The Shift Flow application has been created to meet these requirements, offering an intuitive interface for the assignment, review, and tracking of tasks while promoting efficient communication. The management of employee tasks has become increasingly vital for sustaining workflow efficiency. It necessitates the provision of clear and actionable instructions to employees while ensuring that managers are able to effortlessly monitor and evaluate completed tasks.

This process may frequently encompass complex information; however, by utilizing appropriate tools, managers can streamline task management and improve overall productivity. A good task management tool, like Shift Flow, allows employees to have control over their day-to-day tasks so that tasks get done on time and performance is continuously tracked. As industries persist in embracing digital solutions for workforce management, Shift Flow functions as a comprehensive platform that facilitates communication, task delegation, and performance tracking through various features, including real-time notifications, progress visualization presented in pie charts, and thorough performance reviews depicted in bar charts. This application enables managers to oversee employees'

progress and make informed, data-driven decisions, while also providing employees with a structured method for managing their tasks.

Finally, investment in a complete digital task management application like Shift Flow is considered a strategic step for those organizations that are on the road to achievement in this highly competitive world. Focused much on user experience and efficiency in operation, Shift Flow makes it the most valuable tool for instilling a lifetime relationship among people working together in an organization to make loyalty and success possible at all levels.

CHAPTER 2

LITERATURE SURVEY

[1]AI-powered task prioritization and scheduling system(2022):this app to prioritize tasks according to workload, deadlines, and urgency.It delivers intelligent reminders, schedules tasks automatically by integrating with calendars, and predicts when tasks will be completed. By providing real-time, data-driven job prioritizing, the app maximizes efficiency. Its objective is to increase team productivity and simplify work management.

[2]Task Management App with Goal-Setting and Performance Tracking(2007): enables staff members to associate individual or group goals with particular activities. It provides real-time progress insights and tracks performance based on goal achievement and task completion. For managers and staff alike, the app regularly generates performance reports and comments.

[3]Task Management App with Real-Time Performance Feedback(2023): managers may give prompt feedback on the timeliness and quality of tasks. It has real-time dashboards for tracking parameters related to employee performance and a rating system for tracking advancement over time.

[4]Task Management System with Biometric Authentication(2022): offers secure task access and approvals. With audit records monitoring access and approvals, biometric verification provides enhanced security for critical jobs

[5]Task Management App with Customizable Workflows(2021):It provides task stages that may be customized to meet departmental needs and pre-built templates. It can be integrated with HR and finance systems to enable smooth task tracking.

[6]Task Management App with Integration to Project Management Tools(2024): It interfaces with CRMs like Salesforce to link tasks with customer or sales data, and it supports bi-directional updates, guaranteeing that changes are reflected across all systems.

[7]Task Management System with Augmented Reality (AR)(2022): projects task lists, reminders, and notifications into users' physical settings. For intricate workflows, it offers AR-based task checklists, and field personnel can receive real-time updates.

[8]Task Management App with staff Mood Tracking(2020): workers may record their emotions while they work on assignments, which enables managers to evaluate staff morale and how it affects output. It offers a dashboard with real-time team morale data and statistics to link job completion rates with employee satisfaction and stress levels.

[9]Cross-Platform Task Management App with Offline Access (2019) :

guarantees flawless task management across desktop, web, and mobile platforms. With Electron for desktop apps, it provides cross-platform interoperability via React Native or Flutter. Using light and dark modes, the app improves user experience while IndexedDB handles local storage and synchronization.

[10]Voice-Enabled Task Management App(2021): may add, edit, and finish tasks without using their hands. It works with virtual assistants such as Google Assistant, Siri, Alexa, and others, allowing task descriptions to be converted in real time from voice to text. The software, which was developed with the help of Flutter, Firebase, Google Cloud's Speech-to-Text API or Alexa Skills Kit, streamlines task management through natural voice interaction.

[11]Task Management App with Collaboration Features(2009): Along with task-based comments and file sharing, it provides integrated messaging, discussion threads, and connectivity with Slack, Zoom, and Microsoft Teams. The software, which was developed with React, Firebase, and Twilio or WebRTC, improves teamwork for remote workers.

[12]Blockchain-Based work Management App(2016): offers safe, transparent work tracking with unchangeable records of assignments and revisions. Smart contracts are utilized to automate job verification and milestone approvals, guaranteeing accountability in delicate initiatives such as finance and healthcare. Utilizing web3.js or React, Ethereum or Hyperledger, and Solidity, the application improves task management security and trust.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The majority of the current employee performance management systems rely on antiquated techniques, such manual performance evaluations carried out using spreadsheets or paper forms, which are laborious, biased, and incapable of providing real-time feedback.
Performance data is frequently dispersed over several platforms, which reduces accuracy and results in inadequate evaluations. Due to the limited data visualization capabilities of these systems, it is challenging to effectively identify trends or potential improvement areas. Usually, feedback is withheld until planned reviews, which lowers employee engagement and impedes prompt problem solving.
Furthermore, user adoption is hindered by inadequate accessibility across devices and a lack of customization in templates that does not cater to specific roles or objectives. All things considered, these drawbacks weaken the efficacy of performance management and obstruct initiatives for ongoing improvement.

DISADVANTAGES:

- Limited Decision-Making Insights.
- Poor Accessibility and Adoption
- Fragmented Data Sources

3.2 PROPOSED SYSTEM

The **Proposed Employee Performance Digital Dashboard** is a modern, data-driven tool designed to address the shortcomings of traditional performance management systems. This solution integrates real-time data and advanced

visualization techniques to provide a comprehensive view of employee performance, enhancing decision-making and employee engagement

ADVANTAGES:

- Advanced Data Visualization.
- Continuous Feedback and Alerts.
- Real-Time Data Integration.
- Customization and Flexibility.

3.3 SOFTWARE REQUIREMENTS

- Android Studio
- Java SDK
- Firebase Cloud Messaging

3.4 HARDWARE REQUIREMENTS

- Processor: Intel i5 or higher / AMD equivalent.
- RAM: 8 GB or more
- Storage: Minimum 256 GB SSD.

CHAPTER 4

SYSTEM DESIGN

4.1 USE CASE DIAGRAM

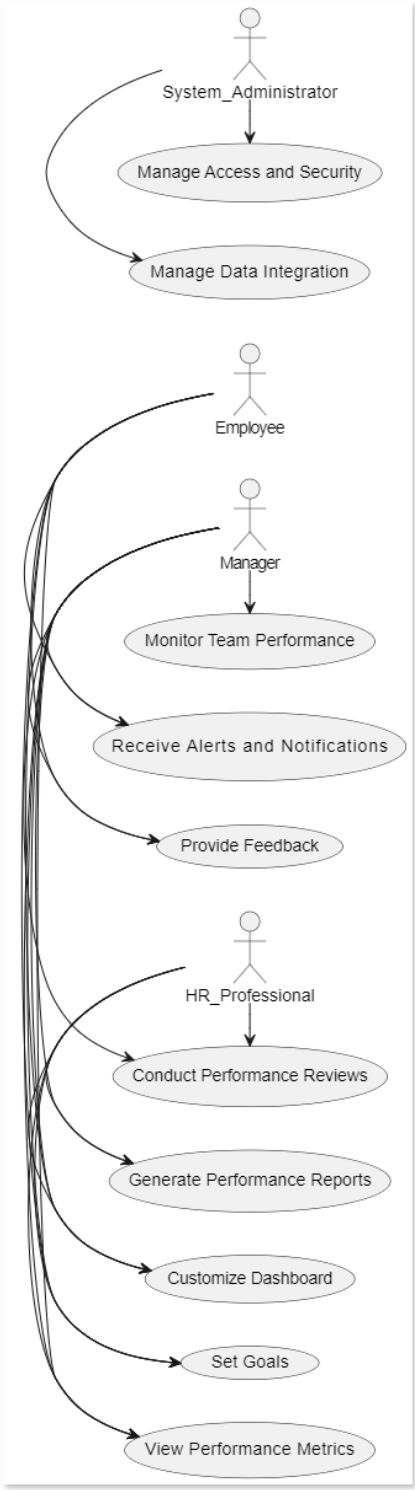

Fig 4.1 Use case diagram

This use case diagram refers to activities done by System and Users and their corresponding use cases.

