THE MAHARAJA SAYAJIRAO UNIVERSITY OF BARODA

DEPARTMENT OF COMPUTER APPLICATIONS

FACULTY OF SCIENCE

An Internship Report

on

HR Management Dashboard

for

IndaPoint Technologies Pvt Ltd

Submitted by

Shlok N. Tilokani

PRN: 8021001295

in partial fulfillment for the award of the degree

of

Bachelors of Computer Application

in the

Department of Computer Applications

April 2024

Internal Guide

Ms. Poonam Shah

External Guide

Mr. Chirag Kansara
CTO
IndaPoint Technologies Pvt. Ltd.

TABLE OF CONTENTS

ACKNOWLEDGEMENT

I extend my sincere appreciation to my mentor, Mr. Chirag Kansara (CTO), for his unwavering guidance and dedication throughout my internship and for providing me with an opportunity to learn new skills. His profound understanding of various technologies, coupled with his adept problem-solving skills, enabled me to explore a myriad of possibilities within the field. His mentorship not only enriched my learning experience but also instilled in me a deeper understanding of the subject matter.

Furthermore, I am grateful to the Department of Computer Applications and Dr. P.K Mehta, Head of Department (HOD), for their continuous support and provision of numerous opportunities throughout my academic journey. His encouragement and belief in my capabilities have been instrumental in my growth and development.

I also wish to express my gratitude to Ms. Poonam Shah for her invaluable guidance and unwavering support. Her insightful suggestions and willingness to lend a helping hand have been like a beacon of light, making the entire experience both enjoyable and enriching.

In summary, I am deeply thankful to all those who have played a significant role in shaping my professional and academic endeavours. Their collective efforts have not only enriched my knowledge but have also inspired me to strive for excellence in all my future pursuits.

CERTIFICATE

This is to certify that **Mr. Shlok N. Tilokani** a student of Bachelors of Computer Application PRN 8021001295 in the Department of Computer Applications, Faculty of Science, The Maharaja Sayajirao University of Baroda, Vadodara, has successfully completed the Internship prescribed for the sixth semester of the program during the period from **25th of Dec 2023** to **10th of April 2024** towards the partial fulfilment for the degree of Bachelors of Computer Applications.

Date:

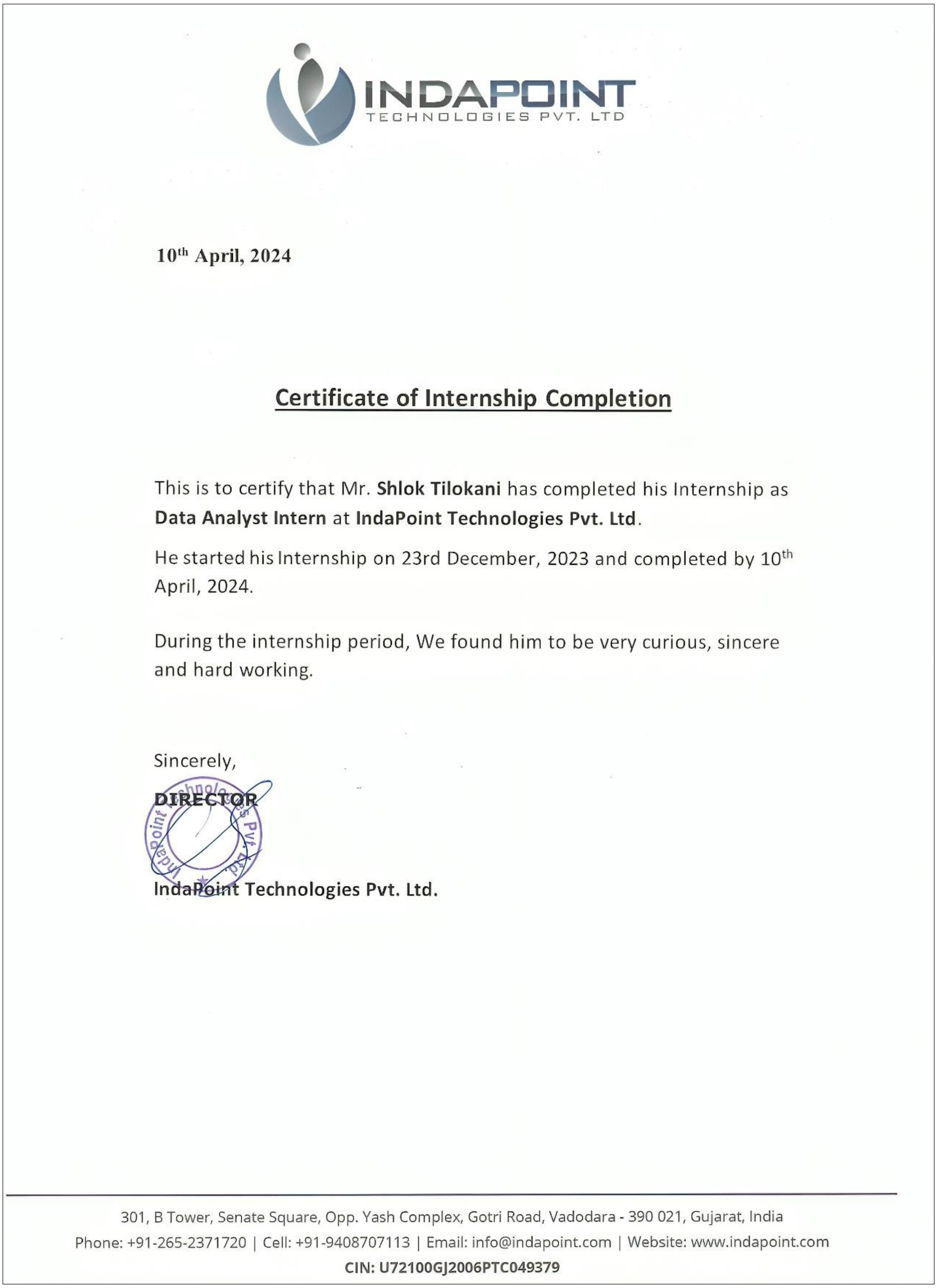
Name & Signature of the Guide

Ms. Poonam Shah

Head, Department of Computer Applications

Dr. P.K. Metha

CERTIFICATE BY ORGANIZATION



INDAPOINT
TECHNOLOGIES PVT. LTD

10th April, 2024

Certificate of Internship Completion

This is to certify that Mr. **Shlok Tilokani** has completed his Internship as **Data Analyst Intern** at **IndaPoint Technologies Pvt. Ltd**.

He started his Internship on 23rd December, 2023 and completed by 10th April, 2024.

During the internship period, We found him to be very curious, sincere and hard working.

Sincerely,

DIRECTOR

IndaPoint Technologies Pvt. Ltd.

IndaPoint Technologies Pvt. Ltd.

301, B Tower, Senate Square, Opp. Yash Complex, Gotri Road, Vadodara - 390 021, Gujarat, India
Phone: +91-265-2371720 | Cell: +91-9408707113 | Email: info@indapoint.com | Website: www.indapoint.com
CIN: U72100GJ2006PTC049379

Chapter 1: - INTRODUCTION TO ORGANIZATION

COMPANY NAME:	IndaPoint Technologies Pvt Ltd
COMPANY ADDRESS:	311-315, Third Floor, Ananta Stallion, Gotri - Sevasi Rd, near Sears Tower, above Westside, Gotri, Vadodara, Gujarat 390021
COMPANY PHONE:	09408707113
COMPANY EMAIL:	info@indapoint.com
COMPANY WEBSITE:	https://www.indapoint.com/
HEAD OFFICE:	311-315, Third Floor, Ananta Stallion, Gotri - Sevasi Rd, near Sears Tower, above Westside, Gotri, Vadodara, Gujarat 390021

IndaPoint Technologies Pvt Ltd, a dynamic Web 2.0 development company, is headquartered in Baroda (Vadodara), Gujarat, India, with a strategic presence in California, USA. With a rich legacy spanning over a decade, the company has been a trailblazer in professional software and web development services, serving a diverse clientele across the globe. Its team comprises seasoned professional's adept in a plethora of technologies, including PHP, .NET, Java, and other cutting-edge web technologies.

The company's journey has been characterized by an unwavering commitment to excellence and innovation. With a solid foundation laid over the years, IndaPoint has emerged as a trusted name synonymous with reliability, quality, and forward-thinking solutions. Its expertise extends beyond conventional software development; it acts as architects of digital transformation, shaping the future of technology.

At the core of its success lies a profound understanding of the constantly evolving tech landscape and a relentless pursuit of excellence. Leveraging its robust business domain experience and technical prowess, IndaPoint delivers progressive end-to-end web solutions that empower businesses to thrive in the digital age. It stays ahead of the curve, continuously adapting to the latest industry trends and technological advancements to ensure its clients remain at the forefront of innovation.

IndaPoint's service portfolio encompasses a comprehensive range of offerings, including cloud development, web development, mobile development, tech stack expertise, framework innovation, next-generation digital solutions, and enterprise mobility solutions. Whether clients are embarking on

new projects or seeking to enhance existing systems, IndaPoint provides tailored solutions that align with their unique business objectives and propel them towards success.

Moreover, IndaPoint is more than just a software company; it serves as a nurturing ground for talent and innovation. It provides a platform for passionate individuals and quick learners to cultivate their careers in information technology. By fostering a culture of continuous learning and growth, IndaPoint empowers its team members to unleash their full potential and drive impactful change.

As it looks to the future, IndaPoint remains steadfast in its commitment to delivering unparalleled value to its clients and fostering a culture of innovation and excellence. It stands ready to partner with businesses on their journey towards digital transformation, providing the expertise, insight, and support needed to turn their vision into reality.

Chapter 2: - Abstract

The HR Management Dashboard is a sophisticated web application developed using Python for its backend functionality, encompassing data analysis, and HTML, CSS, and JavaScript for the frontend user interface. It serves as an essential tool for Human Resources (HR) professionals, offering comprehensive insights derived from data acquired during employee exit interviews.

At its core, the Dashboard aggregates and analyses data collected through exit interviews, which encapsulate employees' sentiments and reasons for departing the organization. This data is pivotal in facilitating a deeper understanding of the underlying factors contributing to employee turnover.

By leveraging the analytical capabilities of the Dashboard, HR personnel can discern recurring patterns and trends within the data. This discernment enables informed decision-making processes aimed at addressing root causes of attrition and fostering strategies to enhance employee retention.

Consequently, the HR Management Dashboard plays a pivotal role in optimizing organizational strategies by providing actionable insights into employee sentiment and turnover dynamics. Its utilization empowers HR departments to implement targeted interventions and initiatives aimed at cultivating a more conducive and fulfilling work environment, thereby mitigating turnover rates and fostering long-term organizational sustainability.

Chapter 3: - Requirement Analysis

Project Purpose:

The objective of this project is to develop a web-based application tailored for HR data analysis. The system will enable users to log in, access HR survey data (such as exit interviews), and leverage the Python-based backend for comprehensive data analysis. The analysed data will then be presented through an intuitive dashboard using HTML.

Technical Specifications:

Backend

- **Language**: Python
- **Primary Libraries**: NumPy, Pandas, Matplotlib, Seaborn, SciPy, Scikit-learn
- **Functions**: Data processing, analysis, and visualization; API development for frontend integration

Frontend

- **Technology**: HTML
- **Features**: User authentication, data upload interface, dashboard for data visualization

Database

- **Database System**: MySQL
- **Purpose**: Store user credentials, raw data, and processed data

Workflow

1. **User Login**: Implement a secure authentication system.
2. **Data Processing**: Utilize Python scripts to process uploaded data.
3. **Analysis and Visualization**: Conduct data analysis and generate visual reports.
4. **Dashboard Display**: Present processed data through charts and tables on the user dashboard.

Types of Reports

1. **Graphical Representations**: Utilize pie charts, bar graphs, and line graphs to illustrate various metrics.
2. **Tabular Reports**: Provide detailed tables with sortable columns.
3. **Summary Reports**: Offer key insights and summaries of survey results.
4. **Trend Analysis Reports**: Analyse trends over time.

Data Analysis Techniques

- **Descriptive Analysis**: Summarize and describe different aspects of the data.
- **Predictive Analysis**: Employ statistical techniques and machine learning to predict future trends.
- **Sentiment Analysis**: Utilize sentiment analysis for qualitative data, such as text responses in surveys.
- **Correlation Analysis**: Determine relationships between different survey variables.

Dashboard Features

- **Interactive Charts**: Enable users to interact with data visualizations.
- **Customizable Views**: Allow users to filter and customize displayed data.
- **Real-time Data Refresh**: Automatically update data as new data is processed.

Additional Suggestions

- **Security Measures**: Implement robust data encryption and user authentication.
- **Responsive Design**: Ensure the dashboard is accessible across various devices.
- **User Feedback System**: Integrate a system for collecting user feedback for continuous improvement.

Potential Challenges

- **Data Privacy**: Ensure confidentiality and integrity of sensitive HR data.
- **Scalability**: Design backend to handle large data volumes efficiently.
- **User Experience**: Create an intuitive and user-friendly interface.

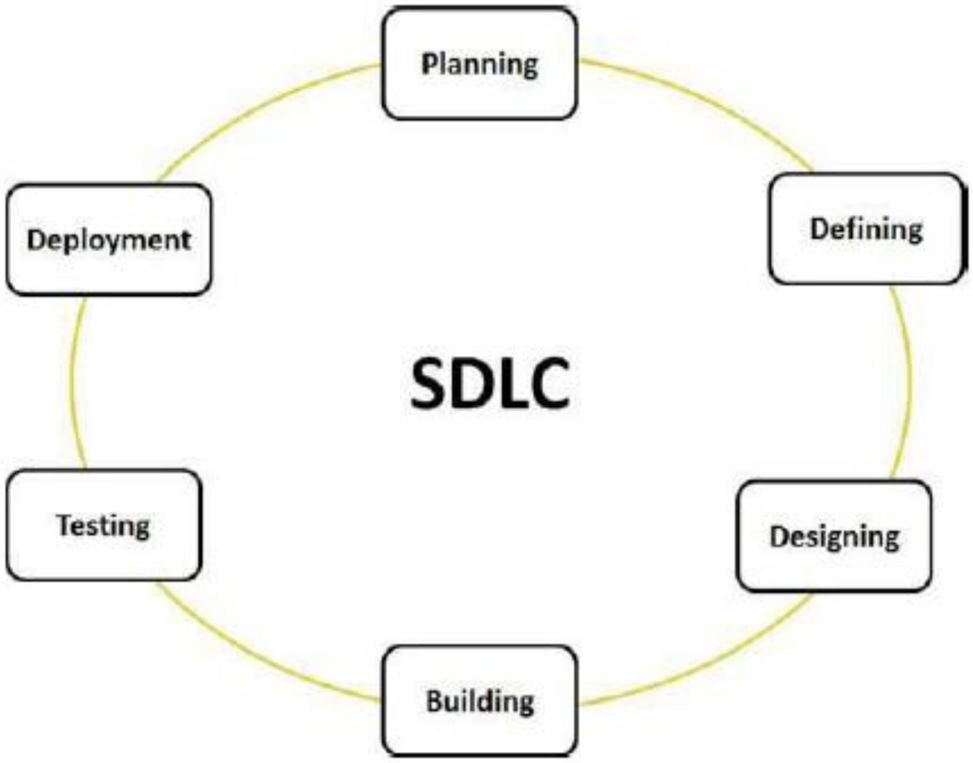
In conclusion, this project represents a significant advancement in the tools available to HR professionals for analysing survey data. By providing a comprehensive and user-friendly tool, it aims to empower HR professionals with the ability to conduct efficient and insightful analyses. With access to detailed data analysis, HR professionals will be better equipped to make informed decisions that are grounded in the insights derived from thorough data examination.

This tool not only streamlines the process of analysing survey data but also enhances the quality and depth of the insights that can be obtained. By utilizing this tool, HR professionals will be able to uncover trends, patterns, and correlations within the data that may not be immediately apparent. These insights will enable them to identify areas for improvement, make strategic decisions, and implement targeted interventions to address organizational challenges or capitalize on opportunities.

Overall, the implementation of this project will contribute to the enhancement of HR practices within the organization, fostering a data-driven approach to decision-making. By leveraging the power of data analysis, HR professionals can effectively navigate the complexities of workforce management, optimize organizational performance, and drive positive outcomes for the organization as a whole.

Chapter 4: - Development Technique (SDLC)

4.1 Software Development Lifecycle (SDLC)



The Software Development Life Cycle (SDLC) comprises all the essential activities involved in creating a software system, along with the specification of artefacts. An artefact is one of several tangible products generated during software development. Certain artefacts, such as use cases, class diagrams, various UML models, requirements, and design documents, aid in delineating the software's function, architecture, and design. A life cycle model, a representation of the SDLC, serves as an abstract depiction of the software development methodology, emphasizing which activities to execute and when. Life cycle models encompass the development process, management process, and supporting process.

The software development life cycle (SDLC) serves as a framework that outlines the tasks to be executed at each stage of the software development process. This life cycle establishes a methodology for enhancing software quality and improving the overall development process. SDLC is a procedural

framework adopted for software projects within software organizations. It encompasses a comprehensive plan detailing the approach for developing, maintaining, replacing, and modifying or enhancing specific software.

STAGE 1: PLANNING AND REQUIREMENT ANALYSIS:

Requirement analysis stands as a crucial initial phase in the SDLC, spearheaded by senior team members in collaboration with inputs from customers, sales departments, market surveys, and domain experts. This phase sets the foundation for planning the project approach and conducting a comprehensive product feasibility study, encompassing economic and technical aspects. It involves delineating quality assurance requirements and identifying project-related risks. The technical feasibility study's outcome defines various feasible technical approaches for implementing the project with minimal risks.

STAGE 2: DEFINING REQUIREMENTS:

Following requirement analysis, the subsequent step involves clearly defining and documenting product requirements, subject to approval from customers or market analysts. This process is facilitated through the Software Requirement Specification (SRS) document, consolidating all product requirements slated for design and development throughout the project lifecycle.

STAGE 3: DESIGNING THE PRODUCT ARCHITECTURE:

The SRS serves as a pivotal reference for product architects to devise the optimal product architecture. Based on the requirements outlined in the SRS, multiple design approaches for product architecture are typically proposed and documented in a Design Document Specification (DDS). The DDS undergoes rigorous stakeholder review, considering parameters such as risk assessment, product robustness, design modularity, budget constraints, and time constraints. The selected design approach delineates all architectural modules of the product, along with communication and data flow representation with external and third-party modules, if applicable. Internal design specifications for all modules are meticulously defined within the DDS.

STAGE 4: BUILDING OR DEVELOPING THE PRODUCT:

This phase marks the commencement of actual development, where the product is constructed. The programming code is generated by the DDS, leveraging coding guidelines stipulated by the organization. Programming tools such as compilers, interpreters, and debuggers are utilized for code

generation. Various high-level programming languages such as C, C++, Pascal, Java, and PHP are employed based on the software's nature.

STAGE 5: TESTING THE PRODUCT:

This stage, typically integrated into all SDLC stages in modern models, focuses solely on product testing. Here, defects are reported, tracked, rectified, and retested until the product aligns with quality standards delineated in the SRS.

