**ITW Corporation**

NguyenTC8Mock Project

**Time Keeping System - Detailed Design**



**Version – 1.0**

**26th Jul 2024**

NNN BishopsXXXX

London

EC2N 4AY

Tel: +44 (0)20 7890 1234

Fax: +44 (0)20 7890 1235

# **Table of Contents**

[Table of Contents 2](#_Toc149334074)

[1 Introduction 5](#_Toc149334075)

[1.1 Scope 5](#_Toc149334076)

[1.2 Definitions, Acronyms and Abbreviations 5](#_Toc149334077)

[1.3 Overview 5](#_Toc149334078)

[1.4 References 6](#_Toc149334079)

[2 Software Requirements 7](#_Toc149334080)

[3 Design 8](#_Toc149334081)

[3.1 System Overview 8](#_Toc149334082)

[3.2 Sequence diagram 9](#_Toc149334083)

[3.2.1 Feature 1 9](#_Toc149334084)

[3.2.2 Feature 2 11](#_Toc149334085)

[3.3 Class diagram 13](#_Toc149334086)

[3.3.1 Feature 1 13](#_Toc149334087)

[3.3.2 Feature 2 13](#_Toc149334088)

[4 Implementation 14](#_Toc149334089)

[4.1 VendorExt-Project Time Keeping System 14](#_Toc149334090)

[4.2 Source code structure 14](#_Toc149334091)

[4.2.1 Source code file name of components: 14](#_Toc149334092)

[4.2.2 Directory structure 14](#_Toc149334093)

[4.3 Common 15](#_Toc149334094)

[4.3.1 Constants 15](#_Toc149334095)

[4.3.2 Structure list 15](#_Toc149334096)

[4.3.3 Function list 16](#_Toc149334097)

[4.4 Features Detail 18](#_Toc149334098)

[4.4.1 Feature 1 18](#_Toc149334099)

[4.4.2 Feature 2 19](#_Toc149334100)

[5 Unit Test 22](#_Toc149334101)

[5.1 Unit Test plan 22](#_Toc149334102)

[5.1.1 Prerequisite 22](#_Toc149334103)

[5.1.2 Testing environment 22](#_Toc149334104)

[5.1.3 Convention 22](#_Toc149334105)

[5.1.4 Assumptions 22](#_Toc149334106)

[5.2 Unit Test cases 23](#_Toc149334107)

[5.2.1 Common Functions Test cases 23](#_Toc149334108)

[5.2.2 Features Test cases 23](#_Toc149334109)

[5.3 Result of the Unit Test. 23](#_Toc149334110)

[6 Open Issues 23](#_Toc149334111)

[7 Appendix 23](#_Toc149334112)

RECORD OF CHANGES

|  |  |  |  |
| --- | --- | --- | --- |
| **Document Version History** | | | |
| **Version** | **Date** | **Author** | **Details** |
| 1.0 | 20th Jul, 2024 | Ho Thi Kim Nga | Main author |

**APPROVERS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Distribution List** | | | | |
| **Name** | **Role** | **Signature** | **Date** | **Version** |
| NguyenTC8 | Trainer |  | 18-07-2024 | 1.0 |

**CONFIDENTIALITY**

This document is distributed on a restricted basis, is commercial in confidence to the recipient, and may not be used for any purpose other than that associated with an ITW Corporation project. The contents of this document may not be disclosed to any third parties without the expressed advance written authorisation of ITW Corporation.

# Introduction

The document is to describe how Project ‘Time Keeping System’ components work in low level view. A system which records check-in time (when employee arrives in office) and check-out time (when employee leaves out office) of all employees each day in a company.

At a periodic or random time, company’s accountants will calculate salary based on the checkin/out time.

