EMPLOYEE SALARY CALCULATION IN PYTHON

A PROJECT REPORT

Submitted by

PIRIYANGA D

(2303811720522038)

in partial fulfillment for the completion of the course

CGB1121- PYTHON PROGRAMMING

in

INFORMATION TECHNOLOGY

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, Affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

JUNE, 2024

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report titled **“EMPLOYEE SALARY CALCULATION IN PYTHON”** is the bonafide work of **PIRIYANGA D (2303811720522038)** who carried out the project under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other project report or dissertation on the basis of which a course was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Mrs.A.Delphin Carolina Rani, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K. Ramakrishnan College of Technology
(Autonomous)

Samayapuram – 621 11

SIGNATURE

Mrs. P.Sudha M.E.,Ph.D.,

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram – 621 112.

Submitted for the viva-voce examination held on …18/06/2024……….

INTERNAL EXAMINER

DECLARATION

I declare that the project report on **“EMPLOYEE SALARY CALCULATION IN PYTHON”** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1121- PYTHON PROGRAMMING**.

Signature

PIRIYANGA D

Place: Samayapuram

Date:18/06/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Mrs. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project guide **Mrs.P.Sudha, M.E.,Ph.D** Department of **ELECTRONICS AND COMMUNICATION ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To emerge as a leader among the top institutions in the field of technical education.

MISSION OF THE INSTITUTION

- Produce smart technocrats with empirical knowledge who can surmount the global challenges.
- Create a diverse, fully-engaged, learner-centric campus environment to provide quality education to the students.
- Maintain mutually beneficial partnerships with our alumni, industry, and Professional associations.

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities

   with an understanding of the limitations

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

This Python project is designed to automate the calculation of employee salaries, incorporating various factors such as basic pay, allowances, deductions, and bonuses. The system aims to provide a reliable and efficient tool for HR departments, ensuring accurate and timely salary computations. The project features a user-friendly interface for data entry, robust backend processing, and comprehensive reporting capabilities. Computes the gross salary by summing the basic pay, house rent allowance (HRA), dearness allowance (DA), and other allowances. Derives the net pay by subtracting total deductions from the gross salary. Utilizes databases for storing employee details, salary components, and transaction histories, ensuring data integrity and ease of access.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

BA	-	Basic Pay
DA	-	Dearness Allowance
HRA	-	House Rent Allowance
TA	-	Total Allowance

CHAPTER 1

INTRODUCTION

12.1 INTRODUCTION TO PYTHON

12.1.1. Overview

Python is a widely-used, high-level programming language renowned for its readability and simplicity, making it an ideal choice for both novice and seasoned programmers. Created by Guido van Rossum and released in 1991, Python's core philosophy emphasizes code readability and straightforward syntax, allowing developers to write clear and concise code more efficiently compared to other languages like C++ or Java.

12.1.2. Programming Paradigms

Python supports various programming paradigms, including procedural, object-oriented, and functional programming. This flexibility, combined with a dynamic type system and automatic memory management, facilitates the development of a wide range of applications, from simple scripts to complex software systems.

12.1.3. Standard Library

The language's comprehensive standard library, often referred to as "batteries-included," provides built-in modules and functions for handling many programming tasks, such as file I/O, system calls, and even web services. This extensive library helps streamline the development process by offering ready-to-use solutions for common programming challenges.

12.1.4. Third-Party Libraries And Frameworks

One of Python's significant strengths is its extensive ecosystem of third-party libraries and frameworks. Popular libraries such as NumPy and Pandas enable efficient data manipulation and analysis, while frameworks like Django and Flask streamline web development. In the realm of machine learning and artificial intelligence, libraries like TensorFlow and PyTorch are widely adopted for building and deploying sophisticated models.

12.1.5. Versions Of Python

Python has undergone significant evolution since its inception, with two major versions in use.

Python 2: Released in 2000, Python 2.x series was a major milestone and widely used for many years. However, it reached its end of life on January 1, 2020, and is no longer maintained.