4.2 CLASS DIAGRAM

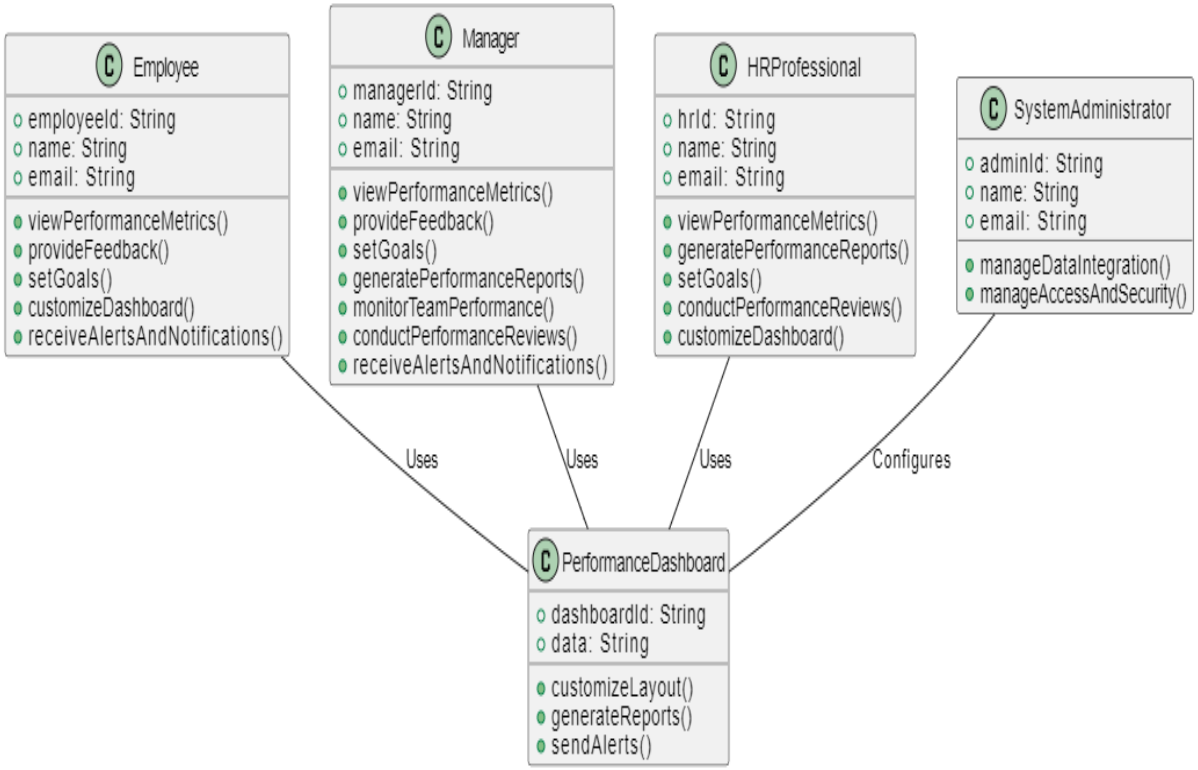

Fig 4.2 Class diagram

The class diagram illustrates the roles and interactions within the Shift Flow system. It defines four key users: Employee, Manager, HRProfessional, and System Administrator, each with specific functions related to task management and performance tracking. These users interact with the Performance Dashboard, which supports operations like report generation, customization, and alerts. The System Administrator configures data integration and access, while other roles use the dashboard to monitor performance and conduct evaluations.

4.3 SEQUENCE DIAGRAM

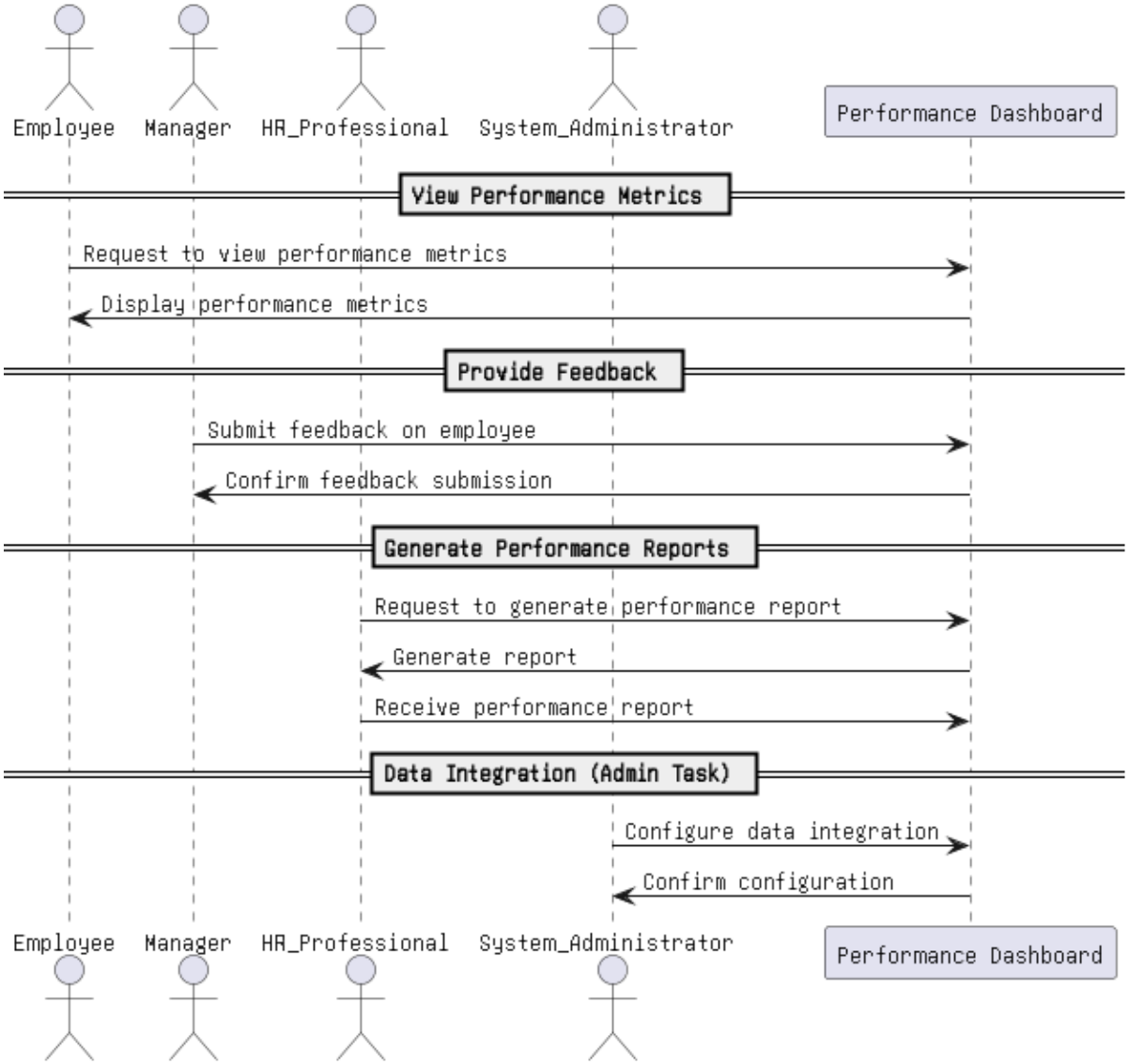

Fig 4.3 Sequence diagram

The sequence diagram shows interactions with the Performance Dashboard for tasks like viewing metrics, providing feedback, generating reports, and data integration. Employees, Managers, and HR handle performance tasks, while the System Administrator manages integration and configuration. Each action involves requests and confirmations for smooth operations.

