|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2-m-logo-word-ppt_1420714889 | | | | **Macromedia University of Applied Sciences** | | | | | | | |  |
|  | | | | | | | | | | | |
|  | | | | Course Title: Advanced Coding Skills  Name of Examiner: Prof. Dr. René Brunner and William Baker Morrison | | | | | | | |
|  | | | | | | | | | | | |
| **To be completed by students:** | | | | | | | | | | | |  |
| 303496 | | | | | | |  | | B-UBr DT ATI 6e 23W | | |  |
| Student ID number | | | | | | |  | | Matriculation | | |  |
| RAZANAMAHOLY | | | | | | |  | | Marvyn | | |  |
| Last name | | | | | | |  | | First name | | |  |
| The student work will be submitted as:  (Please fill in the letter X in the appropriate box) | | | | | | | | | | | |  |
|  | | | | | | | | | | | |  |
|  | X | | Individual work | | | | | | | | |  |
|  | | | | | | | | | | | |  |
|  |  | | Group work | | | | | | | | |  |
|  | | | | | | | | | | | |  |
|  | | | | | | | | | | | |  |
| **Does only apply to group work:** (Complete only if it is a group work)  **If you are submitting a group work, please list the first and last names of all group members. The names must be entered electronically by the respective group members themselves. By entering the name, it is confirmed that the student agrees to submit the paper in the present form.** | | | | | | | | | | | |  |
| If you are submitting a group work, please list the first and last names of all group members. The names must be entered electronically by the respective group members themselves. By entering the name, it is confirmed that the student agrees to submit the paper in the present form. Furthermore, by entering their names it is declared by the individual groups members to have created the project paper (in case of a group work: the part which the respective student has contributed to the paper and has marked accordingly within the paper) on their own, without the help of others. In the process, the student has not used any aids other than those cited in the listing of sources and literature. All passages taken either verbatim or in adapted form from publications are indicated as such. The work has not been submitted to another examination office in the same or a similar form. | | | | | | | | | | | |  |
|  | | | | | | | | | | | |
| 1) | |  | | | |  | | 5) | | |  |
| 2) | |  | | | |  | | 6) | | |  |
| 3) | |  | | | |  | | 7) | | |  |
| 4) | |  | | | |  | | 8) | | |  |
|  | | | | | | | | | | | |
| Assessment of group work:  (Please fill in the letter X in the appropriate box) | | | | | | | | | | | |
|  | | | | | | | | | | | |
|  |  | | I apply for an individual evaluation (i.e. each member of the group will receive an individual mark) | | | | | | | | |
|  | | | | | | | | | | | |
|  |  | | I apply for a group evaluation (i.e. each member of the group receives an identical grade) | | | | | | | | |
|  | | | | | | | | | | | |
| Berlin/ 26.06.2024 | | | | |  | | | | | RAZANAMAHOLY Marvyn | |  |
| Place/Date | | | | |  | | | | | Complete First Name and Last Name | |  |
|  | | | | | | | | | | | |  |
| **To be completed by the examiner:** (Text area for the second examiner) | | | | | | | | | | | |  |
|  | | | | | | | | | | | |  |
| *Title Page for Digital Submission, Macromedia version 1/2020* | | | | | | | | | | | |  |

**Advanced coding skills: Documentation**

**Table of contents**

Overview4

***Structure5***

***How the application works ?6***

***Conclusion……………………………………………………………………………….7***

**Overview:**

The purpose of this document is to provide a comprehensive overview and a detailed explanation of a To-Do List application developed using Python's Tkinter library for the Graphical User Interface (GUI). This application is designed to offer users an efficient way to manage and organize their daily tasks. By using this application, users can add new tasks, delete existing ones, choose specific tasks to focus on, and mark tasks as complete once they are accomplished. The intuitive and user-friendly interface of the application makes it accessible to users of all technical backgrounds.

This project is structured based on Object-Oriented Programming (OOP) principles, incorporating key OOP concepts such as inheritance, encapsulation, and modularization. The application is divided into various classes and modules, each responsible for different functionalities, ensuring a clean and maintainable codebase. By leveraging OOP, the To-Do List application is not only easy to extend and modify but also promotes code reusability.

The primary goal of the To-Do List application is to enhance the user's productivity by providing a simple yet powerful tool to keep track of their tasks and responsibilities. Whether users need to manage their daily chores, professional duties, or personal projects, this application serves as a reliable assistant in their day-to-day life. By enabling users to list their tasks, set priorities, and track their progress, the To-Do List helps them stay organized and focused.

