



Faculty of Computing and Information Technology

AMCS1034 Software Development Fundamentals




Practical Assignment Report

2024/2025 Semester

Tutor : YEN WEI YAN

Programme: DCS1 S2

Tutorial Group: Group 1

No.	Student Names	Student ID	Contribution (%)	Signature	Marks (20%)
1.	TAN WY HANG	2401976	50%		
2.	ANG KAR WAI	2402009	35%		
3.	MAK JUN XIANG	2402035	15%		
Total:			100%		

Date of Submission : 25/12/2024

Date Received by Tutor : _____



FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

AMCS1034 SOFTWARE DEVELOPMENT FUNDAMENTALS

Diploma in Computer Science (DCS)

Year 1 Semester 2

Group 1

PLAGIARISM STATEMENT

Read, complete, and sign this statement to be submitted with the written report.

We confirm that we have read and shall comply with all the terms and conditions of TAR University of Management and Technology's plagiarism policy.

We declare that this assignment is free from all forms of plagiarism and for all intents and purposes is my own properly derived work.

Declaration Statement Acknowledged by




No	Name	Student ID	Signature	Date
1	TAN WY HANG	2401976		24/12/2024
2	ANG KAR WAI	2402009		24/12/2024
3	MAK JUN XIANG	2402035		24/12/2024

Table of Contents

1. SOLARIS App Introduction.....	3
1.1 Overview.....	3
1.2 Application Objectives.....	3
1.2 System Design Utilities.....	4
1.4 Main User Interface.....	5
 2. GPA Calculator.....	 7
2.1 System Description.....	7
2.2 Stepwise Refinement.....	15
2.3 User Manual.....	16
2.4 Test Cases.....	18
 3. Pomodoro Timer.....	 26
3.1 System Description.....	26
3.2 Stepwise Refinement.....	27
3.3 User Manual.....	27
3.4 Test Cases.....	29
 4. To-do List Manager.....	 35
4.1 System Description.....	35
4.2 Stepwise Refinement.....	36
4.3 User Manual.....	37
4.4 Test Case.....	38

1. SOLARIS App Introduction

1.1 Overview

SOLARIS (Student-Oriented Learning Refined Integration System) is a **tab-based** comprehensive productivity and academic management suite designed to enhance students' learning experience and time management capabilities. By integrating three essential academic tools - **GPA Calculator**, **Pomodoro Timer**, and **To-Do List** - SOLARIS creates a unified environment that supports students throughout their academic journey.

SOLARIS is built using the CustomTkinter framework, the application implements a modern, user-friendly interface with a customisable themes and the Windows 11 Mica effect for optimal visual comfort during extended study sessions.

The application utilizes a tabbed interface architecture, allowing seamless navigation between three core modules while maintaining a cohesive user experience. Each module is designed to operate both independently and in conjunction with others, providing a comprehensive study management solution.

1.2 Application Objectives

Academic Performance Management

- Accurate GPA tracking and calculation with support for different grading scales
- Historical grade data maintenance for trend analysis
- Visual representation of academic performance
- Credit hour management and distribution visualization
- Customizable grade mapping system

Time Management

- Implementation of the Pomodoro Technique with customizable intervals
- Focus session tracking with detailed statistics
- Intelligent break scheduling based on work patterns
- Visual and audio notification system
- Progress tracking across multiple study sessions

Task Organization

- Comprehensive task management with priority levels
- Deadline tracking and reminder system
- Category-based organization
- Progress monitoring and completion statistics
- Search and filter capabilities

1.3 System Architecture

1.2 System Design Utilities

SOLARIS employs a modular architecture built on modern Python frameworks:

❖ Frontend Layer

- CustomTkinter for modern UI components
- Tkinter for base GUI functionality
- Dynamic window management system
- Responsive design principles
- Custom widget implementations

❖ Data Layer

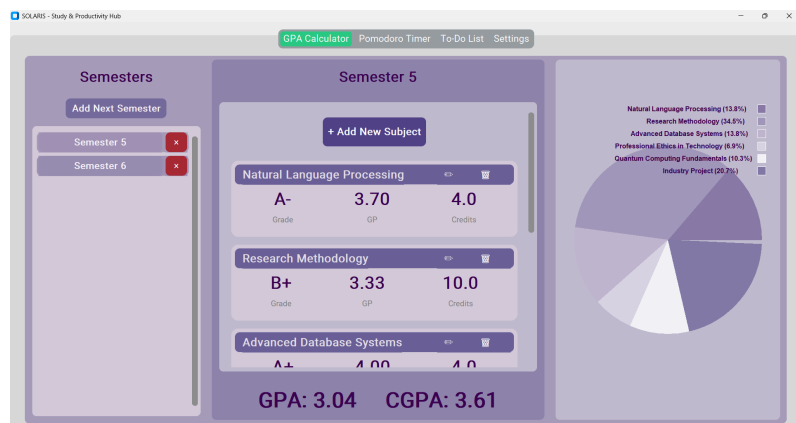
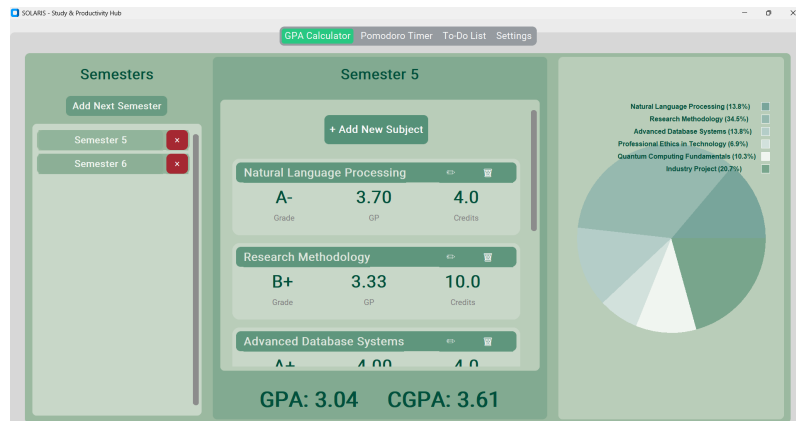
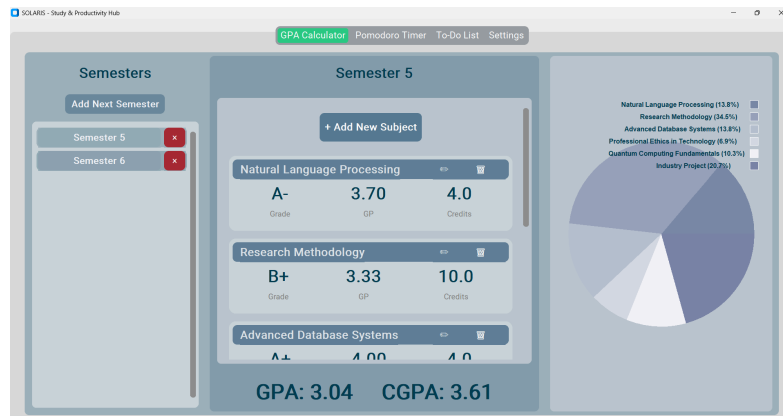
- JSON-based persistent storage
- File system integration
- Data validation and sanitization
- Error handling and recovery
- Backup management

❖ Business Logic Layer

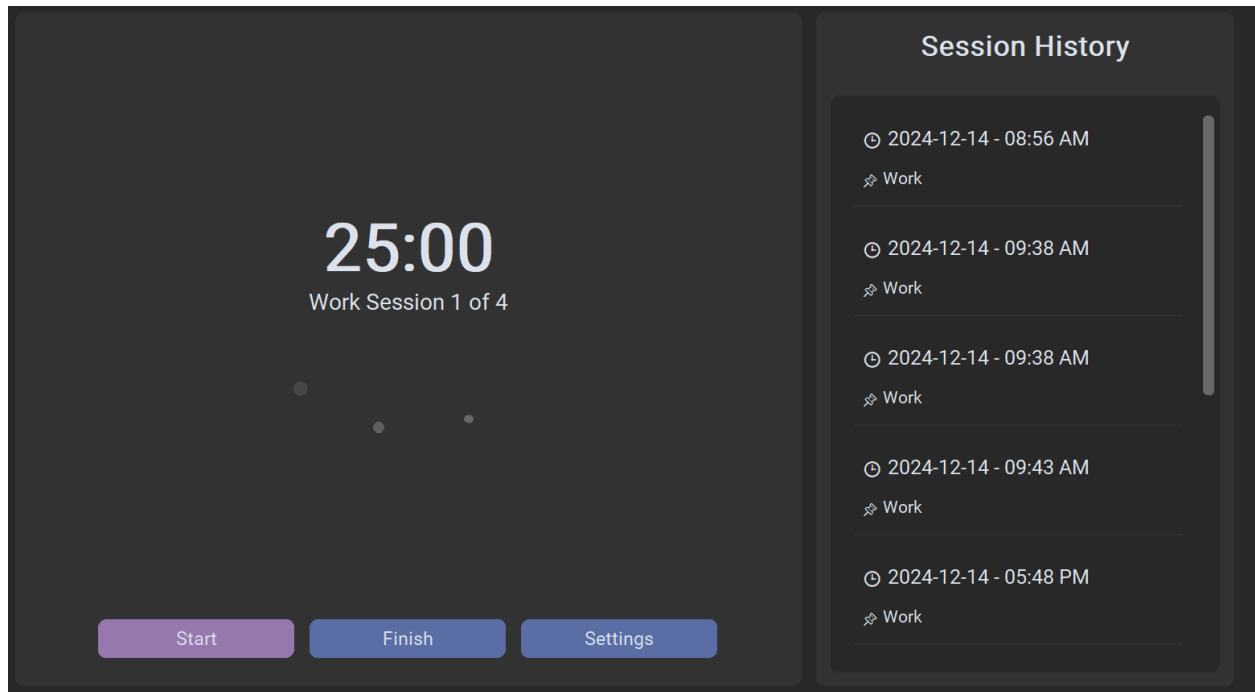
- GPA calculation engine
- Timer management system
- Task organization logic
- Event handling system
- State management