Python 3: Introduced in 2008, Python 3.x series brought substantial improvements and changes to the language, such as better Unicode support, a more consistent syntax, and enhanced standard libraries. Python 3 is the recommended version for all new projects.

12.1.6. Python Tools

Python's ecosystem includes numerous tools that enhance productivity and development experience:

- **IDEs and Code Editors:** Popular options include PyCharm, VS Code, and Jupyter Notebook, which offer features like syntax highlighting, code completion, and debugging.

- **Package Management:** Tools like pip and conda facilitate the installation and management of Python libraries and dependencies.

- **Virtual Environments**: virtualenv and venv allow developers to create isolated environments for different projects, ensuring dependency conflicts are avoided.

- **Testing Frameworks:** unittest, pytest, and nose are commonly used for writing and running tests to ensure code reliability and correctness.

- **Build Tools:** setuptools and wheel help in packaging Python projects, making them easy to distribute and install.
- **Documentation Generators:** Tools like Sphinx are used to create comprehensive documentation for Python projects.

- **Linters and Formatters:** pylint, flake8, and black help maintain code quality and consistency by enforcing coding standards and formatting.

12.1.7. Versatility And Adoption

Python's simplicity and versatility have led to its widespread adoption in various fields, including web development, data science, artificial intelligence, automation, and scientific computing. Its active community continually contributes to a rich repository of resources, tutorials, and documentation, making it easier for developers to learn and apply Python effectively.

CHAPTER 2

PROJECT DESCRIPTION

2.1. PROJECT INTRODUCTION

In the contemporary business environment, the accurate and timely calculation of employee salaries is paramount to maintaining workforce satisfaction and operational efficiency. Manual payroll processing is often fraught with errors, time-consuming, and susceptible to compliance issues. To address these challenges, this Python project aims to develop an automated salary calculation system, streamlining the payroll process for Human Resources (HR) departments.

2.2. PROJECT OBJECTIVE

Develop an automated system for accurate salary computation.Reduce manual workload and save time in payroll processing.Create an intuitive interface for easy data management by HR staff.Generate detailed payslips and summary reports.Ensure secure and accurate storage of employee data.Build a scalable and maintainable system using Python.Maintain compliance with tax laws and regulatory requirements.

2.3. PROBLEM STATEMENT

In many organizations, the process of calculating employee salaries is often manual, time-consuming, and prone to errors. This inefficiency not only increases the workload for HR departments but also poses risks of non-compliance with tax laws and regulatory requirements. Inaccurate salary computations can lead to employee dissatisfaction, financial discrepancies, and potential legal issues.

2.4. LIBRARIES USED

The following Python libraries are utilized in this project to achieve the desired

functionality:

- Pandas:

For data manipulation and analysis.
Used to handle and process employee data, including salary components and calculations.

- NumPy:

For numerical operations.
Facilitates efficient computation of salary components and deductions.

- SQLite:

For database management.
Used to store employee information, salary details, and transaction histories securely.

- SQLAlchemy:

For database ORM (Object Relational Mapping).
Provides a high-level abstraction for database operations, ensuring efficient data retrieval and storage.

- Tkinter:

For GUI (Graphical User Interface) development.
Creates a user-friendly interface for HR personnel to interact with the system.

CHAPTER 3

SYSTEM ANALYSIS

3.1. EXISTING SYSTEM

In many organizations, the existing systems for calculating employee salaries are largely manual or rely on basic spreadsheet software. Salary calculations are performed manually by HR staff, which is time-consuming and prone to human error.

Keeping track of various salary components, deductions, and bonuses manually can lead to mistakes and inconsistencies.

3.1.1. DISADVANTAGES

a. Initial Development and Implementation Costs:

Developing and implementing the system can require significant time and financial investment.Organizations may need to allocate resources for software development, testing, and deployment.

b. Training Requirements:

HR personnel and other users need to be trained to use the new system effectively.

