ICFS2 Summative 1

Task description

For this assignment, you are tasked with designing and building a Minimum Viable Product (MVP) for a GUI application within the Employee Management Systems (EMS) domain. Your application could be an event register, a diversity needs tracker, an employee performance tracker, or any other concept related to Employee Management Systems (EMS). The system you develop must have the following features:

- A GUI to facilitate the input, viewing, and exporting of employee data.
- Capability to read from and write to a CSV file.
- Implementation using object-oriented programming principles.
- Comprehensive commentary, including appropriate docstrings. Exception handling and input validation. Testable logic, exemplified by pure functions for input validation that consistently return the same output given the same parameters.

Task 1: Application Documentation

1.1: Introduction

Within our business, the Call Services team plays a pivotal role in customer interactions, requiring a robust system to monitor and enhance employee performance. Recognizing this need, we have designed an Employee Performance Management System that serves both staff members and managers, aligning their objectives to foster a culture of continuous improvement and exceptional service delivery.

For staff members, whose primary responsibility lies in handling incoming calls through an existing telephony system, the platform offers valuable analytical tools to self-assess performance. Beyond simply logging calls, employees can review their success rates, identify recurring pain points, and contextualize their results by comparing their metrics against team averages. This functionality helps distinguish between isolated challenges and broader trends requiring managerial intervention. Staff also benefit from a detailed call log history, enabling efficient follow-ups with customers when needed. To further streamline their workflow, a quick-access resource library provides immediate answers to frequently encountered issues, reducing downtime and repetitive inquiries to managers.

Customer feedback serves as the cornerstone of performance evaluation, with each interaction rated on a scale from 0 to 1. A score of 0.8 or higher classifies the call as successfully resolved, creating a clear benchmark for service quality. These ratings not only inform individual progress but also feed into broader team analytics, ensuring transparency at all levels.

Managers, tasked with optimizing team output and removing operational obstacles, access a specialized dashboard featuring real-time performance visualizations. These insights highlight overall success rates, recognize high performers, and pinpoint areas where underperforming staff may need additional coaching or resources—all framed as opportunities for growth rather than punitive measures. The system also enables managers to benchmark their team's efficiency against other units, fostering healthy competition and shared best practices. Administrative functionalities allow for seamless updates to employee records, including personal details and adjustable daily targets, ensuring goals remain both challenging and achievable.

The Minimum Viable Product (MVP) is designed to deliver immediate value by bridging the gap between frontline staff and managerial oversight. Through intuitive, role-tailored dashboards, the system transforms raw performance data into actionable insights, empowering users at every level to make informed decisions. Secure authentication via username and password safeguards sensitive information while providing customized interfaces for each user class. By consolidating these features into a unified platform, the MVP not only enhances operational efficiency but also cultivates a collaborative environment where employees and managers work in tandem to elevate customer service standards.