4.4 ACTIVITY DIAGRAM

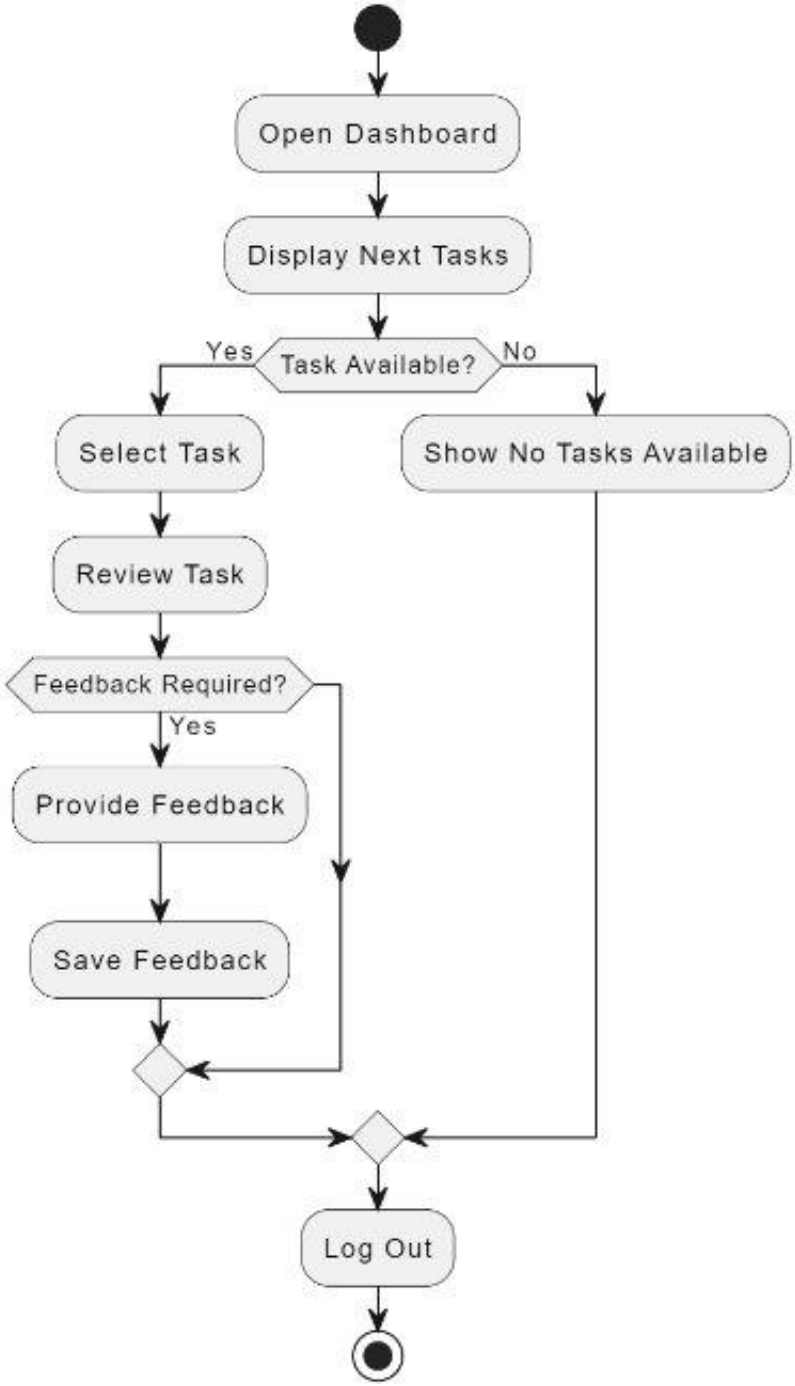

Fig 4.4 Activity diagram

The activity diagram shows the workflow for viewing metrics, providing feedback, generating reports, and data integration. Users initiate actions, followed by system responses and confirmations. The System Administrator handles

integration, while employees, managers, and HR professionals focus on performance tracking.

CHAPTER 5

SYSTEM ARCHITECTURE

5.1 ARCHITECTURE OVERVIEW

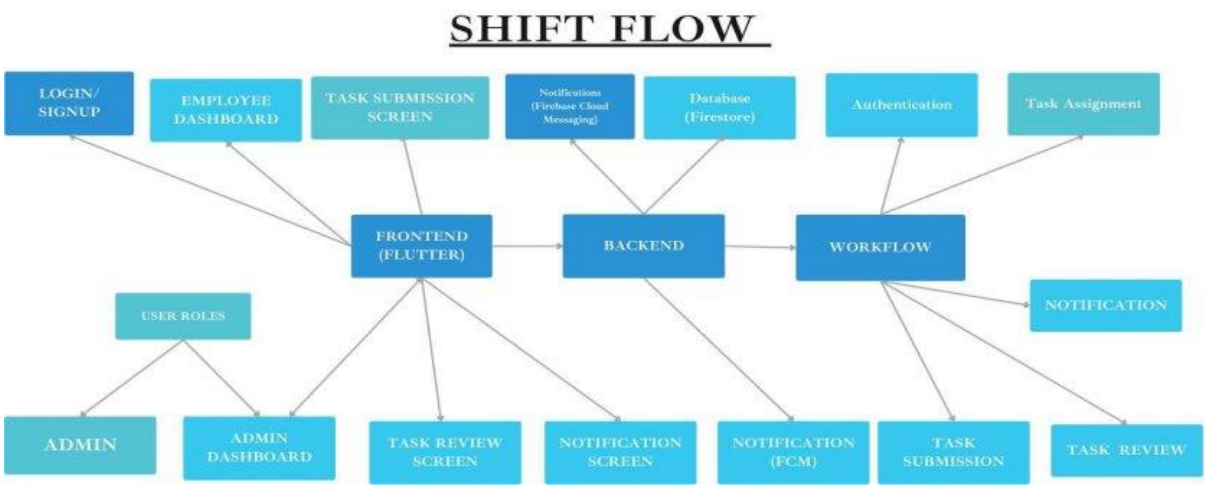

Fig 5.1 System Architecture

The Shift Flow architecture connects the Frontend (Flutter) for user interfaces with the Backend for data storage, notifications, and authentication. The Workflow module manages task assignment, submission, review, and notifications. An Admin Panel offers dashboards for user roles and task management. This setup ensures smooth task handling and efficient communication across the system.

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 CODING FOR PYTHON FILE

pubspec.yaml dependencies:

dependencies:

flutter:

sdk: flutter

firebase_core: latest_version

firebase_auth: latest_version

cloud_firestore: latest_version

provider: latest_version

charts_flutter: latest_version

flutter_local_notifications: latest_version

dependencies:

lib/

|-- main.dart

|-- screens/

| |-- login_screen.dart

| |-- manager_screen.dart

| |-- employee_screen.dart

| |-- task_submission_screen.dart

| |-- task_review_screen.dart

|-- models/

| |-- task.dart

|-- services/

| |-- firebase_service.dart

Main.dart :

```
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'screens/login_screen.dart';

void main() async {
WidgetsFlutterBinding.ensureInitialized();
awaitFirebase.initializeApp();
runApp(ShiftFlowApp());
}

classShiftFlowApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
returnMaterialApp(
title: 'Shift Flow',
theme: ThemeData(
primarySwatch: Colors.grey,
brightness: Brightness.dark,
      ),
home: LoginScreen(),
    );
  }
}
models/task.dart
class Task {
  String id:
```

```
  String title;
  String description;
  String status; // Pending, Completed, etc.
  String employeeId;
DateTimedueDate;

Task({
required this.id,
requiredthis.title,
requiredthis.description,
requiredthis.status,
requiredthis.employeeId,
requiredthis.dueDate,
  });

  Map<String, dynamic>toMap() {
return {
      'id': id,
      'title': title,
      'description': description,
      'status': status,
      'employeeId': employeeId,
      'dueDate': dueDate,
    };
  }

static Task fromMap(Map<String, dynamic> map) {
```

```
return Task(
id: map['id'],
title: map['title'],
description: map['description'],
status: map['status'],
employeeId: map['employeeId'],
dueDate: DateTime.parse(map['dueDate']),
    );
  }
}
```