STAGE 6: DEPLOYMENT IN THE MARKET AND MAINTENANCE:

Upon successful testing, the product is formally released into the market, often deployed in stages aligned with the organization's business strategy. Initial deployment may involve releasing the product to a limited segment for real-world User Acceptance Testing (UAT), followed by feedback-driven enhancements before wider market release. Post-release, maintenance activities cater to the existing customer base, ensuring sustained product performance and functionality.

4.2 Iterative Model

The Iterative Model is employed in this project to achieve its objectives.

In this model, you initiate by establishing certain software specifications and crafting the initial version of the software. If there arises a necessity to modify the software following the initial version, a new iteration is created to accommodate these changes. Each release within the Iterative Model concludes within a predetermined and consistent timeframe termed an iteration.

The Iterative Model permits revisiting previous stages to incorporate respective alterations. The culmination of the project yields a refreshed final output after the Software Development Life Cycle (SDLC) process.

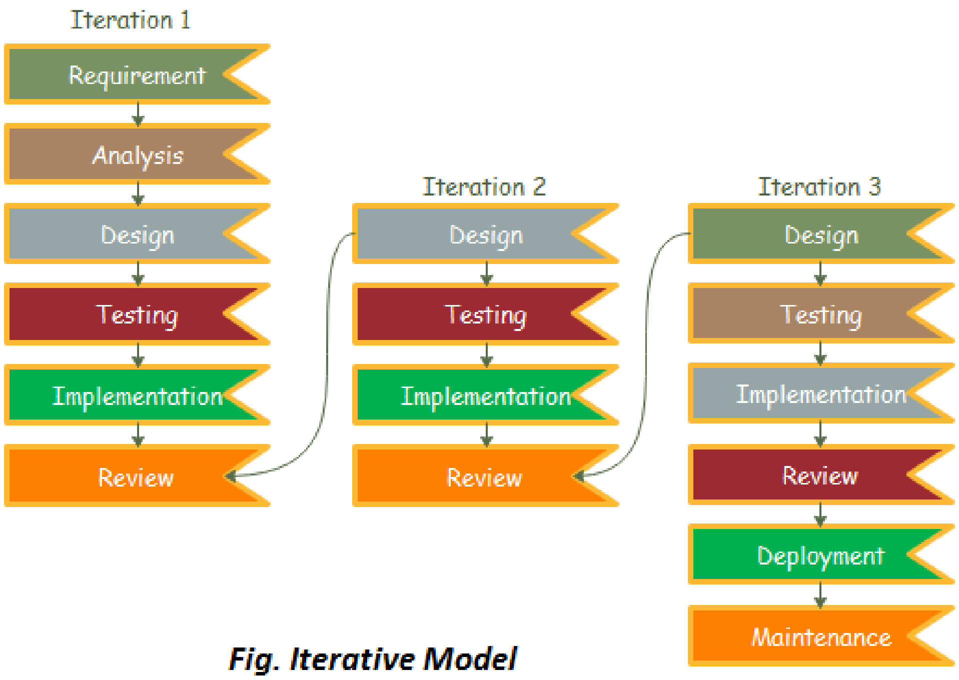


Fig. Iterative Model

STAGE 1: REQUIREMENT GATHERING & ANALYSIS:

During the Requirement Gathering & Analysis phase, the project team interacts with stakeholders, including customers and end-users, to elicit, document, and analyse their requirements. These requirements encompass functional and non-functional aspects of the software system. An analyst scrutinizes these requirements to ensure their feasibility within the project constraints, including budgetary considerations. This phase lays the foundation for the subsequent stages of software development.

STAGE 2: DESIGN:

In the Design phase, the project team conceptualizes and structures the software system based on the gathered requirements. Various modelling techniques and diagrams such as Data Flow diagrams, activity diagrams, class diagrams, and state transition diagrams are employed to represent different aspects of the system architecture. The design phase aims to create a blueprint for the software solution, ensuring alignment with the specified requirements and facilitating efficient implementation.

STAGE 3: IMPLEMENTATION:

Implementation involves the actual coding of the software based on the design specifications. Programmers transform the design artifacts into executable code using programming languages and development tools. This phase is characterized by the creation of software modules, functions, and algorithms that collectively form the software solution. Rigorous adherence to coding standards and best practices ensures the quality and maintainability of the codebase.

STAGE 4: TESTING:

Testing is a critical phase where the developed software undergoes rigorous evaluation to uncover defects, errors, and deviations from the expected behaviour. Various testing methodologies such as white box, black box, and grey box testing are employed to verify the correctness, reliability, and robustness of the software. Test cases are executed, and defects are identified, reported, and addressed iteratively to enhance the software quality.

STAGE 5: DEPLOYMENT:

Upon successful completion of testing and quality assurance activities, the software is deployed to its intended environment, whether it be a production server, client workstation, or cloud platform. Deployment involves the installation, configuration, and setup of the software system to enable end-users to access and utilize its functionalities effectively.

STAGE 6: REVIEW:

The Review phase occurs post-deployment and involves evaluating the performance, usability, and functionality of the software in a real-world environment. Feedback from end-users and stakeholders is collected to assess the software's effectiveness in meeting their needs and expectations. Any identified issues or deficiencies are documented and prioritized for resolution in subsequent iterations or updates.

STAGE 7: MAINTENANCE:

In the Maintenance phase, the software undergoes ongoing support and enhancement to address issues, bugs, and user feedback encountered during its operational lifespan. This phase involves debugging, troubleshooting, and applying patches to resolve software defects and ensure optimal performance. Additionally, new features, updates, and enhancements may be introduced to adapt the software to evolving user requirements and technological advancements. Regular maintenance activities are crucial for sustaining the longevity and relevance of the software solution.

Chapter 5: - Software Modules and Details

1. **Login Module**:
   - This module serves as the entry point for users, allowing them to authenticate and access the dashboard securely. By entering valid credentials, users gain authorized access to the dashboard's functionalities and features.

2. **Analysis Module**:
   - This module combines a MySQL database with Python logic, utilizing various libraries for data analysis and visualization. The MySQL database stores the data, while Python, equipped with libraries like Pandas and Matplotlib, conducts analysis and generates visual representations of the data. This integrated approach streamlines the process, allowing for efficient extraction of insights and informed decision-making.

3. **Graphs Module**:
   - In this section, users can explore a variety of graphical representations derived from data analysis performed using Python. These graphs provide visual insights into various aspects of employee data. Additionally, users can utilize the filter option to focus on specific departments, customizing the displayed graphs according to their preferences.

4. **Data Table Module**:
   - Here, users can access a comprehensive data table containing information gathered from employee exit interviews. The table offers a user-friendly interface, allowing users to easily navigate and explore the dataset. Additionally, the module provides filtering options for all columns, enabling users to refine their search criteria. Moreover, users can make multiple selections to further customize their data view, facilitating efficient data analysis and decision-making.

5. **Feedback Module**:
    - Integrated within the HTML layout, this module enhances user engagement by allowing them to provide feedback on individual records. A feedback button is conveniently placed at the end of each record, enabling users to share comments and insights regarding specific data entries. Clicking on the feedback button opens a new tab, where users can view comments submitted by different users for the corresponding record, fostering collaboration and knowledge sharing among users.

6. **Reports Module**:
    - This module offers comprehensive reports that combine graphical representations and concise data tables. Users can generate detailed reports by applying filters based on departmental criteria. By selecting specific departments, users can generate department-specific reports or opt for a general report encompassing all departments. This functionality enables users to gain valuable insights into departmental performance, facilitating informed decision-making and strategic planning within the organization.

Chapter 6: - Tools and Technologies

1. **Database:**
   - **MySQL Workbench:** MySQL Workbench serves as a visual database design tool, consolidating SQL development, administration, database design, creation, and maintenance within a unified development environment tailored for the MySQL database system.

2. **Frontend:**
   - **HTML:** HyperText Markup Language, commonly known as HTML, stands as the standard markup language for crafting documents intended for web browser display. It delineates the content and structure of web-based content, often complemented by technologies such as Cascading Style Sheets (CSS) and scripting languages like JavaScript.

   - **CSS:** Cascading Style Sheets (CSS) operates as a style sheet language utilized to define the presentation and styling of documents scripted in markup languages like HTML or XML. It serves as a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

   - **JavaScript:** JavaScript, abbreviated as JS, functions as a programming language and essential technology of the Web, alongside HTML and CSS. It finds ubiquitous use across 99% of websites for client-side webpage behaviour. Web browsers incorporate dedicated JavaScript engines responsible for executing client-side code. Additionally, technologies such as ***AJAX*** facilitate content updates without necessitating page reloads, while ***JSPDF*** facilitates customized styling and arrangement of PDF pages.

3. **Backend:**

    - **Flask:** Flask, a micro web framework authored in Python, is characterized by its minimalist nature, requiring no specific tools or libraries. It lacks a database abstraction layer and form validation, yet leverages pre-existing third-party libraries for common functionalities.

    - **MySQL Connector:** MySQL Connector, a software component developed by Oracle Corporation, enables programming languages supporting the ODBC interface to communicate seamlessly with MySQL databases.

4. **Analysis:**

    - **NumPy:** NumPy serves as a library for the Python programming language, enhancing support for large, multi-dimensional arrays and matrices. It boasts a vast collection of high-level mathematical functions tailored for array manipulation.

    - **Pandas:** Pandas stands as a Python software library designed for data manipulation and analysis. It provides specialized data structures and operations for manipulating numerical tables and time series data, and is distributed as free software under the three-clause BSD license.

    - **Matplotlib:** Matplotlib operates as a plotting library for Python and its numerical mathematics extension, NumPy. It offers an object-oriented API for embedding plots into applications, accommodating general-purpose GUI toolkits such as Tkinter, wxPython, Qt, or GTK.

    - **Seaborn:** Seaborn emerges as an open-source Python library dedicated to data visualization, built atop Matplotlib. It finds widespread application in data science and machine learning tasks, facilitating the creation of interactive plots conducive to querying data.

- **Scikit-learn:** Scikit-learn, an open-source Python library, caters to machine learning tasks. It boasts an extensive array of classification, regression, and clustering algorithms, including support vector machines, random forests, gradient boosting, k-means, and DBSCAN. Designed for seamless interoperability with Python's numerical and scientific libraries, namely NumPy and SciPy, it empowers practitioners in their machine-learning endeavours.

5. **Tools:**
    - **MySQL Workbench:** MySQL Workbench serves as a visual database design tool, consolidating SQL development, administration, database design, creation, and maintenance within a unified development environment tailored for the MySQL database system.

    - **VS Code:** Visual Studio Code, also commonly referred to as VS Code, is a source-code editor developed by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

    - **PyCharm:** PyCharm is an integrated development environment used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supports web development with Django. PyCharm is developed by the Czech company JetBrains.

    - **Draw.io:** Draw.io is a cross-platform graph drawing software developed in HTML5 and JavaScript. Its interface can be used to create diagrams such as flowcharts, wireframes, UML diagrams, organizational charts, and network diagrams

- **SequenceDiagram.org:** SequenceDiagram.org is a free online tool that allows you to create UML sequence diagrams using textual notation or drag and drop. The tool has an easy-to-use interface and can convert diagrams to text.

- **Excel:** Microsoft Excel is a spreadsheet editor developed by Microsoft for Windows, macOS, Android, iOS and iPadOS. It features calculation or computation capabilities, graphing tools, pivot tables, and a macro programming language called Visual Basic for Applications. Excel forms part of the Microsoft 365 suite of software.

- **PowerBI:** Microsoft Power BI is an interactive data visualization software product developed by Microsoft with a primary focus on business intelligence. It is part of the Microsoft Power Platform.

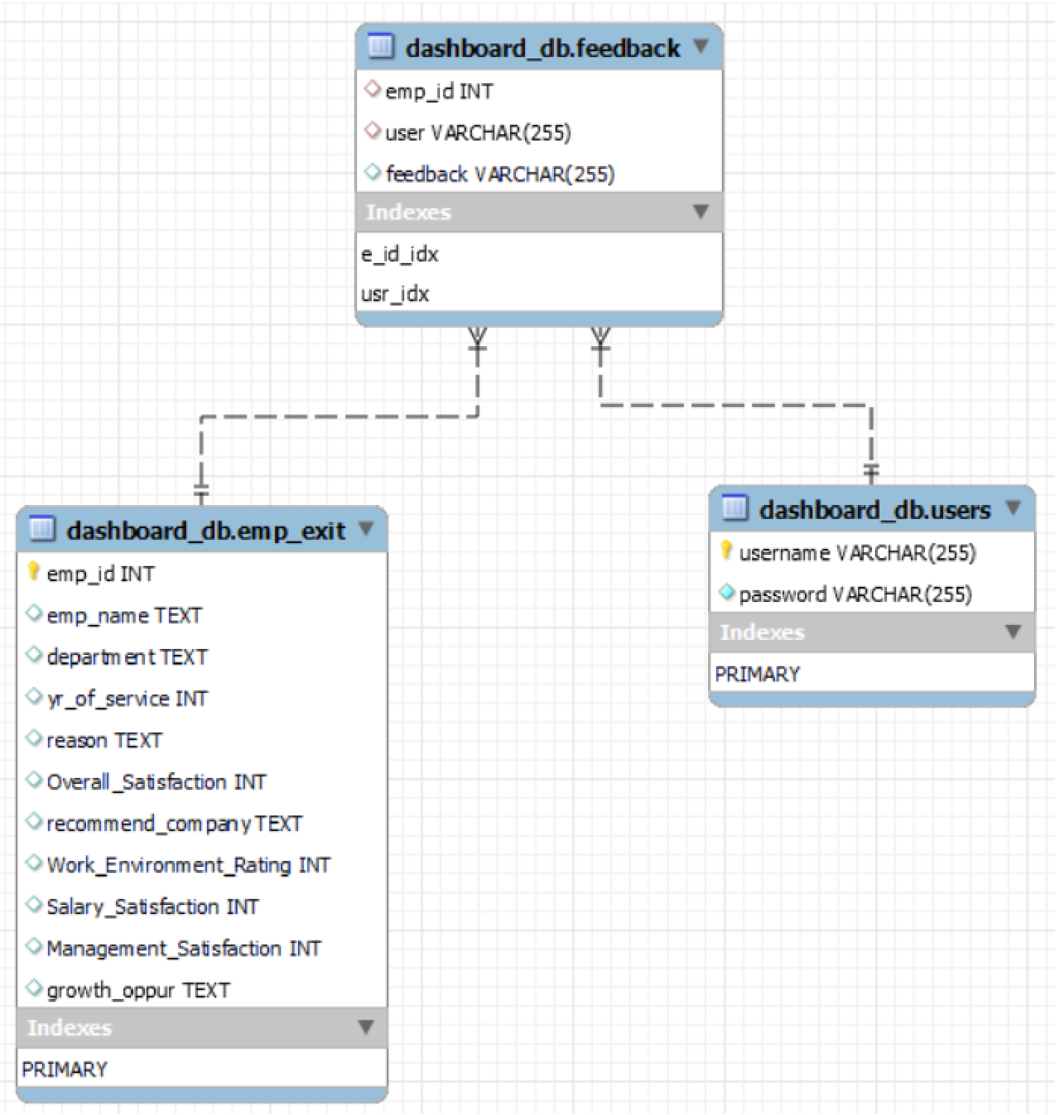
- **Jupyter Notebook:** Jupyter Notebook is a web-based application that allows users to create interactive documents that contain technical and data science content, such as code, visualizations, and media. The notebook combines rich text elements like tables, figures, and equations with computer code, such as Python, SQL, or R. The output of the code is displayed below the cell.

Chapter 7: - System Design of Modules

Diagrams serve as visual depictions of data, concepts, or processes, offering a clear and user-friendly means to communicate intricate information. They manifest in diverse formats, including flowcharts, bar graphs, pie charts, Venn diagrams, and numerous others, each uniquely suited to fulfil its intended function.

7.1 Database Schema

The database schema is the structure of a database described in a formal language supported typically by a relational database management system. The term "schema" refers to the organization of data as a blueprint of how the database is constructed.



Within the dashboard_db database, a conceptual schema is established, comprising three primary tables: users, emp_exit, and feedback.

1. **dashboard_db.users Table:**
   - This table is dedicated to storing user-related information pertinent to the dashboard application. It includes fields for username, password and a primary key for usernames, facilitating in avoiding data duplication.

2. **dashboard_db.emp_exit Table:**
   - The emp_exit table houses details concerning employee departures. It is structured with fields such as emp_id (possibly a foreign key referencing emp_id in the users' table), emp_name, department, yr_of_service, reason for leaving, Overall_Satisfaction, recommend_company (potentially a boolean type), Work_Environment_Rating, Salary_Satisfaction, Management_Satisfaction, and growth_oppur (possibly indicating growth opportunities). Additionally, it is defined with a primary key constraint on the emp_id column.

3. **dashboard_db.feedback Table:**
   - This table is designated to capture user feedback regarding the employee exit interview data. It features columns for emp_id (likely a foreign key referencing emp_id in the users table), user (potentially denoting the username of the feedback provider), and feedback text. Furthermore, indexes are incorporated on both emp_id and user columns to enhance data retrieval efficiency.

Foreign Key Constraints:

1. **emp_id in dashboard_db.emp_exit Table:**
    - This column is expected to serve as a foreign key referencing the emp_id column in the users' table. By enforcing this relationship, data integrity is maintained, ensuring that employee exit records are linked only to existing users within the system. Consequently, a record in the emp_exit table cannot possess an emp_id that lacks corresponding entries in the users' table.

2. **emp_id in dashboard_db.feedback Table:**
    - Similarly, this column is likely configured as a foreign key referencing the emp_id column in the users' table. This constraint safeguards that feedback entries are attributed solely to valid users. Thus, a record in the feedback table cannot feature an emp_id that does not correspond to existing user entries.

This schema delineates the underlying structure of a system aimed at tracking employee departures and aggregating user feedback within a dashboard application.