1.4 Main User Interface

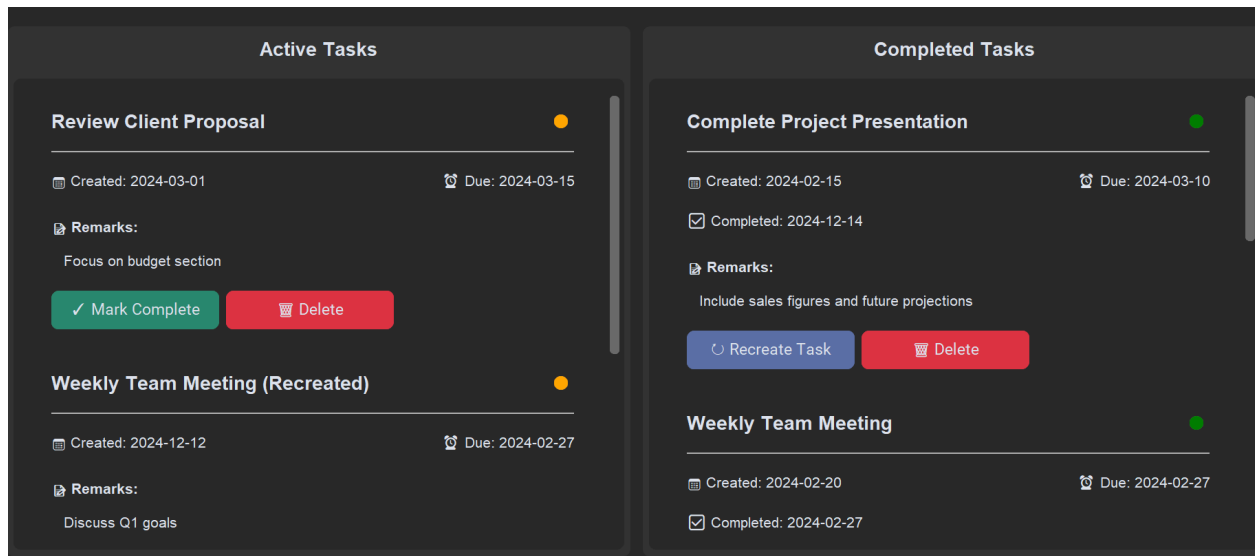
Customizable Theme for Complex GPA Calculator



Immersive Pomodoro-timer



Flexible To-do List Manager



2. GPA Calculator

2.1 System Description

The **GPA Calculator** module is a sophisticated academic performance tracking and analysis system designed to provide students with comprehensive grade management capabilities. This advanced system combines precise mathematical calculations with interactive data visualization to deliver a powerful tool for academic progress monitoring.

1. GPA Calculator Key Features

- ❖ **Dynamic Grade Point System**
 - Comprehensive grade mapping (A+ through F) with corresponding GPA values
 - Standard 4.0 scale implementation with precise decimal values
 - Support for various grade notations including plus/minus grades
 - Visual representation of grade point values
- ❖ **Subject Management**
 - Add multiple subjects with individual credit weightings
 - Dedicated delete functionality for each subject
 - Real-time subject list updates
 - Duplicate subject name detection and prevention
 - Subject name character limit enforcement
- ❖ **Credit Hour Management**
 - Automatic credit hour validation
 - Running total credit calculation
 - Credit distribution visualization
 - Invalid input prevention
- ❖ **Real-time Visualization**
 - Interactive pie chart showing credit distribution
 - Dynamic color coding for different subjects
 - Hover effects with detailed percentage information
 - Animated transitions for data updates
 - Synchronized legend with percentage breakdowns
- ❖ **Data Persistence**
 - Automatic saving of subject data to JSON file
 - Data recovery on application startup
 - Backup creation for data safety
 - Error handling during file operations
 - Data integrity verification
- ❖ **Interactive User Interface**

- Scrollable subject list for unlimited entries
- Color-coded action buttons
- Real-time GPA updates
- Visual feedback for user actions
- Responsive layout adaptation
- ❖ Error Handling
 - Input validation for all fields
 - Clear error messages for invalid inputs
 - Grade format verification
 - Credit hour range checking
 - Duplicate entry prevention
- ❖ Calculation Features
 - Automatic GPA recalculation on any change
 - Weighted average computation
 - Running GPA updates
 - Individual subject GPA tracking
 - Overall GPA precision control
- ❖ Visual Feedback System
 - Status indicators for each subject
 - Clear visual hierarchy
 - Action confirmation messages
 - Error state visualization
 - Success state indication
- ❖ Data Export Options
 - JSON format data storage
 - Subject data persistence
 - GPA calculation history
 - Credit distribution data
 - System state preservation

Core Calculation Engine

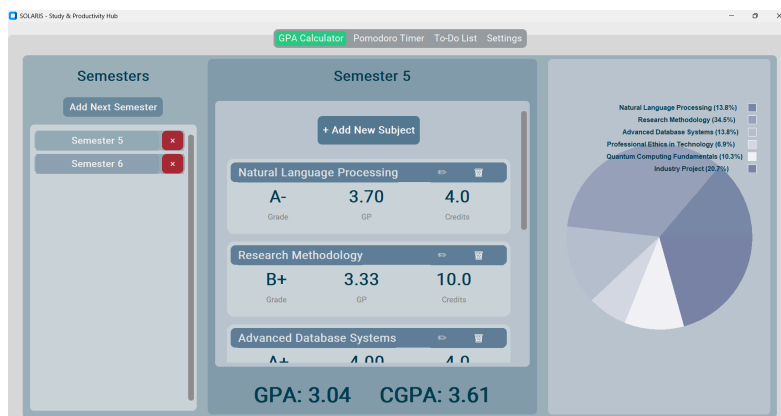
The calculator employs a weighted grade point average system based on the standard 4.0 scale academic grading system. The computation engine processes multiple variables simultaneously, including:

- Individual course grades (A+ through F)
- Running cumulative calculations
- Weighted average computations
- Real-time updates and recalculations

The grade point values are meticulously mapped according to standard academic scales:

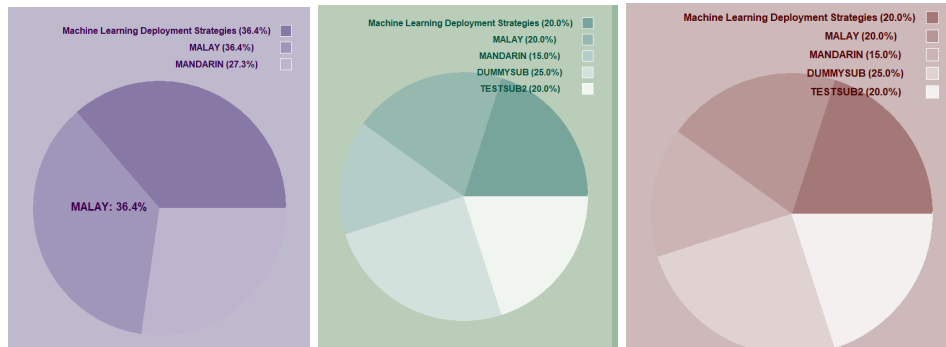
- A+/A (4.0): Representing exceptional performance and complete mastery of course material
- A- (3.75): Indicating excellent achievement with minor areas for improvement
- B+ (3.33): Demonstrating very good achievement with strong understanding
- B (3.0): Showing good achievement and solid comprehension
- C (2.0): Indicating satisfactory achievement and basic understanding
- D (1.0): Representing minimal passing achievement
- F (0.0): Indicating failure to meet minimum requirements

Data Visualization Functionality



The module incorporates an advanced data visualization system featuring a dynamic, interactive pie chart that provides real-time visual representation of credit hour distribution. This sophisticated visualization includes:

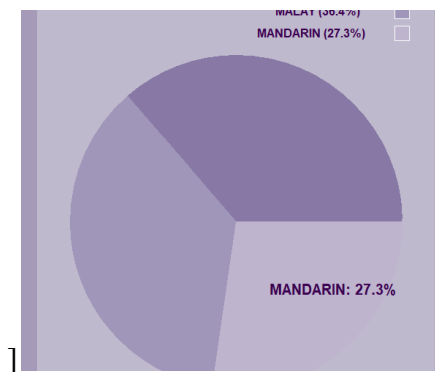
Chart Components



- Dynamically sized segments based on credit hour weightings
- Custom color palette implementation for distinct subject identification
- Smooth animations for all transitions and updates
- Interactive hover effects with precise percentage displays
- Synchronized legend with automatic updates
- Responsive layout adapting to window size

Interactive Features

- Tooltip system showing detailed breakdowns



- Hover state management with smooth transitions
- Dynamic updating with data changes
- **Custom easing arc animation functions for smooth experience**

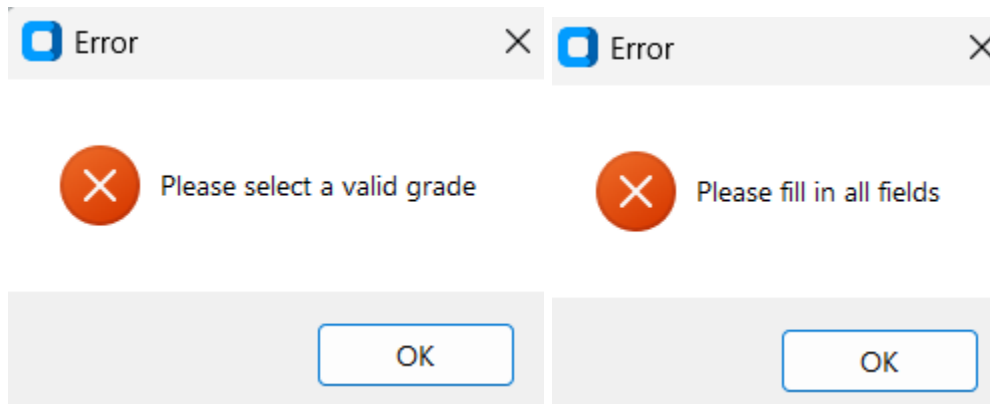
Data Management System

The calculator implements a robust file handling system ensuring data integrity and persistence:

Storage Architecture

- JSON-based data structure for efficient storage
- Real-time synchronization with user inputs
- Automated backup system
- Data validation at multiple levels
- Error recovery mechanisms
- State persistence handling

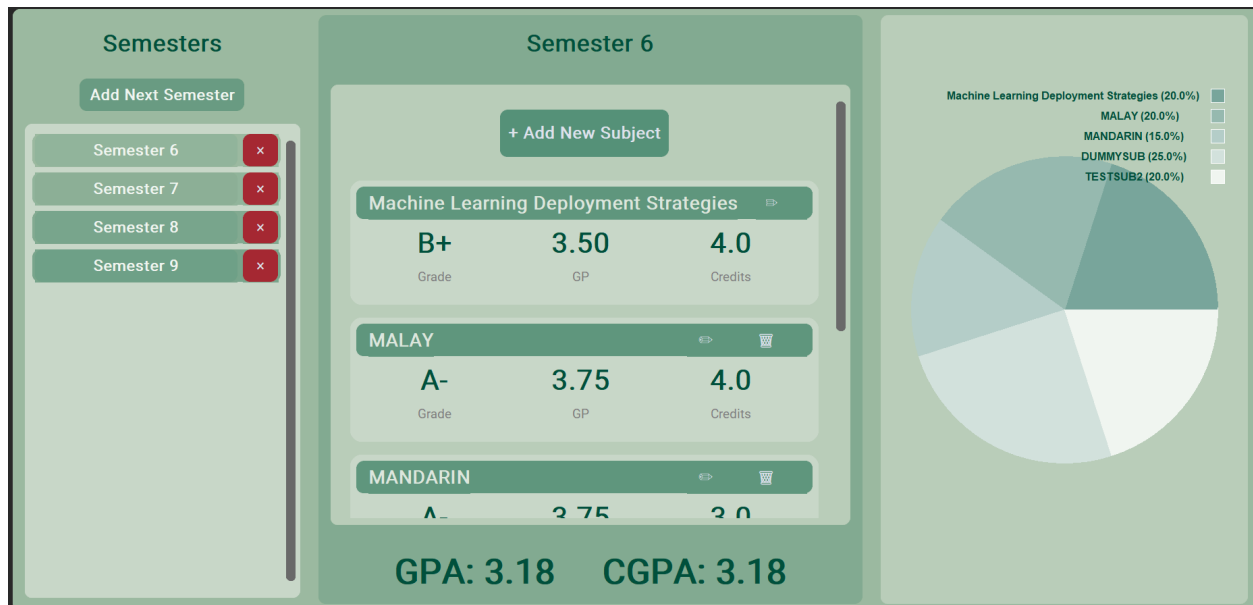
Data Validation



- Input sanitization for all user entries
- Grade format verification
- Credit hour range validation
- Duplicate entry detection
- Error state management
- Recovery procedures

User Interface Components

The interface design prioritizes user experience through thoughtful component organization:

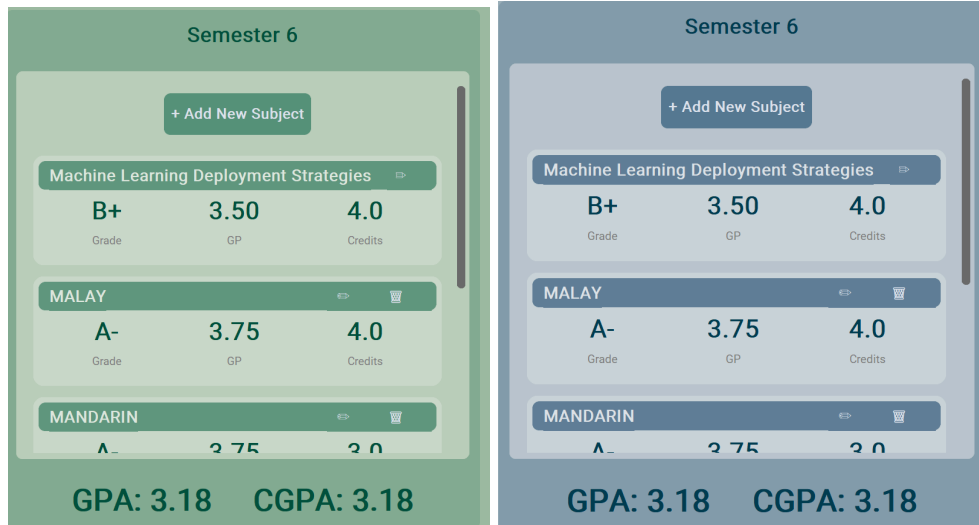


Subject Input System

The 'Add New Subject' modal form is displayed over the 'Semester 6' interface. It contains three input fields: 'Subject:' (a text input), 'Grade:' (a dropdown menu with 'Select Grade' as the selected option), and 'Credits:' (a text input). At the bottom of the modal are two buttons: 'Cancel' (red) and 'Add Subject' (green). The background interface shows the same subject table and GPA/CGPA summary as the previous screenshot.

- Validated subject name entry
- Dropdown grade selection
- Numeric credit hour input
- Add/Delete functionality
- Bulk operation handling
- Error feedback system

Display Components



- Scrollable subject list
- Individual subject cards
- Grade point displays
- Credit hour indicators

Edit Function

Semester 6

A- Grade	3.75 GP	4.0 Credits
MANDARIN ⌵ 🗑		
A- Grade	3.75 GP	3.0 Credits
<div>Grade: A- ⌵</div> <div>Credits: <input type="text" value="3.0"/></div> <div>Save Cancel</div>		
DUMMYSUB ⌵ 🗑		

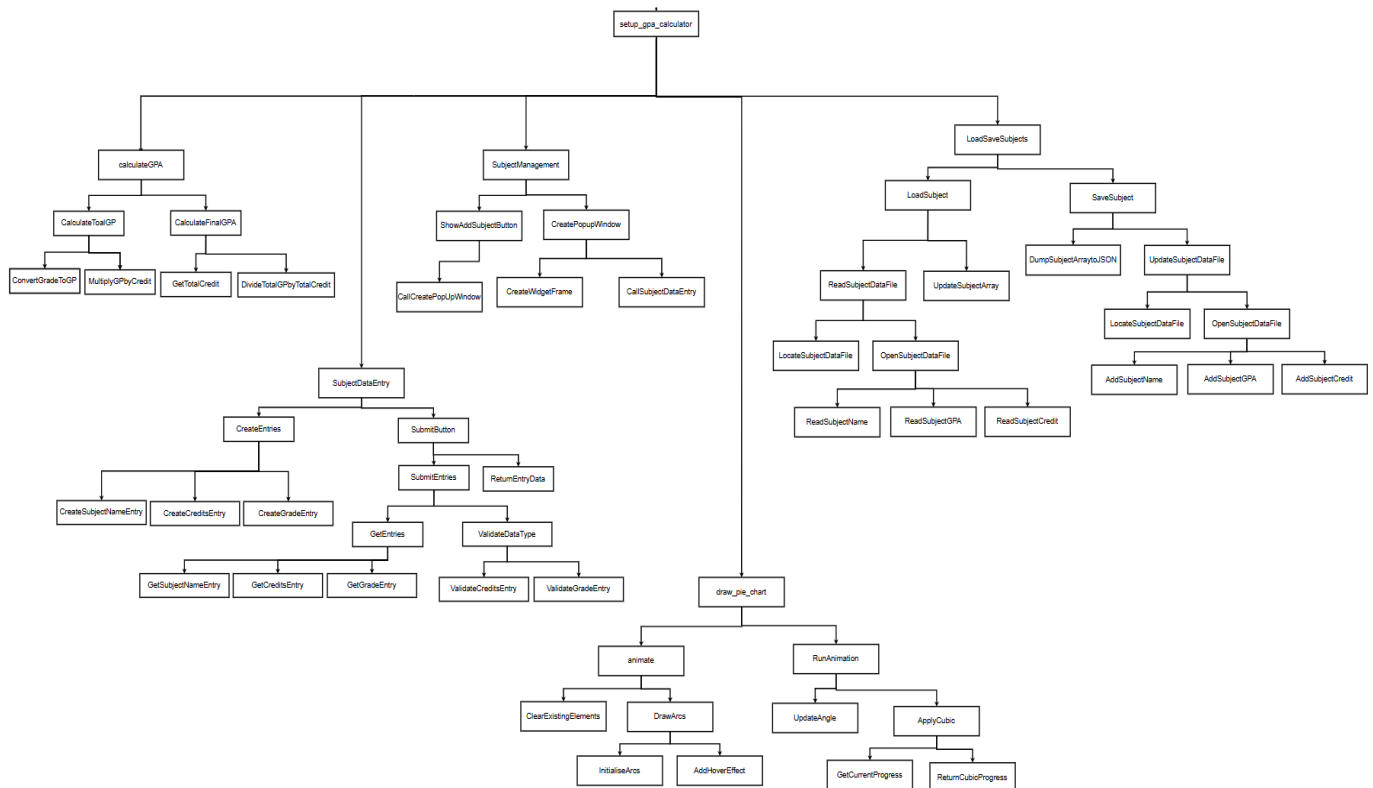
GPA: 3.18 CGPA: 3.18

- on-block subject data editing functionality
- Allows quick editing on typo / mis-inserted data
- Automatically updates system data file after editing