services/firebase_service.dart:

```
import 'package:cloud_firestore/cloud_firestore.dart';
import '../models/task.dart';
classFirebaseService {
finalFirebaseFirestore _firestore = FirebaseFirestore.instance;
  // Fetch tasks for a specific employee
  Stream<List<Task>>getTasksForEmployee(String employeeId) {
return _firestore
        .collection('tasks')
        .where('employeeId', isEqualTo: employeeId)
        .snapshots()
        .map((snapshot) =>snapshot.docs
            .map((doc) =>Task.fromMap(doc.data() as Map<String, dynamic>))
            .toList());
  }
  // Create or update a task
  Future<void>assignTask(Task task) {
```

```
return _firestore.collection('tasks').doc(task.id).set(task.toMap());
  }
  // Review a submitted task
  Future<void>updateTaskStatus(String taskId, String status) {
return _firestore
        .collection('tasks')
        .doc(taskId)
        .update({'status': status});
  }
}
```

screens/login_screen.dart:

```
import 'package:flutter/material.dart';
import 'manager_screen.dart';
import 'employee_screen.dart';
classLoginScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
return Scaffold(
appBar: AppBar(title: Text('Login')),
body: Center(
child: Column(
mainAxisAlignment: MainAxisAlignment.center,
children: [
ElevatedButton(
onPressed: () {
Navigator.push(context,
MaterialPageRoute(builder: (context) =>ManagerScreen()));
```

```
            },
child: Text('Login as Manager'),
            ),
ElevatedButton(
onPressed: () {
Navigator.push(context,
MaterialPageRoute(builder: (context) =>EmployeeScreen()));
            },
child: Text('Login as Employee'),
            ),
          ],
        ),
      ),
    );
  }
}
```

screens/manager_screen.dart:

```
import 'package:flutter/material.dart';
import 'task_submission_screen.dart';
import 'task_review_screen.dart';

classManagerScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
return Scaffold(
appBar: AppBar(title: Text('Manager Dashboard')),
body: Column(
```

```
children: [
ElevatedButton(
onPressed: () {
Navigator.push(context,
MaterialPageRoute(builder: (context) =>TaskSubmissionScreen()));
          },
child: Text('Assign Task'),
        ),
ElevatedButton(
onPressed: () {
Navigator.push(context,
MaterialPageRoute(builder: (context) =>TaskReviewScreen()));
          },
child: Text('Review Tasks'),
        ),
       ],
     ),
   );
  }
}
```

screens/employee_screen.dart:

```
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import '../services/firebase_service.dart';
import '../models/task.dart';

classEmployeeScreen extends StatelessWidget {
```

```
final String employeeId = 'EMPLOYEE_ID'; // Replace with actual employeeId


  @override
  Widget build(BuildContext context) {
return Scaffold(
appBar: AppBar(title: Text('Employee Dashboard')),
body: StreamProvider<List<Task>>(
create: (_) =>
FirebaseService().getTasksForEmployee(employeeId),
initialData: [],
child: TaskListView(),
      ),
    );
  }
}


classTaskListView extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
final tasks = Provider.of<List<Task>>(context);
returnListView.builder(
itemCount: tasks.length,
itemBuilder: (context, index) {
final task = tasks[index];
returnListTile(
title: Text(task.title),
subtitle: Text(task.description),
```

```
trailing: Text(task.status),
        );
      },
    );
  }
}
```

screens/task_submission_screen.dart:

```
import 'package:flutter/material.dart';
import '../services/firebase_service.dart';
import '../models/task.dart';
import 'package:uuid/uuid.dart';

classTaskSubmissionScreen extends StatelessWidget {
final _titleController = TextEditingController();
final _descriptionController = TextEditingController();
final _employeeIdController = TextEditingController();

  @override
  Widget build(BuildContext context) {
return Scaffold(
appBar: AppBar(title: Text('Assign Task')),
body: Padding(
padding: EdgeInsets.all(16.0),
child: Column(
children: [
TextField(
controller: _titleController,
```

```
decoration: InputDecoration(labelText: 'Task Title'),
          ),
TextField(
controller: _descriptionController,
decoration: InputDecoration(labelText: 'Task Description'),
          ),
TextField(
controller: _employeeIdController,
decoration: InputDecoration(labelText: 'Employee ID'),
          ),
ElevatedButton(
onPressed: () {
final task = Task(
id: Uuid().v4(),
title: _titleController.text,
description: _descriptionController.text,
status: 'Pending',
employeeId: _employeeIdController.text,
dueDate: DateTime.now().add(Duration(days: 1)),
              );
FirebaseService().assignTask(task);
            },
child: Text('Assign Task'),
          ),
        ],
      ),
    ),
```

```
    );
  }
}
```