7.1.1 Field Description:

1. dashboard_db.emp_exit Table

Employee ID	States ID of the employee
Name	States employees' name
Department	States the department employee worked in
Years of Service	States Years the employee served before leaving the company
Reason for Leaving	States the reason for which the employee is leaving the company
Overall Satisfaction	Rating is given by the employee about his overall satisfaction experience on a scale of 1 to 5, 5 being the highest
Would Recommend Company	Employee selects either 'Yes' or 'No' if he would recommend the company to other potential candidates
Work Environment Rating	Rating is given by the employee about his Work Environment experience on a scale of 1 to 5, 5 being the highest
Salary Satisfaction	Rating is given by the employee about his Salary satisfaction, if the actual salary meets his desired salary on a scale of 1 to 5, 5 being the highest
Management Satisfaction	Rating is given by the employee about his management experience on a scale of 1 to 5, 5 being the highest
Opportunities for Growth	Employee selects from either option regarding the opportunity he got during his service period. Options – Adequate, Excellent, Limited, None

2. dashboard_db.users Table

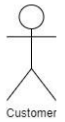
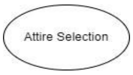
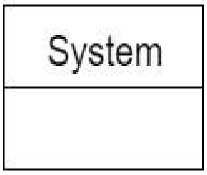
Username	Username of user
Password	Password of user

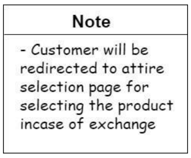
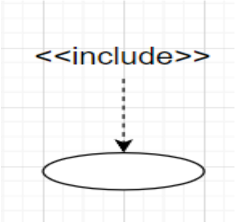
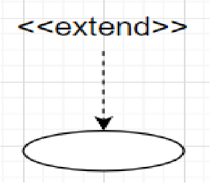
3. dashboard_db.feedback Table

Employee ID	Record in dashboard_db.emp_exit Table for which feedback is given
Username	Username of user giving the feedback
Feedback	Feedbacks given by the users

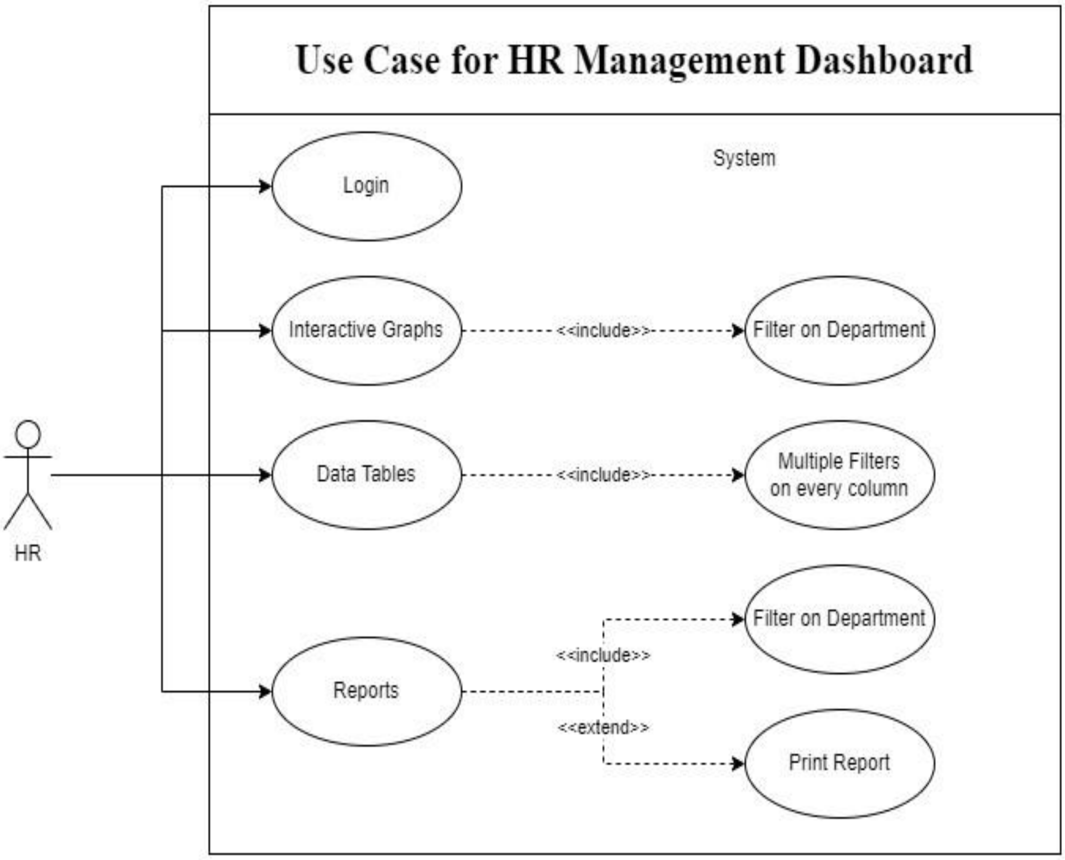
7.2 Use Case Diagram

A use case diagram serves as a comprehensive blueprint illustrating the myriad ways in which users or entities, referred to as actors, can engage with a system. It provides a visual roadmap of the system's capabilities and the diverse users it accommodates, thereby enabling a precise understanding of the system's requirements and ensuring they harmonize with user expectations. Moreover, it captures the essence of the system's functionalities and delineates its scope, facilitating seamless communication between stakeholders and developers. Furthermore, it functions as a proactive mechanism for identifying and rectifying potential errors or misconceptions that may crop up during the developmental phase, thereby streamlining the entire process.

Actor		Actors are external users who interact with a system and can take many forms, including individuals, organizations, or external systems. They must produce or consume data and are essential for the effective operation of the system.
Use Case		Use cases are written descriptions of how users will perform tasks on a website or system. They represent a specific situation in which the product or service could potentially be used and outline the system's behaviour in response to a user's request.
System Boundary		A system boundary is a rectangle in a use-case diagram that separates internal system use cases from external actors.

Note	 Note - Customer will be redirected to attire selection page for selecting the product incase of exchange	Comments in a use-case diagram provide textual information about the system's functionality or the actions being performed. They are an essential part of the diagram and aid in the understanding and interpretation of the model.
Include		An include relationship is a relation in which one use case includes the functionality of another use case
Extend		Extend is another relationship in use case diagrams, where one use case may add additional functionality to another use case.

7.2.1 Use Case for HR Management Dashboard



The use case diagram presented here illustrates how human resource (HR) professionals interact with an HR management dashboard to effectively manage employee information within an organization. Let's delve into each functionality outlined in the diagram:

Login: HR professionals begin by logging into the system securely. This login functionality ensures that only authorized personnel can access the dashboard and its functionalities, maintaining data security and confidentiality.

Filter on Department: Once logged in, HR professionals can utilize the filter on department functionality. This feature enables them to refine the data displayed on the dashboard based on specific departments within the organization. By doing so, HR professionals can focus their attention on particular departments or compare data across different departments, aiding in decision-making processes related to workforce management and resource allocation.

Multiple Filters on Every Column: The dashboard offers the capability to apply multiple filters on every column of the displayed data tables. This feature allows HR professionals to conduct in-depth analyses by narrowing down the dataset based on various criteria simultaneously. For example, they can filter employee data by department, job role, location, or any other relevant parameter, providing a granular view of the information and facilitating data-driven decision-making.

Interactive Graphs: The dashboard presents interactive graphs that visually represent various HR data trends, such as employee turnover rates, recruitment statistics, training progress, and more. These graphs offer a dynamic way to visualize and comprehend complex data patterns, enabling HR professionals to identify trends, anomalies, and areas requiring attention more efficiently. Interactive features may include zooming, panning, or hovering over data points to access detailed information.

Data Tables: In addition to graphical representations, the dashboard includes data tables that provide detailed information on the data depicted in the graphs. These tables offer a comprehensive view of the underlying data, allowing HR professionals to access specific details, conduct further analysis, and extract insights as needed. Data tables may include sortable columns, search functionalities, and other features to enhance usability and data exploration.

Reports: HR professionals can generate reports based on the data displayed in the dashboard. This functionality extends further by allowing users to print reports directly from the dashboard. The inclusion of the print report functionality ensures that HR professionals can document and preserve important insights, trends, and analysis findings for future reference, sharing with stakeholders, or compliance purposes.

Overall, by leveraging these functionalities, HR professionals can gain valuable insights into their organization's workforce dynamics and make informed, data-driven decisions regarding HR initiatives and strategies. For instance, they can use the dashboard to identify departments with high turnover rates, track the effectiveness of training programs, analyze recruitment trends, or monitor compliance with HR policies and regulations.

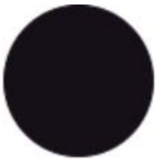
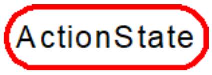
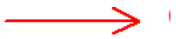
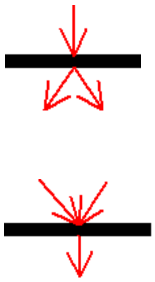
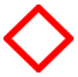
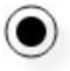
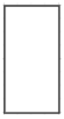
7.3 Activity Diagram

An activity diagram serves as an effective modelling tool for illustrating the dynamic behaviour of a system. It falls under the category of UML (Unified Modelling Language) diagrams, specifically designed to showcase the flow of control among different activities within a system. The primary objective of an activity diagram is to delineate the sequence of activities, decisions, and conditions that transpire throughout a process, from inception to completion.

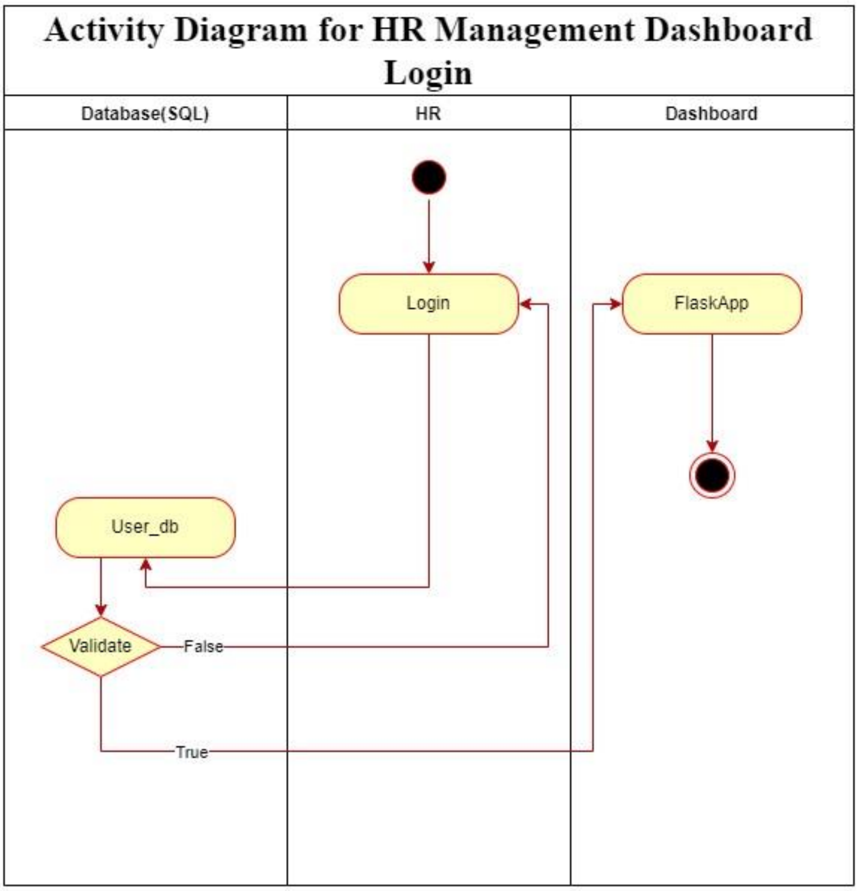
Comprising various shapes and symbols, the diagram represents distinct elements such as activities, transitions, and decision points. Activities embody the tasks to be executed within the system, while transitions illustrate the progression of control from one activity to another. Decision points serve to delineate branching paths within the process, contingent upon specific conditions.

A notable advantage of employing an activity diagram lies in its capacity to provide a comprehensive visualization of the entire system process, encompassing all decision points and potential outcomes. This facilitates the identification of potential issues or bottlenecks within the system and streamlines the process optimization for enhanced efficiency.

In essence, activity diagrams serve as a potent modelling tool, contributing to improved comprehension, design, and documentation of complex systems. By leveraging this diagram, stakeholders and team members can effectively communicate the system's behaviour, ensuring alignment of objectives and concerted efforts towards shared goals.

NAME	SYMBOL	DESCRIPTION
INITIAL STATE		A black circle is standard notation for initial state before an activity takes place. It can either stand alone or can be further elucidated using a note.
ACTION STATE		The activity symbols/ action states are the basic building blocks of an activity diagram and usually have a short description of the activity they represent.
CONTROL FLOW		Arrows represent the direction flow of the flow chart. The arrow points in the direction of progressing activities.
TRANSITION FORK		A transition fork splits one activity flow into two concurrent activities or vice versa.
DECISION		A marker shaped like a diamond is the standard symbol for a decision. There are always at least two paths coming out of a decision and the condition text lets you know which options are mutually exclusive.
FINAL STATE		The black circle that looks like a selected radio button is the symbol for end state of an activity.
SWIM LANES		The lanes are boundaries are drawn and the activities a particular category are drawn in the same lane as that of the category.

7.3.1 Activity Diagram for HR Management Dashboard - Login



This activity diagram delineates the sequential steps involved in the login process for accessing an HR management dashboard. Let's delve into the breakdown of each step with detailed explanations:

1. **Start:**
   - The login process initiates when the user navigates to the login page of the HR management dashboard, signalling the commencement of the authentication sequence.

2. **Login:**
   - Upon reaching the login page, the user is prompted to input their credentials, namely their email address and password, into the designated fields.

3. **Database Validation:**
    - Subsequently, the system proceeds to validate the entered login credentials by querying a designated database, typically referred to as User_db, which contains a repository of registered usernames and their corresponding passwords.

4. **Login Successful:**
    - If the provided credentials match a valid record within the database, the system confirms the login as successful. This validation step grants the user authorized access to the HR management dashboard.

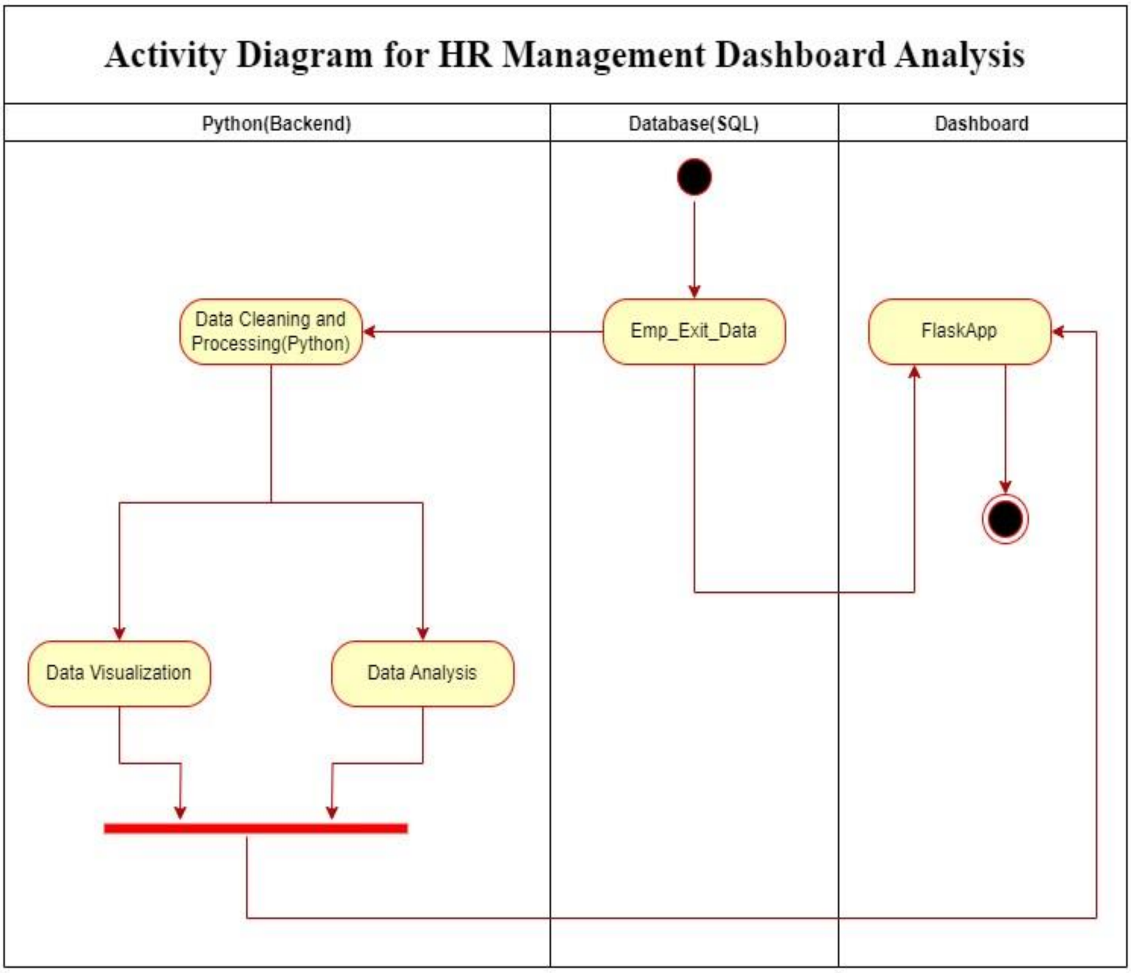
5. **HR Dashboard:**
    - Upon successful authentication, the system seamlessly transitions the user to the HR dashboard interface. Here, the user gains access to HR-related functionalities and data resources tailored to their role and permissions within the organization.

6. **Login Failed:**
    - Conversely, if the provided credentials do not align with any existing records within the database, the system identifies the login attempt as unsuccessful.

7. **End:**
    - The activity diagram concludes at this stage. In the event of a failed login attempt, the user may be presented with an error message indicating the unsuccessful authentication. At this point, the user can opt to retry the login process or seek assistance as necessary.

In essence, this activity diagram elucidates the intricacies of a streamlined login authentication process engineered specifically for the HR management dashboard. Through meticulous validation and seamless transition, users can securely access the dashboard's robust suite of HR management tools and functionalities, facilitating efficient workforce management and decision-making processes within the organization.

7.3.2 Activity Diagram for HR Management Dashboard - Analysis


Activity Diagram for HR Management Dashboard Analysis
Python(Backend)
Database(SQL)
Dashboard
Data Cleaning and Processing(Python)
Emp_Exit_Data
FlaskApp
Data Visualization
Data Analysis

This Activity Diagram illustrates the process of analyzing data within the HR management dashboard. It encompasses data cleaning and processing, data visualization, and data analysis. Let's delve into a detailed breakdown of each step involved:

1. **Start:**
   - The process commences with the existence of data stored in a database, typically SQL, which presumably contains HR-related information.

2. **Table:**
   - The Emp_Exit_Data table is presented within a Flask App interface, offering a user-friendly UI equipped with various options for filtering the table view according to specific criteria.

