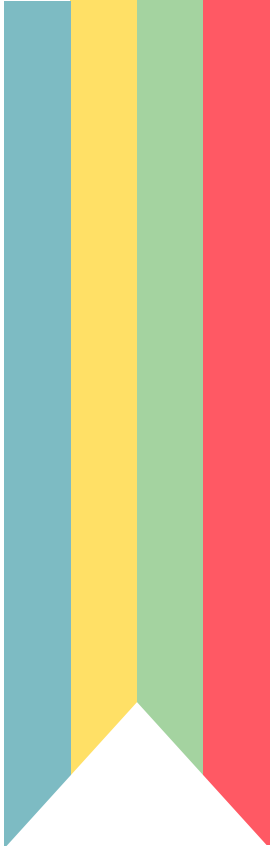

Prio

A Priority Task Management App

Technical Paper

presented to the faculty of

Mapúa Malayan Colleges Mindanao

In partial fulfillment of the requirements in:

CS106P Data Structures & Algorithms

presented by:

Azriel Peter Deduyo

Stacey Andrew Gonzaga

November 2023

INTRODUCTION

A. Company/Office Profile

Figure 1. AchieveLink logo

AchieveLink SEO Services was founded in 2020 with the vision of providing comprehensive search engine optimization services to businesses of all sizes. Its headquarters are in the vibrant heart of Davao City but serve clients worldwide. The team comprises proficient SEO specialists, creative content creators, web developers, and seasoned digital marketing experts who collaborate closely to ensure clients achieve online success.

Company Services:

- SEO Consultation and Strategy Development
- On-Page and Off-Page SEO Optimization
- Content Creation and Marketing
- Social Media Marketing
- Pay-Per-Click Advertising
- Web Development and Design
- Analytics and Reporting

B. Sample Forms/Reports

Figure 2. Sample Report 1

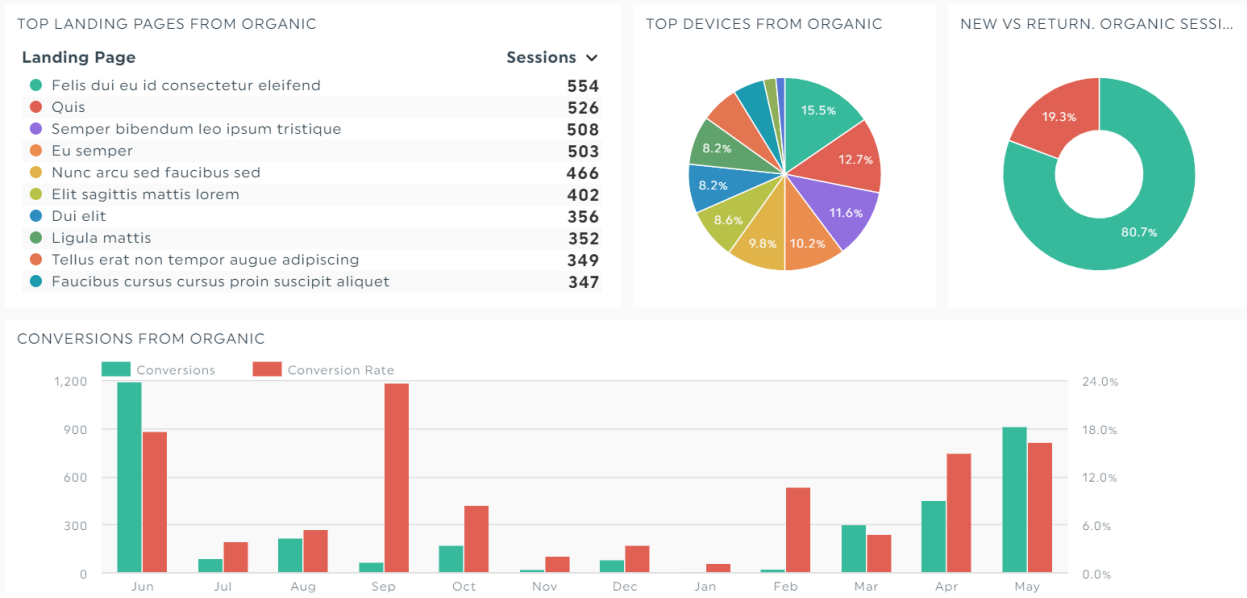

Figure 3. Sample Report 2

C. Statement of the Problem

At AchieveLink SEO Services, we understand that our employees play a critical role in addressing the challenges that businesses face within the digital marketing landscape. The following common challenges are experienced by our team, and we are committed to finding solutions from the employee's perspective:

1. **Disinterest in work and lack of motivation:** Many employees appear disengaged and exhibit a noticeable lack of enthusiasm within the workplace. This issue manifests itself through a variety of signs and behaviors that include a general sense of disinterest in their tasks, decreased motivation, and lower morale.

2. **Uncertainty with roles and responsibilities:** This issue stems from a lack of clarity regarding who is accountable for various aspects of key decisions, which can lead to confusion, inefficiencies, and even potential conflict.

3. **Challenges in tracking and prioritizing tasks:** These result in result in missed deadlines and suboptimal project outcomes. The challenges in tracking and prioritizing tasks often manifest as a lack of visibility into the status of ongoing projects, difficulty in managing competing priorities, and the potential for tasks to fall through the cracks.

When considering the findings and insights from various studies and articles on productivity apps and their effects on well-being and performance, it's evident that these principles can be seamlessly integrated into the Prio Task Organizer Application. Prio, like the discussed productivity apps, can significantly benefit from these insights and help solve the problems mentioned.

D. Project Objectives

The focus on employee engagement and motivation is a core aspect of Prio's mission to help users efficiently prioritize tasks. Much like Goal Guru [1]. Prio can incorporate features that monitor and encourage healthy work habits, such as setting task priorities based on importance, complexity, and deadlines. Users can engage in friendly competitions and challenges within the app, motivating them to improve their task management skills and boosting their overall productivity. In the development of Prio, the following challenges were identified and addressed:

1. **User-friendly interface for employee engagement:** Crafting a user-friendly interface will be helpful in enhancing employee engagement and motivation within the workplace. This, in turn, fosters a positive work environment, encourages active participation, and ultimately contributes to higher levels of motivation and job satisfaction.