1.2: Design

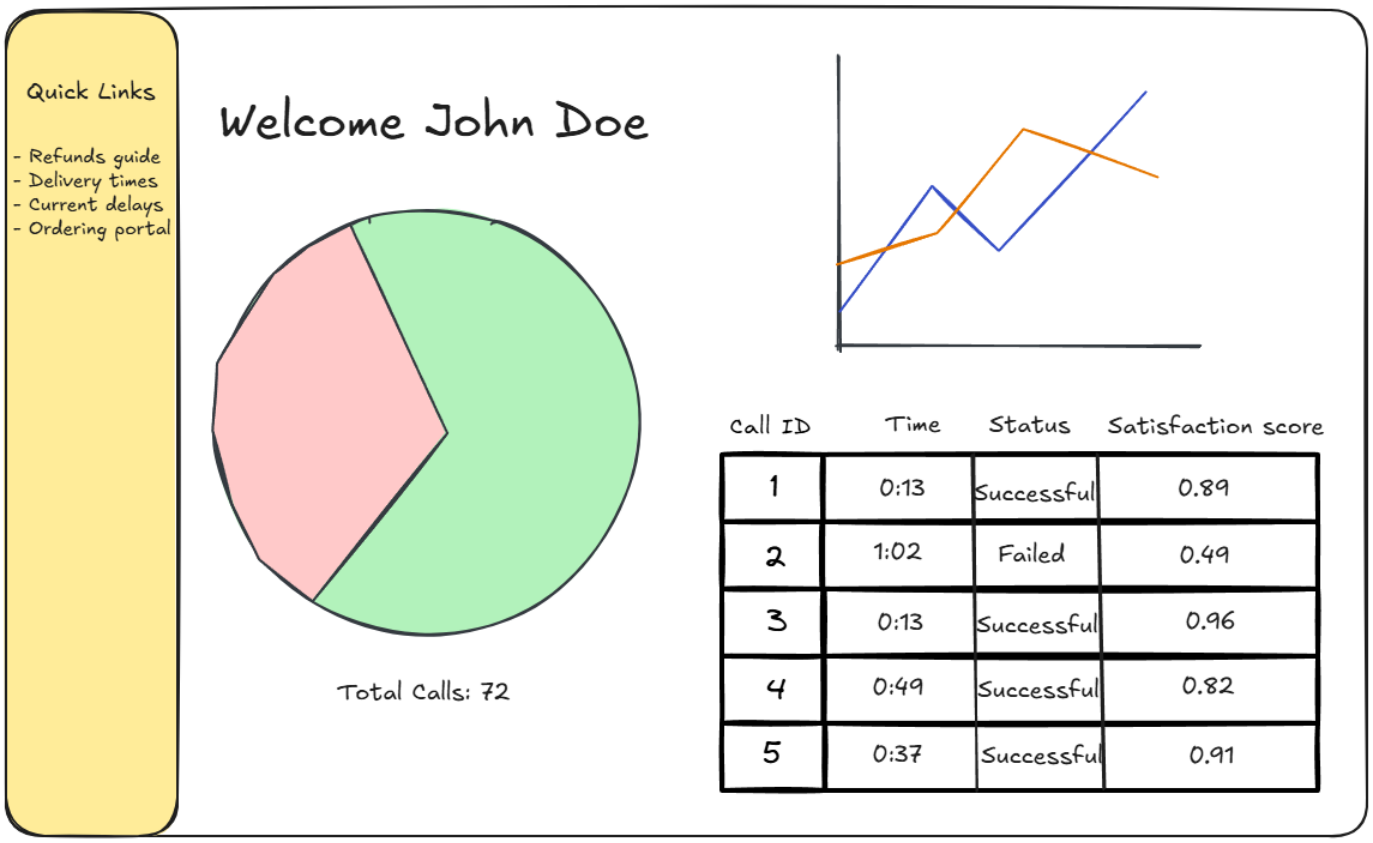
1.21: User requirements

Requirement Code	Description	Linked features
RM1	Search for employees to bring up individual data	Drop down bar at the top of the page to filter by individuals
RM2	See whole team success rate	Pie chart that denotes ratio of successful to non-successful calls
RM3	Filter team success rate by time periods	Slider that controls the time period of the pie chart that shows success %
RM4	See top/worst performers	Table that shows 3 best and worst performers
RM5	See team success rate vs other teams	Graph that shows success rate of local teams
RM6	Add members to team	Separate screen for adding members in an entry form
RM7	Remove members from team	Separate screen that uses a drop down to remove members from team
RM8	Edit staff details	Be able to edit personal details for members of the team
RM9	View staff details	Bring up staff details based on a drop down of IDs
RS1	See individual daily success rate	Pie chart that shows ratio of successful/non-successful calls
RS2	See call history	See a preview in the form of a table containing the last 5 calls
RS3	See average customer satisfaction score history with filter to enable team's average too	A graph that shows the average quality of customer service with quick filters that show year, month and week and team to be able to compare the two lines.
RS4	Quick links section	A section in the navigation bar that takes them to frequently used resources like guides
RS5	See time elapsed	Have a section in the nav bar that shows time elapsed
RS6	Answering call	Drop down in nav bar when a call comes through that updates with a answer option that runs the answer_call function
RS7	Declining call	Drop down in nav bar when active call is detected that runs the end_call function

With requirement codes starting with 'RM', they relate to requirements for the managerial users, whilst requirement codes starting with 'RS' relate to functionality necessary for the staff. All of these codes will start with an 'R' to indicate that it relates to user requirements whilst codes starting with a 'T' will be in accordance with testing requirements.

1.22: User journey prototype/Wireframe

The image below shows a wireframe of what the staff would see on their screen. As mentioned, the quick-links section will provide a set of links to commonly used guides. The pie chart provides a simple and intuitive way to see the current proportion of successful to unsuccessful calls. The line chart on the right hand side contains two lines, one outlining the average satisfaction for the current user, whilst the other line shows the average user satisfaction for the whole team. Below the line graph is a table showing the last 5 calls, with the appropriate details.



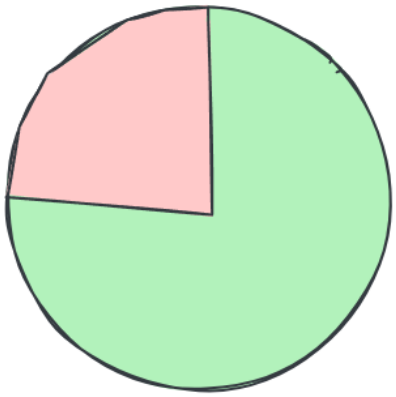
The wireframe below is unique to the managers and whilst it uses the same metrics, it focuses on the wider picture which is more relevant to the supervisors in an effort to identify common pain points that can be improved for the whole team leading to a boost in productivity. Whilst it may seem like a hostile environment of pointing out the best and worst performers, it comes with the purpose of assisting those who require it, and understanding how top performers remain consistent and any tips they may have for their colleagues.