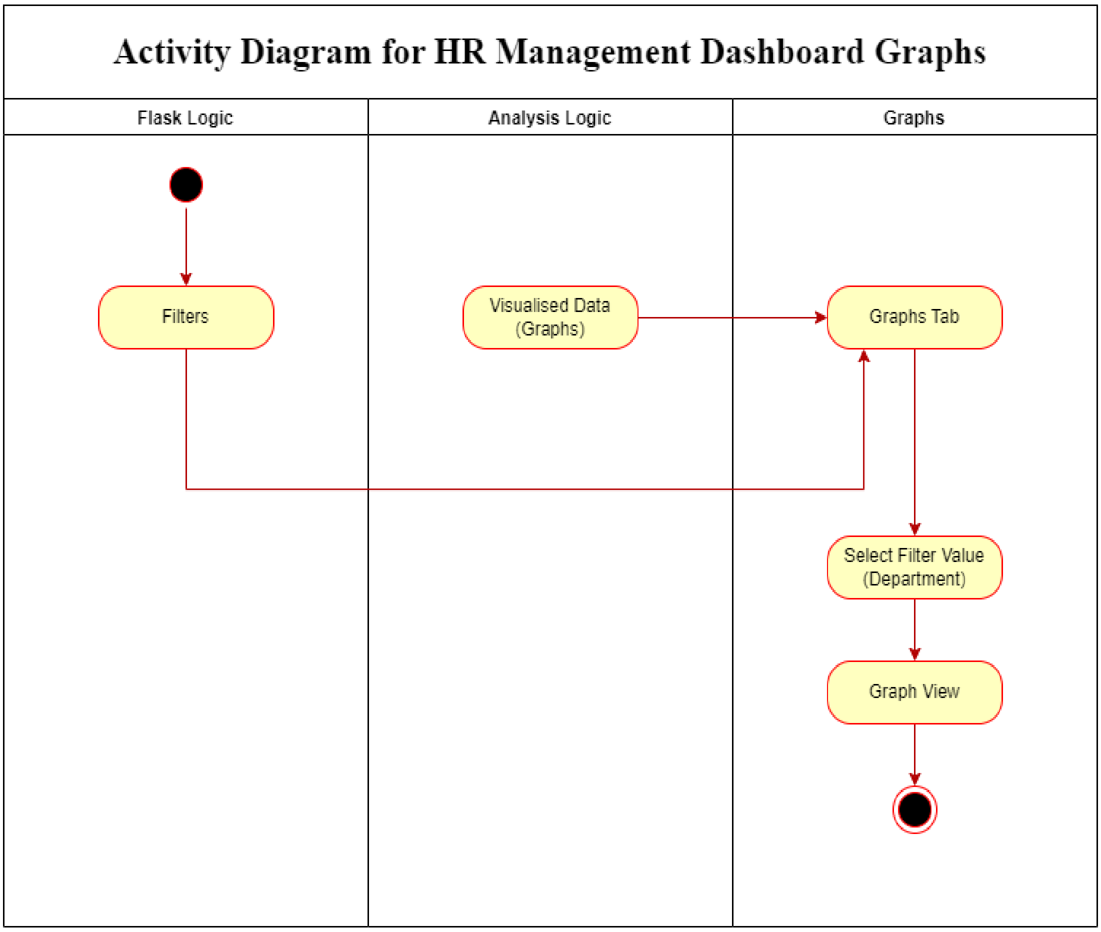
3. **Data Cleaning and Processing (Python):**
    - Subsequently, the data undergoes a transition to a Python backend for cleaning and processing. This phase involves preprocessing the data to enhance its quality and structure, making it suitable for predictive modelling, correlational analysis, and other advanced analytical techniques.

4. **Data Visualization & Data Analysis:**
    - The cleaned and processed data is utilized for performing data analysis and visualization following predefined requirements. Graphs, charts, and small data tables are generated to populate the Flask App interface, facilitating data exploration and insight generation.

5. **End:**
    - The activity diagram concludes at this stage, signifying the completion of the data analysis process. However, it's important to note that data analysis is an iterative process and can be revisited as needed to incorporate additional insights or refine existing analyses.

In essence, this activity diagram elucidates the technical workflow involved in data analysis within the HR management dashboard system. It begins with raw data stored in a database, progresses through cleaning and processing stages facilitated by Python, and culminates in the presentation of visualized data on the dashboard interface to support informed decision-making and strategic planning within the HR domain.

7.3.3 Activity Diagram for HR Management Dashboard - Graphs


Activity Diagram for HR Management Dashboard Graphs
Flask Logic
Analysis Logic
Graphs
Filters
Visualised Data (Graphs)
Graphs Tab
Select Filter Value (Department)
Graph View

This activity diagram depicts the process of creating and managing graphs for an HR management dashboard. It uses a top-down approach, where the high-level process is broken down into smaller sub-processes.

1. **Flask Logic:**
   - This lane represents the backend logic responsible for data processing and manipulation.

2. **Analysis Logic:**
   - This lane depicts the analytical calculations carried out on the data to derive insights.

3. **Graphs:**
   - This lane illustrates the visual representation of the data.

Here's a breakdown of the activities:

1. **Start:**
    - This marks the initiation of the process, involving the creation of graphs for the HR management dashboard commencing from the Analysis Logic lane.

2. **Filters (in the Flask Logic lane):**
    - This activity entails defining filters for data refinement, enabling users to customize graphs by narrowing down data.

3. **Visualized Data (in the Analysis Logic lane):**
    - This activity encompasses processing data based on applied filters and transforming it into visually understandable formats.

4. **Graphs (in the Graphs lane):**
    - This activity involves generating graphs based on the visualized data.

5. **Graphs Tab (in the Graphs lane):**
    - This activity focuses on displaying the generated graphs on a designated tab within the HR management dashboard.

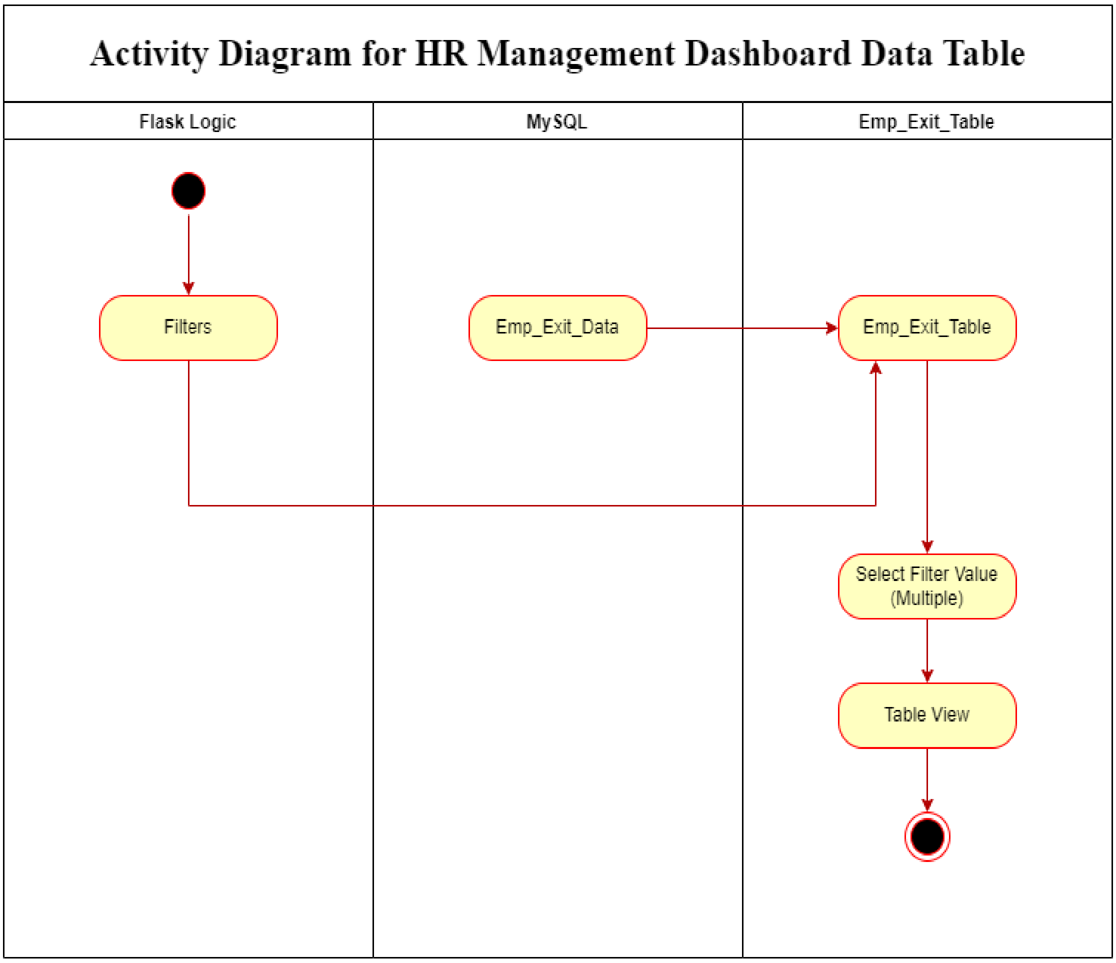
6. **Select Filter Value (Department) (in the Graphs lane):**
    - This external input enables users to choose specific filter values like department.

7. **End:**
    - The process ends with providing HR a general or filtered view of graphs.

The diagram implies that after creating dashboard graphs, users can select filter values triggering analysis logic to process data based on chosen filters, updating graphs on the Graphs tab.

In summary, this activity diagram provides a clear depiction of steps involved in creating and managing graphs for an HR management dashboard, emphasizing the interplay between backend logic, data analysis, and visual data representation.

7.3.4 Activity Diagram for HR Management Dashboard - Table


Activity Diagram for HR Management Dashboard Data Table
Flask Logic
MySQL
Emp_Exit_Table
Filters
Emp_Exit_Data
Emp_Exit_Table
Select Filter Value (Multiple)
Table View

This activity diagram delineates the workflow for an HR management dashboard data table. It leverages Flask logic, assumedly a Python-based web framework, to interface with a MySQL database table named Emp_Exit_Table.

1. **Start:**
   - The process commences with Flask logic, initiating the data retrieval process and adding filter options to the Table Tab.

2. **Filters:**
   - Following Flask logic, filters are applied to the Emp_Exit_Data, allowing users to refine data based on specific criteria.

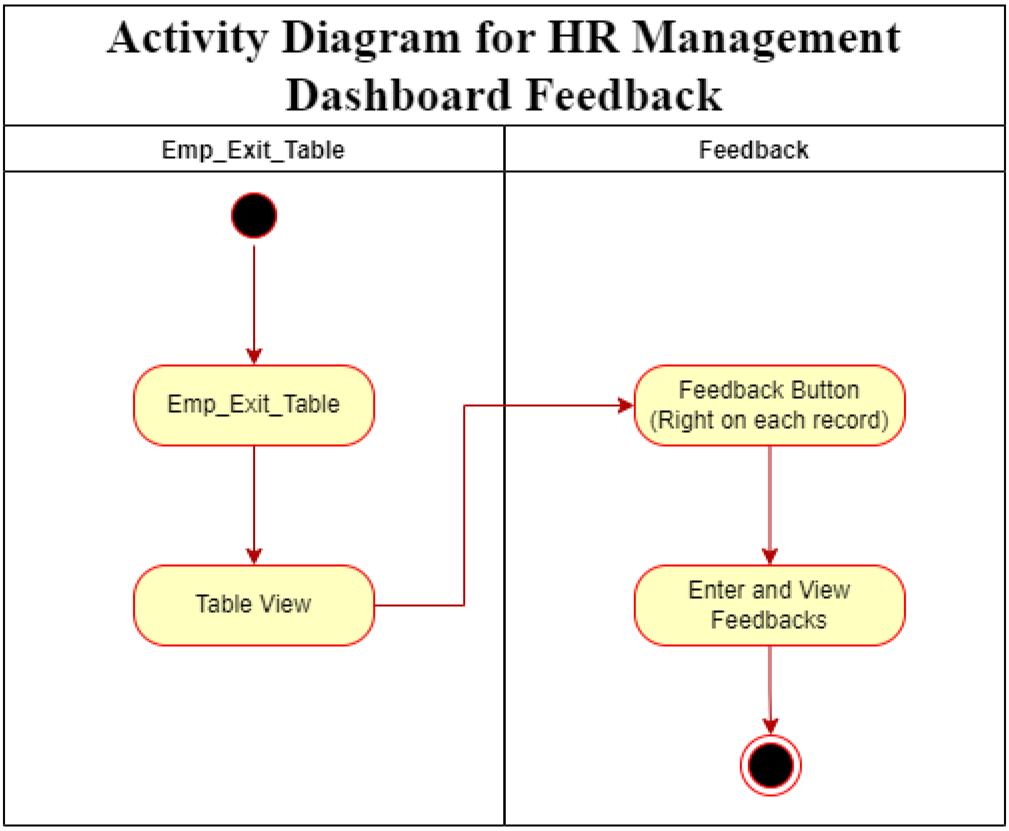
3. **Select Filter Value (Multiple):**
    - This step involves selecting multiple filter values, facilitating a more precise dataset.

4. **MySQL Emp_Exit_Table:**
    - Subsequently, the filtered data is utilized to query the Emp_Exit_Table in the MySQL database.

5. **End:**
    - Ultimately, the retrieved data is presented in a table view, presumably integrated into the HR management dashboard.

In summary, this activity diagram elucidates the steps entailed in filtering and presenting employee exit data on an HR management dashboard.

7.3.5 Activity Diagram for HR Management Dashboard - Feedback


Activity Diagram for HR Management Dashboard Feedback
Emp_Exit_Table
Feedback
Emp_Exit_Table
Table View
Feedback Button (Right on each record)
Enter and View Feedbacks

This is an activity diagram for an HR management dashboard feedback system. It outlines the process for how HR can provide feedback through a feedback button and how that feedback is stored.

1. **Start (Employee Exit Process (assumed)):**
   - An employee leaves the company, triggering an internal process (not shown) that fills the Emp_Exit_Table in the MySQL database. This table likely holds data about exiting employees.

2. **Feedback Button:**
   - Each record in the Emp_Exit_Table probably has a "Feedback Button" linked to it.

3. **HR Initiates Feedback:**
    - When an HR person clicks this button for a specific record, they move to the next step.

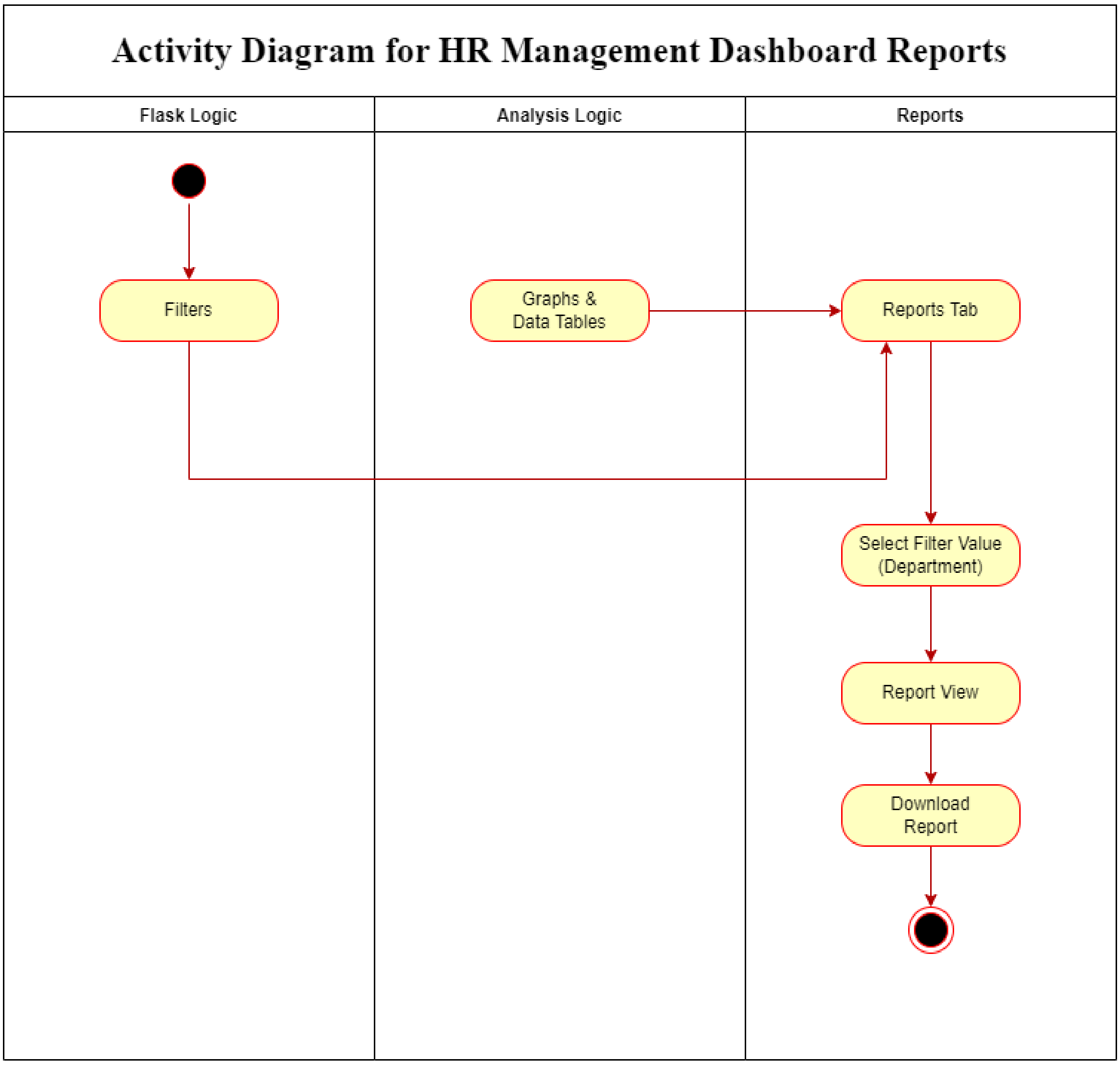
4. **Enter and View Feedback:**
    - This step shows a section where HR can type feedback about the employee's exit experience and see any previous feedback for that employee's record.

5. **Submit Feedback:**
    - Once the HR enters and submits their feedback, it's stored in a separate database.

6. **End:**
    - The process concludes with the employee exit feedback stored in the Feedback database. This data helps HR understand employee experiences and improve company processes.

This functionality facilitates the ability of individual HR personnel to conveniently access and review feedback submitted by their colleagues within a unified tab interface. HR staff can easily view and analyze the feedback provided by other HR members.

7.3.6 Activity Diagram for HR Management Dashboard - Reports



The activity diagram you sent me illustrates the process of generating HR management dashboard reports. It leverages three main components: Flask Logic, Analysis Logic, and Reports. Here's a breakdown of the workflow:

1. **Start:**
    - The process commences with Flask logic, initiating the data retrieval process and adding filter options to the Table Tab.

2. **Flask Logic and Analysis Logic**:
    - These sections work together to filter and analyze data relevant to the HR dashboard reports.

3. **Reports Tab**:
    - The processed data is then directed to the Reports tab.

4. **Filters**:
    - HR can apply additional filters here to refine the data displayed in the reports.

5. **Graphs & Data Tables**:
    - The filtered data is presented in multiple formats including graphs and data tables within the Reports tab.

6. **Download Report (Optional)**:
    - If desired, HR can download the report for further analysis or record keeping**.**

7. **End:**
    - The process concludes with HR viewing or downloading HR management dashboard reports.

This diagram shows how HR dashboard reports are built. Data is analyzed (by Flask Logic and Analysis Logic). Users then see reports with graphs and tables in the Reports tab. They can filter data further, explore it visually, and optionally download the report.