2. **Task prioritization for decision-making workflows:** By carefully ordering and assigning priority to tasks, organizations can ensure that critical decisions receive the necessary attention and resources, leading to more informed and efficient decision-making processes.

3. **Performance Optimization for efficient task and project management:** Achieving performance optimization in task and project management involves streamlining processes and implementing best practices to enhance productivity and resource utilization. Efficiently managing tasks and projects not only boosts overall performance but also enables organizations to meet their goals more effectively, reducing costs and ensuring successful project outcomes.

E. Conceptual Framework

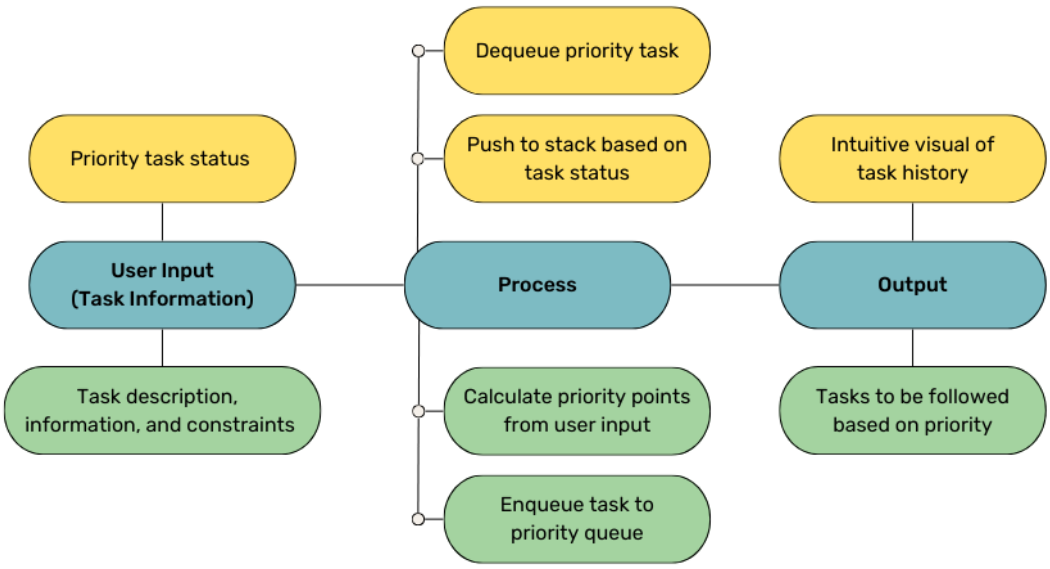

Figure 4. Prio's conceptual framework

Prio seamlessly integrates with our conceptual framework to empower AchieveLink employees with efficient task prioritization, data-driven insights, a

user-friendly interface, and performance optimization. It serves as a valuable tool to enhance productivity and organization in their daily tasks.

F. Scope and Limitations

The scope encompasses the adapted framework, using Prio as a tool, is designed to encompass various aspects of employee task management and productivity within AchieveLink. It is intended to offer comprehensive solutions to enhance their daily workflow. However, the scope remains focused on digital task management and productivity within the digital realm. It does not extend to unrelated areas such as traditional advertising.

Scope:

- Task prioritization using criteria and priority point calculation.
- Viewing task history, including completed and canceled tasks.
- Tracking task progress, distinguishing between tasks in progress and those due.

Limitations:

- Limited functionality beyond task prioritization, history, and progress tracking.
- Does not offer features for advanced project management or collaboration.
- Not designed for integration with external tools or applications.

G. Project Definition

In the context of AchieveLink employees using Prio, the principles of Data Structures and Algorithms (DSA) can be applied to optimize task management. This includes the efficient indexing and management of tasks within the application, the development of algorithms for task prioritization, and the application of data analysis techniques to improve employee performance and task execution.

Priority Queue, Binary Heap, and Stacks for Task Management

Prio leverages the power of priority queues, binary heaps, and stacks to enhance the efficiency of task management. These data structures play a pivotal role in providing employees with a streamlined and productive task management experience:

1. **Priority queue for task prioritization**

   Prio employs a priority queue data structure to prioritize tasks. Each task is assigned a priority based on factors like due date, estimated time, complexity, risk, and other user-defined criteria. The priority queue ensures that the most urgent and important tasks are readily accessible to employees.

2. **Binary heap for efficient retrieval of tasks**
   A binary heap is leveraged for efficient task handling, facilitating swift task enqueueing and dequeuing, which is crucial for optimizing task management and execution.

3. **Stacks for tracking activity and task history**
   The stack data structure is employed to visualize task history by maintaining a chronological record of executed tasks, allowing users to trace their workflow and understand the sequence of completed or canceled activities.

PRIO: Precedence-Regulated Input Optimizer

PRIO, or Precedence-Regulated Input Optimizer, is an algorithm designed to efficiently manage and prioritize user-input tasks. The process involves assigning points to tasks based on user-defined criteria, enqueuing them in a priority queue, and employing a binary heap sort to determine the next task in the queue. This approach ensures that higher-priority tasks are addressed first. Additionally, as tasks undergo status changes, the algorithm organizes and stores them in respective files, contributing to an organized and streamlined task management system. The use of priority queues and binary heap sort enables PRIO to optimize task execution, providing a systematic and responsive solution for handling user inputs.

H. Time and Space Complexity

The time and space complexity of the adapted framework using Prio will vary depending on the complexity of the tasks and projects managed by AchieveLink employees. The goal is to ensure that time and resources are utilized efficiently while delivering high-quality results and maximizing productivity. Prio's performance optimization features contribute to achieving this goal by minimizing time and space complexity in task management through the strategic use of priority queues and stacks.

Optimizing Time and Space Complexity with Priority Queues and Stacks:

- **Priority Queue and Binary Heap for Efficient Task Retrieval**: Prio's use of a priority queue keeps time complexity for task retrieval low, typically *O(log n)* due to the efficient data structure. This ensures that employees can quickly access and prioritize tasks even in scenarios with many tasks.