Definitions, Acronyms and Abbreviations

Table 1: Abbreviations in this document

|  |  |
| --- | --- |
| **Acronym** | **Reference** |
| ACS | Automatic Configuration Server |
| ID | Identification |
| VSCode | Visual Studio Code |

Overview

This document includes following sections:

1. **Introduction:**

The C program is designed to manage employee attendance and payroll, featuring two operational modes: Admin Mode and Normal Mode. Upon startup, users can select either mode from the main menu. In Admin Mode, protected by a default encrypted password, administrators can change the admin password, add new employees with automatically generated IDs, view all employee information, track the working hours of specific employees, and calculate salaries considering both actual working time and penalties for late check-ins or early check-outs. Normal Mode provides employees with the functionality to check in and check out by entering their ID and confirming their identity. The program ensures data security by encrypting passwords and simplifies payroll management through automated calculations and data handling.

1. **Software Requirement:**
2. **Unit test:**

This section includes information relates to Unit Test for Project Time Keeping system as following:

* Description of the Unit Test plan.
* List of Unit Test cases and their description.
* Result of the Unit Test.

References

|  |  |
| --- | --- |
| **Documentation Name** | **Description** |
| HCM24\_FRF\_EMB\_04 - CF Module - Mock Project.pdf | System Requirement Specification document which contains requirement of all features. |
| https://en.wikipedia.org/wiki/SREC\_(file\_format) | Technical specification about the S-Record file format. |

# Software Requirements

**Table 2: Software requirement table for Project Time Keeping System**

|  |  |  |
| --- | --- | --- |
| **Module Time Keeping System** | | |
| **No.** | **Requirement** | **Note** |
| **1** | **Admin mode** |  |
| **1.1** | Enter the administrator password as 12345678. Change the administrator password. |  |
| **1.2** | Adding an employee has 3 sections: Employee code, full name, basic salary (how many VND/hour). |  |
| **1.3** | View information of all employees (employee code, full name, basic salary). |  |
| **1.4** | View the work hours of a specified employee. |  |
| **1.5** | Calculate salaries for all employees. Salary is calculated according to regulations. |  |
| **2** | **Normal mode** |  |
| **2.1** | Check-in. |  |
| **2.2** | Check-out. |  |
| **2.3** | Enter the employee ID, display the employee's full name, and wait for confirmation from the employee. |  |
| **2.4** | Add the corresponding "Check-in time" or "Check-out time" section of this employee and save it to the data file. |  |
| **2.5** | Return to the menu of Normal mode, displaying 2 items Check In and Check Out. |  |
| **3** | **Other** |  |
| **3.1** | Encrypt the administrator password before saving to file. |  |
| **3.2** | The employee's full name has a maximum length of 24 characters. |  |
| **3.3** | Check-in and check-out times are calculated in minutes. |  |

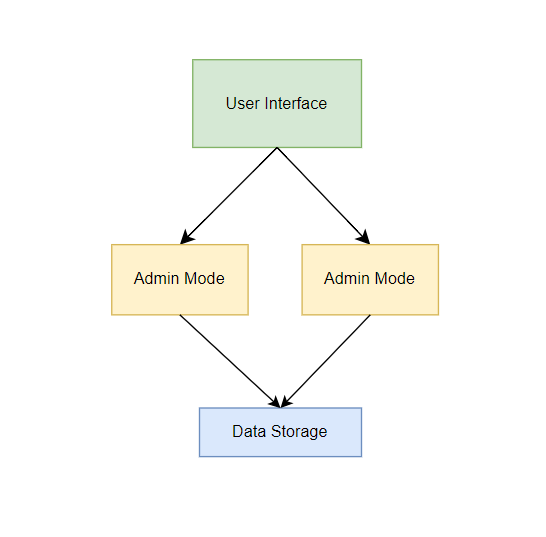
# Design

This section includes diagrams which describe behavior of components in Project Time Keeping System.

The diagrams shown following are written in UML language using StarUML tool.

System Overview

The following diagram shows system architecture of PROJECT Time Keeping System



**Figure 1: PROJECT** Time Keeping System **system block diagram**

Currently, Project Time Keeping System has 2 components. They are: Admin mode; Normal mode.

Behavior outline:

* Admin mode: The admin logs in with the password, then the admin changes the new password. Admin enters employee information, views employee information, calculates employee salary.
* Normal mode: The employee enters the ID, then if the ID is confirmed, the employee will select modes such as check-in or check-out, then the employee's time will be saved for salary calculation.