Welcome John Doe (Manager)

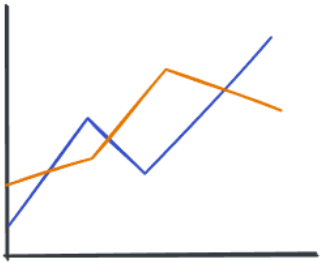
Select Employee

Worst performers:

EmployeeID	Employee Name	Avg Cust Sat Score
101	Archie Smith	0.75
102	Barry Daniels	0.73
103	Clive Thomas	0.65



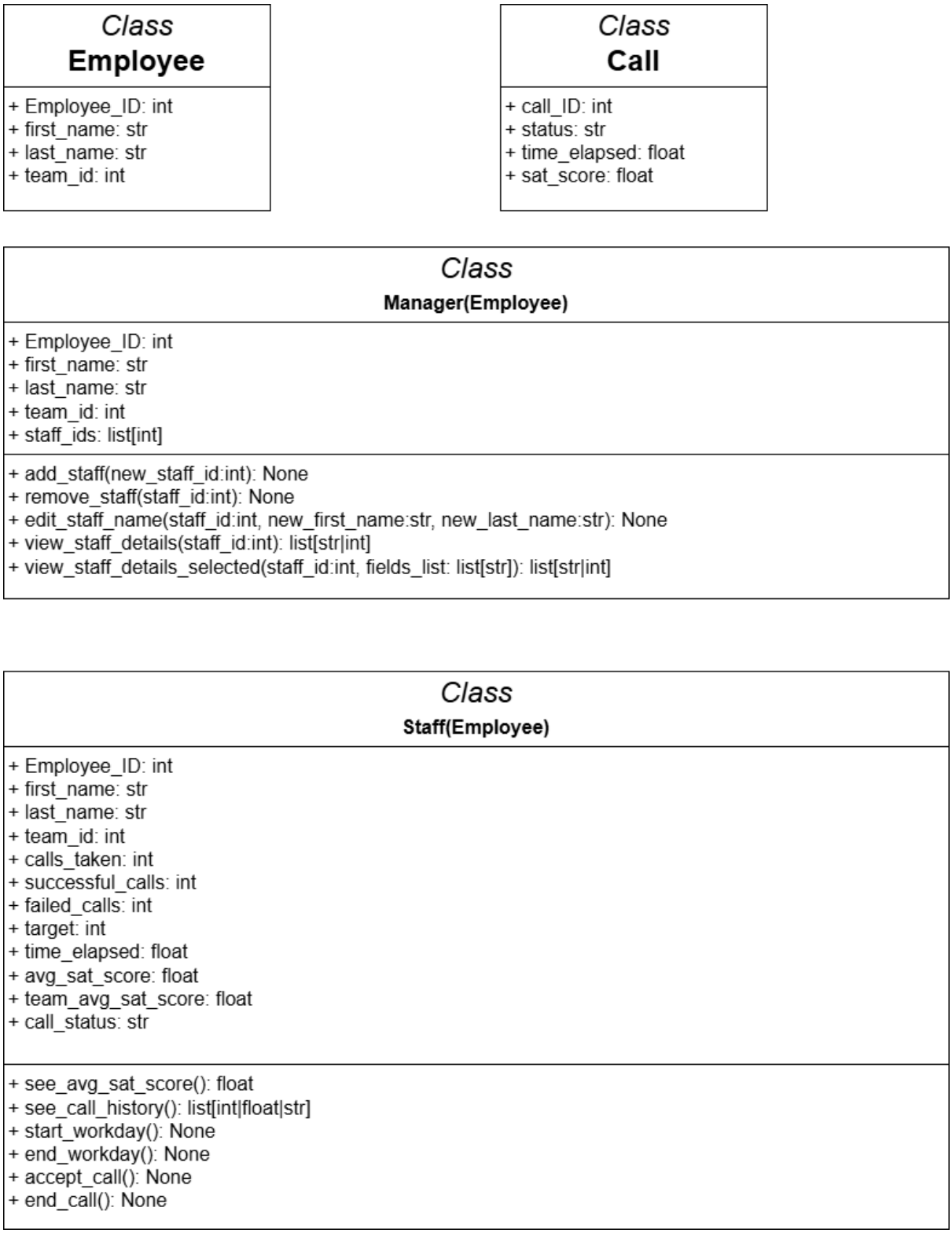
Time frame slider



Best performers:

EmployeeID	Employee Name	Avg Cust Sat Score
104	Dan Summers	0.97
105	Ed Milton	0.96
106	Finley Shane	0.92

1.23: Class diagram



This UML diagram highlights the main classes that will be used to construct this MVP. The two parent classes are Employee and Call where Call has no child classes and a call object has the attributes 'ID, status, time_elapsed, sat_score' where sat_score refers to a customer's satisfaction score of the call. Time_elapsed denotes the duration of the call, as it can suggest whether or not a longer call time is actually beneficial to the customer or not, and how to

manage the time spent with the customer to maximise conversation time whilst retaining customer interaction.