- **Stack for Task Handling**: The stack, with its *O(1)* time complexity for push and pop operations, simplifies task handling. Utilizing the push operation in this sense will allow users to keep track of their activities, keeping a logical and intuitive history of the tasks they've completed.

- **Space Complexity Management**: Prio's approach to space complexity optimizes memory usage by efficiently managing data structures for both the priority queue and stack. The space complexity typically ranges from O(n) to *O(log n)* based on the number of tasks managed.

The combination of these data structures contributes to a well-balanced framework that minimizes time and space complexity, ensuring that AchieveLink employees can efficiently manage their tasks while delivering high-quality results.

PROJECT SOLUTION

A. Project/System Prototyping

ADT Testing

```
62      public static void BinaryHeapSort(int[] data)
63      {
64          int n = data.Length;
65
66          for (int i = n / 2 - 1; i >= 0; i--)
67          {
68              Heapify(data, n, i);
69          }
70
71          for (int i = n - 1; i >= 0; i--)
72          {
73              int temp = data[0];
74              data[0] = data[i];
75              data[i] = temp;
76
77              Heapify(data, i, 0);
78          }
79      }
80
            3 references
81      private static void Heapify(int[] data, int n, int root)
82      {
83          int smallest = root;
84          int left = 2 * root + 1;
85          int right = 2 * root + 2;
86
87          if (left < n && data[left] < data[smallest])
88          {
89              smallest = left;
90          }
91          if (right < n && data[right] < data[smallest])
92          {
93              smallest = right;
94          }
95
96          if (smallest != root)
97          {
98              int temp = data[root];
99              data[root] = data[smallest];
100             data[smallest] = temp;
101
102             Heapify(data, n, smallest);
103         }
104     }
```

Figure 5. Implementation of Binary Heap Sort

```
126     public static void QuickSort(int[] data, int low, int high)
127     {
128         if (low < high)
129         {
130             int partitionIndex = Partition(data, low, high);
131             QuickSort(data, low, partitionIndex - 1);
132             QuickSort(data, partitionIndex + 1, high);
133         }
134     }
135
            1 reference
136     public static int Partition(int[] data, int low, int high)
137     {
138         int pivot = data[high];
139         int i = (low - 1);
140
141         for (int j = low; j < high; j++)
142         {
143             if (data[j] < pivot)
144             {
145                 i++;
146                 Swap(data, i, j);
147             }
148         }
149
150         Swap(data, i + 1, high);
151         return i + 1;
152     }
153
```

Figure 6. Implementation of Quick Sort

```
106     public static void SelectionSort(int[] data)
107     {
108         int n = data.Length;
109         for (int i = 0; i < n - 1; i++)
110         {
111             int minIndex = i;
112             for (int j = i + 1; j < n; j++)
113             {
114                 if (data[j] < data[minIndex])
115                     minIndex = j;
116             }
117             if (minIndex != i)
118             {
119                 int temp = data[i];
120                 data[i] = data[minIndex];
121                 data[minIndex] = temp;
122             }
123         }
124     }
```

Figure 7. Implementation of Selection Sort

```
public static void BinaryHeapSortComparison()
{
    int[] data = GenerateTestData(10000);

    Stopwatch sw = new Stopwatch();
    sw.Start();

    BinaryHeapSort(data);

    sw.Stop();

    Console.WriteLine("Binary Heap Sort Execution Time: " + sw.ElapsedMilliseconds + " ms");
}

1 reference
public static void SelectionSortComparison()
{
    int[] data = GenerateTestData(10000);

    Stopwatch sw = new Stopwatch();
    sw.Start();

    SelectionSort(data);

    sw.Stop();

    Console.WriteLine("Selection Sort Execution Time: " + sw.ElapsedMilliseconds + " ms");
}

1 reference
public static void QuickSortComparison()
{
    int[] data = GenerateTestData(10000);

    Stopwatch sw = new Stopwatch();
    sw.Start();

    QuickSort(data, 0, data.Length - 1);

    sw.Stop();

    Console.WriteLine("Quick Sort Execution Time: " + sw.ElapsedMilliseconds + " ms");
}
```

