# Criterion A:

## The Scenario:

The client, I, has found themselves in a to-do list limbo. There already exist hundreds of to-do list applications and time managers, but none fit the way that the client thinks. Most of those programs were designed for professionals, not students.

The client has found that having a running list of tasks—one wherein all tasks are always visible, regardless of due date—is the most useful way to organize tasks. Paper does this quite well, but it’s bulky and “finished tasks” stay on the page when crossed out. Furthermore, paper is not always on one’s person, but the client always carries a Windows laptop.

Hence, I proposed a software solution that will have a running list of tasks, and, additionally, will include a calendar that can be used.

## The Rationale for the Solution:

This solution addresses the problem because it will be a To-Do manager that fits in the Client’s niche. The Solution should run on Windows, as that is the client’s main operating system. This project will be a standalone application to improve the ease of use for the client. It is intended to be eventually ported for use on an Android phone, so Java was an appropriate language to code in. Additionally, I have been working with mostly Java for most of my previous projects.

## Success Criteria:

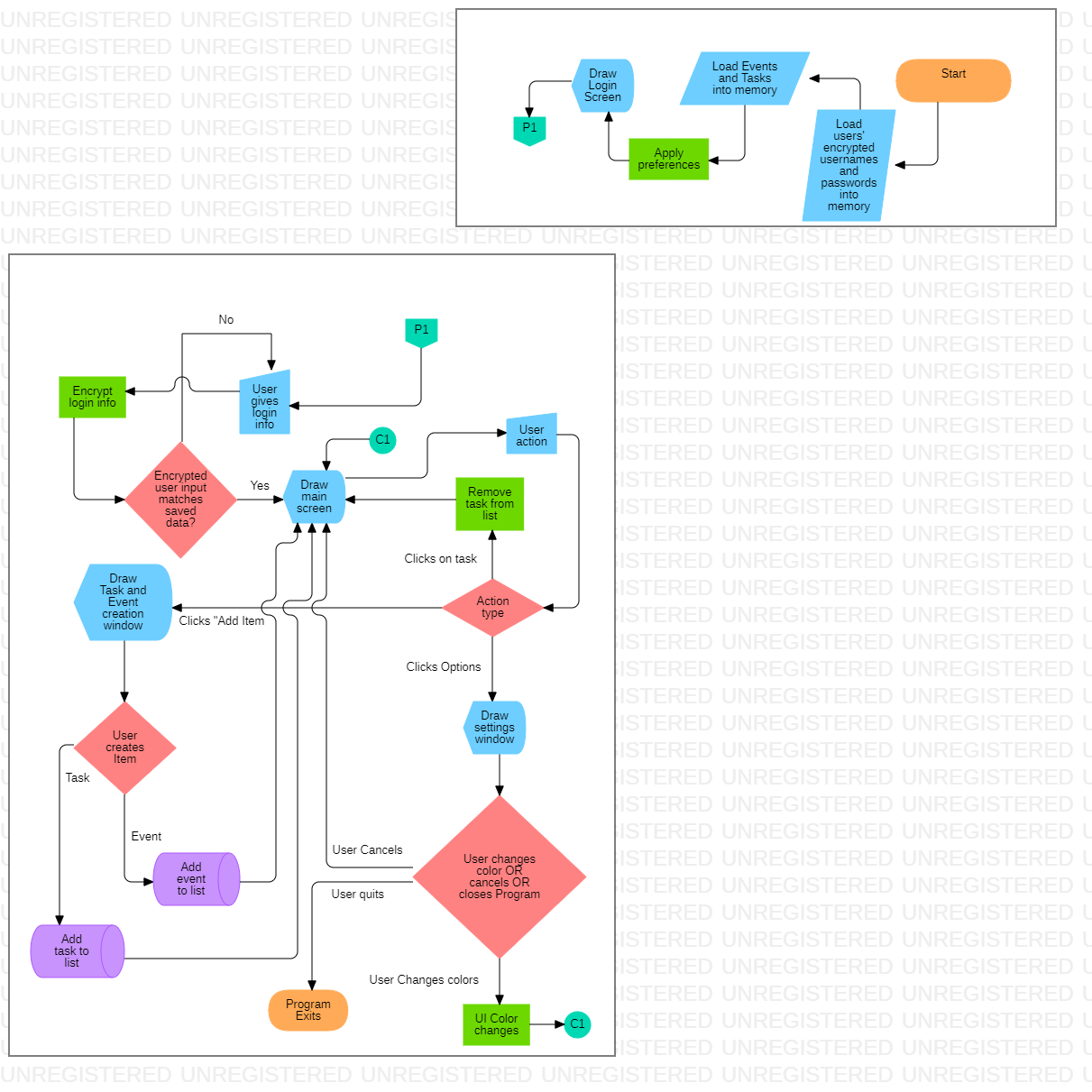
1. Client can add and remove tasks and events with ease.
2. Client can add due dates to tasks.
3. Client can add reminders for events and tasks.
4. Client can add priority levels to their events and tasks.
5. Client can see completed tasks.
6. Client can change the proportion of their UI used by the calendar and task list.
7. Client can change the colors of their UI.