7.4 Sequence Diagram

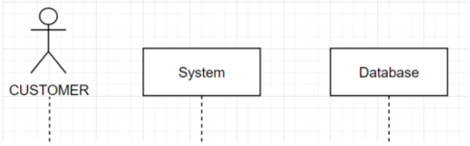
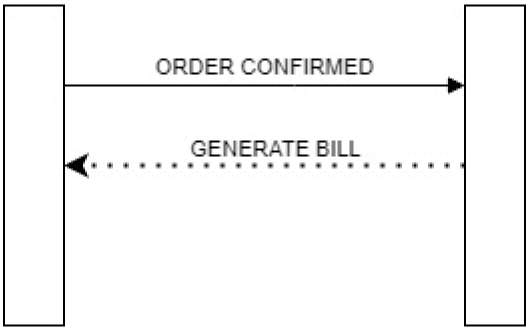
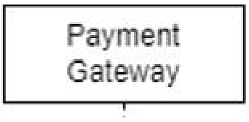
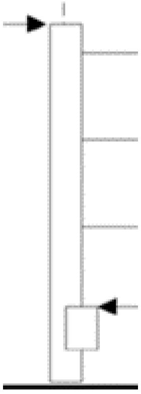
A sequence diagram is a graphical representation that effectively illustrates the interactions between different objects within a system. In this diagram, each object is depicted as a vertical line known as a lifeline, extending downwards on the page. The interactions between these objects are depicted as messages, represented by arrows drawn from the source lifeline to the target lifeline.

This visual representation allows for a clear depiction of the sequence of events and communication pathways between objects in a system. It enables stakeholders to understand how objects interact with each other and the order in which messages are exchanged.

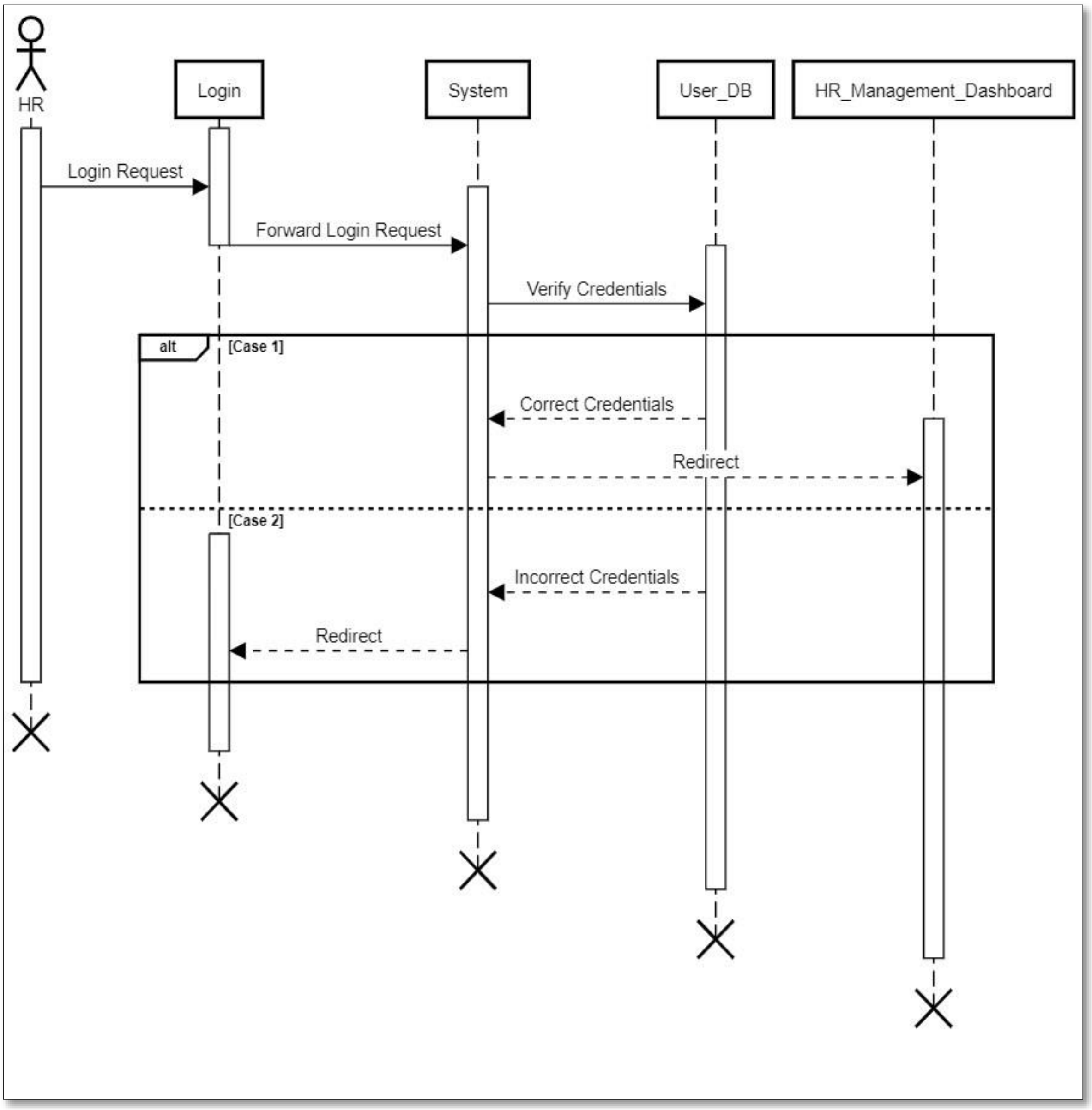
One of the key strengths of sequence diagrams lies in their ability to showcase object communication patterns. By visually representing how objects send and receive messages, sequence diagrams provide insights into the dynamic behavior of the system. This is particularly useful for understanding complex systems where multiple objects interact with each other in intricate ways.

Furthermore, sequence diagrams are invaluable for debugging and system analysis purposes. They provide a detailed overview of the flow of messages within the system, allowing developers to identify potential issues or errors in the communication process. By visualizing the sequence of interactions, developers can pinpoint areas of concern and make necessary adjustments to improve system functionality and performance.

In summary, sequence diagrams are powerful tools for visualizing object interactions within a system. They offer a comprehensive overview of communication patterns and message sequences, making them essential for system design, analysis, and troubleshooting.

NAME	SYMBOL	DESCRIPTION
TIMELINES		A Timeline represents an individual participant in a sequence diagram. A lifeline will usually have a rectangle containing its object name. If its name is "self", that indicates that the lifeline represents the classifier which owns the sequence diagram.
MESSAGES		A message represents an interaction between objects, or between an object and the environment. A message can be an event, a triggered operation, or a primitive operation. In the metamodel, a message defines a specific type of communication.
OBJECT		The sequence diagram consists of a group of objects that are represented by lifelines and the messages that they exchange over time during the interaction
FOCUS OF CONTROL		Focus of control (FOC) is used in sequence diagrams to show the period of time during which an object performs an action. FOC is rendered as a thin, rectangular object that sits on top of object lifelines

7.4.1 Sequence Diagram for HR Management Dashboard - Login



This Sequence Diagram provides an insightful breakdown of the intricate login process within an HR management system. Let's delve deeper into the interactions among the key actors and elements involved:

1. **HR:**
    - This pivotal actor embodies the user attempting to gain access to the HR management system, signifying the human interaction aspect of the login process.

2. **Login:**
    - Serving as a crucial functional component, the Login element encapsulates the mechanism through which users initiate the login process, an input form on the system's user interface.

3. **System:**
    - At the heart of the operation lies the System, representing the overarching infrastructure supporting the HR management dashboard. It serves as the central hub orchestrating interactions between various system components to facilitate seamless user authentication.

4. **User_DB:**
    - This critical element symbolizes the user database housing securely stored login credentials, such as usernames and passwords, which are essential for validating user access.

The sequential progression unfolds as follows:

1. **Login Request:**
    - The login journey commences as the HR user takes the initiative by inputting their credentials and triggering the login action. This action prompts the generation of a login request message, which is then dispatched to the Login element for processing.

2. **Forward Login Request:**
    - Subsequently, the Login component forwards the received login request to the System, transitioning the authentication process to the system level.

3. **Verify Credentials:**
    - Upon receipt of the login request, the System diligently interfaces with the User_DB component to validate the provided credentials. This

crucial step involves cross-referencing the entered username and password with the stored records within the user database.

4. **Correct Credentials (Case 1):**
   - If the provided credentials align precisely with a corresponding record in the User_DB, a confirmation message is relayed back to the System. This affirmative response signals the successful verification of user credentials.

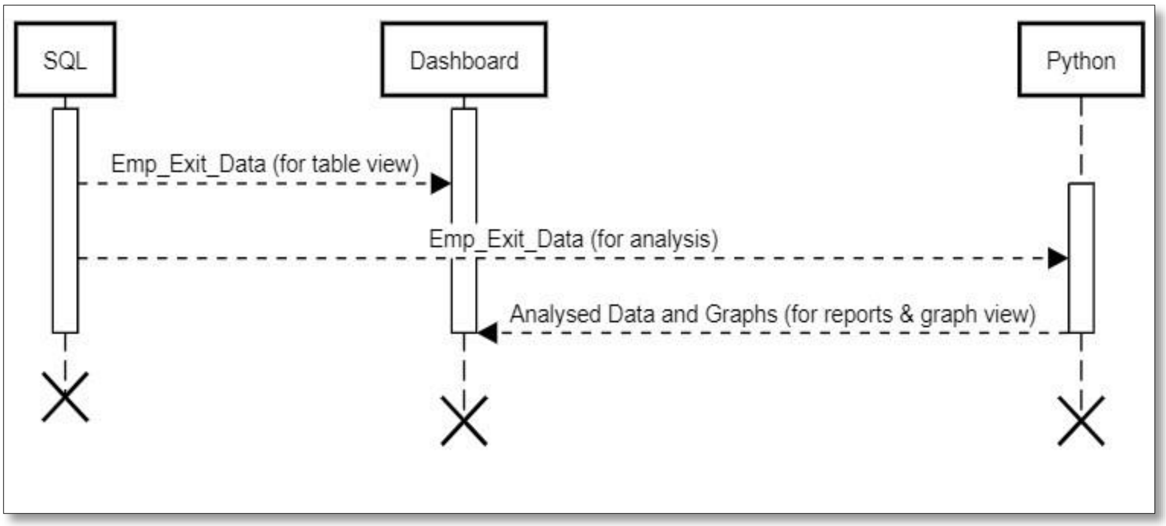
5. **Redirect to HR Management Dashboard (Case 1):**
   - With the confirmation of correct credentials, the System promptly redirects the user to the eagerly awaited HR Management Dashboard. This pivotal moment marks the culmination of the login process, granting the user access to a wealth of HR-related functionalities and data resources.

6. **Incorrect Credentials (Case 2):**
   - Conversely, if the entered username and password fail to match any existing records within the User_DB, an error notification is promptly dispatched back to the System, indicating the presence of incorrect credentials.

7. **Redirect to Login (Case 2):**
   - In response to the notification of incorrect credentials, the System takes proactive measures to guide the user back to the Login element. This redirection serves as a gentle nudge for the user to re-enter their credentials accurately, thereby facilitating a successful login attempt.

In conclusion, this sequence diagram illustrates a typical login process where the system authenticates the user's credentials against a database and redirects them accordingly.

7.4.2 Sequence Diagram for HR Management Dashboard - Analysis


SQL
Dashboard
Python
Emp_Exit_Data (for table view)
Emp_Exit_Data (for analysis)
Analysed Data and Graphs (for reports & graph view)

This sequence diagram provides a detailed portrayal of the process involved in retrieving data from a database, analyzing it using Python, and subsequently displaying it on a dashboard. Let's explore the intricate interaction among the various elements:

1. **SQL:**
    - This element signifies the database system responsible for storing the employee exit data securely.

2. **Dashboard:**
    - Representing the user interface, the Dashboard element is instrumental in presenting the retrieved data and any subsequent analysis results to users in a comprehensible format.

3. **Python:**
    - Symbolizing a Python program, this element plays a crucial role in retrieving, analyzing, and processing the data retrieved from the database.

The sequence unfolds methodically as follows:

1. **Retrieve Employee Exit Data (SQL to Dashboard):**
    - The sequence commences with the activation of the SQL element, which then proceeds to transmit a message labeled "Emp_Exit_Data (for table view)" to the Dashboard. This communication indicates that SQL is transmitting employee exit data to the Dashboard, presumably for display in a tabular format.

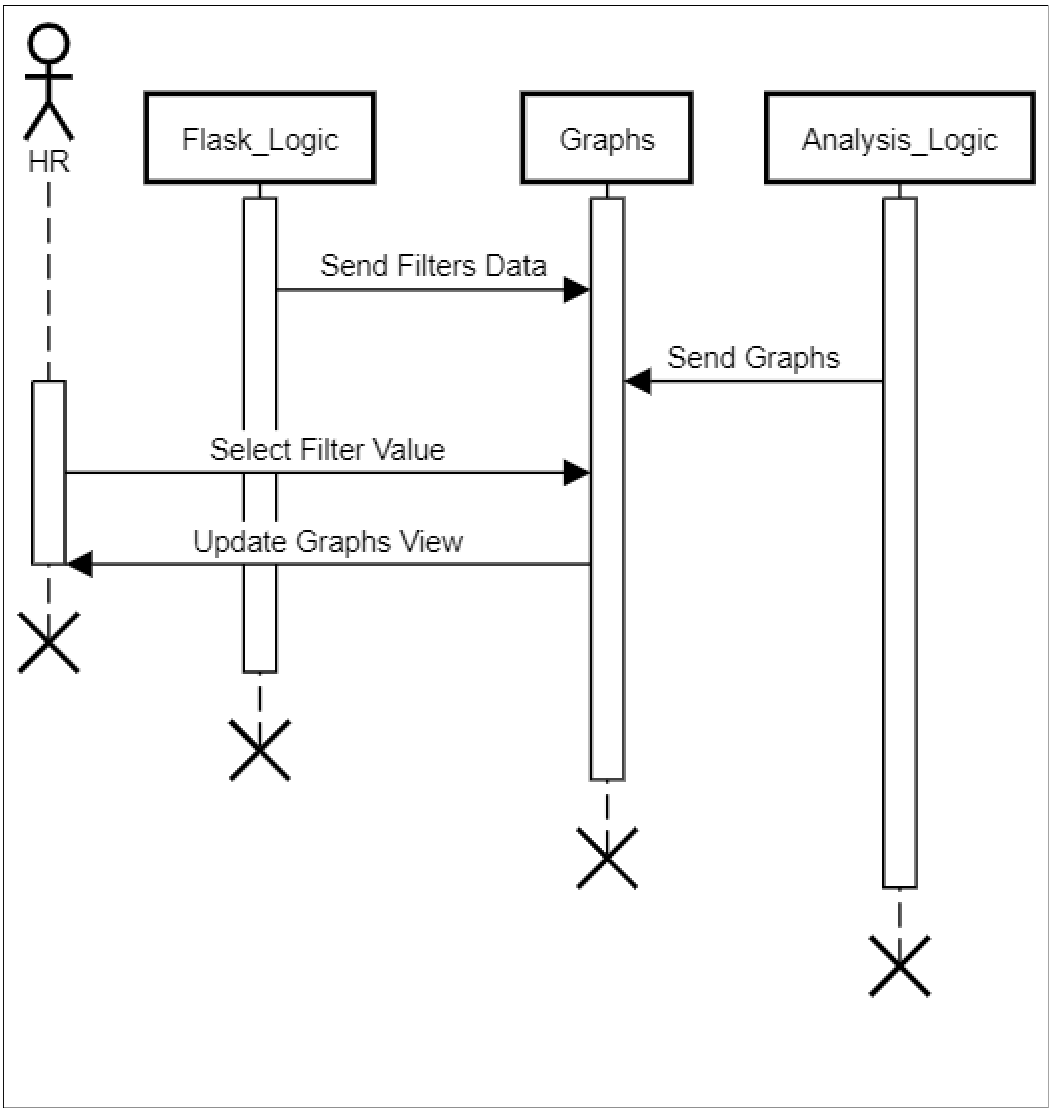
2. **Retrieve Employee Exit Data (SQL to Python):**
    - Concurrently, SQL also dispatches a message labeled "Emp_Exit_Data (for analysis)" to the Python element. This signifies SQL's transmission of the same employee exit data to Python for in-depth analysis purposes.

3. **Analyze Data (Python to Dashboard):**
    - Following the reception of the data, Python engages in thorough analysis and processing. Subsequently, Python dispatches the "Analyzed Data and Graphs (for reports & graph view)" message back to the Dashboard. This communication signifies Python's generation of reports and graphs based on the analysed data, which are then presented to users via the Dashboard.

4. **Deactivate and Destroy Elements:**
    - Once the data retrieval, analysis, and presentation processes are successfully executed, all three elements—SQL, Dashboard, and Python—are systematically deactivated and eventually destroyed, marking the conclusion of the sequence.

In summary, this detailed sequence diagram elucidates a meticulously orchestrated workflow wherein data is retrieved from a database, processed and analyzed by Python, and finally presented on a user-friendly dashboard interface. Through this systematic process, users gain access to insightful analyses and visualizations, enhancing their ability to interpret and derive actionable insights from the data presented.

7.4.3 Sequence Diagram for HR Management Dashboard - Graphs



This sequence diagram illustrates the interaction among four participants: HR, Flask Logic, Analysis Logic, and Graphs, demonstrating the flow of data used in generating graphs.

1. **HR**:
   - This actor represents a Human Resource professional who interacts with the system.

2. **Flask_Logic**:
   - This represents a web framework or a similar component that handles user interaction and data flow.

3. **Graphs**:
    - This represents a component responsible for generating and manipulating data visualizations (like charts).

4. **Analysis_Logic**:
    - This component handles any calculations or transformations needed for the graphs based on the selected filters.

Here is a breakdown of the process:

1. **Activation**:
    - All actors are activated, suggesting they are loaded or prepared for interaction.

2. **Filter Data**:
    - Through Flask logic, filters are applied to the Emp_Exit_Data, allowing users to refine data based on specific criteria.

3. **Send Graphs**:
    - Analysis_Logic is responsible for processing the filtered data and generating the corresponding graphs.

    - It then sends these graphs to the Graphs component for visualization.

4. **HR Interaction and Update**:
    - HR interacts with the Graphs component, by selecting a specific filter value from the available options.