Figure 8. Methods for testing

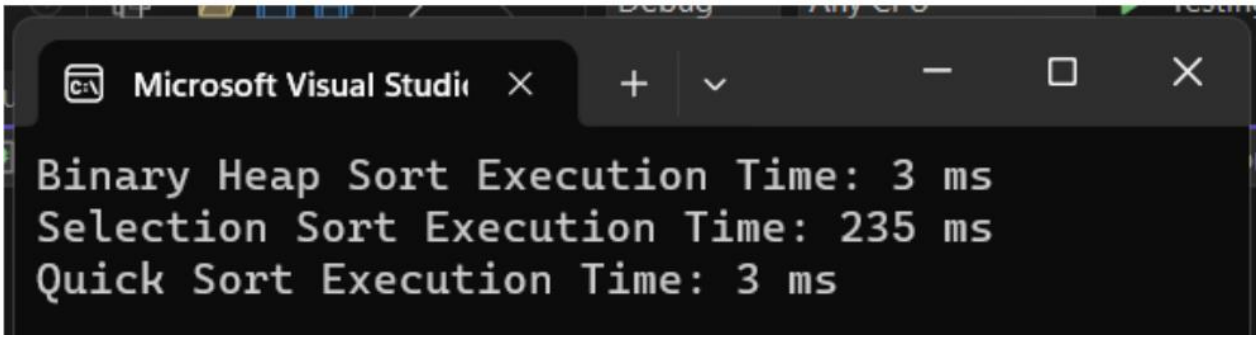

Figure 9. Test results

Prototyping

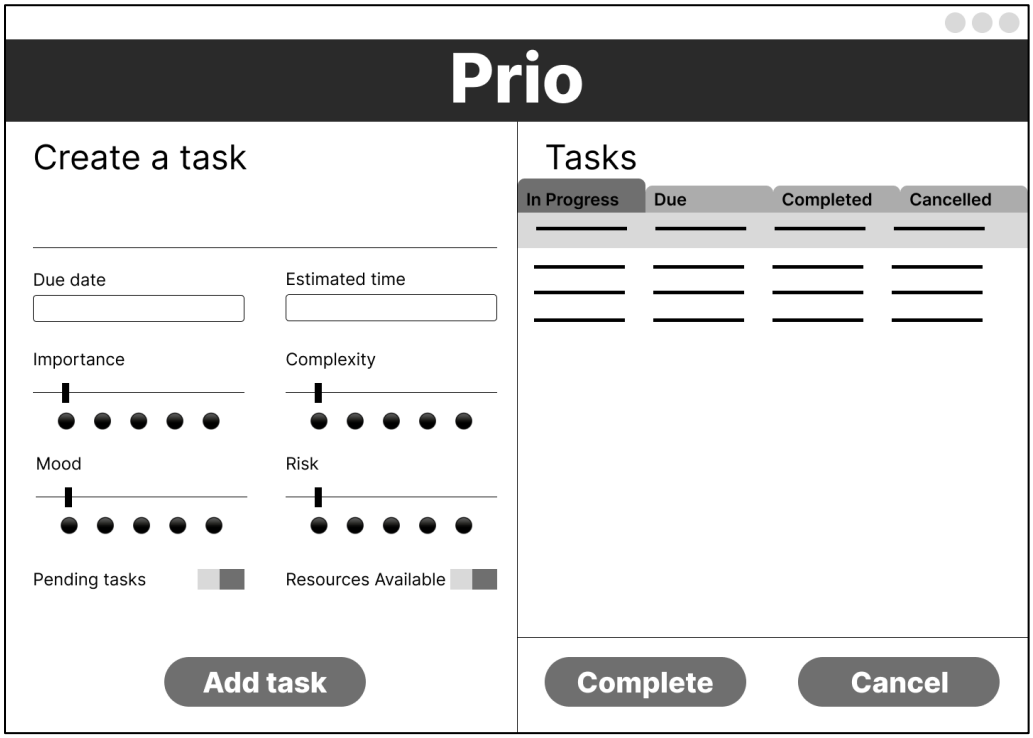

Figure 10. UI Wireframe

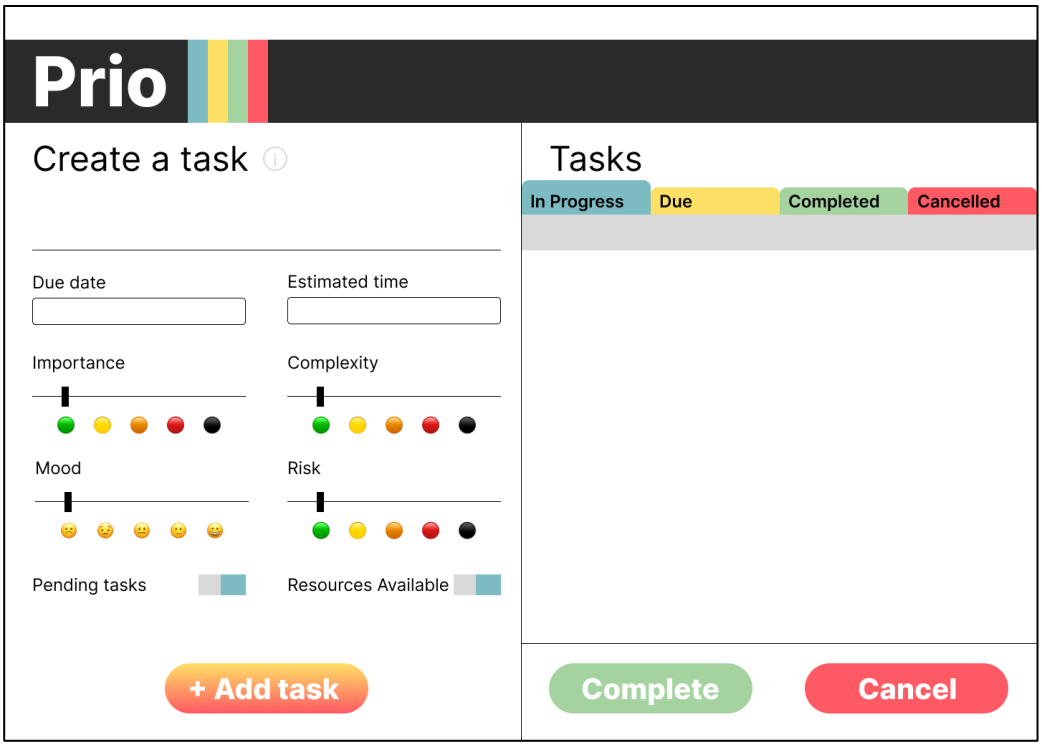

Figure 11. UI Mockup

B. Sample Input/Output

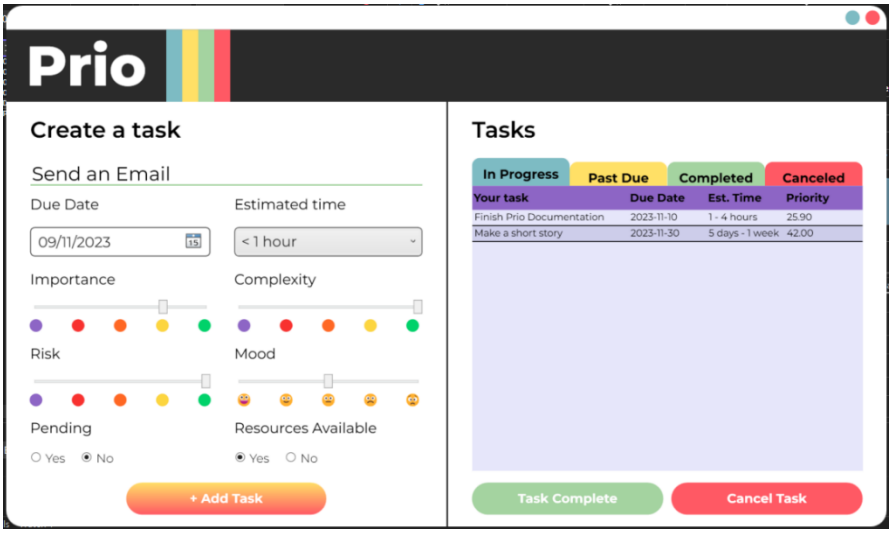

Figure 12. Prio's primary view.