Transitioning from a manual or spreadsheet-based system to an automated one may require time and effort to adapt.

c. Technical Dependencies:

The system relies on various Python libraries and frameworks, which need to be maintained and updated.Dependencies on third-party libraries may introduce compatibility issues or require frequent updates.

d. Data Migration Challenges:

Migrating existing payroll data to the new system can be complex and error-prone.
Ensuring data integrity and accuracy during the migration process is critical.

e. System Maintenance and Support:

Ongoing maintenance is required to address bugs, update libraries, and ensure compatibility with new regulations.

3.2. PROPOSED SYSTEM

The proposed Python-based employee salary calculation system aims to address the limitations of existing manual and spreadsheet-based processes. This system will automate salary computations, enhance accuracy, ensure compliance, and provide robust data management and reporting capabilities.

3.2.1. ADVANTAGES

a. Automated Salary Computation:

Calculate gross and net salaries automatically by incorporating various components such as basic pay, house rent allowance (HRA), dearness allowance (DA), other allowances, bonuses, and standard deductions (income tax, provident fund, professional tax).

b. User-Friendly Interface:

Develop a graphical user interface (GUI) using Tkinter to allow HR personnel to easily input and manage employee data and salary components.Provide features for adding, updating, and deleting employee records.

c. Comprehensive Data Management:

Use SQLite for secure and efficient storage of employee details, salary components, and transaction histories.Implement SQLAlchemy for database operations, ensuring data integrity and ease of access.

d. Advanced Reporting Capabilities:

Generate detailed payslips for individual employees, including a breakdown of earnings and deductions.Create summary payroll reports for organizational analysis and financial planning**.**

e. Compliance and Updates:

Integrate features to automatically adjust calculations based on changes in tax laws and regulatory requirements.Provide tools to easily update tax rates, deduction rules, and other regulatory parameters**.**

f.Data Security:

Implement robust security measures to protect sensitive employee information**.**