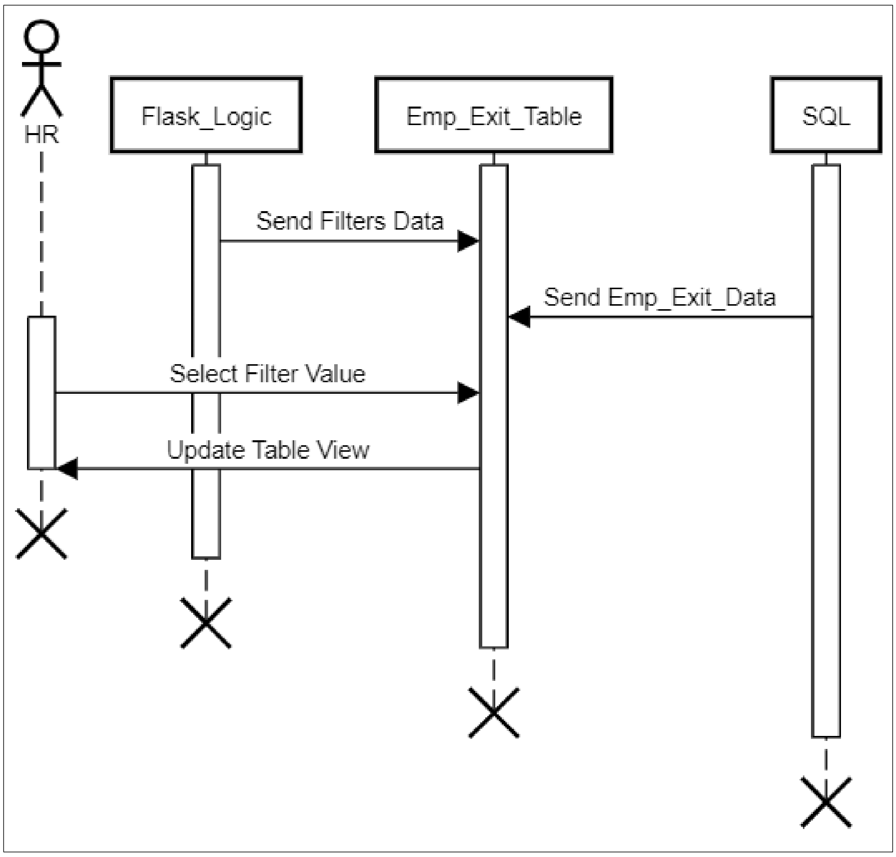
5. **Dynamic Update**:
    - Based on the chosen filter value, Graphs retrieves the relevant data and updates the graphs view for HR.

    - This means the graphs displayed to HR change dynamically based on their selection.

6. **Deactivation and Destruction**:
    - Once the interaction is complete, all actors are deactivated, indicating they are no longer actively needed.

In summary, this sequence diagram depicts an interaction between HR and a data visualization system. Initially, HR employs filters to narrow down the desired data. Subsequently, the system generates graphs based on the applied filters, updating the display accordingly. Through the Graphs tab, users can access real-time graphs and observe filtered views of the data.

7.4.4 Sequence Diagram for HR Management Dashboard - Table



The sequence diagram depicts a system designed for HR professionals to interactively explore employee exit data using filters.

1. **HR**:
   - Represents a human resource professional interacting with the system.

2. **Flask_Logic**:
   - This represents a web framework or a similar component that handles user interaction and data flow.

3. **Emp_Exit_Table**:
   - A component responsible for displaying employee exit data in a tabular format.

4. **SQL**:
   - Represents a database or a component interacting with a database to retrieve employee exit data.

Here is a breakdown of the process:

1. **Activation**:
    - All four actors are activated, suggesting they are prepared for interaction.

2. **Sending Filter Data**:
    - Through Flask logic, filters are applied to the Emp_Exit_Data, allowing users to refine data based on specific criteria.

3. **Retrieving Emp_Exit_Data**:
    - SQL sends employee exit data to Emp_Exit_Table, likely based on the received filter criteria.

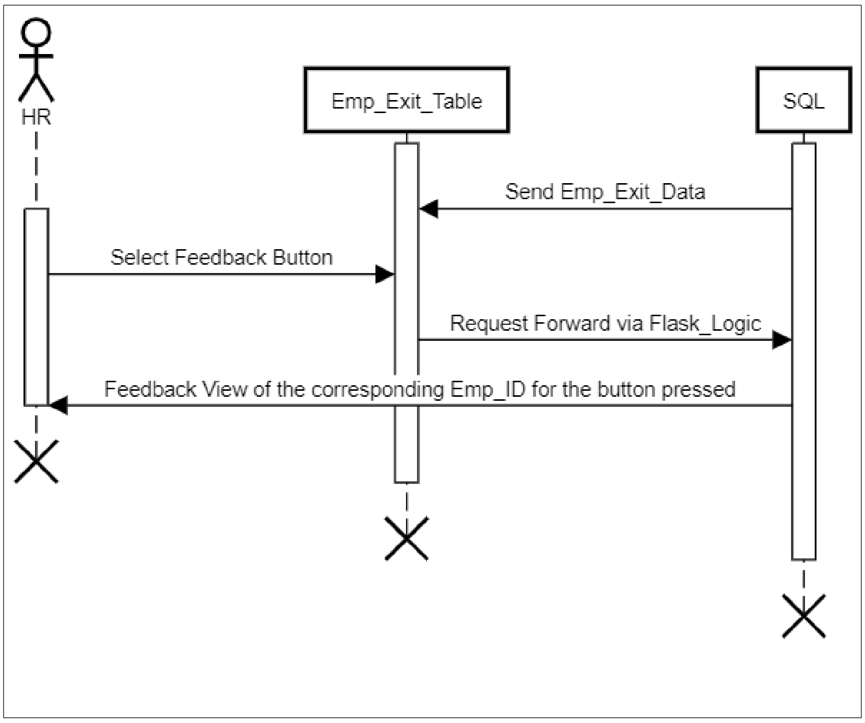
4. **HR Interaction**:
    - HR interacts with Emp_Exit_Table, likely by selecting filter values for further refinement.

5. **Updating Table View**:
    - Emp_Exit_Table dynamically updates the table view for HR, based on the selected filter values and the retrieved data.

6. **Deactivation and Destruction**:
    - Once the interaction is complete, all actors are deactivated and eventually destroyed, potentially for resource management.

In conclusion, the sequence diagram illustrates a system tailored for HR professionals to actively delve into employee exit data through interactive filtering. This system empowers HR to efficiently view, refine, and assess employee departure information, presumably to inform workforce management strategies and decision-making processes.

7.4.5 Sequence Diagram for HR Management Dashboard - Feedback


HR
Emp_Exit_Table
SQL
Send Emp_Exit_Data
Select Feedback Button
Request Forward via Flask_Logic
Feedback View of the corresponding Emp_ID for the button pressed

The sequence diagram depicts a system enabling HR professionals to access and review feedback related to employee exits. It allows HR to easily pinpoint specific employee feedback for evaluation, likely to inform decision-making processes and improve workforce strategies.

1. **HR:**
    - Represents a human resource professional interacting with the system.
2. **Emp_Exit_Table:**
    - A component responsible for displaying employee exit data and providing feedback options.
3. **SQL:**
    - Represents the database storing employee exit data, feedback, and related information.

Here is a breakdown of the process:

1. **Activation:**
    - HR, Emp_Exit_Table, and SQL are activated, suggesting they are prepared for interaction.

2. **Retrieving Exit Data:**
    - SQL sends employee exit data to Emp_Exit_Table, likely for initial display.

3. **HR Interaction:**
    - HR interacts with Emp_Exit_Table, specifically by selecting a feedback button associated with a particular employee.

4. **Feedback Request:**
    - Emp_Exit_Table forwards a feedback request to SQL, likely via a web framework or component called Flask_Logic.

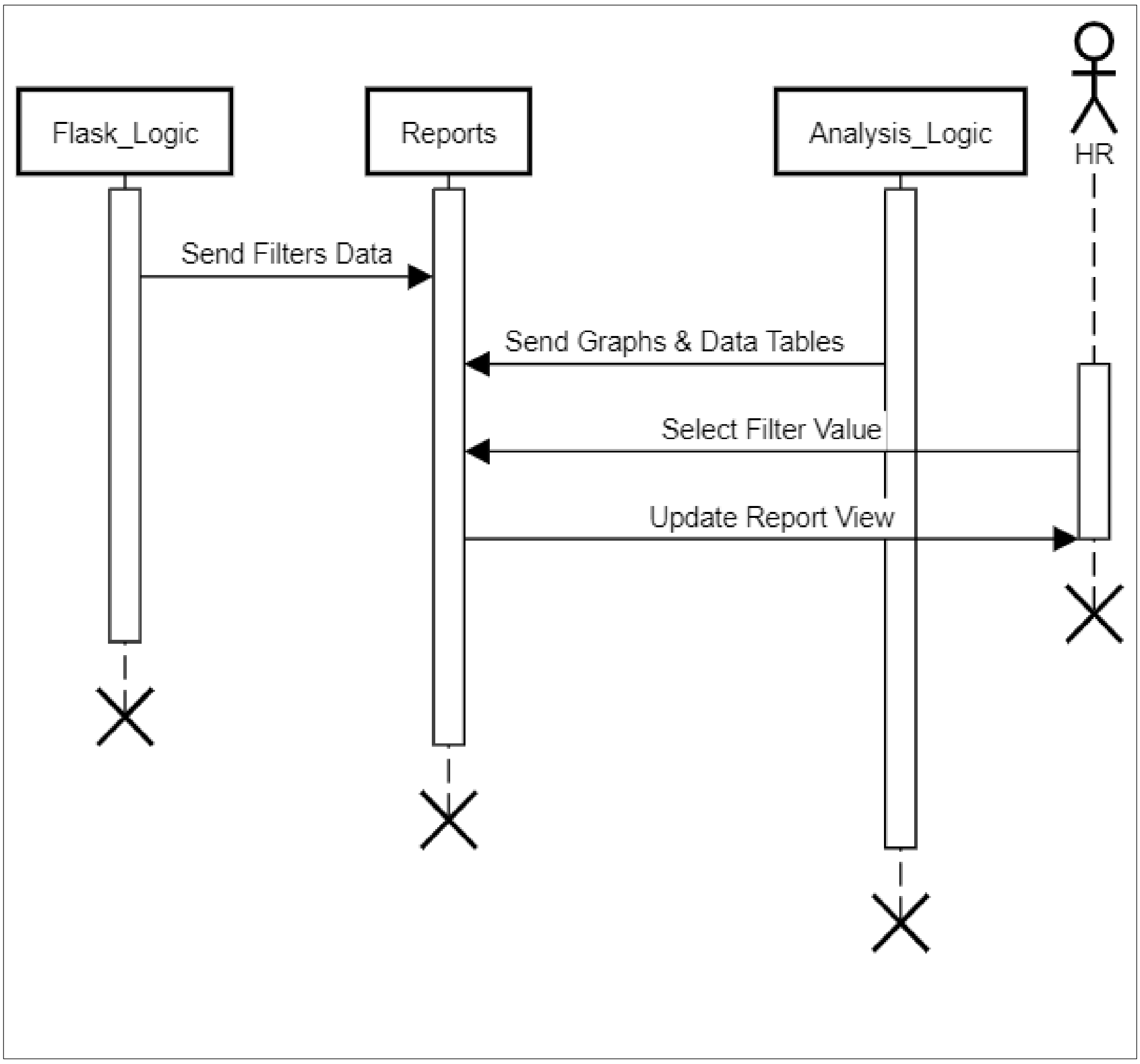
5. **Feedback Retrieval:**
    - SQL retrieves the feedback view corresponding to the employee ID for the button pressed.

6. **Displaying Feedback:**
    - SQL sends the retrieved feedback view to HR for review.

7. **Deactivation and Destruction:**
    - Once the feedback interaction is complete, all actors are deactivated and eventually destroyed, potentially for resource management.

In summary, the sequence diagram portrays a system that empowers HR professionals to access and assess feedback concerning employee departures. It provides HR with a straightforward means to identify particular employee feedback for assessment, presumably contributing to informed decision-making and enhancements in workforce strategies.

7.4.6 Sequence Diagram for HR Management Dashboard - Reports



This sequence diagram depicts a user-driven data visualization and reporting scenario. HR refines the data using filters, and the system updates the reports accordingly for interactive exploration.

1. **HR:**
   - Represents a human resource professional interacting with the system.

2. **Flask_Logic:**
   - This represents a web framework or a similar component that handles user interaction and data flow.

3. **Reports:**
    - A component responsible for generating reports consisting of data tables and graphs.

4. **Analysis_Logic:**
    - This component handles calculations or transformations needed for the reports based on the selected filters.

Here is a breakdown of the process:

1. **Activation**:
    - All four actors are activated, signifying they are prepared to interact.

2. **Sending Filter Data**:
    - Through Flask logic, filters are applied in the Reports Tab, allowing users to refine data based on specific criteria.

3. **Sending Graphs & Data Tables**:
    - Analysis_Logic is responsible for processing the filtered data and generating the corresponding reports, including graphs and data tables.

    - It then sends these reports to the Reports component.

4. **HR Interaction and Update**:
    - HR interacts with the Reports component, by selecting a specific filter value from the available options.

5. **Dynamic Report Update**:
    - Based on the chosen filter value, Reports retrieves the relevant data and updates the report view for HR.

    - This means the reports displayed to HR change dynamically based on their selection.

6. **Deactivation and Destruction**:
    - Once the interaction is complete, all actors are deactivated, indicating they are no longer actively needed.

    - The "destroy after" notation suggests that these actors might be garbage collected after deactivation to free up resources.

In summary, the sequence diagram offers a detailed portrayal of a user-centric approach to data visualization and reporting. Within this scenario, HR actively engages with the system by employing filters to refine the dataset according to specific criteria. Subsequently, the system dynamically adjusts the generated reports to reflect these refined parameters, facilitating an interactive and tailored exploration of the data by HR professionals.

7.5 Interaction Overview Diagram

An Interaction Overview diagram is a type of UML (Unified Modeling Language) diagram that provides an overview of the flow of interactions between elements in a system or within a particular scenario. It combines elements of both activity diagrams and sequence diagrams to depict the high-level interactions between components, objects, or actors in a system.

Key features of Interaction Overview diagrams include:

1. **Interaction Fragments**:
    - Interaction Overview diagrams use interaction fragments to represent individual interactions or sequences of interactions. These fragments can include elements such as messages, method calls, decisions, loops, and parallel behaviors.

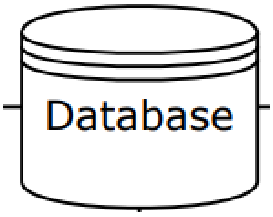
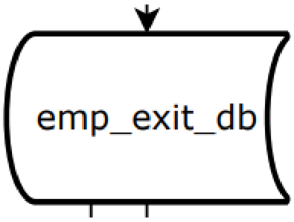
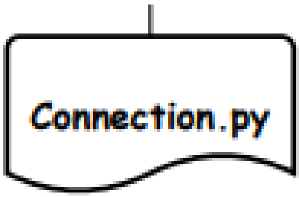
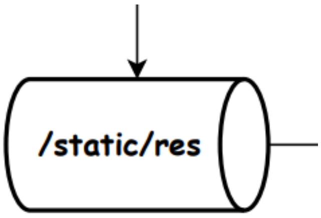
2. **Nodes**:
    - Nodes represent the elements involved in the interactions, such as objects, components, or actors. These nodes are connected by control flows to indicate the flow of control between them.

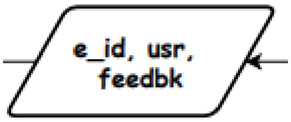
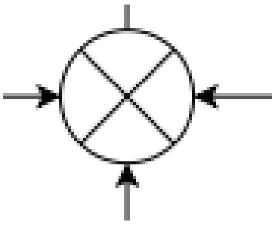
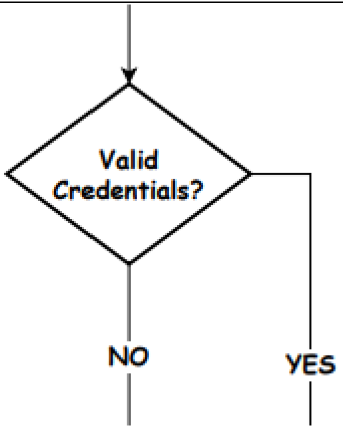
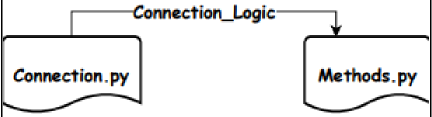
3. **Control Flows**:
    - Control flows show the sequence of interactions between nodes in the diagram. They represent the flow of control from one node to another, indicating the order in which interactions occur.

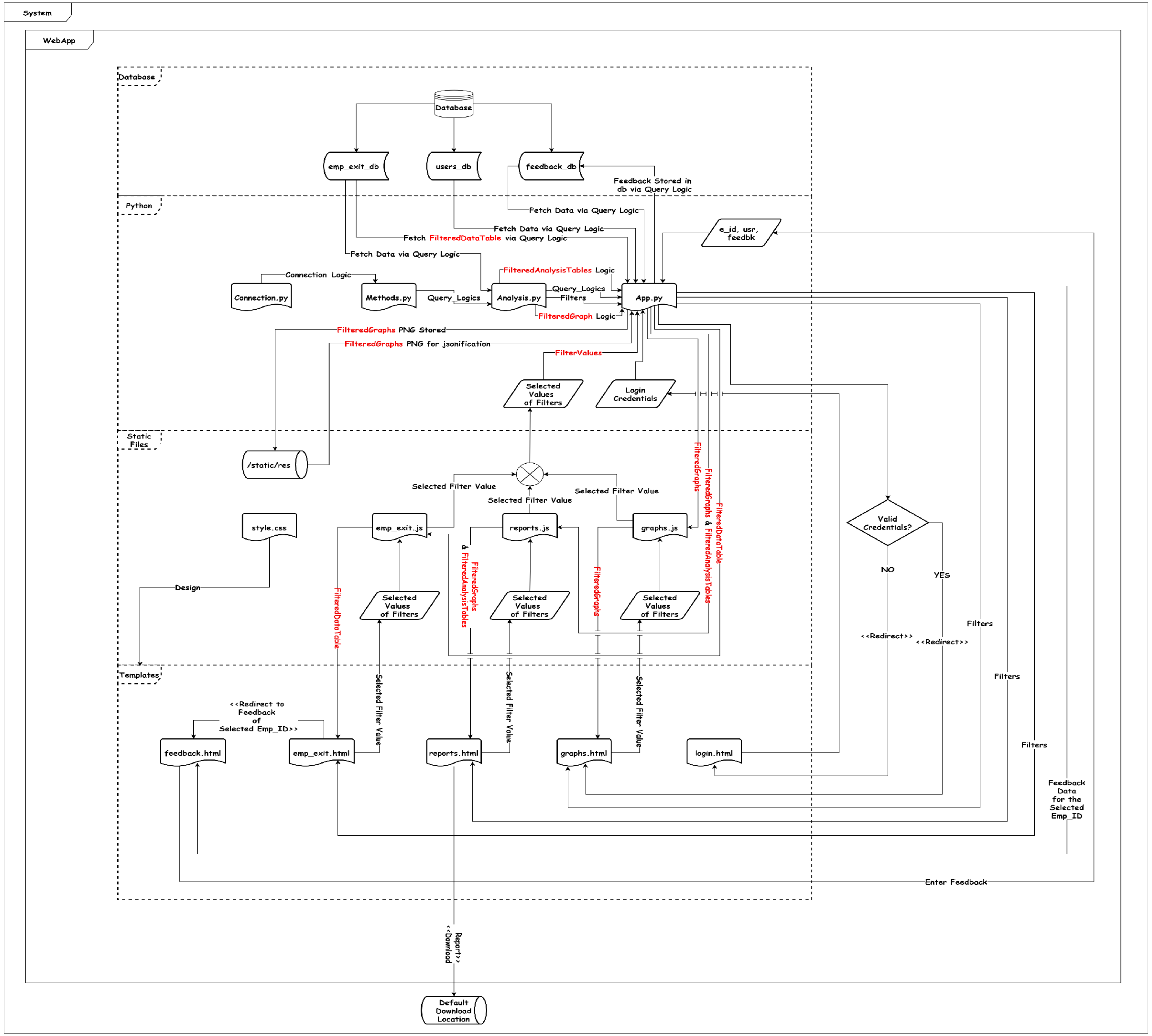
4. **Decision Nodes**:
    - Decision nodes are used to represent conditional behavior within the diagram. They allow for branching based on certain conditions, enabling different paths of interaction depending on the state of the system.

