RPM.md (Release Planning Meeting)

Release Planning Meeting - Moody (To-Do List App)

Release Goal:
Moody aims to help users manage their events, tasks, and productivity in a user-friendly interface. It will allow users to sign up, create and view events, manage tasks, and receive notifications, ultimately improving their organization and time management skills.

Scope of the Project:

1. User Sign-Up

Description: Allow users to "sign up" by entering their name and email, storing this information locally (e.g., in a text file).

Benefits: Provides a personalized experience without complex back-end systems.

2. Basic Event Creation

Description: Users can add events to specific days in their calendar, including event names and simple details like date and time.

Benefits: Easy to implement and helps users start organizing their schedules.

3. Event Viewing (List View)

Description: Display events in a simple list for each day, without complex calendar views.

Benefits: Simplifies the process of viewing events, making it easy to implement without complex UI.

4. Mark Events as Done

Description: Add a checkbox next to each event so users can mark them as done or completed.

Benefits: Gives users a sense of accomplishment without much coding complexity.

5. Basic Task List

Description: Users can create a list of tasks (to-do list) with due dates.

Benefits: Expands the app's functionality while keeping it easy to code.

6. Simple Date Navigation

Description: Users can navigate between different days by clicking "Previous" and "Next" buttons.

Benefits: Allows basic calendar functionality without needing to implement full calendar views.

Participants:

- Product Owner: Abdihakim
- Team Members:
    - Najwa Waqar
    - Abdihakim
    - Hamza
- Scrum Demo TA: Claudia

Agenda

1. Introduction and Objective of Meeting
    - Objective: Define goals and set the scope for the release of the To-Do List App.
2. Review of Project Goals
    - Goal: Improve user organization and productivity through features like event management, task management, and notifications.
3. Scope Definition and Key Features
    - Scope: Identify essential features for this release:
        - User Sign-Up
        - Event Creation
        - Event Viewing (List View)
    - outline each feature briefly, explaining the functionality, benefit, and initial approach for each.
4. Identify User Stories for This Release
    - User Stories: Break down each feature into specific user stories, which should be achievable within the release period.
    - Example user stories:
        - "As a user, I want to sign up with my name and email to have a personalized app experience."
        - "As a user, I want to create an event with a name, date, and time, so I can keep track of my schedule."
    - Compile and clarify each user story, ensuring they’re well-defined, and will facilitate any discussions about breaking down stories further if needed.
5. Define Timeline and Milestones
    - Set deadlines and key milestones for each feature or story.
    - Decide on testing phases and any interim reviews.
6. Identify Dependencies and Risks
    - Discuss any dependencies between features and potential risks that could impact the timeline.
    - Plan for mitigating known risks.
    - identify dependencies, document any risks discussed.
7. Tools and Resources
    - Confirm tools (Trello, GitHub) and document structure.
    - Assign initial tasks and ensure all members have access to resources.
8. Wrap-Up and Action Items
    - Summarize decisions and confirm each member's next steps.
    - Set the date for the next check-in.

Responsibility Summary

- **Najwa**: Project Manager – Leads the meeting, ensures alignment with project goals, confirms priorities, and assigns resources.
- **Hamza**: Lead Developer – Outlines feature details, estimates development timelines, and suggests prioritization based on technical needs.
- **Abdihakim**: QA and Documentation Specialist – Defines user stories, documents dependencies and risks, and identifies QA checkpoints.

sprint1.md (Sprint 1 Planning Meeting)

Sprint Goal:

Complete the foundational features of the calendar app, focusing on user sign-up, event creation, and event viewing.

Spikes:

Spike 1: UI Design and Component Selection

- Investigate and decide on the Java Swing components (such as JList, JCheckBox, and JButton) that will be used for event and task management. Ensure they meet the functional requirements for user interaction.

Spike 2: User Data Storage Options

- Investigate and decide on the best method for storing user data (e.g., sign-up and login information) locally. Options include plain text files, serialized Java objects, or a lightweight embedded database (e.g., SQLite).

Spike 3: Event Data Structure

- Explore and define the most efficient data structure for storing events and tasks. This might involve deciding between lists, arrays, or other data structures based on accessibility and potential sorting needs.