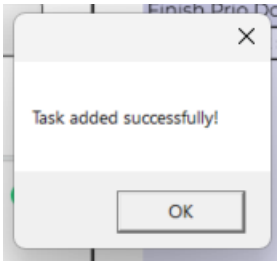

Figure 13. User confirmation: Task has been created.

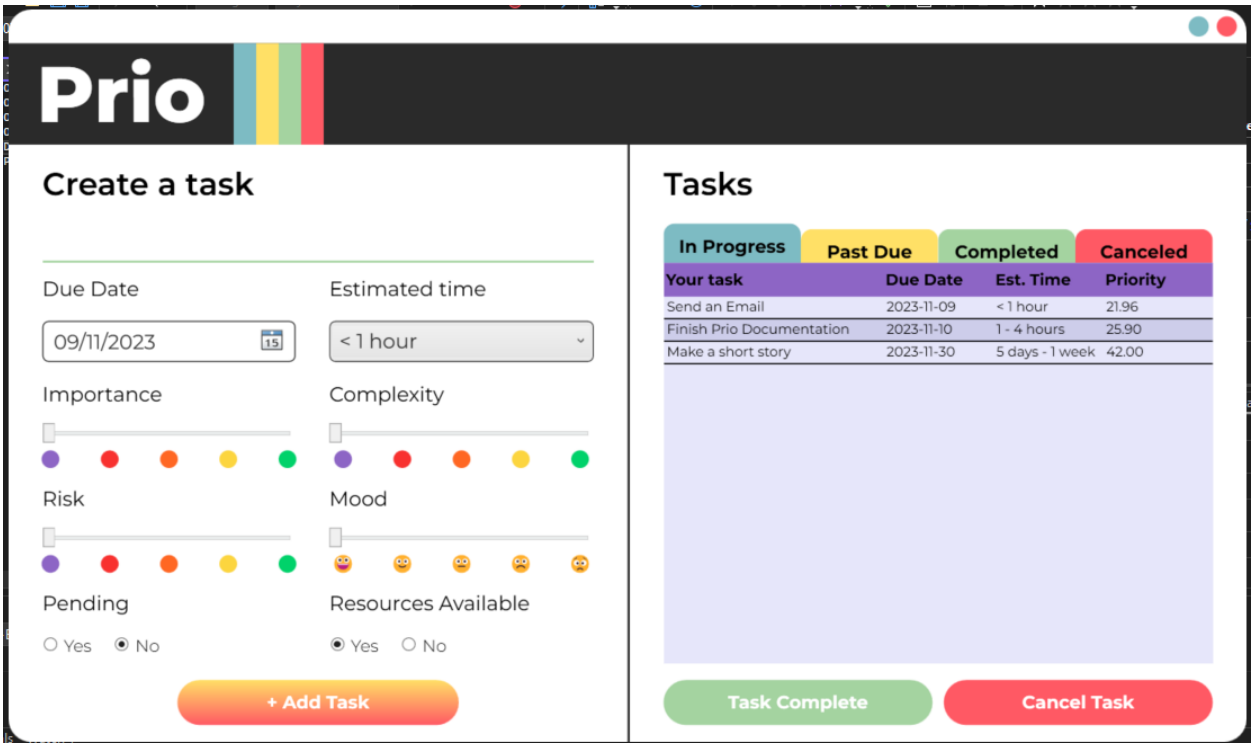

Figure 14. Updated view after task creation.

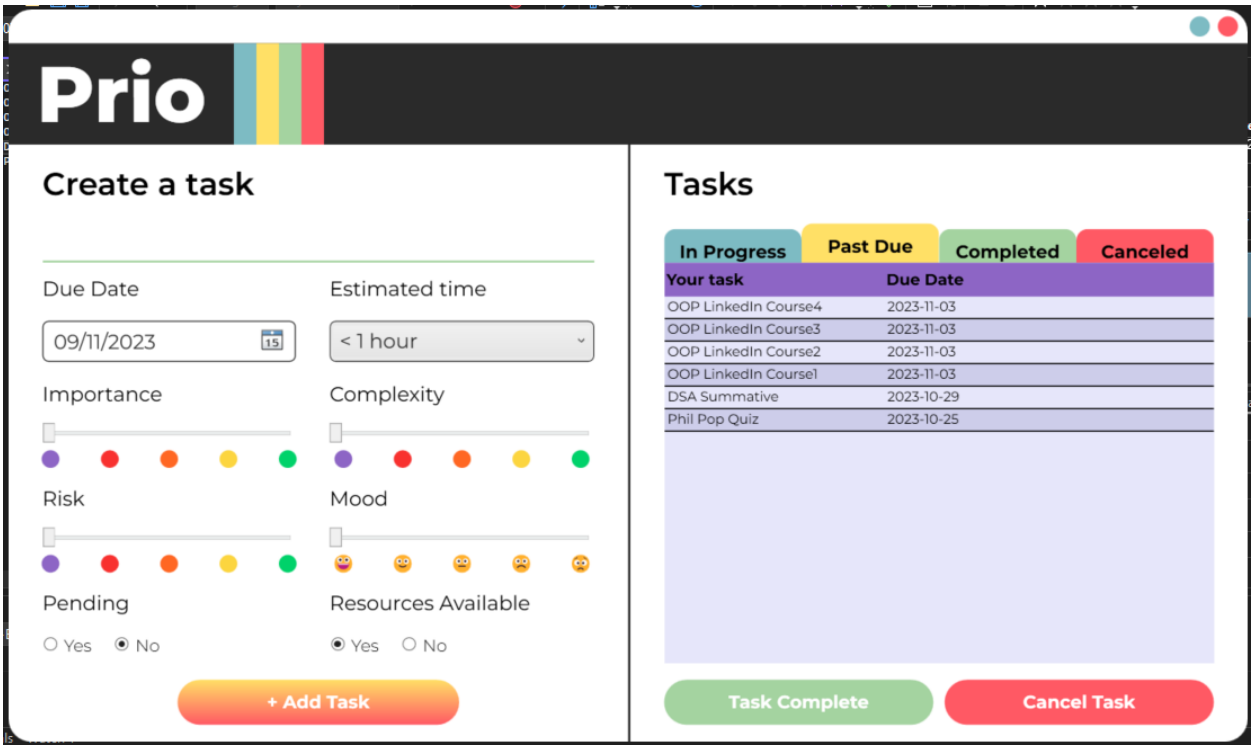

Figure 15. Due tab

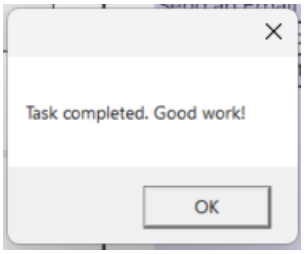

Figure 16. User confirmation: task has been completed.