5. **Merge Nodes**:
    - Merge nodes are used to merge multiple control flows back into a single flow. They are typically used after decision nodes to bring together the different paths of interaction.

Interaction Overview diagrams are particularly useful for providing a high-level overview of complex interactions within a system or between systems. They can be used during the design phase to visualize and analyze the flow of interactions, helping stakeholders to understand the overall behavior of the system and identify potential issues or optimizations.

NAME	SYMBOL	DESCRIPTION
DATABASE		Represents data housed on a storage service that will likely allow for searching and filtering by users.
STORED DATA		The "Stored Data" logo in a code flowchart symbolizes the storage or retrieval of data from a persistent storage medium, typically represented by a rectangle with the word "Data" inside it. This logo indicates a step in the flowchart where data is being stored into or retrieved from a storage location like a database or a file.
DOCUMENT		The "Document" logo in a code flowchart represents actions related to document or file handling within a software application. It is depicted as a rectangle with the word "Document" inside, indicating steps such as creation, reading, updating, or deletion of documents. This symbol serves as a visual cue for operations involving document manipulation within the flowchart.
HARD DISK		The "Hard Disk" logo in a code flowchart represents actions related to storing or retrieving data from a persistent storage medium, like a hard disk drive. It's typically depicted as a rectangle with a small circle or cylinder inside. This symbol indicates steps where data is written to or read from storage, serving as a visual cue for operations involving persistent data storage within the flowchart.

DATA		**The "Data" logo in a code flowchart generally symbolizes the manipulation or processing of data within the flow of a program. It's often represented as a rectangle containing the word "Data" or simply as a rectangle with no text inside. This symbol indicates points in the flowchart where data is input, output, or manipulated in some way. It serves as a visual cue to denote the handling of data within the broader context of the flowchart.**
SUMMING JUNCTION		**The "Summing Junction" logo in a code flowchart symbolizes the consolidation or aggregation of multiple data streams into a single output. It's represented by a circle or triangle with incoming arrows from various sources and a single outgoing arrow indicating the combined result. This symbol visually indicates the merging of data from different inputs within the flowchart.**
DECISION		**The "Decision" logo in a code flowchart, often depicted as a diamond shape, signifies a point where the program evaluates a condition or set of conditions to determine the next course of action. It serves as a visual representation of branching logic, showing different paths the program can take based on the result of the evaluation.**
ARROW		**The "Arrow" symbol in a code flowchart signifies the direction of flow of control or data within the program. It's represented as a straight line with an arrowhead, connecting different elements like decision points or data processing blocks. Arrows visually depict the sequence of steps or operations, aiding in understanding the program's logic and execution flow.**


System
WebApp
Database
Database
emp_exit_db
users_db
feedback_db
Feedback Stored in db via Query Logic
Python
Fetch Data via Query Logic
Fetch Data via Query Logic
Fetch FilteredDataTable via Query Logic
Fetch Data via Query Logic
e_id, usr, feedbk
Connection_Logic
FilteredAnalysisTables Logic
Connection.py
Methods.py
Query_Logics
Analysis.py
Query_Logics
Filters
App.py
FilteredGraph Logic
FilteredGraphs PNG Stored
FilteredGraphs PNG for jsonification
FilterValues
Selected Values of Filters
Login Credentials
Static Files
/static/res
Selected Filter Value
Selected Filter Value
Selected Filter Value
FilteredGraphs
FilteredGraphs & FilteredAnalysisTables
FilteredDataTable
style.css
emp_exit.js
reports.js
graphs.js
Valid Credentials?
NO
YES
Design
FilteredDataTable
FilteredGraphs & FilteredAnalysisTables
FilteredGraphs
Selected Values of Filters
Selected Values of Filters
Selected Values of Filters
Filters
<<Redirect>>
<<Redirect>>
Templates
Selected Filter Value
Selected Filter Value
Selected Filter Value
Filters
<<Redirect to Feedback of Selected Emp_ID>>
feedback.html
emp_exit.html
reports.html
graphs.html
login.html
Filters
Feedback Data for the Selected Emp_ID
Enter Feedback
<<Download Report>>
Default Download Location

This Interaction Overview Diagram shows how all the components are interconnected and how they interact with each other to work seamlessly in harmony and provide a **GOOD User Experience**.

There are 6 major layers where the **“SYSTEM”** is the main layer representing the parent container with one sub-container named **“WebApp”** which has other 4 sub-layers or sub-containers namely **“DATABASE”**, **“PYTHON”**, **“STATIC FILES”** AND **“TEMPLATES”**. Apart from this, the **“SYSTEM”** container also has its **“STORAGE”** named **“DEFAULT DOWNLOAD LOCATION”** and there is a **“STORAGE”** named **“/static/res”** in **“STATIC FILES”**.

1. **SYSTEM:**
   - Represents the primary layer, acting as the parent container. It encompasses the **"WebApp"** sub-container along with additional sub-layers or sub-containers.

2. **WebApp:**
   - Refers to the active instance of the web application running on the system. It comprises various backend components hosted on a server within the same network as the system.

3. **DATABASE:**
   - The MySQL database where essential tables related to the WebApp's functionality are stored.

4. **PYTHON:**
   - Encompasses all Python files responsible for executing diverse functionalities, including API development, analysis, and backend logic, utilizing Flask.

5. **STATIC FILES:**
   - Constitutes components of the Flask App, comprising JavaScript (js) and Cascading Style Sheets (css) files that support HTML files within the TEMPLATES folder.

6. **TEMPLATES:**
   - Components of the FlaskApp, housing frontend files such as HTML.

7. **STORAGE:**
   - Encompasses storage spaces like hard disks, including the **"DEFAULT DOWNLOAD LOCATION"** within the **"SYSTEM"** container and **"/static/res"** within the **"STATIC FILES"** container.

Now let's delve into the internal components of the **WebApp** in detail.

1. **DATABASE**

    - **emp_exit_db** (Employee Exit Data)
    - **users_db** (Users' Login Data)
    - **feedback_db** (Feedback of Users)

2. **PYTHON**

    - **Connection.py** (file): Contains the Connection Logic.
    - **Methods.py** (file): Comprises Query functions utilized for communication with the database.
    - **Analysis.py** (file): Houses functions for preprocessing, Analysis, creating graphs, and data tables.
    - **App.py** (file): Encompasses the complete backend logic programmed in Flask.
    - **Selected Values of Filters** (variable): Points to **App.py**, with its value fetched from HTML files via a related JS file.
    - **Login Credentials** (variable): Points to **App.py**, with its value fetched from login.html.
    - **e_id, usr, feedbk** (variable): Points to **App.py**, with their values fetched from **feedback.html**.

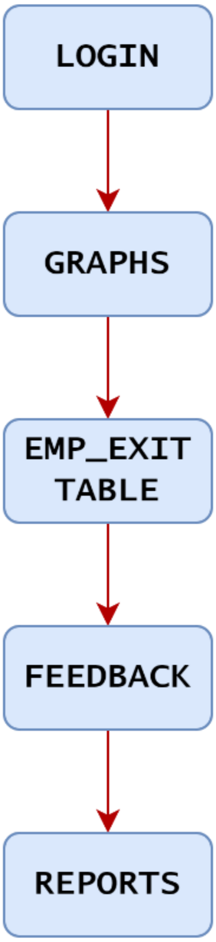
3. **STATIC FILES**

    - **style.css**: Provides design to all the HTML files.
    - **emp_exit.js**: JS file for **emp_exit.html**.
    - **reports.js**: JS file for **reports.html**.
    - **graphs.js**: JS file for **graphs.html**.

4. TEMPLATES

- **feedback.html**: Displays feedbacks from the database.
- **emp_exit.html**: Displays Data Table from the database.
- **reports.html**: Displays reports generated via **Analysis.py** (file).
- **graphs.html**: Displays graphs generated via **Analysis.py** (file).
- **login.html**: Used to take login credentials inputs.

Let's assume the navigation of the user through the WebApp as follows:



Steps:

1. **Establishing Database Connection:**
    - Begins with implementing **“Connection.py”** to establish a connection with the database.

2. **User Authentication:**
    - Upon accessing the front end, the user encounters **“login.html”** to input credentials.

3. **Credential Validation:**
    - Credentials are forwarded to **“App.py”** for validation against the database using functions from **“Methods.py”**.

4. **Authentication Outcome:**
    - Invalid credentials prompt an Alert Box on **“login.html”**.

    - Valid credentials redirect the user to **“graphs.html”**.

5. **Data Visualization ('graphs.html'):**
    - Upon redirection, **“App.py”** initiates the following actions:

        a) Creation of filter options based on 'department' via **“filters”** functions in **“Analysis.py”**.

        b) Default filter value is set to **“All”**.

        c) Utilization of ajax in **“graphs.js”** to transmit the filter value to **“App.py”**.

        d) Generation of graphs based on available data and storage in **“/static/res”** in PNG format.

        e) Presentation of PNG images in **“App.py”** through file handling for JSON serialization.

        f) Transmission of JSON dumps back to **“graphs.js”** for interpretation and placement within designated <**div**> elements.

        g) Reiteration of steps (c) to (f) upon user selecting a different filter value.

6. **Employee Exit Data Presentation ('emp_exit.html'):**
   - User navigates to **“emp_exit.html”**, triggering the following backend actions by **“App.py”**:

     a) Generation of filter options based on multiple fields using **“filters”** functions in **“Analysis.py”**.

     b) Default values for all filters are set to **“All”**.

     c) Utilization of ajax in **“emp_exit.js”** to convey current filter values to **“App.py”**.

     d) Retrieval of data from **“emp_exit_db”** using dynamic functions from **“Methods.py”**.

     e) Conversion of retrieved dataset into JSON dumps.

     f) Transmission of JSON dumps back to **“emp_exit.js”** for interpretation and tabular data presentation.

     g) Addition of a **‘Feedback’** button within each record's rightmost cell to redirect users to **“feedback.html”**.

     h) Reiteration of steps (c) to (g) upon user selecting a different filter value.

7. **Feedback Submission ('feedback.html'):**
   - User is redirected to the feedback page of the selected employee record upon clicking the respective **‘Feedback’** button.

   - Backend actions performed by **“App.py”** include:

     a) Receipt of the **“emp_id”** associated with the clicked **‘Feedback’** button.

     b) Utilization of feedback query logic from **“Methods.py”** to fetch feedback based on the provided **“emp_id”**.

     c) Direct transmission of feedback data to **“feedback.html”**.

     d) Provision for users to add feedback using a textbox.

     e) Submission of user-entered feedback data ('e_id', 'usr', 'feedbk') to **“App.py”**.

f) Utilization of query logic from **“Methods.py”** to add the record to the database.

g) Implementation of a **“Back”** button to redirect users to **“emp_exit.html”**.

8. **Report Generation ('reports.html'):**
   - User access to **“reports.html”** triggers backend actions by **“App.py”**:

a) Generation of filter options based on 'department' using **“filters”** functions in **“Analysis.py”**.

b) Default filter value is set to **“All”**.

c) Utilization of ajax in **“reports.js”** to transmit the current filter value to **“App.py”**.

d) Generation of graphs and data tables based on available data.

e) Storage of graphs in **“/static/res”** as PNG images and conversion of data tables into JSON format.

f) Presentation of PNG images in **“App.py”** through file handling for JSON serialization.

g) Transmission of JSON dumps of graphs and data tables back to **“reports.js”** for interpretation and placement within designated <**div**> elements.

h) Reiteration of steps (c) to (g) upon the user selecting a different filter value.

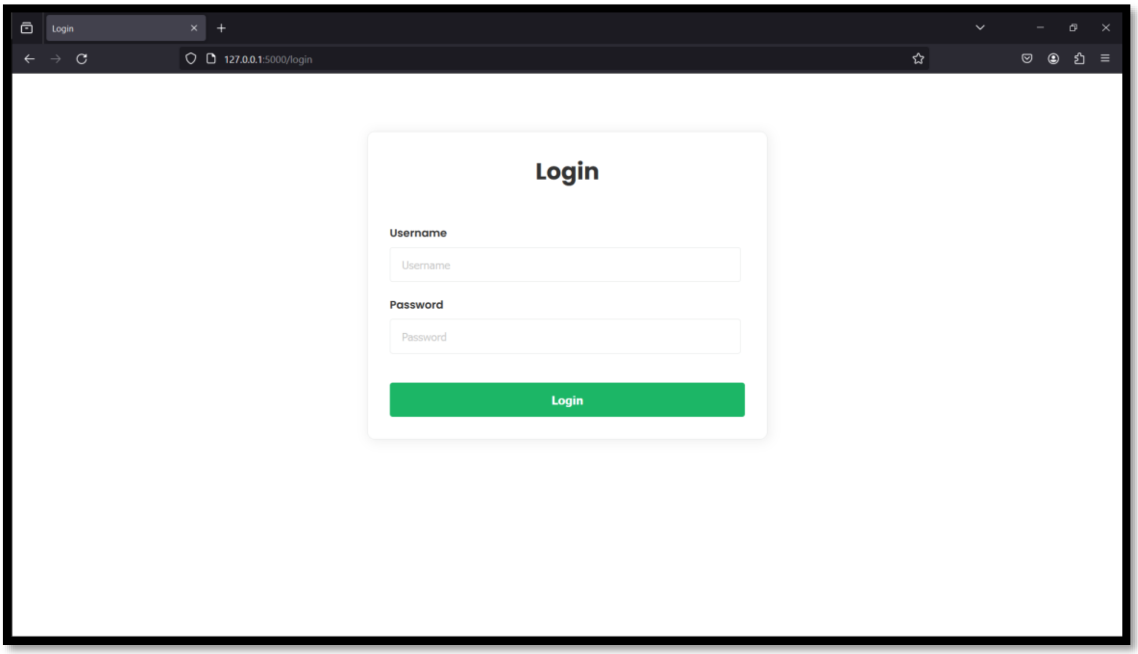
i) Provision for users to download the report using a **“Download”** button.

j) Integration of JSPDF for customized styling and arrangement of pages in the PDF report.

k) Saving the PDF report in the default download location of the system running the “**WebApp**”.

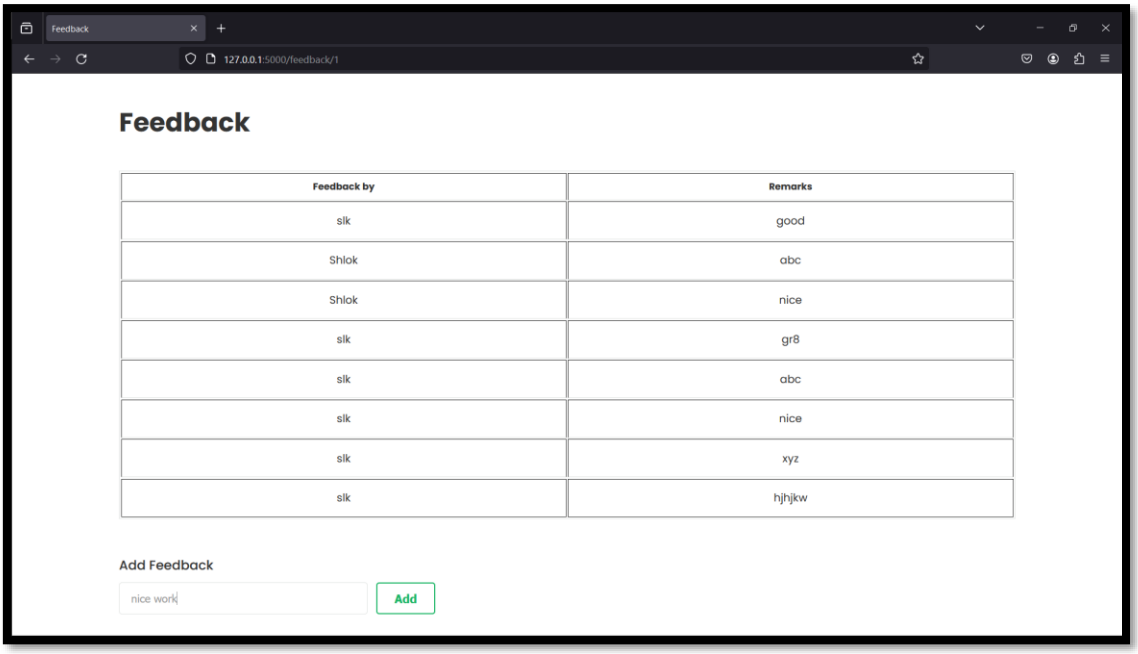
Chapter 8: - GUI Screenshots

8.1 Login Page



This module serves as the entry point for users, allowing them to authenticate and access the dashboard securely. By entering valid credentials, users gain authorized access to the dashboard's functionalities and features.