Participants/Team Capacity Recording:

All Team members have been present during our set meeting time for every week and made equal contributions.

Team Capacity: Each member spent at least 10 hours on the project.

User Stories for Sprint 1:

User Story 1: User Sign-Up

- Story: As a fitness enthusiast, I want to sign up by entering my name and email so that I can have a personalized experience with the app and track my progress.
- Acceptance Criteria:
    - Input fields for name and email
    - "Sign Up" button to submit
    - Data is saved locally in a text file
    - Confirmation message displayed upon successful sign-up

Tasks

1. Task 1: Set up JTextField input fields for name and email.
    - Assigned to: Najwa
    - Estimation: 2 points (1 hour)
2. Task 2: Create a "Sign Up" button for submitting user data.
    - Assigned to: Abdihakim
    - Estimation: 1 point (30 minutes)
3. Task 3: Implement functionality to save user data (name and email) to a text file.
    - Assigned to: Hamza
    - Estimation: 3 points (1.5 hours)
4. Task 4: Display a confirmation message upon successful sign-up (e.g., using JOptionPane).
    - Assigned to: Najwa
    - Estimation: 1 point (30 minutes)

Total Estimate for User Story 1: 3 points

User Story 2: Basic Event Creation

- Story: As a busy student, I want to add events to specific days in my calendar so that I can keep track of important deadlines and manage my schedule effectively.
- Acceptance Criteria:
    - Date selection for event creation
    - Pop-up window for event details (name, date, and time)
    - Event details saved in an array or list
    - Events displayed on the selected date in the calendar

Tasks

1. Task 1: Enable date selection to create an event.
    - Assigned to: Hamza
    - Estimation: 2 points (1 hour)
2. Task 2: Set up a pop-up (using JOptionPane) to enter event details (name, date, and time).
    - Assigned to: Abdihakim
    - Estimation: 2 points (1 hour)

3. Task 3: Save event details (name, date, time) to an array or list.
    - Assigned to: Najwa
    - Estimation: 2 points (1 hour)
4. Task 4: Display events on the selected date in the calendar view.
    - Assigned to: Hamza
    - Estimation: 3 points (1.5 hours)

Total Estimate for User Story 2: 4 points

STANDUP:

[October 9, 2024] - Sprint #1 Standup #1

1. **What did you work on since the last standup?**
    - Reviewed the rubric for planning meetings, specifically focusing on the RPM.md and sprintN.md documentation requirements. Each team member evaluated their contributions to ensure comprehensive records in RPM.md, covering meeting participation, release goals, and user story references.
    - Outlined expectations for sprintN.md, including sprint goal documentation, task breakdowns, and recording team capacity.
2. **What do you commit to next?**
    - We will draft RPM.md using today's meeting notes.
    - Each team member will add a brief summary of their contributions for the RPM.md participant record.
    - The team will review and refine the user stories on Trello to align them with our sprint goals and complete task breakdowns for sprintN.md.
3. **When do you think you'll be done?**
    - The draft RPM.md and sprintN.md will be ready for review by the next standup and finalized for submission by the end of the day.
4. **Do you have any blockers?**
    - No blockers currently, though we may need to double-check the user stories on Trello to ensure all tasks align with our goals and rubric requirements.

[October 16th, 2024] - Sprint #2 Standup #2

1. **What did you work on since the last standup?**
   Worked on implementing the user sign-up feature using JTextField for input and saving user data locally in a text file.
2. **What do you commit to next?**
   Next, we will start developing the basic event creation feature, which allows users to add events with name, date, and time.
3. **When do you think you'll be done?**
   We aim to have the event creation feature completed by the next standup, ensuring it's ready for team review.
4. **Do you have any blockers?**
   No blockers currently. Just need to finalize the format for storing events so we can maintain consistency across the team.

[October 30th, 2024] - Sprint #3 Standup #3

1. **What did you work on since the last standup?**
   Finalized the "Mark Events as Done" feature. Added checkboxes next to each event in the event list, allowing users to mark events as completed. Ensured that the event status updates seamlessly when the checkbox is toggled.
2. **What do you commit to next?**
   We will be finalizing the "Event Viewing (List View)" feature, completing the setup of the JList to display events by day with navigation buttons to switch between dates.
3. **When do you think you'll be done?**
   We expect everything to be fully wrapped up by the end of the sprint, with all features functioning as intended.
4. **Do you have any blockers?**
   No blockers currently. We resolved the issue with dynamically updating the event list, so everything is on track for finalization.