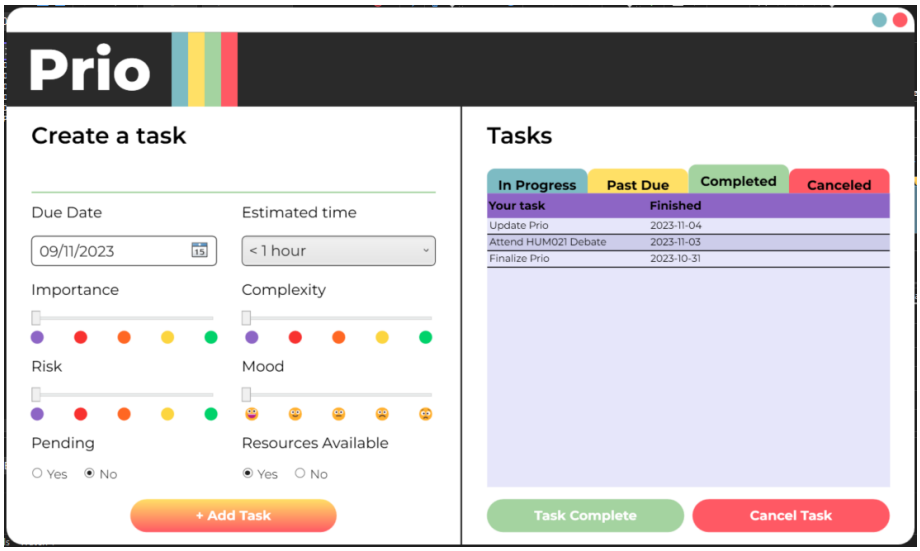

Figure 17. Completed tab

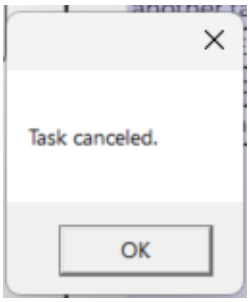

Figure 18. User confirmation: Task has been canceled.

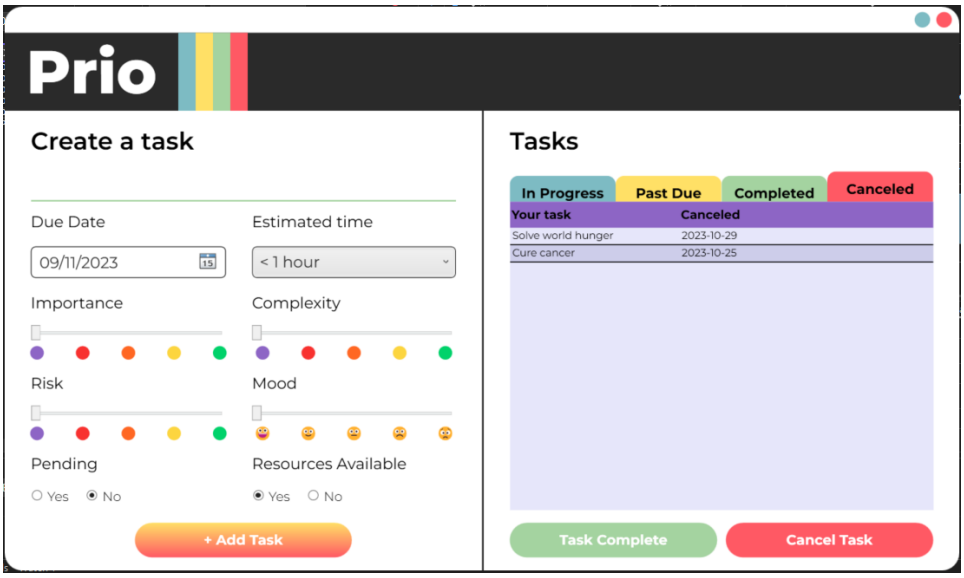

Figure 19. Canceled tab

C. Source Code

BinaryHeap.cs

```
using System;
using System.Collections.Generic;
using System.Linq;

public class BinaryHeap
{
    public static List<string> Sort(List<string> lines)
    {
        var lineData = new List<(string Line, decimal Value)>();

        foreach (var line in lines)
        {
            var parts = line.Split(',');

            if (parts.Length == 4)
            {
                if (decimal.TryParse(parts[3], out decimal value))
                {
                    lineData.Add((line, value));
                }
            }
        }

        BuildMaxHeap(lineData);

        int n = lineData.Count;
        for (int i = n - 1; i > 0; i--)
        {
            Swap(lineData, 0, i);
            MaxHeapify(lineData, 0, i);
        }

        var orderedLines = lineData.Select(ld => ld.Line).ToList();
        return orderedLines;
    }

    private static void BuildMaxHeap(List<(string Line, decimal Value)> arr)
    {
        int n = arr.Count;

        for (int i = n / 2 - 1; i >= 0; i--)
        {
            MaxHeapify(arr, i, n);
        }
    }

    private static void MaxHeapify(List<(string Line, decimal Value)> arr, int i,
int n)
    {
        int largest = i;
        int left = 2 * i + 1;
```

```
        int right = 2 * i + 2;

        if (left < n && arr[left].Value > arr[largest].Value)
        {
            largest = left;
        }

        if (right < n && arr[right].Value > arr[largest].Value)
        {
            largest = right;
        }

        if (largest != i)
        {
            Swap(arr, i, largest);
            MaxHeapify(arr, largest, n);
        }
    }

    private static void Swap(List<(string Line, decimal Value)> arr, int i, int j)
    {
        var temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }
}
```