2.2 Stepwise Refinement

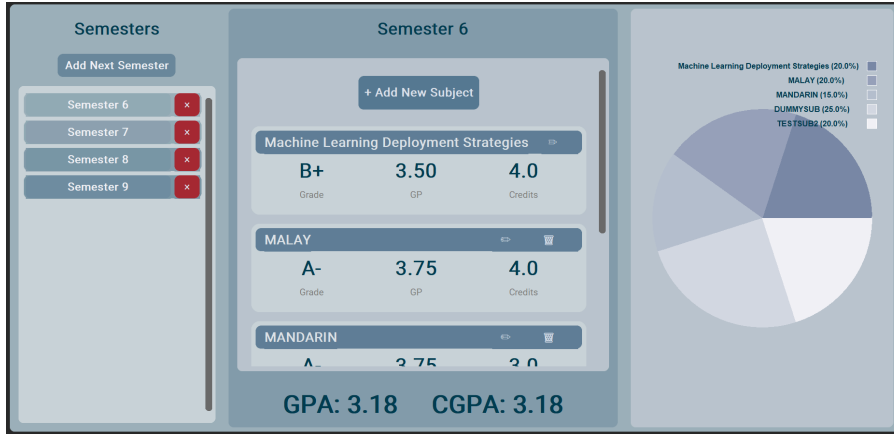
The GPA Calculator module in SOLARIS employs a hierarchical architecture divided into four primary components: GPA Calculation Engine, Subject Management System, Data Entry Interface, and Data Persistence Layer, with an additional Visualization System for data representation. The Calculation Engine handles all GPA computations through a series of nested functions, starting from grade conversion to final GPA calculation. Subject Management oversees the creation and handling of subject entries through a modal interface system. The Data Entry Interface manages user input through a systematic validation process, ensuring data integrity at every step. The Data Persistence Layer handles all file operations, maintaining data continuity through JSON-based storage and retrieval mechanisms. The Visualization System creates and manages an interactive pie chart showing credit distribution with animated transitions and hover effects. These components are tightly integrated through a comprehensive data flow system, where user inputs are validated, processed, stored, and visualized in real-time, with robust error handling at each level ensuring system reliability. This modular structure allows for efficient data management while providing a seamless user experience through immediate feedback and interactive visualizations.

Visualisation of stepwise refinement in the form of structure chart as follows;



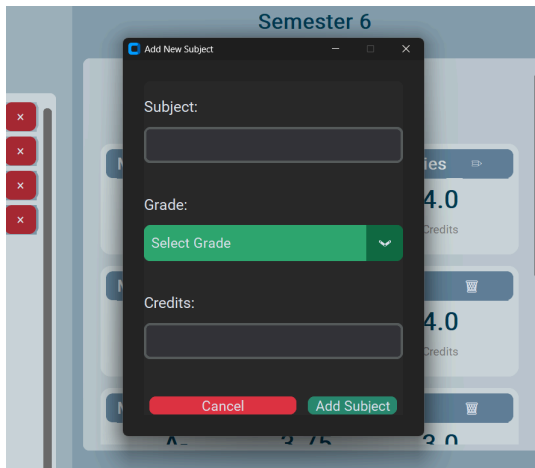
2.3 User Manual

Getting Started



1. Launch the SOLARIS application
2. Navigate to the "GPA Calculator" tab

Adding Subjects



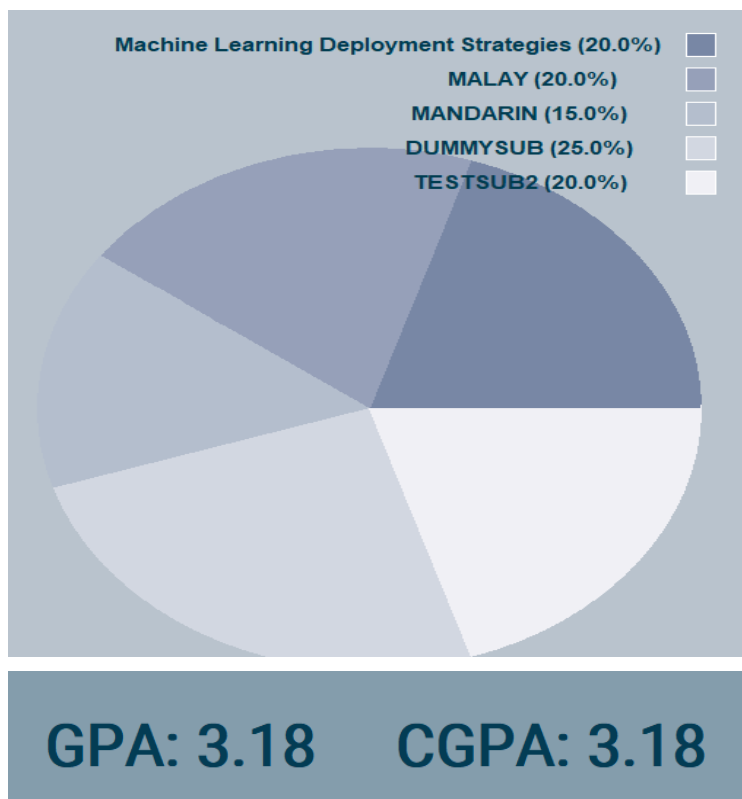
1. Navigate to the desired semester or "Add new semester"
2. Fill in all required fields
3. Click "Add Subject"
4. Verify the subject appears in the list
5. Check the updated GPA calculation

Deleting Subjects



1. Locate the subject in the list
2. Click the "Delete" button
3. Confirm deletion when prompted
4. Verify GPA recalculation

Viewing Statistics



- GPA is displayed prominently at the bottom
- Credit distribution is shown in the pie chart
- Hover over pie chart segments for detailed percentages

2.4 Test Cases

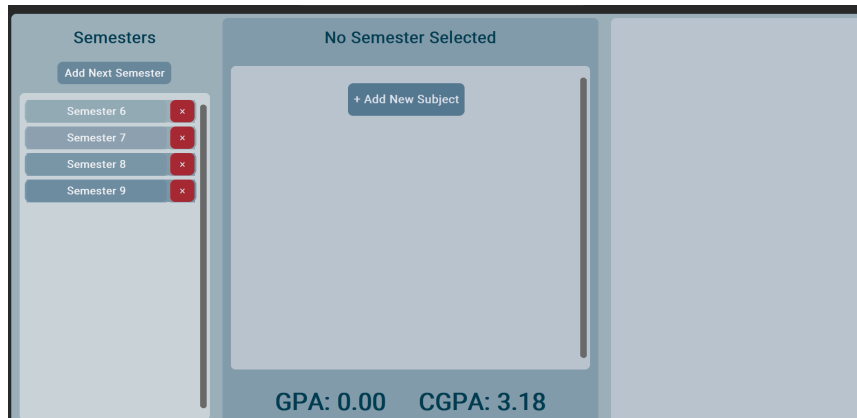
Tabulated Test Case Data:

Functionality	Scenario	Description	Notes	Tested By	Automated	Status
Startup	App loads	Initial load of GPA calculator tab with empty fields and previous data if exists	Should load JSON data if available	Tanwyhang	No	Pass
Add Subject	New subject entry	Enter subject name, select grade A, input 3 credits	Should update pie chart and GPA	Tanwyhang	No	Pass
Invalid Input	Invalid credit hours	Enter non-numeric value in credits field	Should show error message	Tanwyhang	No	Pass
Delete Subject	Remove existing subject	Click delete button on subject row	Should update GPA and pie chart	Tanwyhang	No	Pass
Pie Chart	Visual update	Add multiple subjects to test pie chart segments	Should show proportional distribution and correct colors	Tanwyhang	No	Pass
Data Persistence	Save and reload	Close application and reopen to check data persistence	Should maintain all subject data	Tanwyhang	No	Pass
Empty Fields	Submit empty form	Try to add subject with missing information	Should prevent submission and show error	Tanwyhang	No	Pass
Grade Selection	Grade dropdown	Test all possible grade selections	Should accept all valid grades and calculate correctly	Tanwyhang	No	Pass
Duplicate Subject	Add duplicate subject	Attempt to add subject with existing name	Should update existing subject instead of creating duplicate	Tanwyhang	No	Pass
Multiple Operations	Rapid changes	Perform multiple add/delete operations quickly	Should handle all operations smoothly	Tanwyhang	No	Pass

GPA Calculator Test Case Descriptions

Test Case GPA-1: Application Startup

Purpose: Verify proper initialization of the GPA Calculator module **Test Steps:**

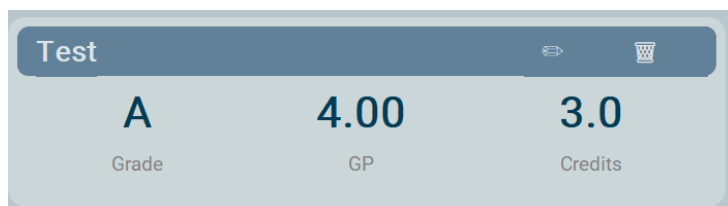


1. Launch SOLARIS application
2. Navigate to GPA Calculator tab
3. Check for empty input fields
4. Verify JSON data loading if available **Expected Outcome:** Application should display empty calculator interface or load previous data if available