CHAPTER 4
SYSTEM DESIGN & MODULES

4.1. BLOCK DIAGRA

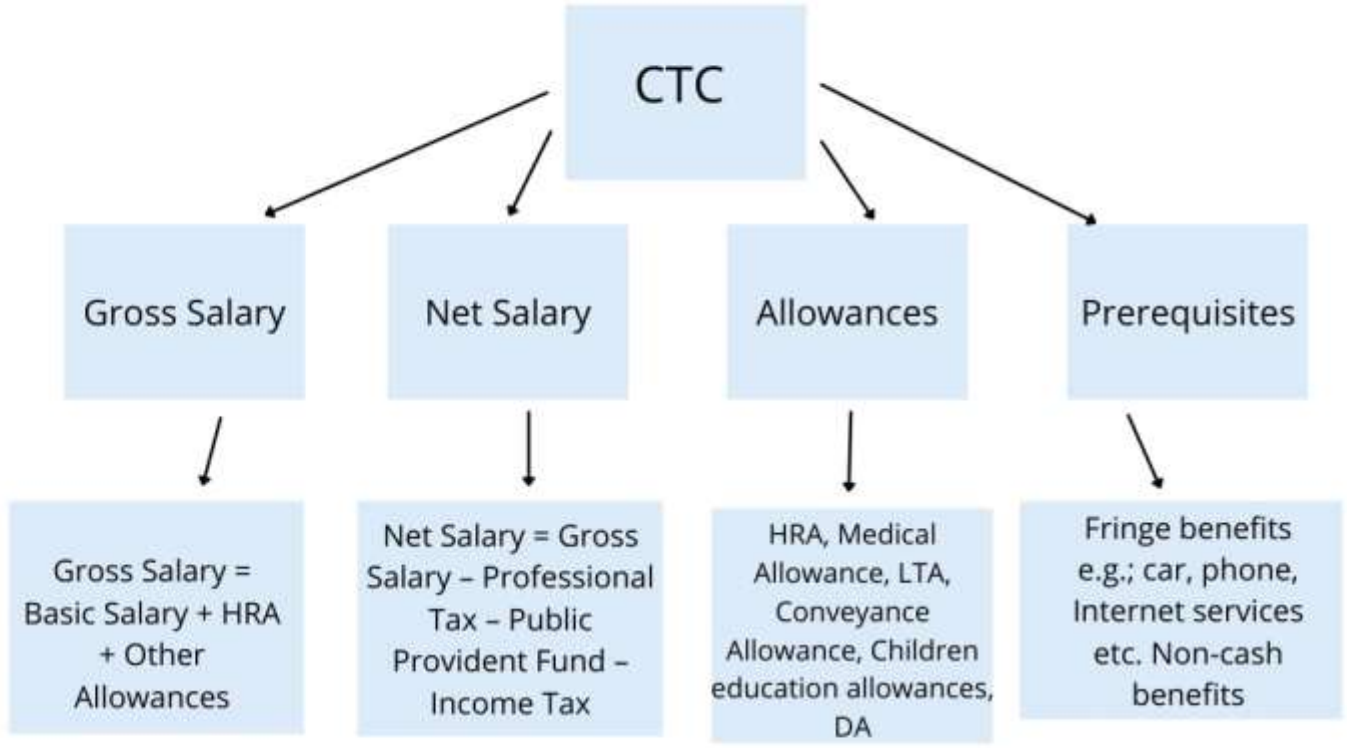

CTC
Gross Salary
Net Salary
Allowances
Prerequisites
Gross Salary = Basic Salary + HRA + Other Allowances
Net Salary = Gross Salary – Professional Tax – Public Provident Fund – Income Tax
HRA, Medical Allowance, LTA, Conveyance Allowance, Children education allowances, DA
Fringe benefits e.g.; car, phone, Internet services etc. Non-cash benefits

Fig. 4.1. Block Diagram

4.1 MODULE DESCRIPTION

4.1.1. INPUT TEXT MODULE

The input text module in the employee salary calculation Python project handles the acquisition and validation of employee data. This module is crucial as it ensures accurate and reliable input from HR personnel for processing salary calculations.

- **User Interface (UI):**

Utilizes Tkinter or another GUI framework to create input forms for HR personnel.
Provides fields for entering employee details such as name, ID, basic pay, allowances, deductions, and bonuses.

- **Input Validation:**

Validates user inputs to ensure data integrity and accuracy before processing.
Checks for:Numeric values for salary components (basic pay, allowances, bonuses).
Valid formats for employee ID and other identifiers. Mandatory fields to prevent
missing data.

4.1.2. MAIN MODULE

The main module of the employee salary calculation Python project serves as the central component that orchestrates the entire salary calculation process. It integrates various functionalities and modules to ensure accurate computation of employee salaries based on input data.

- **Initialization and Setup:**

Initializes necessary libraries, configurations, and database connections (if applicable). Sets up GUI components (if using Tkinter or another GUI framework) for user interaction.

4.1.3. EMPLOYEE MODULE

In the context of the employee salary calculation Python project, the "Employee Module" refers to the component responsible for managing individual employee information and details pertinent to salary calculations. This module plays a crucial role in collecting, storing, and processing employee-specific data required for accurate salary computations.

- **Data Input and Validation:**

Provide methods or functions to input and validate employee data:

Validate numeric values for salary components.

Ensure formats (e.g., employee ID format) are correct.

Check mandatory fields to prevent missing data.

- **Calculation Methods:**

Implement methods to calculate key salary components:

Gross Salary Calculation: Summation of basic pay, allowances, and bonuses.

Deductions Calculation: Compute income tax, PF, professional tax, and other deductions.

4.1.4. OUTPUT TEXT MODULE

In the employee salary calculation Python project, the output text module is responsible for generating and formatting various types of textual outputs, such as payslips and summary reports, based on the processed salary data. This module plays a crucial role in presenting the computed salary information in a clear and structured format for HR personnel and employees.

- **Payslip Generation:**

Create formatted payslips for individual employees that include:

Employee details (name, ID, department, etc.)