EstimatedTime.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Prio.Model
{
    public class EstimatedTime
    {
        public static string[] EstimatedTimeArray =
            {"< 1 hour", "1 - 4 hours", "Half a day", "1 day", "2 - 4 days", "5 days
- 1 week", "2 - 3 weeks", "1 month"};
    }
}
```

PrioCalculator.cs

```
using Prio.Model;
using System;

namespace Prio.Model
{
```

```
    public class PrioCalculator
    {
        public static decimal CalculatePoints(int daysDifference, decimal estTime,
int pending,
            int resourcesAvailable, int importance, int complexity, int risk, int
mood)
        {
            decimal result = daysDifference - estTime +
                (decimal)(importance + complexity + risk + mood) +
                pending + resourcesAvailable;

            return result;
        }
    }
}
```

DataItem.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Prio.Model
{
    public class DataItem
    {
        public string Task { get; set; }
        public string Date { get; set; }
        public string Time { get; set; }
        public string Priority { get; set; }
    }
}
```

Stacks.cs

```
using System;
using System.IO;
using System.Collections.Generic;
using System.Linq;
using System.Windows;

public class Stacks
{
    public static void CompleteTopTask(string entryPath, string exitPath)
    {
        try
        {
            string[] lines = File.ReadAllLines(entryPath);

            if (lines.Length > 0)
```

```
            {
                string[] parts = lines[0].Split(',');

                if (parts.Length >= 1)
                {
                    string firstPart = parts[0];
                    string currentDate = DateTime.Today.ToString("yyyy-MM-dd");
                    string lineToAdd = $"{firstPart},{currentDate}, , ";

                    List<string> completedLines =
File.ReadAllLines(exitPath).ToList();
                    completedLines.Insert(0, lineToAdd);
                    File.WriteAllLines(exitPath, completedLines);

                    var remainingLines = lines.Skip(1).ToArray();

                    File.WriteAllLines(entryPath, remainingLines);

                    MessageBox.Show("Task completed. Good work!");
                }
            }
            else
            {
                MessageBox.Show("No tasks to dequeue.");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("An error occurred while dequeuing the task: " +
ex.Message);
        }
    }

    public static void CancelTopTask(string entryPath, string exitPath)
    {
        try
        {
            string[] lines = File.ReadAllLines(entryPath);

            if (lines.Length > 0)
            {
                string[] parts = lines[0].Split(',');

                if (parts.Length >= 1)
                {
                    string firstPart = parts[0];
                    string currentDate = DateTime.Today.ToString("yyyy-MM-dd");
                    string lineToAdd = $"{firstPart},{currentDate}, , ";

                    List<string> canceledLines =
File.ReadAllLines(exitPath).ToList();
                    canceledLines.Insert(0, lineToAdd);
                    File.WriteAllLines(exitPath, canceledLines);

                    var remainingLines = lines.Skip(1).ToArray();
                    File.WriteAllLines(entryPath, remainingLines);

                    MessageBox.Show("Task canceled.");
```

```
                }
            }
            else
            {
                MessageBox.Show("No tasks to dequeue.");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("An error occurred while dequeuing the task: " +
ex.Message);
        }
    }

    public static void StackDueTasks(string entryPath, string exitPath)
    {
        try
        {
            string[] lines = File.ReadAllLines(entryPath);
            List<string> dueLines = new List<string>();
            List<string> remainingLines = new List<string>();

            foreach (var line in lines)
            {
                var parts = line.Split(',');

                if (parts.Length == 4)
                {
                    string taskDate = parts[1];

                    if (DateTime.TryParse(taskDate, out DateTime dueDate) && dueDate
< DateTime.Today)
                    {
                        dueLines.Add($"{parts[0]},{parts[1]}, , ");
                    }
                    else
                    {
                        remainingLines.Add(line);
                    }
                }
            }

            if (dueLines.Count > 0)
            {
                var existingDueLines = File.ReadAllLines(exitPath).ToList();
                existingDueLines.InsertRange(0, dueLines);
                File.WriteAllLines(exitPath, existingDueLines);
                File.WriteAllLines(entryPath, remainingLines);
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("An error occurred while stacking due tasks: " +
ex.Message);
        }
    }
}
```