Test Case GPA-2: Add New Subject

Purpose: Validate subject addition functionality **Test Steps:**

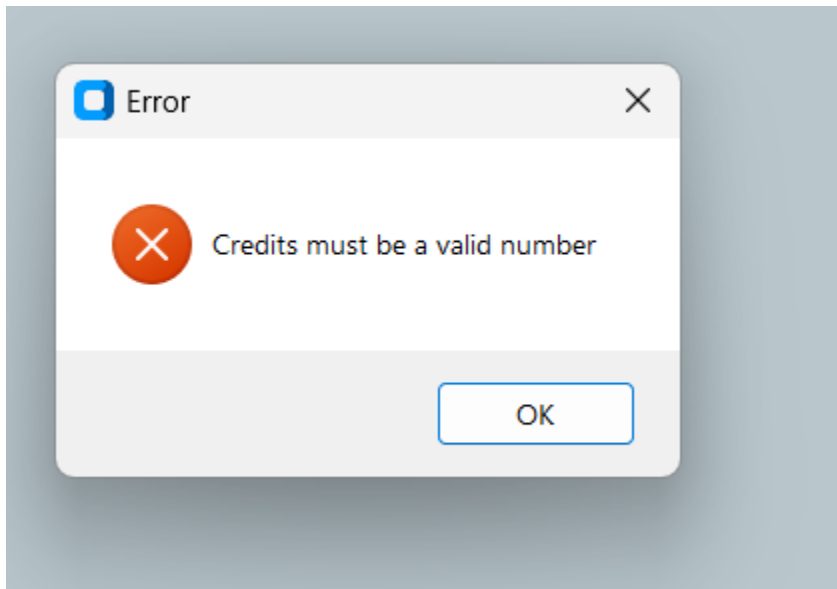
1. Enter subject name in the subject field
2. Select grade 'A' from dropdown menu
3. Input '3' in credits field
4. Click "Add Subject" button **Expected Outcome:**
 - New subject appears in list
 - Pie chart updates with new segment
 - GPA recalculates and displays
 - Subject data saves to JSON



Test Case GPA-3: Invalid Credit Hours

Purpose: Test input validation for credit hours **Test Steps:**

1. Enter valid subject name
2. Select valid grade
3. Enter non-numeric value (e.g., "abc") in credits field
4. Attempt to submit **Expected Outcome:**
 - Error message displays
 - Form submission prevented
 - Invalid input highlighted



Test Case GPA-4: Delete Subject

Purpose: Verify subject deletion functionality **Test Steps:**

1. Locate existing subject in list
2. Click delete button on subject row
3. Confirm deletion in popup **Expected Outcome:**
 - Subject removes from list
 - GPA recalculates
 - Pie chart updates
 - Changes save to JSON

Test		
A	4.00	3.0
Grade	GP	Credits

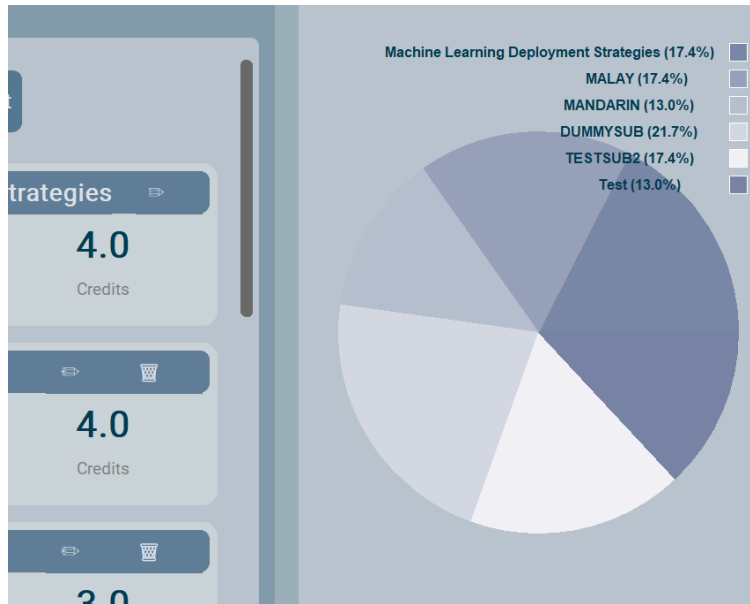
MANDARIN		
A-	3.75	3.0
Grade	GP	Credits
DUMMYSUB		
C	2.00	5.0
Grade	GP	Credits
TESTSUB2		
B+	3.33	4.0
Grade	GP	Credits
GPA: 3.14 CGPA: 3.14		

Context: Test is deleted and GPA is recalculated with the correct value

Test Case GPA-5: Pie Chart Visualization

Purpose: Test visual representation of credit distribution **Test Steps:**

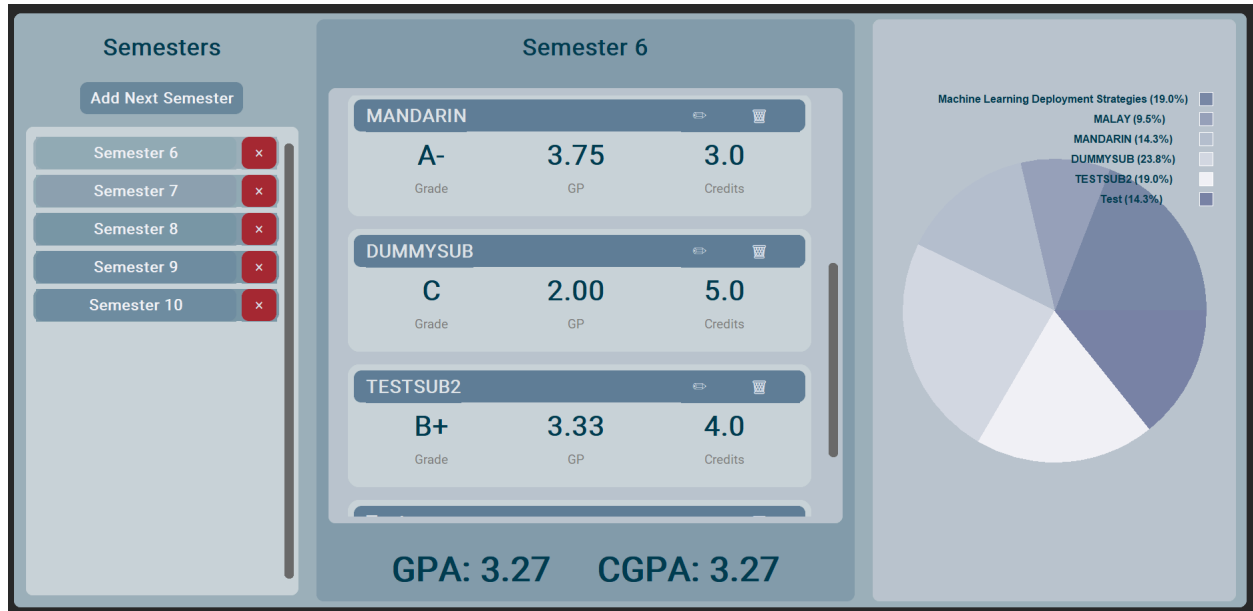
1. Add multiple subjects with different credits
2. Observe pie chart updates
3. Verify segment proportions
4. Check color assignments **Expected Outcome:**
 - Chart shows proportional segments
 - Colors distinct and consistent
 - Hover effects work
 - Segments match credit values



Test Case GPA-6: Data Persistence

Purpose: Verify data saving and loading functionality **Test Steps:**

1. Add several subjects with varying data
2. Close application completely
3. Reopen application
4. Navigate to GPA Calculator **Expected Outcome:**
 - All subjects reload correctly
 - GPA maintains accuracy
 - Pie chart restores properly
 - No data loss occurs



Context: SOLARIS is restarted, data persistence is assured

Test Case GPA-7: Empty Field Validation

Purpose: Test form validation for empty fields **Test Steps:**

1. Leave subject name empty
2. Try submitting with missing grade
3. Attempt submission with empty credits **Expected Outcome:**
 - Error message for each empty field
 - Form submission prevented
 - Clear indication of required fields

Test Case GPA-8: Grade Selection Options

Purpose: Verify grade selection functionality **Test Steps:**

1. Click grade dropdown
2. Select each available grade
3. Verify GPA calculation for each
4. Test grade changes on existing subjects **Expected Outcome:**
 - All grades selectable
 - Correct GPA calculations
 - Proper grade point assignment

- Updates reflect immediately

The image shows a user interface for selecting a grade. On the left, a dropdown menu titled 'Select Grade' is open, listing grades from A+ to F. The 'A+' grade is highlighted. To the right, a 'Grade:' label is followed by a green box containing 'A+'. Below this, a summary bar for a subject named 'MALAY' displays the selected grade as 'A+', the GPA as '4.00', and the credits as '2.0'.

Context: A+ is chosen, GP updates reflect immediately upon selection

Test Case GPA-9: Duplicate Subject Handling

Purpose: Test handling of duplicate subject names **Test Steps:**

1. Add initial subject
2. Attempt to add subject with identical name
3. Verify update behavior **Expected Outcome:**
 - Update existing subject
 - No duplicate entries
 - Clear user feedback
 - Proper data handling

Test Case GPA-10: Performance Testing

Purpose: Verify system performance under rapid operations **Test Steps:**

1. Perform quick succession of adds/deletes
2. Make rapid grade changes
3. Test multiple pie chart updates **Expected Outcome:**

- Smooth operation handling
- No lag or freezing
- Accurate data updates
- Stable visualization

Additional Testing Notes

- All tests performed **manually** by **Tanwyhang**
- Testing environment: Windows 11 platform
- Testing conducted on full screen resolution

3. Pomodoro Timer

3.1 System Description

The Pomodoro Timer module is a comprehensive time management implementation based on the renowned Pomodoro Technique, designed to enhance study efficiency and productivity. The system orchestrates focused work sessions of 25 minutes followed by strategic break periods, incorporating both short 5-minute breaks and extended 30-minute breaks after completing four consecutive work sessions. The module features a sophisticated state management system that handles various timer states including active, paused, and break periods, complemented by a robust notification system to guide users through their study sessions. The interface presents a large-format digital display for clear time tracking, alongside intuitive control buttons for session management. The system maintains accurate session counting and provides visual feedback through status indicators and progress tracking, ensuring users can effectively monitor their study patterns and maintain productive work cycles.