Breakdown of earnings (basic pay, allowances, bonuses)

Detailed deductions (income tax, PF, professional tax) Net
salary calculation and any additional notes or remarks.

Summary Reports:

Generate organizational payroll summaries and financial analysis reports that provide:Aggregated salary data (total payroll expenses, average salaries, etc.)

Comparative analysis (year-to-date summaries, department-wise breakdowns)

Visualizations (optional, using Matplotlib for graphs/charts).

4.1.5.EMPLOYEE MODULE

In the context of the employee salary calculation Python project, the "Employee Module" refers to the component responsible for managing individual employee information and details pertinent to salary calculations. This module plays a crucial role in collecting, storing, and processing employee-specific data required for accurate salary computations.

- **Data Input and Validation:**

Provide methods or functions to input and validate employee data:

Validate numeric values for salary components.

Ensure formats (e.g., employee ID format) are correct.

Check mandatory fields to prevent missing data.

- **Calculation Methods:**

Implement methods to calculate key salary components:

Gross Salary Calculation: Summation of basic pay, allowances, and bonuses.

Deductions Calculation: Compute income tax, PF, professional tax, and other deductions.

4.1.5. OUTPUT TEXT MODULE

In the employee salary calculation Python project, the output text module is responsible for generating and formatting various types of textual outputs, such as payslips and summary reports, based on the processed salary data. This module plays a crucial role in presenting the computed salary information in a clear and structured format for HR personnel and employees.

- **Payslip Generation:**

Create formatted payslips for individual employees that include:

Employee details (name, ID, department, etc.)

Breakdown of earnings (basic pay, allowances, bonuses)

Detailed deductions (income tax, PF, professional tax)

Net salary calculation and any additional notes or remarks.

Summary Reports:

Generate organizational payroll summaries and financial analysis reports that provide:
Aggregated salary data (total payroll expenses, average salaries, etc.)
Comparative analysis (year-to-date summaries, department-wise breakdowns)
Visualizations (optional, using Matplotlib for graphs/charts).

4.1.6.EMPLOYEE MODULE

In the context of the employee salary calculation Python project, the "Employee Module" refers to the component responsible for managing individual employee information and details pertinent to salary calculations. This module plays a crucial role in collecting, storing, and processing employee-specific data required for accurate salary computations.

- **Data Input and Validation:**

Provide methods or functions to input and validate employee data:
Validate numeric values for salary components.
Ensure formats (e.g., employee ID format) are correct.
Check mandatory fields to prevent missing data.

- **Calculation Methods:**

Implement methods to calculate key salary components:
Gross Salary Calculation: Summation of basic pay, allowances, and bonuses.
Deductions Calculation: Compute income tax, PF, professional tax, and other deductions.

4.1.6. OUTPUT TEXT MODULE

In the employee salary calculation Python project, the output text module is responsible for generating and formatting various types of textual outputs, such as payslips and summary reports, based on the processed salary data. This module plays a crucial role in presenting the computed salary information in a clear and structured format for HR personnel and employees.

- **Payslip Generation:**

Create formatted payslips for individual employees that include:
Employee details (name, ID, department, etc.)
Breakdown of earnings (basic pay, allowances, bonuses)
Detailed deductions (income tax, PF, professional tax)
Net salary calculation and any additional notes or remarks.

Summary Reports:

Generate organizational payroll summaries and financial analysis reports that provide:
Aggregated salary data (total payroll expenses, average salaries, etc.)
Comparative analysis (year-to-date summaries, department-wise breakdowns)
Visualizations (optional, using Matplotlib for graphs/charts).

Input Handling:

4.1.7. EMPLOYEE MODULE

In the context of the employee salary calculation Python project, the "Employee Module" refers to the component responsible for managing individual employee information and details pertinent to salary calculations. This module plays a crucial role in collecting, storing, and processing employee-specific data required for accurate salary computations.

- **Data Input and Validation:**

Provide methods or functions to input and validate employee data:

Validate numeric values for salary components.