User Story 2: Quick Event Addition for Busy Professionals

Story: As a working professional with a busy schedule, I want to quickly add and view important work events or deadlines in my calendar so that I can effectively plan my day and avoid missing any deadlines.

Acceptance Criteria:

- The user can select a specific date in the calendar view.
- A pop-up (e.g., using JOptionPane) allows the user to enter details like event name and time.
- After confirming, the event is saved and displayed in the calendar view on the selected date.
- If the user tries to save an event without entering both the event name and time, an error message prompts them to complete all fields.

Tasks:

- **Task 1**: Develop a date selection feature within the calendar view.
  Assigned to: Najwa
  Estimation: 2 points (1 hour)
- **Task 2**: Add a pop-up for entering event name and time.
  Assigned to: Abdihakim
  Estimation: 2 points (1 hour)
- **Task 3**: Write functionality to save event details to a calendar list.
  Assigned to: Hamza
  Estimation: 3 points (1.5 hours)
- **Task 4**: Display saved events on selected dates in the calendar view.
  Assigned to: Najwa
  Estimation: 2 points (1 hour)
- **Task 5**: Implement error handling for incomplete fields (event name and time).
  Assigned to: Abdihakim
  Estimation: 1 point (30 minutes)

Total Estimate for User Story 2: 10 points

4o

ChatGPT can make mistakes. Check important info.

Sprint Goal:

Complete the foundational features of the calendar app, focusing on user sign-up, event creation, and event viewing.

Spikes:

Spike 1: UI Design and Component Selection

Investigate and decide on the Java Swing components (such as JList, JCheckBox, and JButton) that will be used for event and task management. Ensure they meet the functional requirements for user interaction.

Spike 2: User Data Storage Options

Investigate and decide on the best method for storing user data (e.g., sign-up and login information) locally. Options include plain text files, serialized Java objects, or a lightweight embedded database (e.g., SQLite).

Spike 3: Event Data Structure

Explore and define the most efficient data structure for storing events and tasks. This might involve deciding between lists, arrays, or other data structures based on accessibility and potential sorting needs.

Participants/Team Capacity Recording:

All Team members have been present during our set meeting time for every week and made equal contributions.

Team Capacity: Each member spent at least 10 hours on the project.

User Stories for Sprint 1:

User Story 1: User Sign-Up

- Story: As a fitness enthusiast, I want to sign up by entering my name and email so that I can have a personalized experience with the app and track my progress.

- Acceptance Criteria:

  - Input fields for name and email

  - "Sign Up" button to submit

  - Data is saved locally in a text file

  - Confirmation message displayed upon successful sign-up

Tasks

Task 1: Set up JTextField input fields for username and password.

- Assigned to: Najwa

- Estimation: 2 points (1 hour)

Task 2: Create a "Login" button for submitting user data.

- Assigned to: Abdihakim

- Estimation: 1 point (30 minutes)

Task 3: Implement functionality to save user data (username and password) to a text file.

Assigned to: Hamza

Estimation: 3 points (1.5 hours)

Task 4: Display a confirmation message upon successful sign-up (e.g., using JOptionPane).

Assigned to: Najwa

Estimation: 1 point (30 minutes)

Total Estimate for User Story 1: 3 points

User Story 2: Basic Event Creation

- Story: As a busy student, I want to add events to specific days in my calendar so that I can keep track of important deadlines and manage my schedule effectively.

- Acceptance Criteria:

  - Date selection for event creation

  - Pop-up window for event details (name, date, and time)

  - Event details saved in an array or list

  - Events displayed on the selected date in the calendar

Tasks

Task 1: Enable date selection to create an event.

Assigned to: Hamza

Estimation: 2 points (1 hour)

Task 2: Set up a pop-up to enter event details (name and priority).

Assigned to: Abdihakim

Estimation: 2 points (1 hour)

Task 3: Save event details to an array or list.

