Title: Home Maintenance Scheduler
Course: CIS152 Data Structures
Student: David Strong
Instructor: Dr. Michelle E. Ruse

Final Project Report

Introduction

- Home Maintenance Scheduler is designed to address the common challenges homeowners face in maintaining their properties. By automating task scheduling and providing clear, actionable insights into the health of various home maintenance categories, the application aims to prevent costly repairs and enhance property upkeep.

Project Description

- The project centers around a dynamic user interface that visualizes scheduled maintenance tasks through list and in the future calendar views. It uses various data structures and algorithms to efficiently manage and display tasks based on priority and due dates. The inclusion of a visual health status graph, although simplified from the initial proposal, allows homeowners to quickly assess maintenance areas needing attention.

Time/Change Logs

- Week of March 20, 2024
  - Accomplishments: Project setup. Initial planning and configuring the development environment in PyCharm.
  - Blockers: None encountered.
  - Scope Changes: None at this stage.
  - Next Steps: Outline the basic structure for key classes and initialize a Git repository.
- Week of March 26, 2024
  - Accomplishments: Outlined basic classes for Task, Scheduler, User, and Category Health. Set up Git repository and began writing unit tests.
  - Blockers: Difficulty in choosing the optimal data structure for task scheduling.
  - Scope Changes: Opted for using Python's built-in list and dictionary structures over a priority queue or skip list for simplifying the implementation.
  - Next Steps: Finish unit tests, begin developing the user interface focusing on task display and interaction.
- Week of April 2, 2024
  - Accomplishments: No progress due to personal health issues (food poisoning).
  - Blockers: Having issues with GitHub updates.
  - Scope Changes: N/A
  - Next Steps: Catch up on the delayed user interface development.
- Week of April 9, 2024
  - Accomplishments: Resumed work, completed unit tests for all core classes, and started development on the user interface.
  - Blockers: Initial delay in integrating backend with the user interface.
  - Scope Changes: Decision to adjust visualizations from dynamic graphs to simpler tabular displays for task and health status management.
  - Next Steps: Integrate user interface fully with backend logic, begin implementing task loading from JSON.
- Week of April 16, 2024
  - Accomplishments: Completed integration of user interface with backend. Implemented task loading from JSON and manual task additions through GUI.
  - Blockers: Challenges in getting the GUI to update dynamically when tasks were added or modified.
  - Solutions: Enhanced signal-slot connections within the PySide6 framework to ensure UI updates were handled correctly.
  - Scope Changes: Simplified some UI elements to enhance usability based on user feedback.
  - Next Steps: Extensive testing of GUI functionality, start debugging.
- Week of April 23, 2024
  - Accomplishments: Conducted thorough testing, began debugging identified issues particularly around task scheduling and status updates.
  - Blockers: Found bugs in task completion and rescheduling functionalities.
  - Solutions: Refined task scheduling algorithms, fixed issues with task status updates affecting category health calculations.

Title: Home Maintenance Scheduler
Course: CIS152 Data Structures
Student: David Strong
Instructor: Dr. Michelle E. Ruse

    - Scope Changes: None additional.
    - Next Steps: Complete debugging, finalize documentation, and prepare for project presentation.
- Week of April 30, 2024
    - Accomplishments: Final preparations for project submission. Completed documentation, made last-minute tweaks to the user interface, and confirmed functionality.
    - Blockers: Minor issues during deployment testing.
    - Solutions: Resolved outstanding deployment issues.
    - Scope Changes: None.
    - Final Steps: Submit the completed project.

Development Timeline

- Initial Setup (March 20 - 26, 2024): Configured development environment and outlined basic software architecture.
- Integration and Backend Development (March 27 - April 9, 2024): Developed core backend functionalities including the scheduler and the task management system, focusing on unit tests and basic UI components.
- UI Development and Integration (April 10 - 23, 2024): Focused on developing and refining the GUI, integrating backend services with frontend displays.
- Final Testing and Refinement (April 24 - 30, 2024): Conducted extensive testing and debugging, finalized documentation and prepared the project for submission.

Key Changes and Adjustments

- Simplified data structures for task management to lists and dictionaries for better integration and ease of development.
- Adjusted task sorting from the proposed TimSort to Python's built-in sorting methods, leveraging built-in efficiencies.
- Shifted from dynamic graphical representations to tabular views for health statuses to streamline the development and enhance performance.

Lessons Learned:

- **Project Scope and Evolution**
    - The project's scope evolved significantly, from a high-concept dynamic visualization tool to a more streamlined, functional application focusing on essential maintenance scheduling tasks. This shift was largely due to practical development and focusing on achievable goals within the project timeline.
    - One significant challenge was integrating the backend logic with the frontend GUI in a way that ensured real-time updates and responsiveness. This was addressed by enhancing the signal-slot connections within the PySide6 framework, ensuring that changes in the backend were immediately reflected in the GUI.

CODE Including Comments

- **GitHub Repository:** Home Maintenance Scheduler
    - https://github.com/StrongNavy83/CIS152-Final-Data-Structures.git

User's Manual

- **Running the Application**
    - To run the Home Maintenance Scheduler, ensure that Python and PySide6 are installed, and execute the main_gui.py script.
    - The application's interface will guide the user through scheduling tasks, viewing upcoming maintenance requirements, and assessing the health of various home maintenance categories.
- **Interaction Guide**
    - Users can add new tasks via the "Add New Task" button, view tasks in a list, and check off completed tasks to update category health statuses. The system automatically adjusts task priorities and due dates based on user interactions.

Conclusion/Summary

Title: Home Maintenance Scheduler
Course: CIS152 Data Structures
Student: David Strong
Instructor: Dr. Michelle E. Ruse

- The Home Maintenance Scheduler effectively simplifies the management of home maintenance tasks, providing a user-friendly interface to track and schedule essential tasks. This project has not only fulfilled initial objectives but has also provided a foundation for future enhancements.

MERUSE Principles Applied:

- **Modularity:** The application is built with multiple classes, each handling specific aspects of the maintenance scheduling process.
- **Efficiency:** Utilizes efficient data handling and sorting mechanisms to minimize resource usage while maintaining scalability.
- **Readability:** Code is self-commenting with clear naming conventions and detailed comments throughout.
- **Usability:** Designed with an intuitive interface ensuring ease of use for non-technical users.
- **Stability:** Includes error handling to manage exceptions gracefully, ensuring the application does not crash unexpectedly.
- **Elegance:** Maintains a consistent coding style and uses efficient algorithms to enhance both performance and maintainability.