Ensure formats (e.g., employee ID format) are correct.

Check mandatory fields to prevent missing data.

- **Calculation Methods:**

Implement methods to calculate key salary components:

Gross Salary Calculation: Summation of basic pay, allowances, and bonuses.

Deductions Calculation: Compute income tax, PF, professional tax, and other deductions.

4.1.8. OUTPUT TEXT MODULE

In the employee salary calculation Python project, the output text module is responsible for generating and formatting various types of textual outputs, such as payslips and summary reports, based on the processed salary data. This module plays a crucial role in presenting the computed salary information in a clear and structured format for HR personnel and employees.

- **Payslip Generation:**

Create formatted payslips for individual employees that include:

Employee details (name, ID, department, etc.)

Breakdown of earnings (basic pay, allowances, bonuses)

Detailed deductions (income tax, PF, professional tax)

Net salary calculation and any additional notes or remarks.

Summary Reports:

Generate organizational payroll summaries and financial analysis reports that provide:

Aggregated salary data (total payroll expenses, average salaries, etc.)

Comparative analysis (year-to-date summaries, department-wise breakdowns)

Visualizations (optional, using Matplotlib for graphs/charts).

CHAPTER 5

CONCLUSION AND FUTURE ENCHANCEMENT

5.1 CONCLUSION

The employee salary calculation Python project aims to streamline and automate the process of computing employee salaries within an organization. Implemented a robust calculation engine that automates the computation of gross salary, deductions, and net salary based on predefined rules and inputs. Ensured accuracy in salary calculations while adhering to tax laws,regulatory requirements, and organizational policies.Validated inputs and computations to minimize errors and discrepancies.Improved efficiency in handling employee data and generating payslips and reports. Implemented measures to secure sensitive employee information and ensure data integrity throughout the salary calculation process.Protected against unauthorized access and maintained confidentiality of payroll data.

5.2. FUTURE ENHANCEMENT

The employee salary calculation Python project can be enhanced in several ways to improve functionality, usability, and scalability. Enhance integration capabilities to connect with external APIs for real-time updates on tax rates, regulatory changes, and financial data.

Integrate with HR management systems (e.g., SAP, Workday) for seamless data synchronization and improved workflow automation.

- **Advanced Reporting and Analytics:**

Implement advanced reporting features using data visualization libraries (e.g., Matplotlib, Plotly) to create interactive charts, graphs, and dashboards.

Provide insights into payroll trends, cost analysis, and forecasting to support strategic decision-making.