Assigned to: Najwa

Estimation: 2 points (1 hour)

Task 4: Display events

Assigned to: Hamza

Estimation: 3 points (1.5 hours)

Total Estimate for User Story 2: 4 points

User Story 3: Quick Event Addition for Busy Professionals

- Story: As a working professional with a busy schedule, I want to quickly add and view

important work events or deadlines in my calendar so that I can effectively plan my day and avoid missing any deadlines.

- Acceptance Criteria:

- The user can select a specific date in the calendar view.

- A pop-up (e.g., using JOptionPane) allows the user to enter details like event name and time.

- After confirming, the event is saved and displayed in the calendar view on the selected date.

- If the user tries to save an event without entering both the event name and time, an error message prompts them to complete all fields.

Tasks:

Task 1: Develop a date selection feature within the calendar view.

Assigned to: Najwa

Estimation: 2 points (1 hour)

Task 2: Add a pop-up for entering event name and time.

Assigned to: Abdihakim

Estimation: 2 points (1 hour)

Task 3: Write functionality to save event details to a calendar list.

Assigned to: Hamza

Estimation: 3 points (1.5 hours)

Task 4: Display saved events on selected dates in the calendar view.

Assigned to: Najwa

Estimation: 2 points (1 hour)

Task 5: Implement error handling for incomplete fields (event name and time).

Assigned to: Abdihakim

Estimation: 1 point (30 minutes)

Total Estimate for User Story 2: 10 points

System Design Document

Project Team: Najwa, Abdihakim, Hamza
Product Owner: Abdihakim
Scrum Demo TA: Claudia
Sprint 1

Table of Contents

1. **Introduction**
2. **Goals**
3. **System Architecture**
4. **Class-Responsibility-Collaborator (CRC) Cards**
5. **Software Architecture Diagram**
6. **Detailed Design**
7. **User Interface Design**
8. **Future Enhancements**

1. Introduction

Overview

Moody, a To-Do List Application is a GUI-based Java program that enables users to manage their tasks based on priority, track completed tasks, and modify user settings. This application

provides a user authentication feature with a login screen, task creation and categorization by priority, task completion tracking, and user settings management.

2. Goals

- **Allow users to log in, create and manage tasks, and reset passwords.**
- **Categorize tasks into different priority levels and provide tabs for easy navigation.**
- **Provide a simple user interface using Java Swing components.**

3. System Architecture

The application is a standalone desktop GUI using Java Swing for user interface components. It is composed of two main parts:

1. **Login Module – Handles user authentication and displays the login page.**
2. **To-Do List Module – Allows users to add, categorize, and manage tasks within different priority tabs and settings**

4. Class-Responsibility-Collaborator (CRC) Cards

Class	Responsibilities	Collaborators
LoginPage	Display login form; verify user credentials against `User` data; redirect to `ToDoListPage` upon successful login.	`User`, `ToDoListPage`
User	Manage and store user credentials; validate and update passwords.	`LoginPage`, `ToDoListPage`
ToDoListPage	Display main UI for task management; organize tasks into priority tabs; manage completed tasks and user settings.	`User`, `LoginPage`
Task	Represent individual tasks with details like name, priority, and completion status; manage task behaviors like updating completion.	`ToDoListPage`
UserSettings	Handle user settings management, such as password reset and logout;	`User`, `ToDoListPage`

	interact with `User` for credential updates.	

5. Software Architecture Diagram

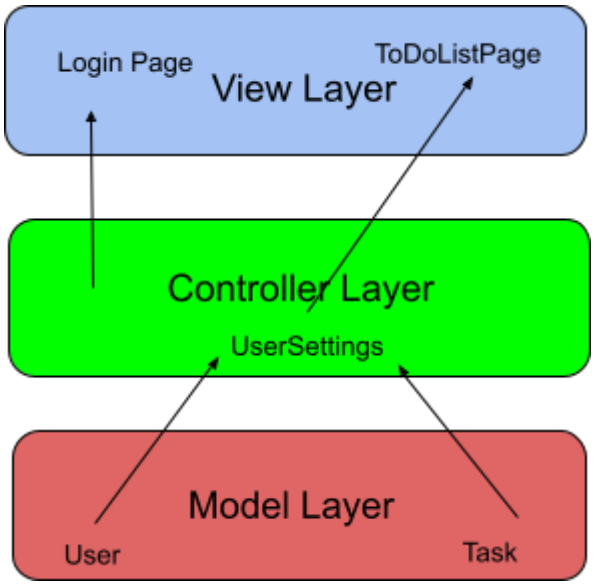

Diagram Outline

Here's how the diagram structure would look:

- **Top Layer (User Interface):**
  - `LoginPage` **and** `ToDoListPage` **(View layer)**
- **Middle Layer (Controller):**
  - `UserSettings` **(Controller layer)**
- **Bottom Layer (Model):**
  - `User` **and** `Task` **(Model layer)**

Interactions:

- `LoginPage` **calls methods on** `User` **to verify credentials.**
- **Upon successful login,** `LoginPage` **redirects to** `ToDoListPage`**.**
- `ToDoListPage` **interacts with** `UserSettings` **for tasks and settings changes.**
- `UserSettings` **invokes** `User` **for password updates and handles logouts.**
- `ToDoListPage` **accesses** `Task` **to manage task statuses and organization.**

6. Detailed Design

6.1 Login Flow

1. **User Input: The `LoginPage` displays text fields for username and password entry.**
2. **Authentication: Upon clicking the login button, the `LoginPage` verifies the input password against the `User` object's stored password.**
3. **Access Control: If verification succeeds, the application closes the login page and opens the `ToDoListPage`.**

6.2 Task Management Flow

1. **Task Creation:**
    - **User enters task details and selects priority in the `ToDoListPage`.**
    - **Clicking "Add Task" triggers `addTask()` to create a task entry panel.**
    - **The new task is displayed in the appropriate tab based on its priority.**
2. **Task Completion:**
    - **Selecting the checkbox next to a task marks it as completed.**
    - **Completed tasks are moved to the "Completed Tasks" tab through `moveToCompletedTasks()`, and the checkbox is disabled.**
3. **Task Deletion: This feature could be added in future versions if needed, allowing users to remove tasks.**

6.3 User Settings Flow

1. **Password Reset:**
    - **In the User Settings tab, the user enters the current password and a new password.**
    - **Clicking "Reset Password" verifies the current password; if correct, it updates the password using `setPassword()`.**
    - **A confirmation message is displayed for success or failure.**
2. **Logout:**
    - **Clicking "Logout" closes the `ToDoListPage` and reopens the `LoginPage` for authentication.**

7. User Interface Design

Login Page (LoginPage)

- **Components:**
    - **Username and password fields.**
    - **Login button to validate credentials.**
    - **Forgot password button (future enhancement).**

Main Task Management Page (ToDoListPage)

- **Components:**
  - **Tabbed Pane for categories (Stressed, Productive, Completed Tasks, User Settings).**
  - **Task Input Panel with text field, priority dropdown, and add button.**
  - **Priority Panels for displaying tasks based on their priority, using scrollable sections.**

User Settings Panel

- **Components:**
  - **Fields to enter the current and new password.**
  - **Reset password button to update credentials.**
  - **Logout button to close the `ToDoListPage` and reopen `LoginPage`.**

8. Future Enhancements

- **Task Due Dates: Add fields for task deadlines to help users manage time-bound tasks.**
- **Recurring Tasks: Enable users to set recurring tasks that reappear after completion.**
- **Reminder Notifications: Provide alerts or reminders for high-priority or due tasks.**
- **Persistent Data Storage: Integrate file-based or database storage to save tasks and user credentials across sessions.**
- **Multi-user Support: Allow multiple users to log in with separate accounts.**

Video Demo Script:

This is Team Cloudwaves sprint 1 demo of our to do list application “Moody List” which revolves around balancing tasks while also taking into consideration and maintaining ones mental health to not overstress yourself

When we run our application we are greeted with a simple login page. Here we can see 2 fields to input a username and a password. For demo purposes we will be using the generic admin login. There is also a forgot password feature which allows a user to change their password incase they forget it.

If someone tries to enter invalid credentials a popup will deny them from accessing the application.

Once valid credentials are inputted a pop up will inform that the credentials are valid and then we are greeted with the main application interface.