Sequence diagram

### Feature 1

#### Get Info

Figure 2: Get Info sequence diagram

This system has 2 main modes: Administrator Mode and Normal Mode.

First, the main function will display the menu and ask the user to select a mode. When the user selects the number '1', the program will enter admin mode and require user interaction for the mode to perform its functions.

Similarly, when the user selects the number '2' the program will enter normal mode and require user interaction. The values ​​updated by the user will then be saved to the data file for storage.

In the employee salary calculation section, there is periodic salary calculation and any salary calculation.

#### Get Device Status

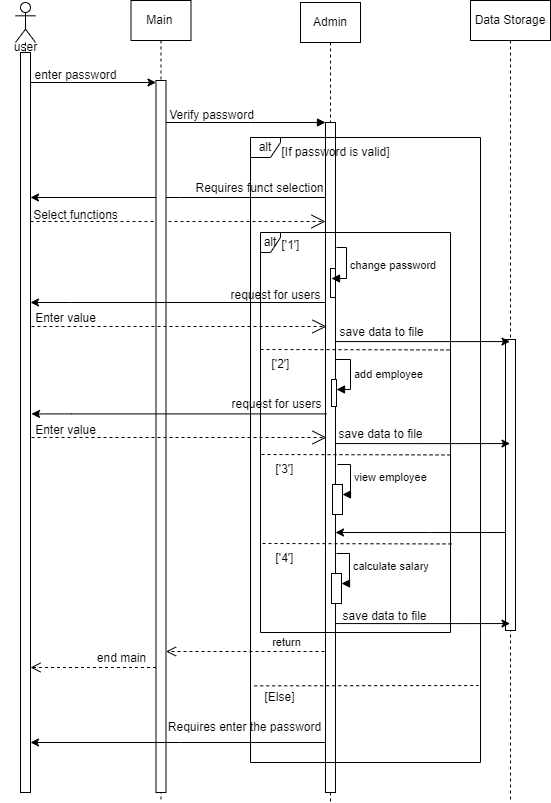


Figure 3: Admin mode sequence diagram

When the user chooses to enter admin mode. The program will ask the user to enter a password to enter admin mode. If the user enters the correct password, the program will display the admin mode menu.

Admin will select the corresponding functions to perform the job such as: changing another password, adding employee information, viewing employee working hours and calculating salary. After the admin enters the data and the calculated salary will be calculated. Save to file Data. If the user enters the wrong password, the program will ask the user to re-enter.

### Feature 2

#### Perform Self-Test

Figure 4: Normal mode sequence diagram

When the user selects normal mode, the program will ask the user to enter the employee's ID. If the employee ID is valid, the program will display the Normal mode menu.

That employee will choose to enter Check-in or Check-out mode, then the employee's real-time data will be saved and put into a Data file. If the user enters an invalid ID, the program will ask the employee to re-enter the ID.

Structure diagram

### Feature 1

Figure 6: class diagram

The class diagram illustrates the relationships and interactions between the various components and functions within the application. At the top level, the `main` function serves as the entry point, offering a menu to choose between Admin Mode and Normal Mode. In Admin Mode, several key functions are encapsulated: `change\_admin\_password` for modifying the admin password, `add\_employee` for adding new employee records with unique IDs, `view\_employees` for listing all employees' details, `view\_working\_time` for displaying the working hours of specific employees, and `calculate\_salaries` for computing employee salaries based on their working hours and penalties. Normal Mode encompasses the `check\_in` and `check\_out` functions, which manage employee attendance by recording their check-in and check-out times. The `Employee` structure is central to the diagram, containing attributes such as `id`, `name`, `basic\_salary`, `check\_in\_time`, and `check\_out\_time`, which are used across various functions to manage employee data and calculate salaries. The diagram highlights the hierarchical and functional dependencies, ensuring a clear understanding of how different parts of the program interact to achieve the desired functionalities.

# Implementation

Common

General function: printf(), scanf(), time().

Datatypes: int, char, float, struct, void.