- **Enhanced Security Features:**

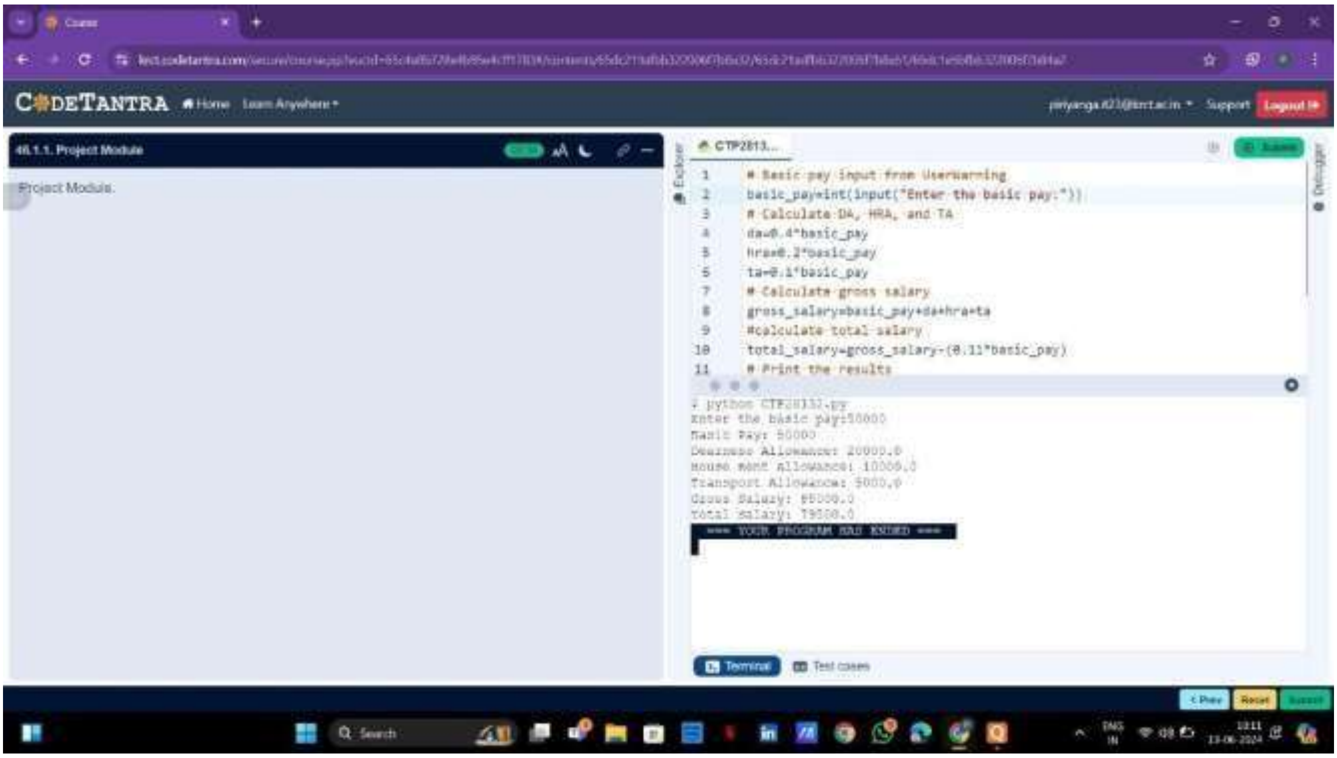
Strengthen security measures to protect sensitive employee information and ensure compliance with data protection regulations (e.g., GDPR, CCPA)