VIZ.PY

```
import pandas as pd
importmatplotlib.pyplot as plt
importseaborn as sns
fromwordcloud import WordCloud, STOPWORDS, ImageColorGenerator
fromsrc import sql
importnumpy as np
fromvaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

# Plot configurations
FIG_W = 10 # Width of plots
FIG_H = 5 # Height of plots
ROT = 0 # Rotation of x-axis labels
TS = 15 # Title size

defbarplot_channel_video_count(df_all, channel_ids):
    '"Create a barplot and save the image to a folder. Return image name. Take a dataframe with videodata  as input. Input channel_ids to render image name."'

channel_ids_string = '_'.join(channel_ids)
image_name = f'static/images/{channel_ids_string}_barplot_channel_video_count.png'
```

```
plt.figure(figsize=(FIG_W, FIG_H))
    df_all.groupby('channel_title').size().sort_values(ascending=False).plot.bar()
plt.xticks(rotation=ROT)
plt.xlabel("Channel Name")
plt.ylabel("Video Count")
plt.title('Video Counts per Channel', fontdict = {'fontsize' : TS})
plt.savefig(image_name, dpi=100)

returnimage_name

defhistogram_video_duration_count(df_all, channel_ids):
    '"Create a histogram and save the image to a folder. Return image name. Take a dataframe with videodata  as input. Input channel_ids to render image name.'"

df_all['duration_min'] = df_all['duration_sec'].astype('int') / 60

    # Calculate outlier and clean them
    outlier = (df_all['duration_min'].describe()['75%'] - df_all['duration_min'].describe()['25%']) * 1.5 + df_all['duration_min'].describe()['75%']
df_all = df_all[df_all['duration_min'] <= outlier]

bin_size = int(df_all['duration_min'].max())
labels = df_all['channel_title'].unique()

data = []
for channel in labels:
```

```
video_durations = df_all[df_all['channel_title'] == channel]['duration_min'].to_numpy()
data.append(video_durations)

    # Create image name
channel_ids_string = '_'.join(channel_ids)
image_name=f'static/images/{channel_ids_string}_histogram_video_duration_count.png'

plt.figure(figsize=(FIG_W, FIG_H))
plt.hist(data, bins=bin_size, alpha=0.5)
plt.legend(labels)
plt.xlabel('Duration of videos in minutes')
plt.ylabel('Videos count')
plt.title('Video counts of durations', fontdict = {'fontsize' : TS})
plt.savefig(image_name, dpi=100)

returnimage_name

defhistogram_video_duration_count_single(df_all, channel_id, channel_title=None):
    '''Create a histogram and save the image to a folder. Return image name. Take a dataframe with videodata  as input. Input channel_ids to render image name.'''

df_all = df_all[df_all['channel_id'] == channel_id]

    # Calculate outlier and clean them
```

```
   outlier = (df_all['duration_sec'].describe()['75%'] - df_all['duration_sec'].describe()['25%']) * 1.5 + df_all['duration_sec'].describe()['75%']

df_all = df_all[df_all['duration_sec'] <= outlier]


df_all['duration_min'] = df_all['duration_sec'] / 60

df_all['duration_min'] = df_all['duration_min'].astype('int32')


bin_size = df_all['duration_min'].max()

ifbin_size< 1:

bin_size = 1

image_name = f'static/images/{channel_id}_histogram_video_duration_count.png'


plt.figure(figsize=(FIG_W, FIG_H))

plt.hist(df_all['duration_min'], bins=bin_size, alpha=0.5, edgecolor='black', linewidth=1)

plt.legend(df_all['channel_title'].unique())

plt.title(f'Video Counts of Durations for "{channel_title}"', fontdict = {'fontsize' : TS})

plt.xlabel('Video Duration in Minutes')

plt.ylabel('Video Count')

plt.xlim(0,bin_size)

plt.savefig(image_name, dpi=100)


returnimage_name


defbarplot_links(video_df, channel_ids):
```

```
    '''Create a barplot with counts on how many video descriptions hae clickable links. Save the plot as image.'''


    # Check if there is 'http' in description and insert result
video_df['Links in decription'] = video_df['description'].str.contains('http').apply(lambda x: 'Clickable Link' if x else 'No clickable Link')


channel_ids_string = '_'.join(channel_ids)
image_name = f'static/images/{channel_ids_string}_barplot_links.png'


video_df = video_df.groupby(['channel_title', 'Links in decription'])[['video_id']].count().reset_index()
sns.set(style="whitegrid")
    g = sns.catplot(x="channel_title",
                    y="video_id",
hue="Links in decription",
data=video_df,
height=6,
kind="bar",
palette="muted"
    )
g.despine(left=True)
g.set_xlabels("Channel Name")
g.set_ylabels("Video Count")
    # g.set_title('Links in Video Descriptions', fontdict = {'fontsize' : TS})
plt.savefig(image_name, dpi=100)


returnimage_name
```

```
defcreate_wordcloud(text, stopwords=STOPWORDS,video_id=None, channel_title=None):
    '''Return a word cloud image name and save the image. Take as input a string of text and a video id or a channel name for creating the title.'''

wordcloud = WordCloud(
max_font_size=50,
min_font_size=10,
max_words=100,
prefer_horizontal=1,
        # fortransparnt background: mode='RGBA', background_color=None
        # mode="RGBA",
background_color="white",
stopwords=stopwords,
        # Increase for lager images
scale=2.0,
        # Disable word pairs
collocations=False
    ).generate(text)

    # Create filename
ifvideo_id == None:
temp_id = sql.set_temp_id()
image_name = f'static/images/{temp_id}_wordcloud.png'
else:
image_name = f'static/images/{video_id}_wordcloud.png'
```

```
ifchannel_title:
title = channel_title
else:
title = video_id

plt.figure(figsize = (FIG_W, FIG_H))
plt.title(f'Wordcloud for "{title}"', fontdict = {'fontsize' : TS})
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.savefig(image_name, dpi=100)

returnimage_name

defsplit_sentiment_pos_neg(comment_sentiment):
    '"Split dataframe into positive, neutral and negative dataframes. Used for plotting.'"

comment_sentiment.sort_values(by='published_at', inplace=True)
comment_sentiment['count'] = 1
comment_sentiment['cumsum'] = comment_sentiment['count'].cumsum()

    # Select negative and poitive comments
neg_sent = comment_sentiment[comment_sentiment['compound'] < -0.5]
neg_sent['count'] = 1
neg_sent['cumsum'] = neg_sent['count'].cumsum()
pos_sent = comment_sentiment[comment_sentiment['compound'] > 0.5]
pos_sent['count'] = 1
pos_sent['cumsum'] = pos_sent['count'].cumsum()
```

returncomment_sentiment, pos_sent, neg_sent

deflineplot_cumsum_video_comments(comment_sentiment, video_id):

"'Create and save a lineplot for the cumsum of video comments over time. Return image name.'"

image_name = f'static/images/{video_id}_lineplot_cumsum_video_comments.png'

plt.figure(figsize=(FIG_W, FIG_H))

plt.plot(comment_sentiment['published_at'], comment_sentiment['cumsum'])

plt.xticks(rotation=ROT)

plt.title('Cumulative sum of comments over time', fontdict = {'fontsize' : TS})

plt.xlabel('Date')

plt.ylabel('Sum of comments')

plt.grid(b=True)

plt.savefig(image_name, dpi=100)

returnimage_name

deflineplot_cumsum_video_comments_pos_neg(comment_sentiment, pos_sent, neg_sent, video_id):

"'Create and save a lineplot for the cumsum of positive and negative sentiments of video comments over time seperately. Return image name.'"

image_name = f'static/images/{video_id}_lineplot_cumsum_video_comments_pos_neg.png'

```
plt.figure(figsize=(FIG_W, FIG_H))

plt.plot('published_at', 'cumsum', data=pos_sent, marker='', color='green', linewidth=1, linestyle='-', label="Positive Sentiment")

plt.plot('published_at', 'cumsum', data=neg_sent, marker='', color='red', linewidth=1, linestyle='-', label="Negative Sentiment")

plt.legend()

plt.title('Cumulative sum of comments over time', fontdict = {'fontsize' : TS})

plt.xlabel('Date')

plt.ylabel('Sum of comments')

plt.xticks(rotation=ROT)

   # plt.grid(b=True)

plt.savefig(image_name, dpi=100)


returnimage_name


defscatterplot_sentiment_likecount(comment_sentiment, pos_sent, neg_sent, video_id):

   '''Create a scatterplot with like counts vs.sentiment. Save image.Return image name. Take as input the output of "split_sentiment_pos_neg()" and a video id.'''


image_name = f'static/images/{video_id}_scatterplot_sentiment_likecount.png'


fig = plt.figure(figsize=(FIG_W, FIG_H))

plt.scatter(comment_sentiment['compound'], np.log1p(comment_sentiment['like_count']), label='Neutral Sentiment')

plt.scatter(pos_sent['compound'], np.log1p(pos_sent['like_count']), color='green', label='Positive Sentiment')

plt.scatter(neg_sent['compound'], np.log1p(neg_sent['like_count']), color='red', label='Negative Sentiment')

plt.xticks(rotation=ROT)
```

```
plt.title('Sentiment / Like count', fontdict = {'fontsize' : TS})
plt.xlabel('Sentiment')
plt.ylabel('Logarithm of Like count')
plt.legend()
   # plt.grid(b=True)
fig.savefig(image_name, dpi=100)

returnimage_name

deftop_videos(video_df, metric='view', n=5):
   '"Return a table with top n videos of all channels in the dataframe considering a given metric. Possible metrics are like, dislike, comment and view"'

df_table = video_df.sort_values(by=f'{metric}_count',ascending=False).groupby('channel_title').head(n).sort_values(by=f'{metric}_count', ascending=False)[['channel_title', 'title', f'{metric}_count']].rename(columns={'channel_title':'Channel Title', 'title':'Video Title', f'{metric}_count':f'{metric.capitalize()} Count'}).set_index('Channel Title')
df_table = df_table.reset_index()
df_table = df_table.set_index(df_table.index + 1)
df_table = df_table.reset_index()
df_table = df_table.rename(columns={'index':'Rank'})
df_table = df_table.set_index(['Rank', 'Channel Title', 'Video Title',f'{metric.capitalize()} Count'])
df_table.reset_index(inplace=True)

returndf_table
```