### Constants

|  |  |  |
| --- | --- | --- |
| **Constant Name** | **Value** | **Description** |
| PROJECT\_Time Keeping System\_ADMINPASSWORD\_LENGTH | 8 | Maximum length of an interface |
| PROJECT\_ Time Keeping System \_NAME\_LENGTH | 24 | Maximum length of an interface |
| PROJECT\_ Time Keeping System \_TIMER\_CHECKIN | 8 | Maximum time to check in |
| PROJECT\_ Time Keeping System \_TIMER\_CHECKOUT | 17 | Maximum time to check out |
| PROJECT\_ Time Keeping System \_MINUS | 20000 | The amount of money deducted for being late |
| PROJECT\_ Time Keeping System \_DAY\_IN\_MONTH\_NUMBER | 31 | Maximum number of an interface |
| PROJECT\_ Time Keeping System \_TIME\_LUNCH\_START | 12 | Lunch break start and end times |
| PROJECT\_ Time Keeping System \_TIME\_LUNCH\_END | 13 |

### Structure list

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| PROJECT\_ Time Keeping System \_EMPLOYEE\_INFO: Structure for employee information which is used by any component of Project Time Keeping System | | |
| ID | PROJECT \_Time Keeping System \_INT | Employee’ ID |
| NAME [25] | PROJECT \_Time Keeping System \_char | Employee’ name, Array field |
| checkInTime[DAYS\_IN\_MONTH] | PROJECT \_Time Keeping System \_struct tm | Timer status |
| checkOutTime[DAYS\_IN\_MONTH] | Timer status |
| Related constants: Yes  - fiels name[24] | | |
| *STRUCT\_NAME*: Employee | | |
| ID | int |  |
| name[25] | char | Array field |
| basic\_salary | float |  |

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| PROJECT\_ Time Keeping System \_WorkingTime: used to define a data structure that can group different data types together under a single name. | | |
| Employee\_id | PROJECT \_Time Keeping System \_int | Employee’ ID |
| check\_in | PROJECT \_Time Keeping System \_struct tm | Timer status |
| check\_out | PROJECT \_Time Keeping System \_struct tm | Timer status |
| Related constants: No | | |
| *STRUCT\_NAME*: WorkingTime | | |
| Employee\_id | int |  |
| check\_in | struct tm | Timer status |
| check\_out | struct tm | Timer status |

### Function list

|  |
| --- |
| int main() |
| **Description:**   * Outline:   This function is automatically generated from template file of Project Time keeping system.  The program begins and displays the main menu.  Select Mode: Admin Mode, Normal Mode  Exit |
| **Flowchart:** |

## Features Detail

### Feature 1

#### Notes

Feature 1’ main functionality: Admin\_mode

* Functionality 1: Enter the administrator password. Change the administrator password.
* Functionality 2: Adding an employee.
* Functionality 3: View information of all employees.
* Functionality 4: View the work hours of a specified employee.
* Functionality 5: Calculate salaries for all employees.

#### Function list

|  |  |  |  |
| --- | --- | --- | --- |
| void Admin\_mode() | | | |
| **Arguments** | admin\_mode  change\_admin\_password  add\_employee  view\_employees  view\_working\_time  calculate\_salary  check\_in  check\_out  save\_data  load\_data  encrypt\_password  get\_current\_time | void  void  void  void  void  void  void  void  void  void  void  void |  |
| **Description:**   * Outline:   The function is called when the feature wants to initiate this component.  The user is prompted to enter the Admin password.  Verify Password  Display Admin Menu  Select function  After completing a function, return to the Admin Menu to continue with another selection or exit.  If the Admin chooses to exit, the program terminates. | | | |
| **Flowchart:** | | | |

### Feature 2

#### Notes

Feature 2’ main functionality: Normal mode.

* Functionality 1: Check-in.
* Functionality 2: Check-out.

#### Function list

|  |  |  |  |
| --- | --- | --- | --- |
| Time Keeping System \_Normal Time Keeping System Normal(); | | | |
| **Arguments** | Check-in  Check-out | void  void |  |
| **Description:**   * Outline:   Display Normal Mode Menu  Select Function  Return to Normal Mode Menu  If the user chooses to exit, the program terminates. | | | |
| **Flowchart:** | | | |