# Criterion B:

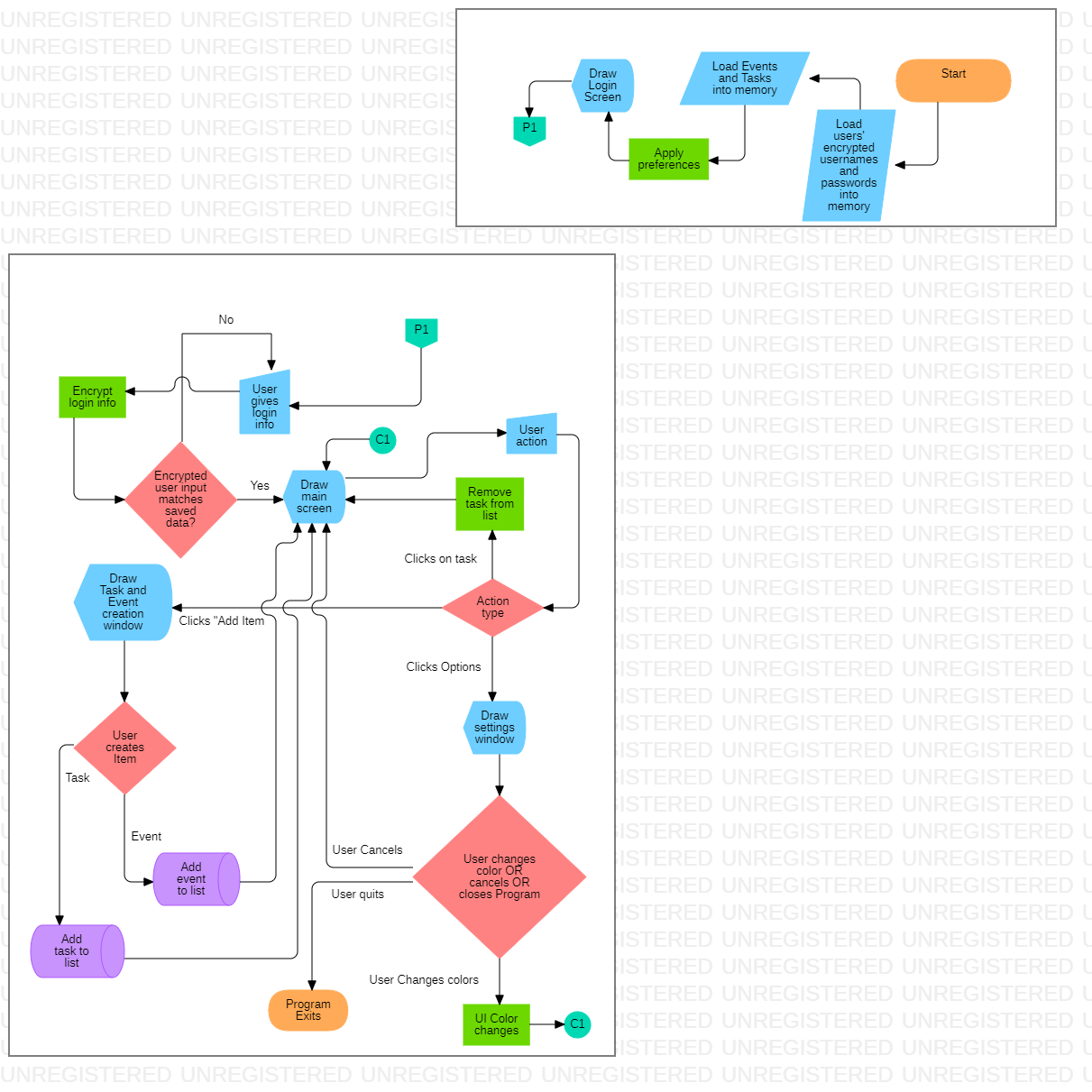
## Tests:

|  |  |  |
| --- | --- | --- |
| Criteria | Test | Result |
| 1. Can the client add and remove tasks and events with ease? | Check that tasks and events can be added with ease. | Pressing the add button brings up a menu where tasks are added to the task list. |
| 2. Can the client add due dates to tasks? | Check that tasks can have due dates added to them. | Clients can be notified when tasks are due. |
| 3. Can the client add reminders to events and tasks? | Check that events and tasks can be given reminders. | In the “Add item” dialog there is a place to add reminders. |
| 4. Can the client add priority levels to their events and tasks? | Check that tasks can be given priority levels. | In the “Add item” dialog there is a JSlider to choose priority levels. |
| 5. Can the client see completed tasks? | Check that completed tasks are still accessible to the client. | Clients can press a button and be shown completed tasks. |
| 6. Can the client change the proportion of their UI used by the calendar and task list? | Check that the calendar and task lists are resizable. | The JSplitPane has a built in resize bar. |
| 7. Can the client change the colors of their UI? | Check that UI colors are user-mutable. | The “Options” dialog has color settings. |

## Flow Charts

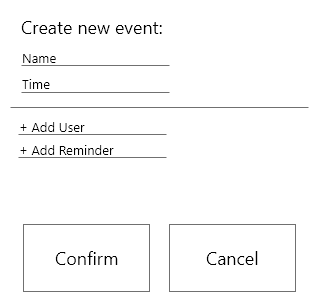


## Flow Charts

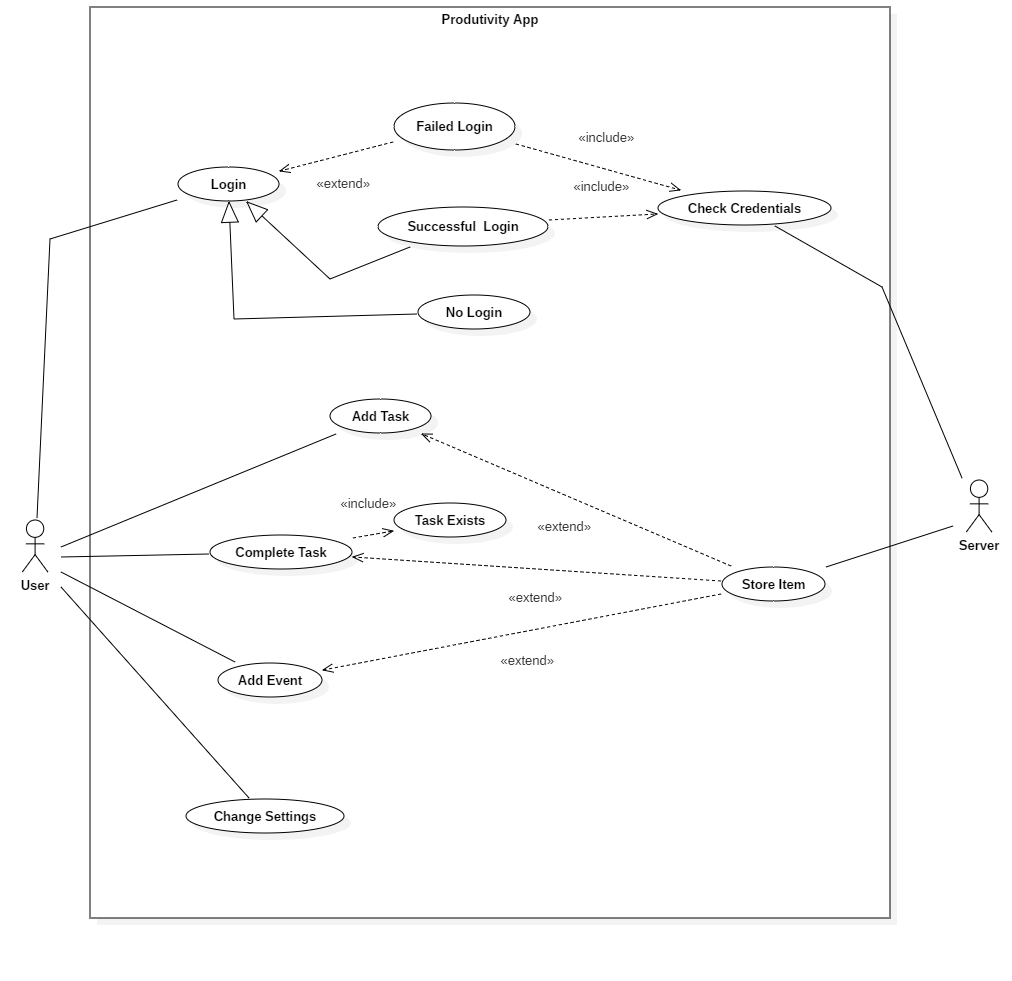


## Planned Graphics

## 



## Planning Documents



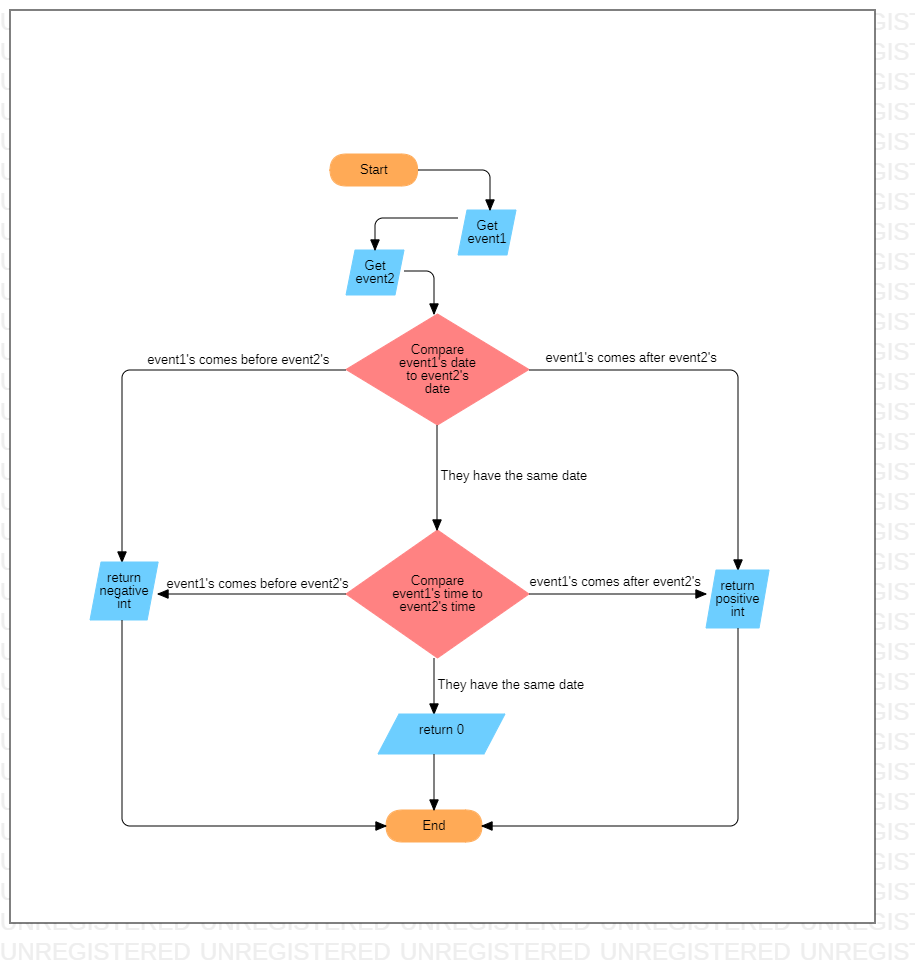
## Key Algorithms

TaskComparator

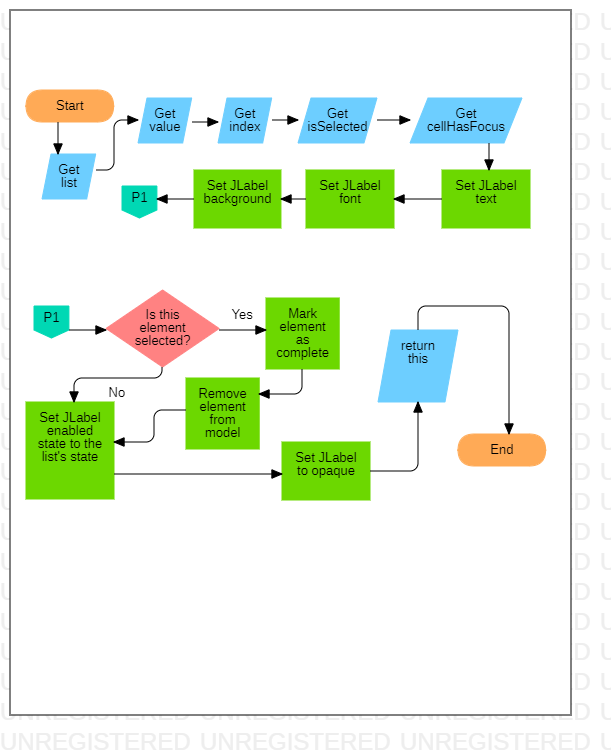
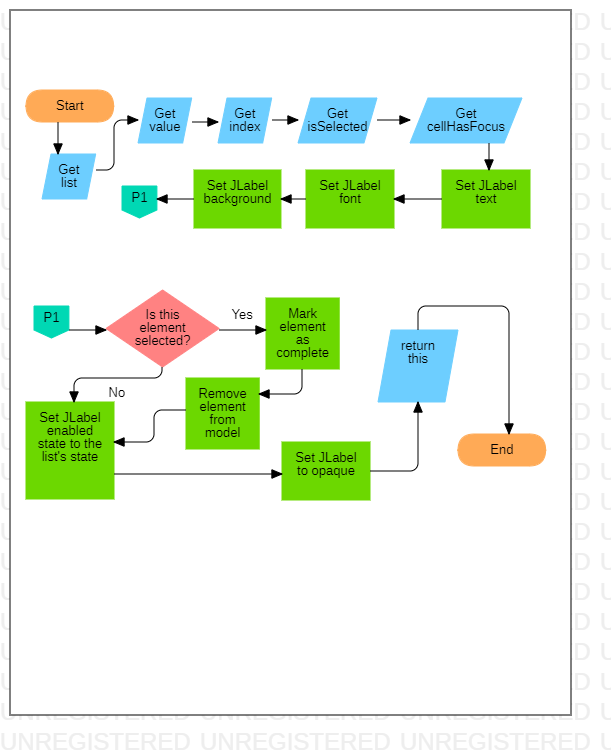
I had to implement a comparator for my custom “Task” class so I could order my tasks by priority in Java Swing. This comparator compares task priorities and then, as a last resort, to settle differences, task names.

### TaskComparator

EventComparator  
I had to implement a comparator for my custom “Event” class so I could order my events by date in Java Swing. This comparator compares event dates and then, as a last resort, to settle differences, event times. This ensures the proper ordering of events.



### TaskCellRenderer

In order to draw tasks in Java swing, I had to create a cell renderer. This extends JLabel so I can use the rendering methods and properties that exist in the JLabel class. This renderer essentially just takes the Task’s name and displays it in a JLabel. However, it does have an important functionality where, when the cell is selected, that Task is removed from the JList model (this removes it from the list of drawn tasks, but not from the task list or drive. 

# Criterion C:

UML Diagram

Complex Code Screenshots

Explain 6-8 uses of advanced techniques

Sources

Harvard Referencing

# Criterion D:

Video

# Criterion E:

Evaluation

Check Success Criteria

Client Opinions

Future Improvements

Resources (Remove later and place in separate appendix file):

Window Builder

Java Documentation

Big Java

Random Indian dude (https://www.youtube.com/watch?v=r8Qiz9Bn1Ag)

Some girl (<https://www.youtube.com/watch?v=zG8CrISqPpU>)

StarUML

Adobe XD

<https://javahungry.blogspot.com/2013/06/calendar-implementation-gui-based.html>

<https://www.geeksforgeeks.org/java-swing-jdialog-examples/>

<https://www.javatpoint.com/java-string-format>

<https://www.vogella.com/tutorials/JavaRegularExpressions/article.html>

<https://regexr.com/>

<http://regexlib.com/DisplayPatterns.aspx?cattabindex=4&categoryId=5>

<https://www.youtube.com/watch?v=dRX6qO46l44>