APP.PY

```
from flask import Flask, render_template, request
fromsrc import YouTube_data_module as ydt
fromsrc import viz
import pandas as pd
importos
import logging
import sys

logger = logging.getLogger('app_logger')
handler = logging.StreamHandler(sys.stderr)
logger.addHandler(handler)
logger.setLevel(logging.INFO)

API_KEY = ('AIzaSyB93OAAvNhwVzPzoyKNeNVR0MYQOjQhCxs')

app = Flask(__name__)

@app.route('/')
def home():
returnrender_template('layout.html')

@app.route('/select_video')
defselect_video():
    '''This page returns search results, when a user hits the 'Search Video' button'''
```

```
result_dictionary = request.args
query = result_dictionary['query']
YouTube = ydt.YouTubeAPIkey(API_KEY)
query_result = ydt.YouTubeSearchListStatistics(YouTube, q=query)

returnrender_template(
        'select_video.html',
query_result=query_result,
query=query
    )

@app.route('/video_comments')
defvideo_comments():
    '"This page returns a video comment analysis, when a user hits the 'See video comment analysis' button"'
video_id = request.args.get('video_id')

YouTube = ydt.YouTubeAPIkey(API_KEY)

logger.info('Getting all comments')
all_snippets = ydt.get_all_comments(YouTube, video_id)
logger.info('Writing comments to dict')
comment_dict = ydt.extract_comments(all_snippets)

image_names = []
logger.info('Generating wordcloud')
comment_string = ydt.concat_comments(comment_dict)
video_title = video_id
```

```
image_names.append(viz.create_wordcloud(comment_string, stopwords=None, video_id=video_id, channel_title=video_title))

comment_df = ydt.comments_to_df(all_snippets)

comment_sentiment = ydt.analyze_comment_sentiments(comment_df)

    comment_sentiment2, pos_sent, neg_sent = viz.split_sentiment_pos_neg(comment_sentiment)


image_names.append(viz.lineplot_cumsum_video_comments(comment_sentiment2, video_id))


image_names.append(viz.lineplot_cumsum_video_comments_pos_neg(comment_sentiment2, pos_sent, neg_sent, video_id))


image_names.append(viz.scatterplot_sentiment_likecount(comment_sentiment2, pos_sent, neg_sent, video_id))

    # Calculate correlation

like_count_sentiment_corr = round(comment_sentiment2.corr().loc['like_count'][5],2)


returnrender_template(

        'video_comments.html',

image_names=image_names,

like_count_sentiment_corr=like_count_sentiment_corr

    )


@app.route('/select_channels', methods=['GET', 'POST'])

defselect_channels():

    '"This page return search results for the channel queries a user inputs and hits the 'Search Channels' button"'

result_dictionary = request.args

channel_names = []
```

```
forchannel_name in result_dictionary:
iflen(result_dictionary.get(channel_name)) > 0:
channel_names.append(result_dictionary[channel_name])

YouTube = ydt.YouTubeAPIkey(API_KEY)
query_results = {}

forcn in channel_names:
result = ydt.YouTubeSearchList(YouTube, channel_id=None, q=cn, maxResults=5, type='channel')
query_results[cn] =  result

returnrender_template(
      'select_channels.html',
query_results=query_results
   )

@app.route('/channels', methods=['GET', 'POST'])
def channels():
   '''This page returns the channel coparison analysis when a user selects at least one channel with a radio button and hits "Compare channels now"'''
result_dictionary = request.args

channel_ids = []
forc_id in result_dictionary:
iflen(result_dictionary[c_id]) == 24:
channel_ids.append(result_dictionary[c_id])
```

```
YouTube = ydt.YouTubeAPIkey(API_KEY)
video_df = ydt.get_channel_video_df(YouTube, channel_ids)

image_names = []
image_names.append(viz.barplot_channel_video_count(video_df, channel_ids))
image_names.append(viz.barplot_links(video_df, channel_ids))

channel_titles = []
forchannel_id in channel_ids:
channel_video_df = video_df[video_df['channel_id'] == channel_id]
channel_title = channel_video_df['channel_title'].unique()[0]
channel_titles.append(channel_title)

image_names.append(viz.histogram_video_duration_count_single(channel_vid
eo_df, channel_id, channel_title=channel_title))
channel_video_series = channel_video_df['tags']
wordcloud_string = ydt.concat_listelements(channel_video_series)
image_names.append(viz.create_wordcloud(wordcloud_string,
stopwords=None, video_id=channel_id, channel_title=channel_title))

df_table = viz.top_videos(video_df, metric='view', n=5)

returnrender_template(
        'channels.html',
result_dictionary=result_dictionary,
video_df=video_df,
image_names=image_names,
```

```
channel_ids=channel_ids,
channel_titles=channel_titles,
tables=[df_table.to_html(index=False, classes='table-striped')],
    )

if __name__ == '__main__':
app.run(port=3000, debug=True)
```

SQL.PY

```
import time
import random

defset_temp_id():
    '''Create a random string by using current unix time and random integer of 4 digits.'''
time_id = str(int(time.time()))
rand_id = str(random.randint(1000,9999))
temp_id = time_id + "_" + rand_id

returntemp_id
```

CHAPTER 6

6. CONCLUSION

6.1 CONCLUSION

The Shift Flow management application will use new digital technologies in optimizing task assignment and tracking execution to help organizations have a right flow management.Just as banks have procedures known as KYC to ensure that customers are real, the Shift Flow will enable managers and employees to

communicate effectively and authenticate submissions of tasks so that operations can perform their services better.Shift Flow is advancing the culture of trust and transparency byreducing the build of a seamless, user-friendly digital experiencethat fosters accountability and productivity within teams. That includes things like real-time notifications and secured ways ofhandling information about tasks.Finally, investment in a complete digital task management application like Shift Flow is considered a strategic step for those

organizations that are on the road to achievement in this highlycompetitive world. Focused much on user experience andefficiency in operation, Shift Flow makes it the most valuable tool for instilling a lifetime relationship among people working together in an organization to make loyalty and success possible at all levels.

6.2 RESULTS AND DISCUSSIONS

By providing a number of significant advantages, the Employee Performance Digital Dashboard successfully tackles the drawbacks of conventional performance management systems. Real-time performance tracking is made possible by it, giving managers and staff members the ability to continuously track development and promptly provide feedback in order to recognize accomplishments and address problems more quickly. By combining performance-related data from timesheets, feedback platforms, HR systems, and project management tools, the solution guarantees centralized data integration. This eliminates data silos and produces assessments that are more accurate. The dashboard displays performance indicators through interactive graphs, charts, and KPIs with enhanced data visualization, which makes it simpler to see patterns and potential improvement areas. Role-based customisation is made possible by the customized goal-setting and feedback capabilities, which let managers and staff create individual goals that are in line with company objectives.

With the introduction of real-time tracking, centralized data management, and intelligent visualizations, the suggested Employee Performance Digital

Dashboard completely changes the way businesses handle employee performance. The capacity to give ongoing feedback promotes employee engagement and synchronizes individual performance with corporate objectives.

Performance reviews are more relevant because of the customization possibilities that meet the unique requirements of various professions. Furthermore, the dashboard's compatibility with various devices boosts user adoption and guarantees managers' and staff members' regular use. Report automation lowers administrative burden and facilitates effective data-driven decision-maker.

Implementation may present certain difficulties, though, such as merging data from several different systems that are currently in place or making sure that every user efficiently uses the platform. It will be essential to have proper training and change management techniques.

6.3 FUTURE ENHANCEMENTS

- Introduce features that enable real-time collaboration between team members, such as chat functionalities, video calls, and shared documents within the dashboard.

- This will facilitate more effective communication and teamwork, enhancing the overall performance management process.

- Provide even more customizable dashboards and reports tailored to the specific needs of different roles, departments, or teams.

- Users can choose what metrics to display, how they want the information visualized, and set personal notifications based on their preferences.

- Expand integration capabilities with other HR tools and software, such as learning management systems (LMS), recruitment platforms, and payroll systems, to create a comprehensive HR ecosystem.

- This holistic approach will ensure that all HR-related data is interconnected and easily accessible.