Key Features:

Timer Management

- 25-minute focused work sessions
- 5-minute short breaks
- 30-minute long breaks after 4 sessions
- Pause/Resume capability
- Session reset function

Session Tracking

- Four-session cycle monitoring
- Automated break determination
- Session completion validation
- Progress persistence
- Cycle reset handling

Break System

- Automatic break initiation
- Break type determination
- Break duration management

- Skip break option
- Break completion handling

User Interface

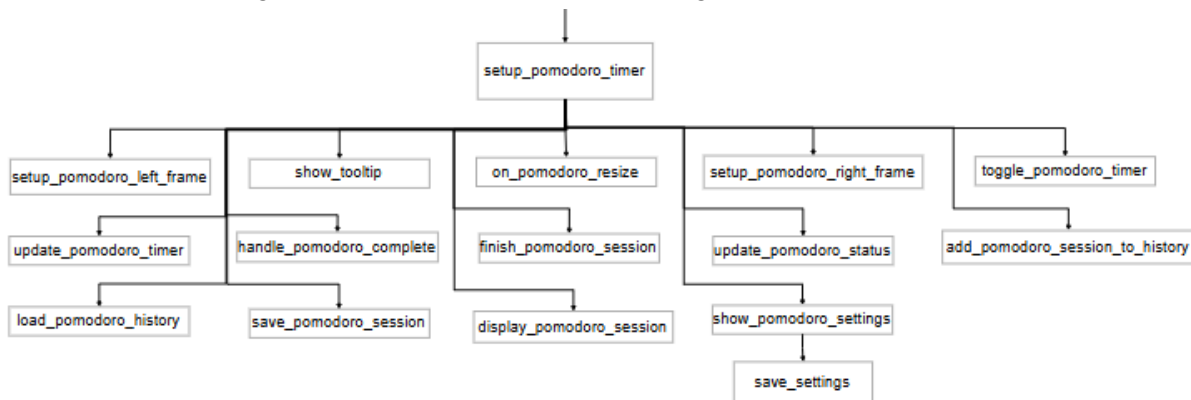
- Large digital countdown display
- Session progress indicators
- Status message display
- Control button panel
- Visual state indicators

State Management

- Active session handling
- Pause state control
- Break state management
- Error recovery
- Session state persistence

3.2 Stepwise Refinement

Structure chart showing process structure of the sub-program:



3.3 User Manual

Getting Started

1. Navigate to the "Pomodoro Timer" tab in SOLARIS

2. The timer will be set to default 25:00 minutes
3. Use the control buttons at the bottom to manage your sessions

Basic Controls

1. Start Button
 - 1.1. Click "Start" to begin a Pomodoro session
 - 1.2. Button changes to "Pause" during active sessions
 - 1.3. Click "Resume" if session was paused
2. Reset Button
 - 2.1. Returns timer to 25:00
 - 2.2. Clears current session progress
 - 2.3. Can be used during any timer state
3. Finish Button
 - 3.1. Immediately ends current session
 - 3.2. Triggers break timer if appropriate
 - 3.3. Requires confirmation to prevent accidental clicks

Session Types

1. Work Session (25 minutes)
 - 1.1. Standard Pomodoro work period
 - 1.2. Status shows "Focus Time!"
 - 1.3. Timer counts down from 25:00
2. Short Break (5 minutes)
 - 2.1. Occurs after each Pomodoro session
 - 2.2. Status shows "Short Break - Take 5!"
 - 2.3. Timer counts down from 05:00
3. Long Break (30 minutes)
 - 3.1. Triggers after completing 4 Pomodoros
 - 3.2. Status shows "Long Break - Time to recharge!"
 - 3.3. Timer counts down from 30:00

Progress Tracking

- Session counter shows "Pomodoros Completed: X/4"
- Status message updates automatically
- Visual indicators show current timer state
- Break notifications appear when sessions complete

Tips for Use

1. Complete full 25-minute sessions when possible
2. Take all scheduled breaks

3. Use reset only when necessary
4. Watch status messages for guidance
5. Complete full 4-session cycles for best results

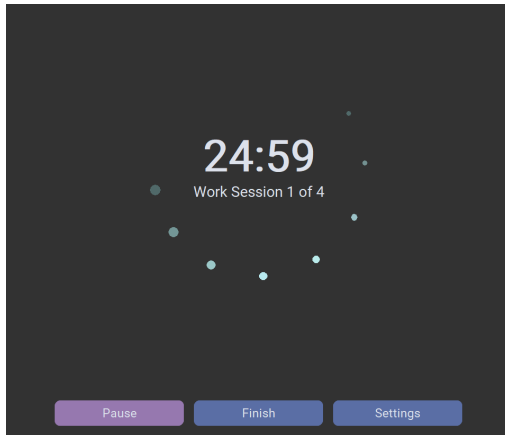
3.4 Test Cases

Functionality	Scenario	Description	Notes	Tested By	Automated	Status
Timer Start	Initial timer start	Click start button with default 25:00	Should begin countdown	Tanwyhang	No	Pass
Break Timer	Session completion	Timer reaches 00:00	Should trigger break timer	Tanwyhang	No	Pass
Pause Timer	Pause during session	Click pause during countdown	Should freeze timer	Tanwyhang	No	Pass
Session Counter	Complete 4 sessions	Finish 4 full pomodoro cycles	Should trigger long break	Tanwyhang	No	Pass
Reset Function	Mid-session reset	Click reset button during active session	Should reset to 25:00 and clear current progress	Tanwyhang	No	Pass
Break Skip	Skip break timer	Use finish button during break	Should end break and prepare next session	Tanwyhang	No	Pass
Status Display	Status updates	Check status message changes during different states	Should show appropriate status messages	Tanwyhang	No	Pass
Multiple Sessions	Long usage session	Complete multiple pomodoro cycles	Should maintain accurate count and timing	Tanwyhang	No	Pass
Button States	Button availability	Check button states during different timer phases	Should enable/disable appropriate buttons	Tanwyhang	No	Pass
Timer Accuracy	Long duration test	Run timer for extended period	Should maintain accurate timing	Tanwyhang	No	Pass

Test Case POM-1: Timer Start

Purpose: Verify timer initialization and start functionality **Test Steps:**

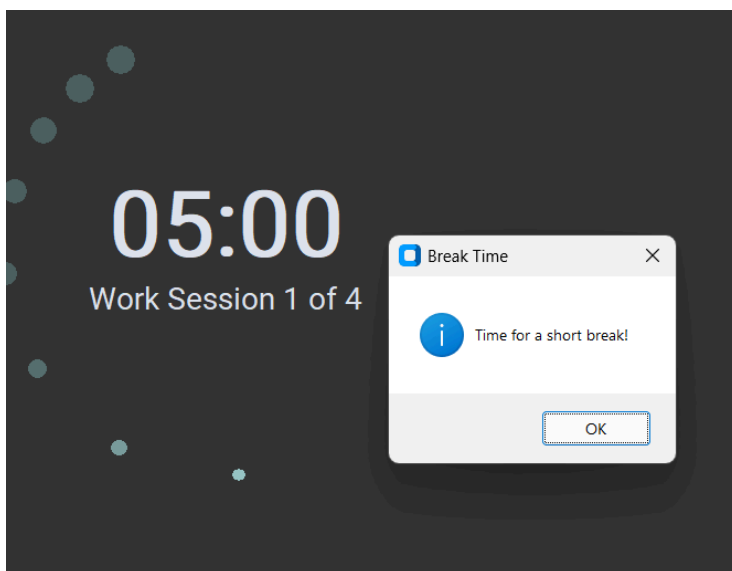
1. Navigate to Pomodoro tab
2. Check initial 25:00 display
3. Click start button
4. Observe countdown **Expected Outcome:** Timer should start from 25:00 and count down correctly



Test Case POM-2: Break Timer

Purpose: Verify break timer activation after session **Test Steps:**

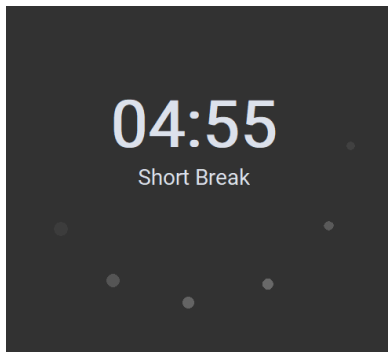
1. Complete one pomodoro session
2. Wait for timer to reach 00:00
3. Observe break timer activation **Expected Outcome:**
 - 5-minute break timer starts automatically
 - Status changes to break mode
 - Break notification appears



Test Case POM-3: Pause Timer

Purpose: Test pause functionality during session **Test Steps:**

1. Start timer session
2. Click pause during countdown
3. Wait 30 seconds
4. Check timer value **Expected Outcome:**
 - Timer stops at clicked time
 - Start button changes to Resume
 - Time remains frozen

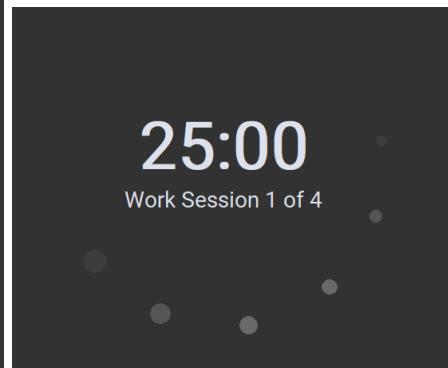
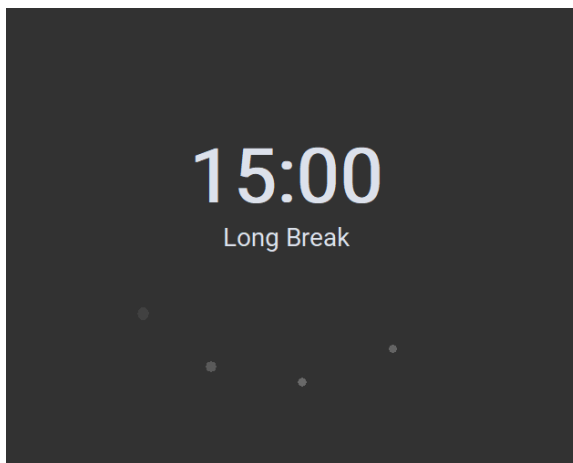