The Employee class is a skeleton class upon which the Manager and Staff class are constructed and the attributes are inherited such as employee_id since all employees will have their own ID, and a first_name and last_name attribute since this will be common to everyone. The methods in the Manager class are actions that require elevated permissions such as a managerial position, and this could involve modifying staff details as an example. On the other hand, the attributes in the Staff class are crucial to interpreting a worker's efficiency, and to perform critical analysis to understand pain points.

1.24: Tech Stack Outline

Use Python for the back-end

Streamlit as a front-end for user interaction as it is a library built into Python and can easily be modified with reusable components

The choice for using Streamlit over Tkinter is a personal one, but the more modern look of Streamlit's UI has a better user experience than Tkinter's outdated boxy designs. Additionally, Streamlit runs in the client's browser and the processing is performed server side, allowing for anyone with an internet connection to make use of the product without having to rely on powerful hardware. This is crucial in a business situation as it's unlikely that all users will be supplied with the most expensive and power-heavy devices, and universal compatibility is crucial for product adoptability.

1.3: Testing Strategy

1.31: Testing code

I have broken down the testing into two sections since it's important to test both the code's functionality in the form of white testing the classes and their methods, but also the GUI's functionality and whether it's meeting its intended purpose. Below is a table showing the tests for the classes, and this will be made into a separate testing file as it's reusable and can be implemented into a CI (Continuous Integration) pipeline to perform automated testing and regression testing. Whilst the front end of the code may change over time, the classes themselves contain the majority of the functionality and it's important to test whether it works as a stand alone file full of classes.

Test Code	Description	Link to requirement code	Expected Result	Achieved?
TC1	Setting up test csv files	N/A	Creates files if doesn't exist	Yes
TC2	Initialising a manager	N/A	Creates manager object if it doesn't exist	Yes
TM3	Viewing staff details	RM9	Returns all details for a single staff	Yes
TM4	Viewing selected details for staff	RM9	Returns selected details for a single staff	Yes
TM5	Viewing staff details for non-existent staff	RM9	Returns error message stating that staff_id was not found	Yes

TM6	Adding staff details	RM6	Instantiate Staff objects	Yes
TM7	Adding existing staff details	RM6	Return error message saying staff_id already exists	Yes
TM9	Removing staff	RM7	Removing a staff if they exist in records	Yes
TM10	Removing non-existent staff	RM7	Return error message stating that staff_id was not found	Yes
TC11	Creating a call object	N/A	Creating a call object if that ID has not been used	Yes
TC12	Creating a staff object	N/A	Creating a staff object if that ID has not been used	Yes
TS13	Checking time elapsed	RS5	Returning the time elapsed of the user working	No
TS14	Answering a call	RS6	Call is answered and navbar has option to end current call while showing current time	No
TS15	Declining a call	RS7	Call is ended and navbar adjusts appropriately. CSV file should be updated	No
TS16	Ending a workday	RS5	Time_elapsed should have total working hours and be updated on CSV	Yes

Above there are test codes starting with 'TC', which relate to common functionality for both managerial users and staff users. All testing requirements have been marked with the code 'T' and distinguish with user requirements which have been marked starting 'R'.

1.32: Testing GUI

Testing the GUI is more of a non-functional series of tests, since it tests how well the code is achieving the desired output from the front-end simulating what a user would see. The tests below

Requirement Code	Description	Functioning?
RM1	Search for employees to bring up individual data	Yes
RM2	See whole team success rate	Yes
RM3	Filter team success rate by time periods	Yes
RM4	See top/worst performers	Yes
RM5	See team success rate vs other teams	Yes
RM6	Add members to team	Yes
RM7	Remove members from team	Yes
RM8	Edit staff details	Yes
RM9	View staff details	Yes
RS1	See individual daily success rate	Yes
RS2	See call history	Yes last 3 calls
RS3	See average customer satisfaction score history with filter to enable team's average too	No
RS4	Quick links section	Yes
RS5	See time elapsed	Yes but doesn't live update
RS6	Answering call	Can simulate incoming call
RS7	Declining call	Yes