8.2 Feedbacks Page



Feedback

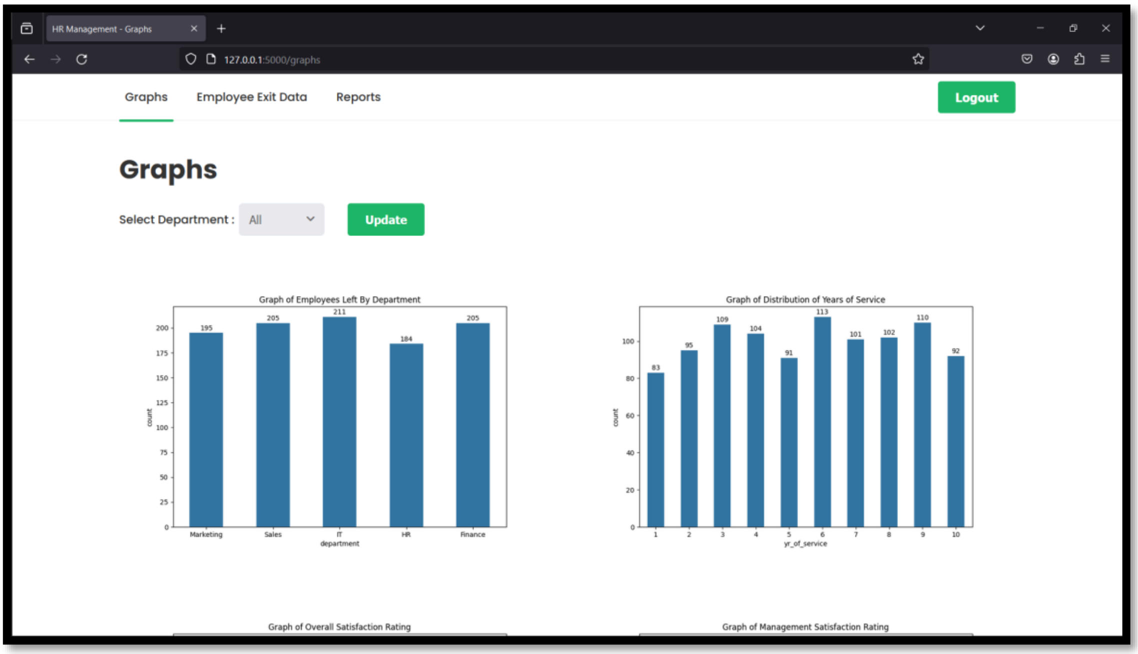
Feedback by	Remarks
slk	good
Shlok	abc
Shlok	nice
slk	gr8
slk	abc
slk	nice
slk	xyz
slk	hjhjkw

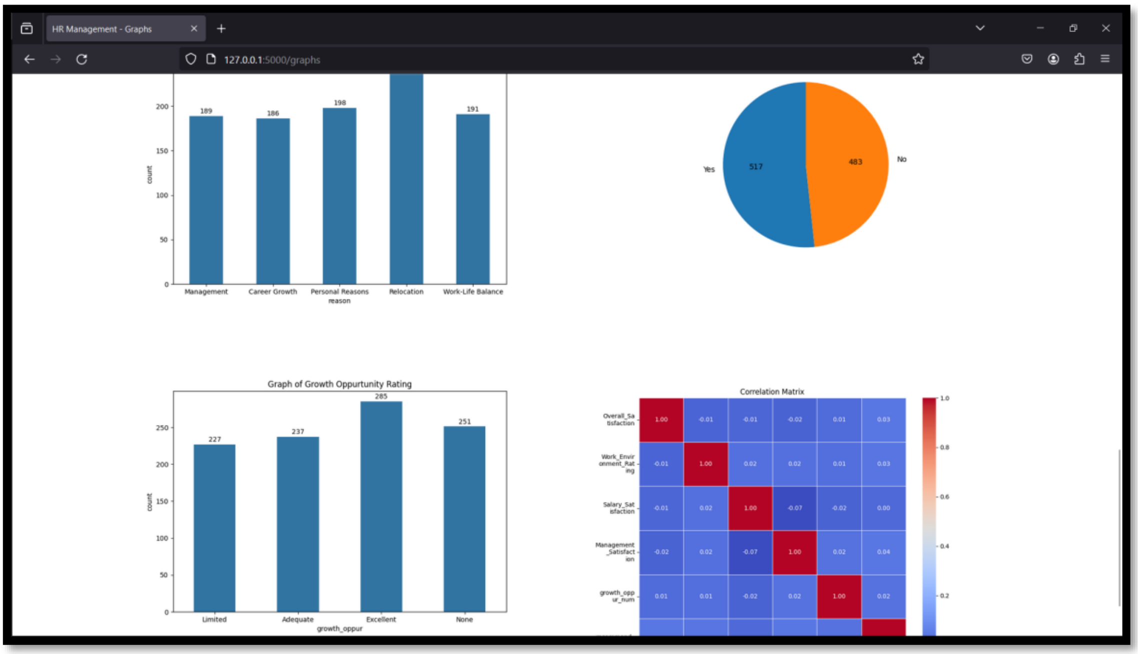
Add Feedback

nice work

This module, integrated into the HTML layout, enhances user engagement by allowing feedback on individual records. Users can click a conveniently placed feedback button at the end of each record to share comments and insights on specific data entries. This opens a new tab where users can view comments from different users for the corresponding record, promoting collaboration and knowledge sharing.

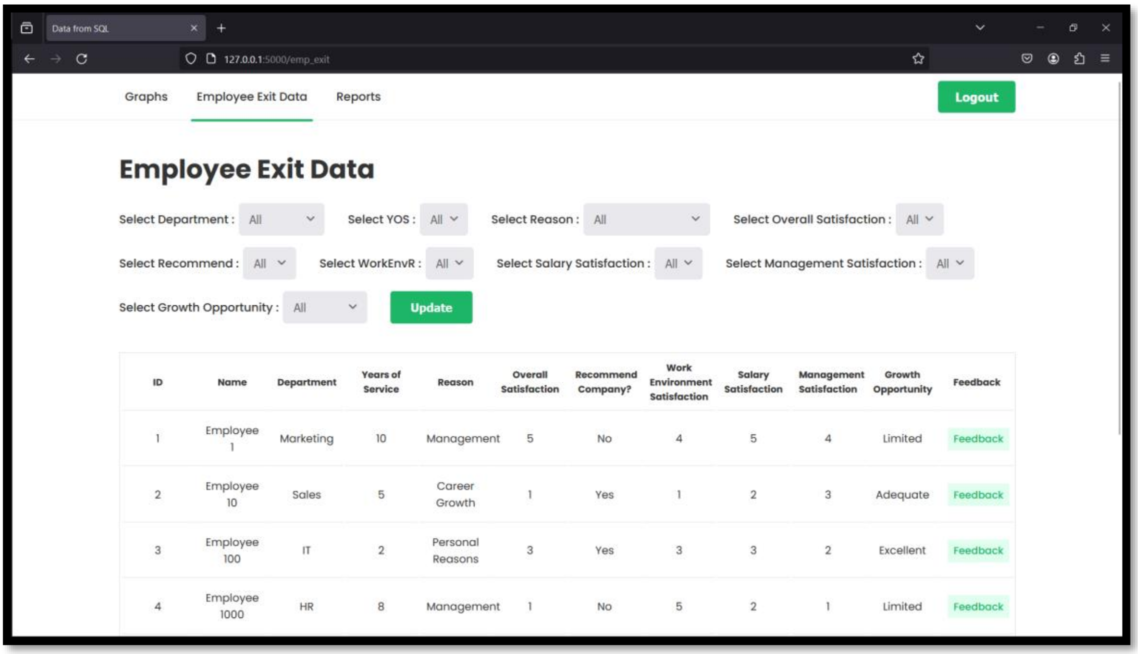
8.3 Graphs Page





In this section, users can delve into a range of graphical representations generated from Python data analysis. These graphs offer visual insights into different facets of employee data. Moreover, users can employ the filter feature to concentrate on particular departments, tailoring the displayed graphs to their preferences.

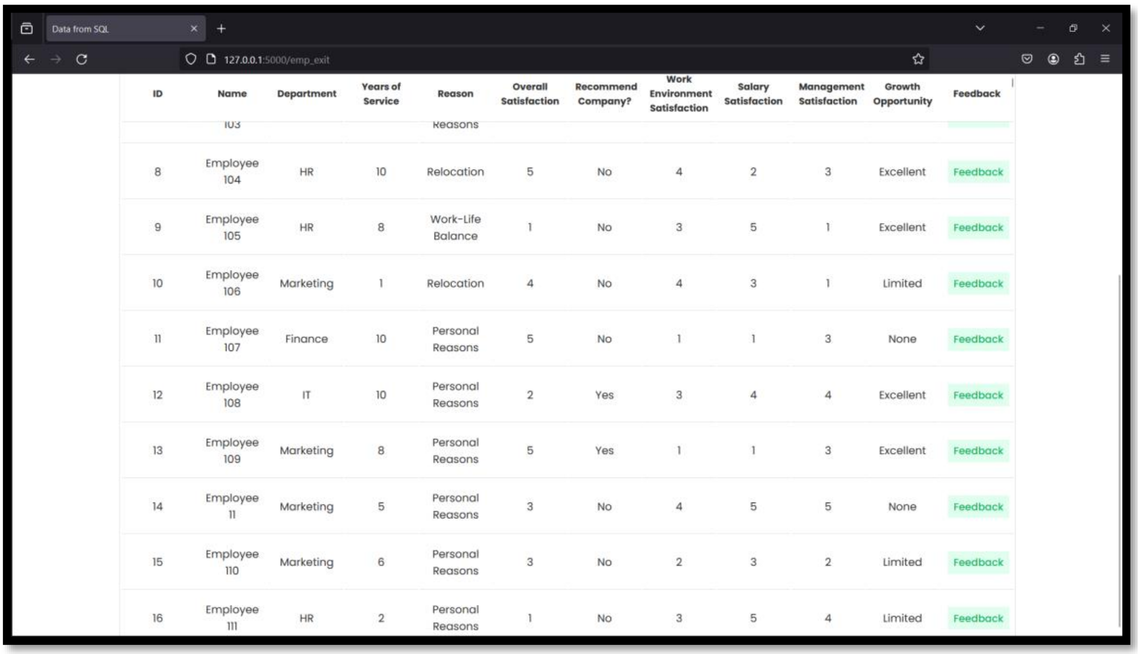
8.4 Employee Exit Details Table



Employee Exit Data

Select Department : All
Select YOS : All
Select Reason : All
Select Overall Satisfaction : All
Select Recommend : All
Select WorkEnvR : All
Select Salary Satisfaction : All
Select Management Satisfaction : All
Select Growth Opportunity : All

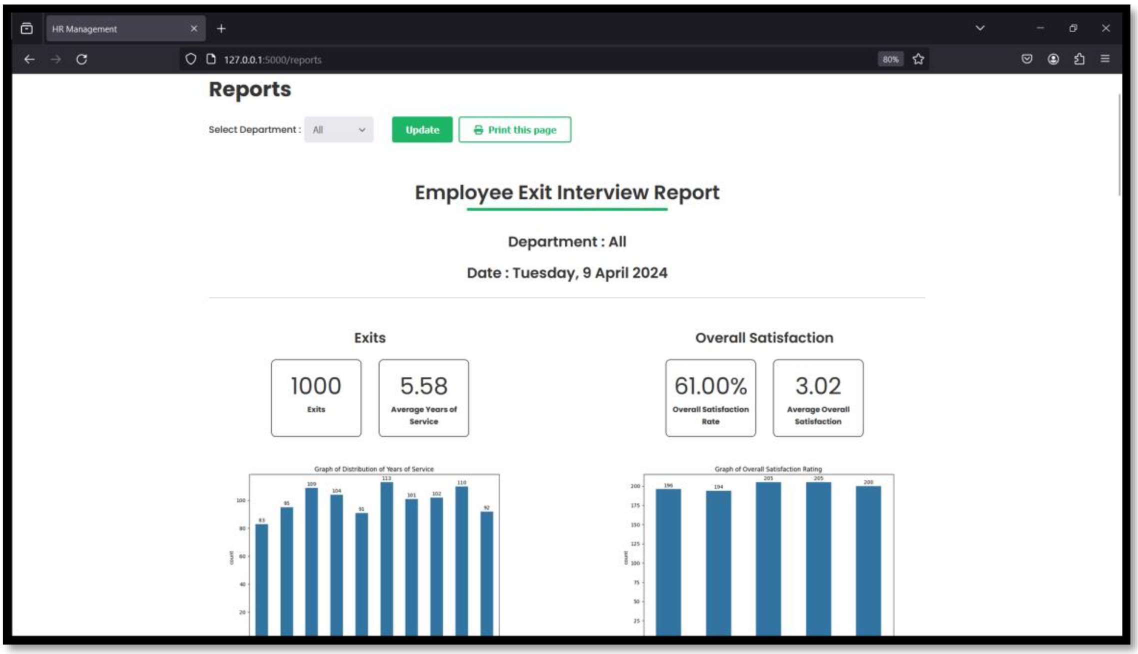
ID	Name	Department	Years of Service	Reason	Overall Satisfaction	Recommend Company?	Work Environment Satisfaction	Salary Satisfaction	Management Satisfaction	Growth Opportunity	Feedback
1	Employee 1	Marketing	10	Management	5	No	4	5	4	Limited	Feedback
2	Employee 10	Sales	5	Career Growth	1	Yes	1	2	3	Adequate	Feedback
3	Employee 100	IT	2	Personal Reasons	3	Yes	3	3	2	Excellent	Feedback
4	Employee 1000	HR	8	Management	1	No	5	2	1	Limited	Feedback



ID	Name	Department	Years of Service	Reason	Overall Satisfaction	Recommend Company?	Work Environment Satisfaction	Salary Satisfaction	Management Satisfaction	Growth Opportunity	Feedback
	103			Reasons							
8	Employee 104	HR	10	Relocation	5	No	4	2	3	Excellent	Feedback
9	Employee 105	HR	8	Work-Life Balance	1	No	3	5	1	Excellent	Feedback
10	Employee 106	Marketing	1	Relocation	4	No	4	3	1	Limited	Feedback
11	Employee 107	Finance	10	Personal Reasons	5	No	1	1	3	None	Feedback
12	Employee 108	IT	10	Personal Reasons	2	Yes	3	4	4	Excellent	Feedback
13	Employee 109	Marketing	8	Personal Reasons	5	Yes	1	1	3	Excellent	Feedback
14	Employee 11	Marketing	5	Personal Reasons	3	No	4	5	5	None	Feedback
15	Employee 110	Marketing	6	Personal Reasons	3	No	2	3	2	Limited	Feedback
16	Employee 111	HR	2	Personal Reasons	1	No	3	5	4	Limited	Feedback

In this section, users have access to a detailed data table comprising data collected from employee exit interviews. The table features a user-friendly interface, enabling easy navigation and exploration of the dataset. Furthermore, the module includes filtering options for all columns, empowering users to refine their search criteria. Additionally, users can make multiple selections to personalize their data view further, facilitating efficient data analysis and decision-making.

8.5 Reports Page



Reports

Select Department : All

Update | Print this page

Employee Exit Interview Report

Department : All

Date : Tuesday, 9 April 2024

Exits

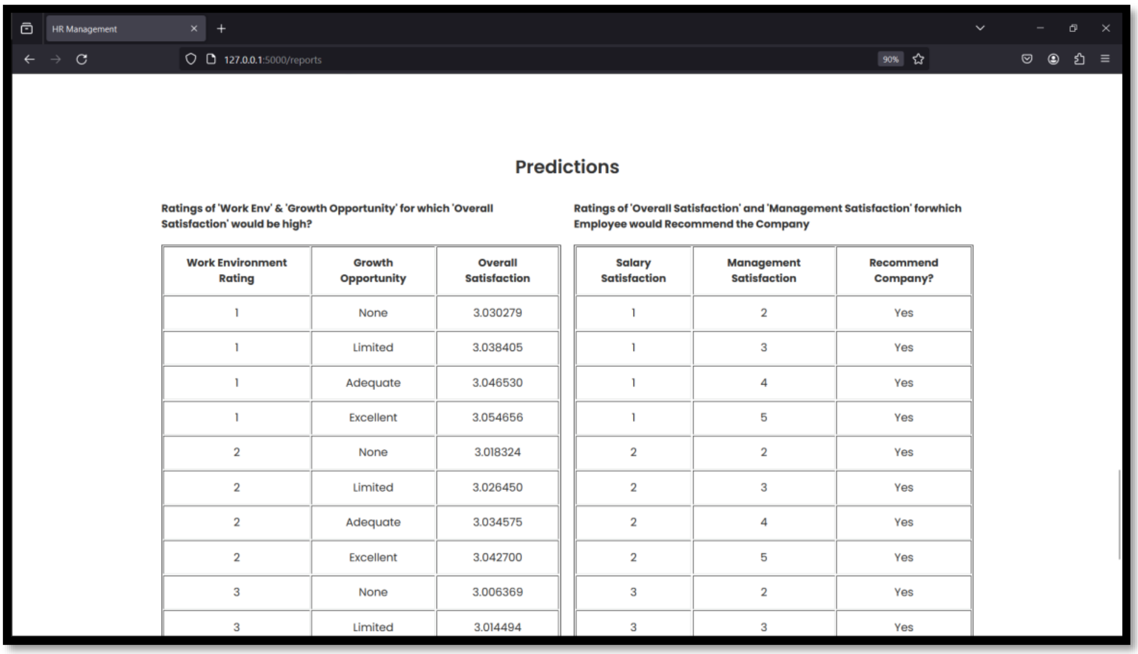
1000 Exits

5.58 Average Years of Service

Overall Satisfaction

61.00% Overall Satisfaction Rate

3.02 Average Overall Satisfaction



Predictions

Ratings of 'Work Env' & 'Growth Opportunity' for which 'Overall Satisfaction' would be high?

Work Environment Rating	Growth Opportunity	Overall Satisfaction
1	None	3.030279
1	Limited	3.038405
1	Adequate	3.046530
1	Excellent	3.054656
2	None	3.018324
2	Limited	3.026450
2	Adequate	3.034575
2	Excellent	3.042700
3	None	3.006369
3	Limited	3.014494

Ratings of 'Overall Satisfaction' and 'Management Satisfaction' forwhich Employee would Recommend the Company

Salary Satisfaction	Management Satisfaction	Recommend Company?
1	2	Yes
1	3	Yes
1	4	Yes
1	5	Yes
2	2	Yes
2	3	Yes
2	4	Yes
2	5	Yes
3	2	Yes
3	3	Yes

This module provides comprehensive reports that blend graphical representations with concise data tables. Users can create detailed reports by applying filters according to departmental criteria. By choosing specific departments, users can generate reports tailored to each department or opt for a general report covering all departments. This feature empowers users to glean valuable insights into departmental performance, aiding informed decision-making and strategic planning within the organization.

Chapter 9: - EXPERIENCES OF CORPORATE AND WORKING ATMOSPHERE

My time at IndaPoint Technologies Pvt Ltd has offered me invaluable experiences and insights into corporate dynamics and workplace environments. Throughout my four-month tenure, I underwent significant personal and professional growth, acquiring proficiency in various technologies, analytical abilities, and development methodologies while actively contributing to a live project.

Serving as a Data Analyst, my primary role involved analyzing data from exit interviews to mitigate employee turnover rates. Additionally, I undertook the task of developing a web-based dashboard, expanding my skill set as a backend developer and broadening my scope of work.

During the initial phase, my focus centred on crafting APIs to facilitate seamless communication across project components, necessitating meticulous planning and execution for optimal data exchange and functionality. This period also allowed me to delve into emerging web development technologies and bridge theoretical knowledge with practical application, emphasizing the vital role of theoretical understanding in effective real-world solutions.

IndaPoint Technologies Pvt Ltd fostered a collaborative environment conducive to continuous learning and innovation, wherein I collaborated with talented peers who generously shared their expertise and mutual support. The company's culture prioritized open communication and transparency, nurturing a feedback-driven ecosystem that nurtured both technical prowess and personal development.

One notable aspect of the company culture was its dedication to work-life balance, offering flexible schedules and remote work options to promote employee well-being amidst the fast-paced tech landscape. Moreover, the company championed diversity and inclusivity, fostering an environment where individual differences were respected and celebrated.

In summary, my stint at IndaPoint Technologies Pvt Ltd was profoundly enriching, providing me with not only technical skills and expertise but also imparting invaluable lessons on the significance of positive corporate culture and conducive work environments. Additionally, I forged a network of supportive professionals during this internship, which will undoubtedly prove instrumental in my future endeavours.

This experience will remain a cornerstone of my career journey, guiding my growth and evolution as I progress further in my professional endeavours.

References

1. MySQL
2. Ajax
3. JsPDF
4. Flask
5. MySQL-Connector
6. NumPy
7. Pandas
8. Matplotlib
9. Seaborn
10. Scikit-Learn
11. Scikit-Learn Playlist
12. SequenceDiagram.org
13. GeeksforGeeks
14. Tutorials Point