To ensure the application is user-friendly, the GUI designed with Tkinter includes clear labels, intuitive buttons, and a visually appealing layout. The use of Tkinter not only simplifies the development process but also allows for the creation of a responsive and interactive interface. The application window is designed to be resizable, adapting to various screen sizes and user preferences.

**Structure:**

The structure of the To – Do – List Application is divided in 3 modules: Main Entry point, UI (user interface), and Application Logic.

1. Main Entry Point :

* The Main Entry Point initializes the main window and runs the application.

1. User interface :

* The user interface takes care of the graphical part of the application, such as, buttons, frames, title and more other widgets.

1. Application Logic :

* The application logic takes care of all the actions that is available in the application. Such as, adding, deleting, managing, marking, etc…

With the use of inheritance two classes are present in the code. One parent class called “ToDoAppLogic” and a child class called “ToDoAppUI”.

***ToDoAppLogic:***

It is acting as the backbone of the application, handles the core functionalities. This includes managing the list of tasks, adding new tasks, deleting tasks, marking tasks as complete, and other essential operations. Methods such as **add\_task, delete, delete\_all, choose\_random, number\_of\_task, and mark \_complete** are central to this class, ensuring all task-related functionalities are well-organized and efficient.

***ToDoAppUI:***

This class brings the application to life with a user-friendly graphical interface using the tkinter library. This class, which inherits from ToDoAppLogic, is responsible for creating and managing all **UI elements, like buttons, labels, entry fields, and list boxes**. It ensures that user interactions, such as button clicks and key presses, are correctly linked to the functions provided by the ToDoAppLogic class. The init\_ui\_components method sets up all the necessary UI components, while update\_listbox keeps the displayed task list current. By keeping the logic and UI separate, the application remains clean, maintainable, and easy to enhance in the future.

**How the application works?**

***Initialization:*** The application starts off by creating a tkinter.Tk () root window, which is then passed to the ToDoAppUI class. This class sets up the user interface and initializes all necessary components.

***UI Components*:** The init\_ui\_components method in ToDoAppUI is responsible for creating and configuring various widgets, such as labels, frames, entry fields, and buttons. It also binds specific actions to events, like pressing the "Enter" key or double-clicking a task.

***Task Management:*** The ToDoAppLogic class manages the core functionality of adding, deleting, and handling tasks. It includes methods like add\_task, delete, delete\_all, choose\_random, number\_of\_task, and mark\_complete.

***Event Handling****:* User interactions with the interface, such as button clicks and key presses, trigger the relevant methods in the ToDoAppLogic class. These methods update the task list and display the appropriate messages.

***Updating the UI****:* The update\_listbox method in ToDoAppUI ensures the list of tasks displayed in the listbox widget is kept up-to-date, with completed tasks highlighted for clarity.

**Conclusion:**

The To-Do List application is a well-organized program that effectively separates its user interface from the core logic using object-oriented programming principles. This design ensures that the code is structured, easy to maintain, and can be scaled up as needed. The application starts by creating a `tkinter.Tk()` root window, which is then passed to the `ToDoAppUI` class. Within this class, the user interface is set up and all necessary components are initialized. By inheriting from `ToDoAppLogic`, the `ToDoAppUI` class is responsible for creating and configuring various widgets like labels, frames, entry fields, and buttons through the `init\_ui\_components` method. It also binds actions to specific events, such as pressing the "Enter" key or double-clicking a task, to enhance the user experience. The core functionality of task management is handled by the `ToDoAppLogic` class, which includes methods for adding, deleting, deleting all, choosing a random task, displaying the number of tasks, and marking tasks as complete. User interactions with the interface trigger relevant methods in the `ToDoAppLogic` class, which update the task list and display appropriate messages. The ToDoAppUI's update\_listbox method ensures that the list of tasks shown in the listbox widget is always current, with completed tasks highlighted for easy visibility. This approach not only simplifies development but also allows for easier future enhancements and upkeep. By utilizing the tkinter library, the application offers a dependable and user-friendly method for managing tasks, making it a practical and effective tool for users.



Sworn  
Statement

I, **Razanamaholy Marvyn**

Born on **24/10/2005**

Hereby declare that I have prepared this thesis independently and without external assistance. In doing so, I have not used any aids other than those mentioned in the enclosed list of sources.

All points that have been taken from publications literally or adapted form have been identified as such by me.

**.Berlin......**......................., ……**26/06/2024…**…………… ……….. …Dessins manuscrits
Dessins manuscrits


[Place] [Date] [Signature]