CHAPTER 7

APPENDICES

7.1 SCREENSHOTS

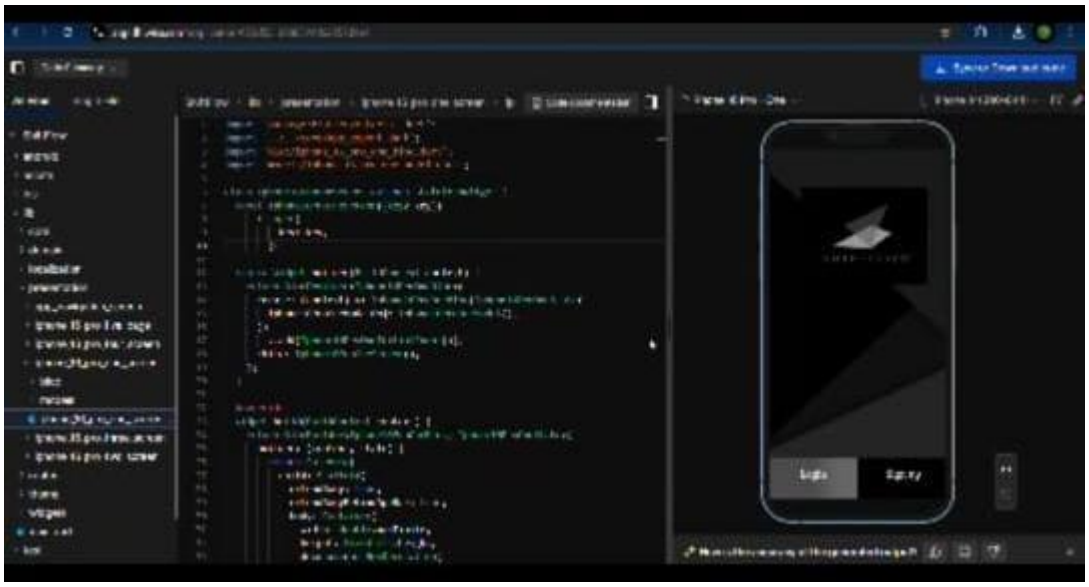

Fig. 7.1 : LOGIN PAGE

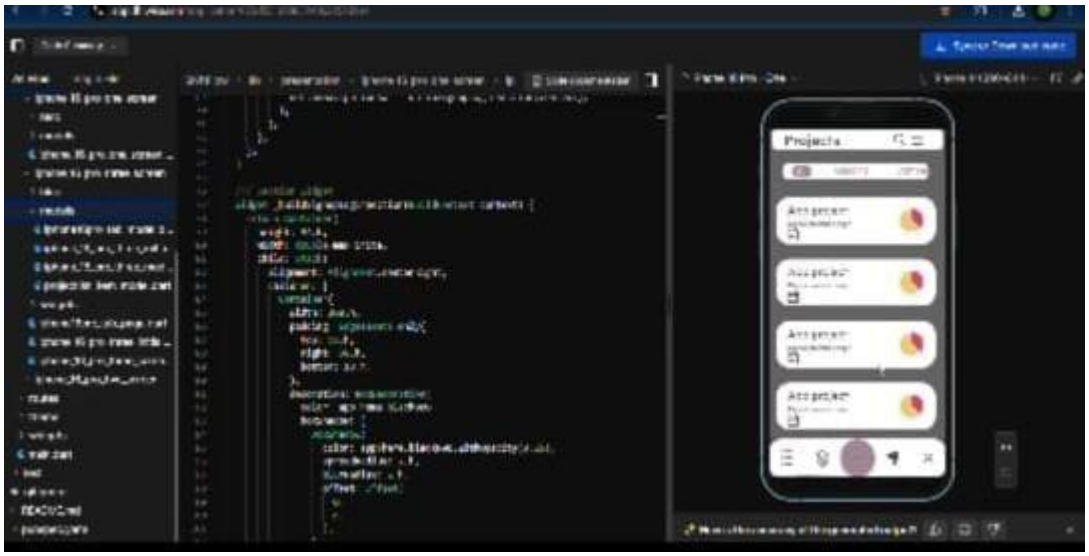

Fig. 7.2 :EMPLOYEE PERFORMANCE SUBMISSION DASHBOARD

7.2 REFERENCES

- Y. Zhang, X. Li, and M. Wang, "Employee Recognition, Task Performance, and OCB: Mediated and Moderated by Pride," *Sustainability*, vol. 14, no. 3, pp. 1631, Jan. 2022. DOI: 10.3390/su14031631.

- A. K. Gupta, R. Singh, and N. Sharma, "Development of a digital dashboard for employee performance evaluation," *2021 IEEE International Conference on Data Science and Systems (DSS)*, 2021, pp. 45-52. DOI: 10.1109/DSS.2021.9402300.

- P. Kumar, "AI-Driven Task Management Systems: Enhancing Productivity and Engagement," *IEEE Access*, vol. 8, pp. 120513-120526, Aug. 2020. DOI: 10.1109/ACCESS.2020.2993847.

- J. C. Chen and S. D. Silverstone, "Performance measurement with dashboards and KPIs: Balancing efficiency and employee engagement," *Journal of Business Research*, vol. 135, pp. 564-576, 2022. DOI: 10.1016/j.jbusres.2021.10.012.

- **Berhane, B. D., & Gudfinnsson, K.** (2021). *Utilizing Business Intelligence Systems for Performance Management: Challenges and Solutions*.

- **Rajesh, R., & Anand, N.** (2019). *KPI-Based Performance Monitoring in the Supply Chain: A Framework for Decision-Making*.

- **Franceschini, F.** (2017). *Performance Measurement Systems for Organizational Success*. This book examines various models and tools, including dashboards, used to align employee goals with company strategy.

- **Al-Mashari, M., & Zairi, M.** (2019). *Real-Time KPI Dashboards for Monitoring Employee Engagement*. The research discusses how organizations use KPI-driven dashboards to enhance transparency and engagement across departments.

    - Available in the *Journal of Business Management*.

- **Kucukaltan, B., & Irani, Z.** (2021). *Using Dashboards for Continuous Improvement in Employee Performance*. The paper addresses the integration of operational, tactical, and strategic dashboards within business intelligence frameworks..

- **Gonçalves, C. T., & Gonçalves, M. J. A.** (2023). *Developing Integrated Performance Dashboards Visualisations Using Power BI as a Platform*.

    .

- **Kumar, P.** (2020). *AI-Driven Task Management Systems: Enhancing Productivity and Engagement*. This paper emphasizes AI applications in task management and performance tracking.

• **Chen, J. C., & Silverstone, S. D.** (2022). *Performance Measurement with Dashboards and KPIs: Balancing Efficiency and Employee Engagement*. This study discusses how dashboards can align KPIs with employee and business objectives.

  • DOI: 10.1016/j.jbusres.2021.10.012.

• **Wixom, B. H., & Watson, H. J.** (2014). *Business Intelligence Dashboards: Key Trends and Impact on Decision Making*. This research outlines how BI dashboards facilitate faster and more effective decision-making by integrating multiple data sources.

  • Available on ResearchGate.

• **Zhang, Y., & Wang, M.** (2022). *Employee Recognition and Performance: The Role of Dashboards in Motivating Employees*. This paper examines the impact of transparent performance dashboards on employee motivation and performance.

COPYRIGHT:

Copyright Office
Government of India

Acknowledgement Slip
(Date: 15/10/2024)

Diary Number: **32255/2024-CO/SW**	Form Received: **Online**
Copyright Reg. of: **Computer Software**	Titled: **Shiftflow**

Communication Address

Name	Address	Phone Number
PRATHIKSHA J	2/3 MUNICIPAL COLONY ANUMANTHAPUTHERI-603002	08825795376

Financial Details

Payment ID	Amount	Bank Name	Payment Mode	Payment Date
370274	500			15/10/2024

* For future communication please mention this DIARY No.

INSTRUCTIONS

For the purpose of processing the application, following documents are mandatory to send by post along with the acknowledgement slip(Office Copy).