Timer remains at same state after 30 seconds

Test Case POM-4: Session Counter

Purpose: Verify four-session cycle tracking **Test Steps:**

1. Complete four full pomodoro sessions
2. Monitor session counter
3. Check long break activation **Expected Outcome:**
 - Counter shows correct progress
 - Long break (30 min) activates after fourth session
 - Counter resets after cycle



context: Counter Resets after long break

Test Case POM-5: Reset Function

Purpose: Test timer reset functionality **Test Steps:**

1. Start a timer session
2. Let run for several minutes
3. Click reset button
4. Check timer and counter state **Expected Outcome:**
 - Timer returns to 25:00
 - Session progress clears
 - Counter maintains previous value

Test Case POM-6: Break Skip

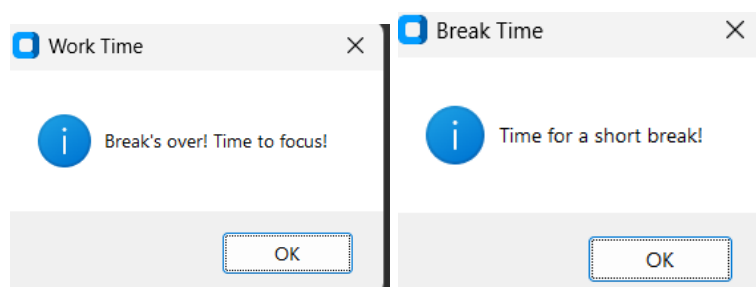
Purpose: Verify break skip functionality **Test Steps:**

1. Complete one pomodoro session
2. Enter break mode
3. Click finish button
4. Confirm skip **Expected Outcome:**
 - Break ends immediately
 - New session ready to start
 - Counter updates correctly

Test Case POM-7: Status Display

Purpose: Test status message updates **Test Steps:**

1. Monitor status through full cycle
2. Check different state messages
3. Verify message accuracy **Expected Outcome:**
 - Correct status for work sessions
 - Accurate break status messages
 - Clear state indicators



Test Case POM-8: Multiple Sessions

Purpose: Test extended usage reliability **Test Steps:**

1. Complete multiple full cycles
2. Monitor all transitions
3. Check counter accuracy **Expected Outcome:**
 - Consistent timing
 - Accurate session tracking
 - Proper break scheduling

Session History
🕒 2024-12-14 - 08:56 AM ⚙️ Work
🕒 2024-12-14 - 09:38 AM ⚙️ Work
🕒 2024-12-14 - 09:38 AM ⚙️ Work
🕒 2024-12-14 - 09:43 AM ⚙️ Work
🕒 2024-12-14 - 05:48 PM ⚙️ Work

Test Case POM-9: Button States

Purpose: Verify button behavior in different states **Test Steps:**

1. Check button states during session
2. Verify during breaks
3. Test during transitions **Expected Outcome:**
 - Correct button availability
 - Proper state changes
 - Clear visual feedback

Test Case POM-10: Timer Accuracy

Purpose: Verify timing precision **Test Steps:**

1. Run complete session
2. Compare with external timer
3. Check multiple intervals **Expected Outcome:**
 - Accurate countdown
 - Precise interval timing
 - Consistent performance

4. To-do List Manager

4.1 System Description

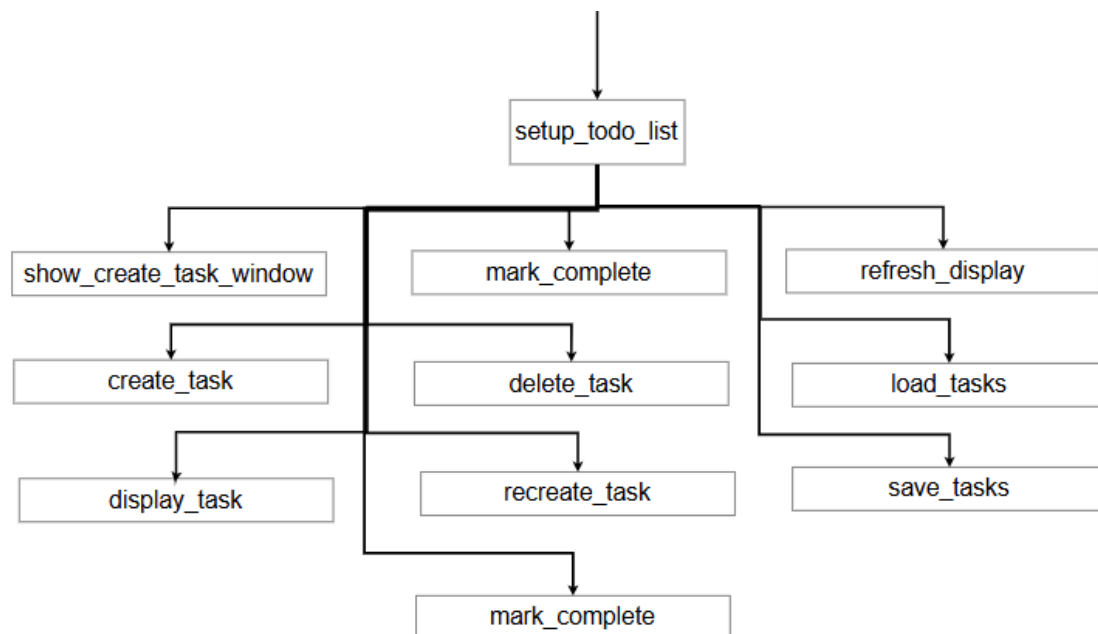
The To-Do List Manager represents an advanced task organization system designed to help students effectively manage and track their academic responsibilities. The module implements a dual-panel interface that distinctly separates active and completed tasks, providing clear visual organization of pending and finished work. It features a comprehensive task creation system with calendar integration for deadline management, detailed remarks capability, and robust data persistence using JSON storage. The system incorporates sophisticated state management for task transitions, allowing users to track task completion, recreate completed tasks, and maintain a clear overview of their academic workload. Visual feedback mechanisms and interactive elements enhance the user experience, while the underlying data structure ensures reliable task persistence and retrieval.

Core Functionality

1. Task Management
 - Task creation interface
 - Deadline setting with calendar
 - Task status tracking
 - Task recreation capability
 - Bulk task operations
2. Data Organization
 - Active tasks panel
 - Completed tasks panel
 - Task categorization
 - Priority management
 - Visual status indicators
3. Data Persistence
 - JSON-based storage
 - Automatic data saving
 - State preservation
 - Data recovery
 - Backup management
4. Interface Components
 - Task creation modal
 - Calendar integration

- Interactive task cards
 - Status notifications
 - Action buttons
5. Task Operations
- Mark as complete
 - Task deletion
 - Task recreation
 - Deadline modification
 - Remarks editing

4.2 Stepwise Refinement



4.3 User Manual

Getting Started

1. Navigate to the "To-Do List" tab
2. Click "Create New Task" to add tasks
3. View tasks in Active and Completed sections

Creating Tasks

1. Click "Create New Task" button
2. Fill in required information:
 - Task name (required)
 - Deadline using calendar
 - Additional remarks (optional)
3. Click "Create Task" to save

Managing Tasks

1. Active Tasks
 - View all pending tasks
 - Mark tasks complete using checkmark
 - Delete tasks using trash icon
 - All changes save automatically
2. Completed Tasks
 - View finished tasks
 - Recreate tasks if needed
 - Delete completed tasks
 - Track completion dates

Calendar Usage

1. Select date from calendar popup
2. Dates save with task creation

Task Operations

1. Marking Complete
 - Click checkmark on active task
 - Task moves to completed section
 - Completion date is recorded
 - Status updates automatically
2. Task Recreation
 - Find task in completed section

- Click recreate button
 - New task appears in active section
 - Original task remains in completed
3. Deletion
- Click trash icon on any task
 - Confirm deletion in popup
 - Action cannot be undone
 - Removes task permanently

Tips for Use

1. Always set realistic deadlines
2. Use remarks for important details
3. Regularly clean up completed tasks
4. Check tasks daily
5. Update task status promptly

Data Management

- All changes save automatically
- Data persists between sessions
- No manual saving required
- Regular backups recommended

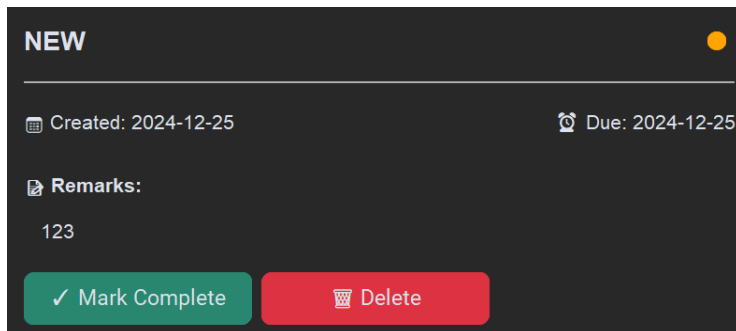
4.4 Test Case

ID	Functionality	Scenario	Description	Notes	Tested By	Automated	Status
TODO-1	Task Creation	New task entry	Create task with name, deadline, and remarks	Should appear in active tasks	Tanwyhang	No	Pass
TODO-2	Task Completion	Mark task complete	Click complete button on active task	Should move to completed section	Tanwyhang	No	Pass
TODO-3	Invalid Date	Past deadline	Set task deadline to past date	Should show warning	Tanwyhang	No	Pass
TODO-4	Task Recreation	Recreate completed task	Click recreate on completed task	Should create new active task	Tanwyhang	No	Pass
TODO-5	Data Persistence	Save and reload	Close and reopen application	Should maintain all task data	Tanwyhang	No	Pass
TODO-6	Empty Fields	Submit empty form	Try to create task with missing information	Should prevent creation and show error	Tanwyhang	No	Pass
TODO-7	Calendar Interface	Date selection	Test calendar date picker functionality	Should allow valid date selection	Tanwyhang	No	Pass
TODO-8	Multiple Tasks	Bulk creation	Create multiple tasks in succession	Should handle multiple creations smoothly	Tanwyhang	No	Pass
TODO-9	Task Deletion	Delete tasks	Remove tasks from both active and completed sections	Should remove tasks properly	Tanwyhang	No	Pass
TODO-10	Long Remarks	Extended text	Add task with very long remarks text	Should handle text wrapping properly	Tanwyhang	No	Pass