APPENDICES

APPENDIX A-SOURCE CODE

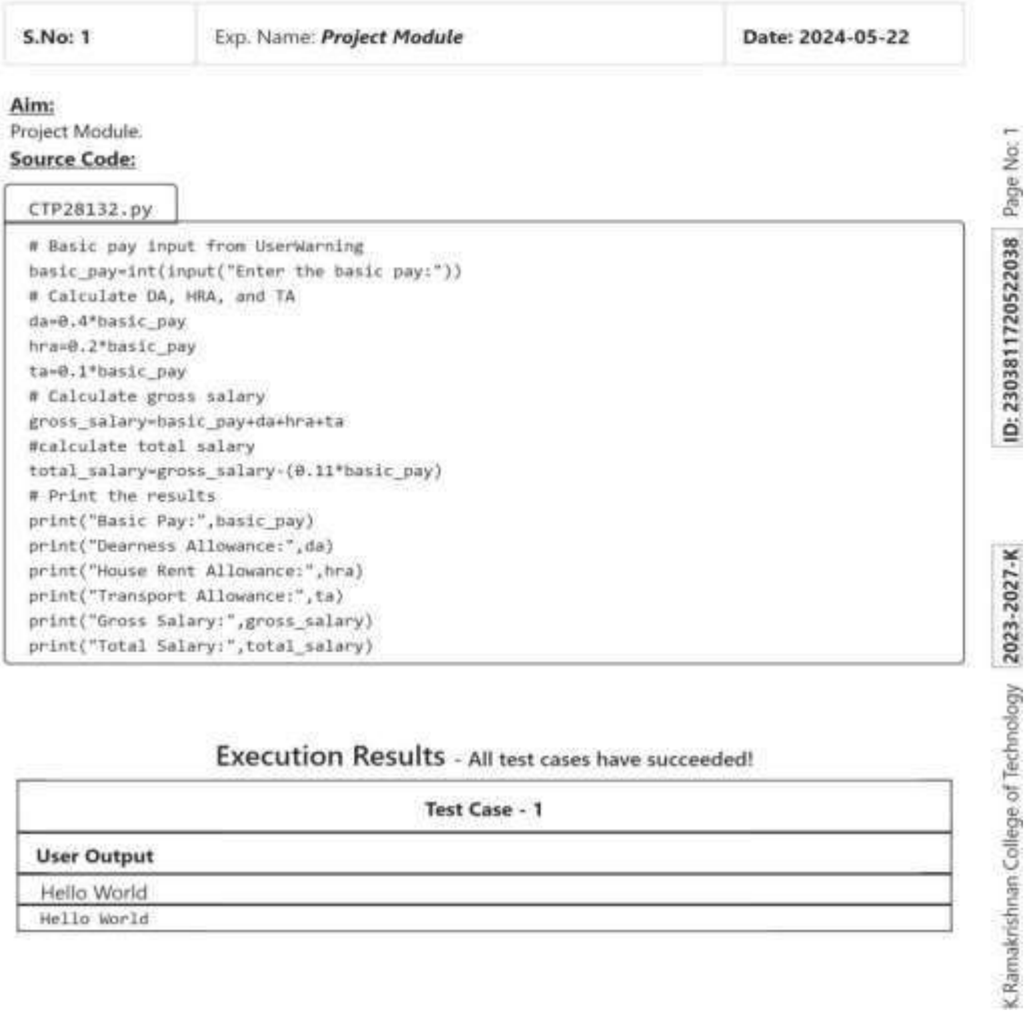
```
# Basic pay input from user
basic_pay = int(input("Enter the basic pay: "))

# Calculate DA, HRA, and TA
da = 0.4 * basic_pay
hra = 0.2 * basic_pay
ta = 0.1 * basic_pay

# Calculate gross salary
gross_salary = basic_pay + da + hra + ta
# Calculate total salary
total_salary = gross_salary - (0.11 * basic_pay)

# Print the results
print("Basic Pay: ", basic_pay)
print("Dearness Allowance: ", da)
print("House Rent Allowance: ", hra)
print("Transport Allowance: ", ta)
print("Gross Salary: ", gross_salary)
print("Total Salary: ", total_salary)
```

APPENDIX B -SCREEN SHOTS



```python
# Basic pay input from UserWarning
basic_pay=int(input("Enter the basic pay:"))
# Calculate DA, HRA, and TA
da=0.4*basic_pay
hra=0.2*basic_pay
ta=0.1*basic_pay
# Calculate gross salary
gross_salary=basic_pay+da+hra+ta
#calculate total salary
total_salary=gross_salary-(0.11*basic_pay)
# Print the results
```

```
$ python CTP28132.py
Enter the basic pay:50000
Basic Pay: 50000
Dearness Allowance: 20000.0
House Rent Allowance: 10000.0
Transport Allowance: 5000.0
Gross Salary: 85000.0
Total Salary: 79500.0
=== YOUR PROGRAM HAS ENDED ===
```



S.No: 1	Exp. Name: ***Project Module***	**Date: 2024-05-22**

Aim:

Project Module.

Source Code:

CTP28132.py

```
# Basic pay input from UserWarning
basic_pay=int(input("Enter the basic pay:"))
# Calculate DA, HRA, and TA
da=0.4*basic_pay
hra=0.2*basic_pay
ta=0.1*basic_pay
# Calculate gross salary
gross_salary=basic_pay+da+hra+ta
#calculate total salary
total_salary=gross_salary-(0.11*basic_pay)
# Print the results
print("Basic Pay:",basic_pay)
print("Dearness Allowance:",da)
print("House Rent Allowance:",hra)
print("Transport Allowance:",ta)
print("Gross Salary:",gross_salary)
print("Total Salary:",total_salary)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
Hello World
Hello World
```

Page No: 1

ID: 2303811720522038

2023-2027-K

K.Ramakrishnan College of Technology