1.	2 Copies of work
2.	DD/IPO of Rs.(as applicable) per work favouring Registrar Copyright Office payable at New Delhi (Not applicable for online payment)
3.	Authorization from author/publisher
4.	If the work is being used on goods or capable of being used on the goods
5.	If the application is being field through attorney, a specific power of attorney in original duly signed by the applicant and accepted by the attorney
6.	Search Certificate from Trade Mark Office(TM-60) (*Only in case of Artistic work*).
7.	Applicant must take a print out of the application, sign it and send along with the other documents.

Kindly send the above documents within 30 Days from the date of online submission on the following address given by herewith: .

Office of the Registrar of Copyrights
Copyright Office, Department for Promotion of Industry & Internal Trade
Ministry of Commerce and Industry
Boudhik Sampada Bhawan,
Plot No. 32, Sector 14, Dwarka,
New Delhi-110078
Email Address: copyright@nic.in
Telephone No.: 011-28032496

APPLICANT'S COPY
THANK YOU

Copyright Office
Government of India

Acknowledgement Slip
(Date:15/10/2024)

Diary Number: **32255/2024-CO/SW**	Form Received: **Online**
Copyright Reg. of: **Computer Software**	Titled: **Shiftflow**

Communication Address

Name	Address	Phone Number
PRATHIKSHA J	2/3 MUNICIPAL COLONY ANUMANTHAPUTHERI-603002	08825795376

Financial Details

Payment ID	Amount	Bank Name	Payment Mode	Payment Date
370274	500			15/10/2024

* For future communication please mention this DIARY No.

INSTRUCTIONS

For the purpose of processing the application, following documents are mandatory to send by post along with this acknowledgement slip(Applicant's Copy).

1.	2 Copies of work
2.	DD/IPO of Rs.(as applicable) per work favouring Registrar Copyright Office payable at New Delhi (Not applicable for online payment)
3.	Authorization from author/publisher
4.	If the work is being used on goods or capable of being used on the goods
5.	If the application is being field through attorney, a specific power of attorney in original duly signed by the applicant and accepted by the attorney
6.	Search Certificate from Trade Mark Office(TM-60) (*Only in case of Artistic work*).
7.	Applicant must take a print out of the application, sign it and send along with the other documents.

Kindly send the above documents within 30 Days from the date of online submission on the following address given by herewith: .

Office of the Registrar of Copyrights
Copyright Office, Department for Promotion of Industry & Internal Trade
Ministry of Commerce and Industry
Boudhik Sampada Bhawan,
Plot No. 32, Sector 14, Dwarka,
New Delhi-110078
Email Address: copyright@nic.in
Telephone No.: 011-28032496

OFFICE COPY
THANK YOU

FORM XIV
APPLICATION FOR REGISTERATION OF COPYRIGHT
[SEE RULE 70]

Diary Number: 32255/2024-CO/SW

To

The Registrar of Copyrights,
Copyright Office,
Department of Industrial Policy & Promotion,
Ministry of Commerce and Industry,
Boudhik Sampada Bhawan,
Plot No. 32, Sector 14, Dwarka,
New Delhi-110075
Email Address: copyright@nic.in
Telephone No.: (Office) 011-28032496, 08929474194

Sir,

In Accordance with Section 45 of the Copright Act, 1957 (14 of 1957), I hereby apply for registration of Copyright and request that enteries may be made in the Register of Copyrights as in the enclosed Statement of Particulars.

1. I also send herewith duly completed the Statement of further Particulars relating to the work. (for Literary/Dramatic, Musical, Atristic works only) **Computer Software works**

2. In accordance with rule 16 of the Copyright Rules, 1958, I have sent by prepaid registered post copies of this letter and of the Statement of Particulars and Statement of Further Particulars to other parties concerned as shown below:

[See columns 7,11,12, and 13 of the Statement of Particulars and party referred in col.2 (e) of the Statement of Further Particulars.]

3. The prescribed fee has been paid, as per details below: **500/-**

Payment ID	Payment Date	Amount	Bank Name	Payment Mode
370274	15/10/2024	500		

4. Communications on this subject may be addressed to:

PRATHIKSHA J
2/3 MUNICIPAL COLONY
ANUMANTHAPUTHERI-603002
08825795376

5. I hereby declare that to the best of my knowledge and belief, no person, other than to whom a notice has been sent as per paragraph 2 above any claim or interest or dispute to my copyright of this work or its use by me.

6. I hereby verify that the particulars given in this Form and the Statement of Particulars and Statment of Further Particulars are true to the best of my knowledge, belief and information and nothing has been concealed there from.

List of Enclosures:

1. 2 Copies of Work
2. DD/IPO of Rs.500 Per Work
3. Authorization from author/publisher
4. If the application is being filed through attorney , a specific Power of Attorney in original duly signed by the applicant and accepted by the attorney

Place:

Date: **15/10/2024**

For : PRATHIKSHA J

STATEMENT OF PARTICULARS

Diary Number: 32255/2024-CO/SW

1.	Registration Number	
2.	Name, Address and Nationality of the Applicant	NAME: PRATHIKSHA J, ADDRESS: 2/3 MUNICIPAL COLONY ANUMANTHAPUTHERI-603002, Indian
3.	Nature of the Applicant's interest in the Copyright of the work	Author
4.	Class and description of the work	Computer Software Work
5.	Title of the work	Shiftflow
6.	Language of the work	Java
7.	Name, Address and Nationality of the Author and if the Author is deceased, the date of decease.	NAME: PRATHIKSHA J, ADDRESS: 2/3 MUNICIPAL COLONY ANUMANTHAPUTHERI-603002, Indian,
8.	Whether the work is Published or Unpublished	Unpublished
9.	Year and Country of first publication, and Name, Address and Nationality of the publisher	N/A
10.	Year and Countries of subsequent publications, if any, and Name, Address and Nationality of the publisher	N/A
11.	Name, Address and Nationality of the Owners of the various rights comprising the copyright in the work and extent of rights held by each, together with particulars of assignments and licence. If any	NAME: PRATHIKSHA J, ADDRESS: 2/3 MUNICIPAL COLONY ANUMANTHAPUTHERI-603002, Indian
12.	Name and address and nationality of other persons, if any authorized to assign or licence the rights comprising the copyright	N/A
13.	If the work is an 'Artistic work', the location of the original work, including name, address and nationality of the person in possession of the work, (In the case of an architectural work, the year of completion of the work should also be shown)	N/A
14.	If the work is an 'Artistic work' which is used or capable of being used in relation to any goods or services, the application should include a certification from the Registrar of Trade Marks in terms of the provision to Sub-Section (i) of Section 45 of the Copyright Act, 1957	N/A
15.	If the work is an 'Artistic work' whether it is registered under the Desings Act 2000 if yes give details.	N/A
16.	If the work is an 'Artistic work' capable of being registrar as a design under the Designs Act 2000, whether is has been applied to an article though an industrial process and,if yes ,then number of times it is reproduced	N/A
17.	Remarks, if any	

Place:
Date: **15/10/2024**

For : PRATHIKSHA J

STATEMENT OF FURTHER PARTICULARS

(For Literary/Dramatic, Musical and Artistic works only)

Diary Number: 32255/2024-CO/SW

1. Is the work to be registered

(a) an orginal work? : Yes

(b) a translation of a work in the public domain? : N.A.

(c) a translation of a work in which Copyright subsists? : N.A.

(d) an adaptation of a work in the public domain? : N.A.

(e) an adaptation of a work in which Copyright subsists? : N.A.

2. If the work is a translation or adaptation of a work in which copyright subsists

(a) Title of the original work : N.A.

(b) Language of the original work : N.A.

(c) Name, address, and nationality of the author of the original work and if the author is deceased, the date of decease : N.A.

(d) Name, address, and nationality of the publisher, if any, of the original work : N.A.

(e) Name, address, and nationality of the publisher, or adaptation including the name, address and nationality of party authorizing : N.A.

3. Remarks, if any

Place:
Date: **15/10/2024**

For : PRATHIKSHA J

Proprietor