Test Case TODO-1: Task Creation

Purpose: Verify task creation functionality **Test Steps:**

1. Click "Create New Task"
2. Fill all fields
3. Submit form **Expected Outcome:**
 - Task appears in active list
 - All data displays correctly
 - Form clears after submission

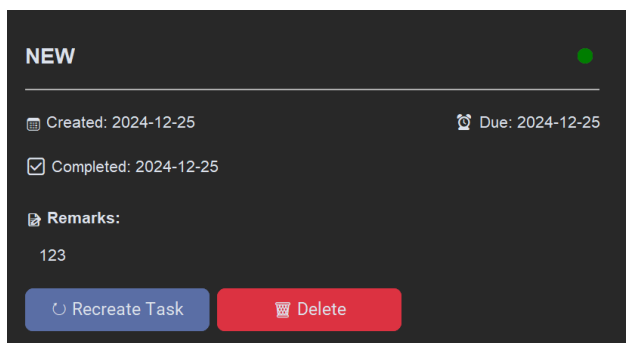


A screenshot of a task creation form titled "NEW" with a yellow dot in the top right corner. The form has a dark background. It displays "Created: 2024-12-25" and "Due: 2024-12-25". Below this is a "Remarks:" section with the text "123". At the bottom, there are two buttons: a green "✓ Mark Complete" button and a red "🗑 Delete" button.

Test Case TODO-2: Task Completion

Purpose: Test task completion process **Test Steps:**

1. Find active task
2. Click complete button
3. Check completed section **Expected Outcome:**
 - Task moves to completed section
 - Completion date added
 - Status updates correctly



A screenshot of a task completion form titled "NEW" with a green dot in the top right corner. The form has a dark background. It displays "Created: 2024-12-25" and "Due: 2024-12-25". Below this is a "Completed: 2024-12-25" section with a checkmark icon. Below that is a "Remarks:" section with the text "123". At the bottom, there are two buttons: a blue "🔄 Recreate Task" button and a red "🗑 Delete" button.

Test Case TODO-3: Invalid Date

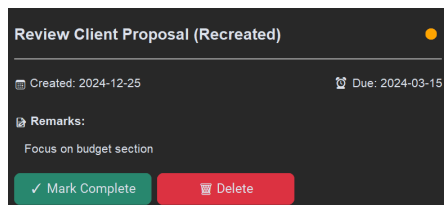
Purpose: Verify date validation **Test Steps:**

1. Create new task
2. Select past date
3. Attempt to save **Expected Outcome:**
 - Warning message appears
 - User notified of invalid date
 - Prevention of invalid entry

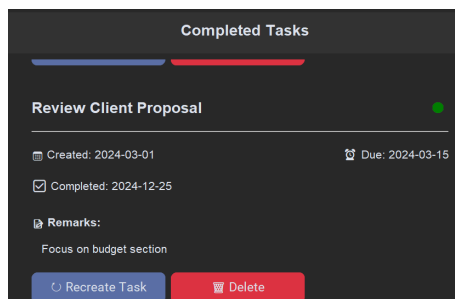
Test Case TODO-4: Task Recreation

Purpose: Test task recreation function **Test Steps:**

1. Find completed task
2. Click recreate button
3. Check active tasks **Expected Outcome:**
 - New task in active section
 - Original remains in completed
 - Correct data transfer



Recreated from:

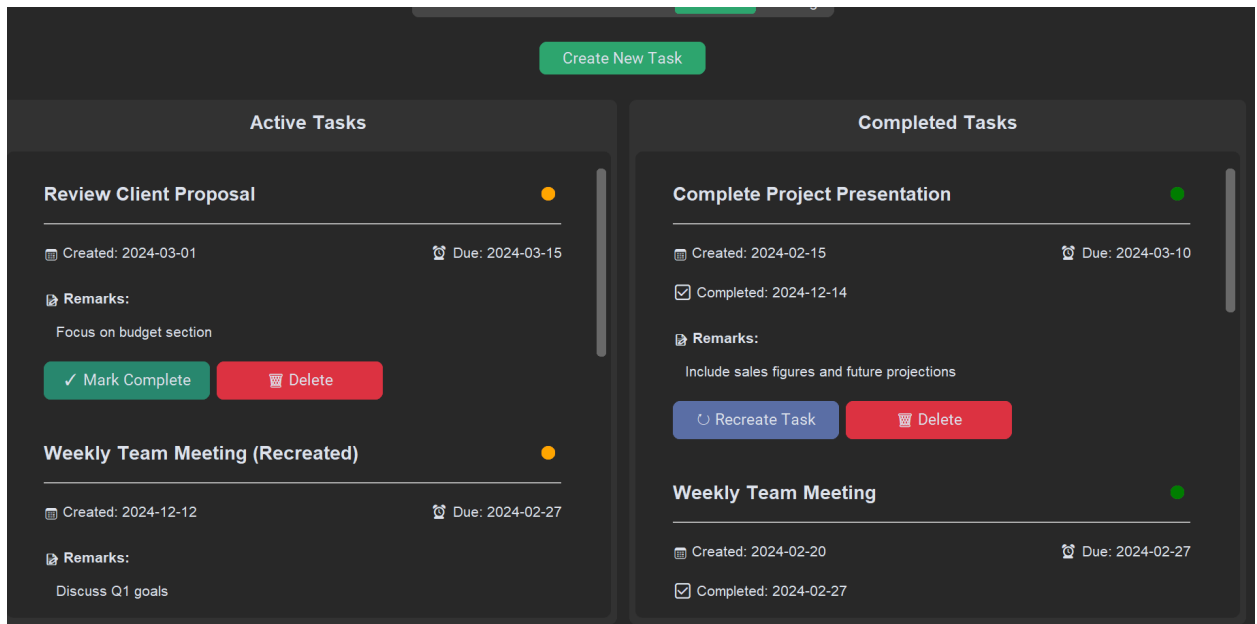


Test Case TODO-5: Data Persistence

Purpose: Verify data saving **Test Steps:**

1. Create multiple tasks
2. Close application
3. Reopen and check data **Expected Outcome:**
 - All tasks present
 - Data integrity maintained
 - Correct section placement

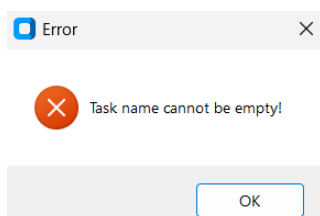
App is reopened:



Test Case TODO-6: Empty Fields

Purpose: Test empty field validation **Test Steps:**

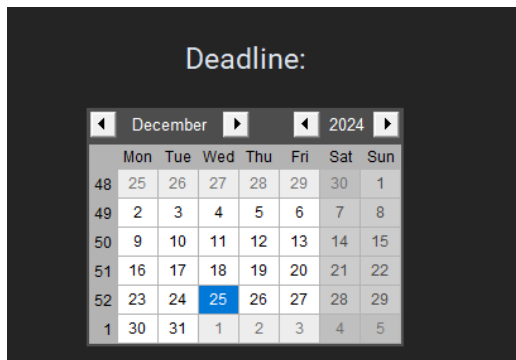
1. Open new task form
2. Leave fields empty
3. Try to submit **Expected Outcome:**
 - Error messages show
 - Submission prevented
 - Required fields indicated



Test Case TODO-7: Calendar Interface

Purpose: Verify calendar functionality **Test Steps:**

1. Open date picker
2. Test date selection
3. Verify date format **Expected Outcome:**
 - Calendar opens properly
 - Date selects correctly
 - Format consistent



Test Case TODO-8: Multiple Tasks

Purpose: Test bulk task handling **Test Steps:**

1. Create several tasks quickly
2. Check all entries
3. Verify order **Expected Outcome:**
 - All tasks save properly
 - Correct ordering
 - Smooth operation

Test Case TODO-9: Task Deletion

Purpose: Verify deletion functionality **Test Steps:**

1. Select task to delete
2. Click delete button
3. Confirm deletion **Expected Outcome:**
 - Task removes from list
 - Confirmation prompt shows
 - No data artifacts

Test Case TODO-10: Long Remarks

Purpose: Test text handling **Test Steps:**

1. Create task with long remarks
2. Check display
3. Verify scrolling **Expected Outcome:**
 - Text wraps properly
 - Scrolling works
 - No display issues

Test

Created: 2024-12-25

Due: 2024-12-25

Remarks:

baskdbkjsbdkjasbdkjasbdkjabdkjasbdkjabdsjkbajksbdkj
asbdjkasbdjkabsdjkbdkasbdjkbaskdbakjsdbjkasbdkjasbdjkabdkjas
bdjkbaskjdbkajsdjbkjasbdkasbdjkadbkjabdjkasbdjkabdkjabdsdkjbkjdb
ajksdbkjasbdkjasbdkjabdkjaskbdjasbdkjasd

✓ Mark Complete

🗑 Delete