MainWindow.xaml.cs

```
using Prio.Model;
using Prio.ViewModel;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.IO;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;

namespace Prio.View
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {

        // Initializing variables ----------------------------------------------
------------------------------------------------------------------------------
        private int resourcesAvailable, pending, daysDifference;
        private string selectedValue;
        private DateTime selectedDate;
        private ObservableCollection<DataItem> dataItems = new
ObservableCollection<DataItem>();

        // Constructor ---------------------------------------------------------
------------------------------------------------------------------------------
        public MainWindow()
        {
            InitializeComponent();

            // Load data from the file
            StackDue();
            LoadDataFromFile(@"..\..\Files\IP.txt");

            var viewModel = new MainViewModel();
            DataContext = dataItems;

            DateTime maxSelectableDate = DateTime.Today.AddDays(30);
            datePicker.DisplayDateEnd = maxSelectableDate;
        }

        // Loading datagrids ---------------------------------------------------
------------------------------------------------------------------------------
        private void LoadDataFromFile(string filePath)
        {
            try
            {
                // Read the lines from the file
                var lines = File.ReadAllLines(filePath);
                var data = new List<DataItem>();
```

```
                foreach (var line in lines)
                {
                    // Split the line by the comma character
                    var parts = line.Split(',');

                    if (parts.Length == 4)
                    {
                        var item = new DataItem
                        {
                            Task = parts[0],
                            Date = parts[1],
                            Time = parts[2],
                            Priority = parts[3]
                        }; data.Add(item);
                    }
                    else
                    {
                        MessageBox.Show("Invalid data format in the file.");
                        return;
                    }
                }

                // Set the DataGrid's ItemsSource to the list of data
                PrioDataGrid.ItemsSource = data;
            }
            catch (Exception ex)
            {
                MessageBox.Show("An error occurred while loading data from the file:
" + ex.Message);
            }
        }

        // Computing points (basis for ordering elements) ------------------------
-------------------------------------------------------------------------------
        public decimal CalculatePoints()
        {

            decimal result = PrioCalculator.CalculatePoints(daysDifference,
EstimatedTime(),
                pending, resourcesAvailable, (int)Importance.Value,
(int)Complexity.Value,
                (int)Risk.Value, (int)Mood.Value);

            return result;
        }

        public decimal EstimatedTime()
        {
            if (comboBoxPriority.SelectedItem != null)
            {
                string selectedValue = comboBoxPriority.SelectedItem.ToString();

                switch (selectedValue)
                {
                    case "< 1 hour": return 0.04m;
                    case "1 - 4 hours": return 0.16m;
                    case "Half a day": return 0.5m;
```

```
                    case "1 day": return 1m;
                    case "2 - 4 days": return 4m;
                    case "5 days - 1 week": return 7m;
                    case "2 - 3 weeks": return 21m;
                    case "1 month": return 30m;
                    default: return 7m;
                }
            }
            return 7m;
        }

        private void datePicker_SelectedDateChanged(object sender,
SelectionChangedEventArgs e)
        {
            selectedDate = datePicker.SelectedDate.Value;
            TimeSpan difference = selectedDate - DateTime.Today;
            daysDifference = ((int)difference.TotalDays) + 1;
        }

        private void PendingRadio_Checked(object sender, RoutedEventArgs e)
        {
            if (sender is RadioButton radioButton)
            { pending = radioButton.Content.ToString() == "No" ? 1 : 3; }
        }

        private void ResourcesRadio_Checked(object sender, RoutedEventArgs e)
        {
            if (sender is RadioButton radioButton)
            { resourcesAvailable = radioButton.Content.ToString() == "Yes" ? 3 : 1;
}
        }

        // Enqueue and dequque tasks ------------------------------------------
------------------------------------------------------------------------------
        private void EnqueueTask()
        {
            try
            {
                if (string.IsNullOrWhiteSpace(txtTask.Text) || txtTask.Text ==
"Hello!")
                {
                    MessageBox.Show("Enter a valid task description");
                    return;
                }

                if (!datePicker.SelectedDate.HasValue)
                {
                    MessageBox.Show("Please select a due date.");
                    return;
                }

                if (comboBoxPriority.SelectedItem == null)
                {
                    MessageBox.Show("Please select a time estimate.");
                    return;
                }
```

```
                string newLine =
$"{txtTask.Text},{datePicker.SelectedDate.Value:yyyy-MM-dd}," +

$"{comboBoxPriority.SelectedItem.ToString()},{CalculatePoints():F2}";

                File.AppendAllLines(@"..\..\Files\IP.txt", new[] { newLine });

                MessageBox.Show("Task added successfully!");

                txtTask.Text = "";
                txtTask.Focus();
                Importance.Value = 1;
                Complexity.Value = 1;
                Risk.Value = 1;
                Mood.Value = 1;
            }
            catch (Exception ex)
            {
                MessageBox.Show("An error occurred while enqueueing task to the
file: " + ex.Message);
            }
        }

        private void StackComplete()
        {
            Stacks.CompleteTopTask(@"..\..\Files\IP.txt",
@"..\..\Files\Completed.txt");
            LoadDataFromFile(@"..\..\Files\IP.txt");
        }

        private void StackCancel()
        {
            Stacks.CancelTopTask(@"..\..\Files\IP.txt",
@"..\..\Files\Canceled.txt");
            LoadDataFromFile(@"..\..\Files\IP.txt");
        }

        private void StackDue()
        {
            Stacks.StackDueTasks(@"..\..\Files\IP.txt", @"..\..\Files\Due.txt");
        }

        // Ordering by Priority ------------------------------------------------
------------------------------------------------------------------------------
        private void ReorderFile(string filePath)
        {
            try
            {
                var lines = File.ReadAllLines(filePath).ToList();
                var orderedLines = BinaryHeap.Sort(lines);
                File.WriteAllLines(filePath, orderedLines);
            }
            catch (Exception ex)
            {
                MessageBox.Show("An error occurred while reordering the file: " +
ex.Message);
            }
        }
```

```
private void Window_MouseDown(object sender, MouseButtonEventArgs e)
{
    if (e.LeftButton == MouseButtonState.Pressed)
    {
        DragMove();
    }
}

private void btnMinimize_Click(object sender, RoutedEventArgs e)
{
    WindowState = WindowState.Minimized;
}

private void btnClose_Click(object sender, RoutedEventArgs e)
{
    Application.Current.Shutdown();
}

private void tabIP_Click(object sender, RoutedEventArgs e)
{
    tabIP.Height = 35;
    tabDue.Height = 30;
    tabComplete.Height = 30;
    tabCancel.Height = 30;

    btnComplete.IsEnabled = true;
    btnCancel.IsEnabled = true;

    DateColumn.Header = "Due Date";
    TimeColumn.Header = "Est. Time";
    PointsColumn.Header = "Priority";

    LoadDataFromFile(@"..\..\Files\IP.txt");
}

private void tabDue_Click(object sender, RoutedEventArgs e)
{

    tabIP.Height = 30;
    tabDue.Height = 35;
    tabComplete.Height = 30;
    tabCancel.Height = 30;

    btnComplete.IsEnabled = false;
    btnCancel.IsEnabled = false;

    DateColumn.Header = "Due Date";
    TimeColumn.Header = "";
    PointsColumn.Header = "";

    LoadDataFromFile(@"..\..\Files\Due.txt");

}

private void tabComplete_Click(object sender, RoutedEventArgs e)
{
    tabIP.Height = 30;
```

```
            tabDue.Height = 30;
            tabComplete.Height = 35;
            tabCancel.Height = 30;

            btnComplete.IsEnabled = false;
            btnCancel.IsEnabled = false;

            DateColumn.Header = "Finished";
            TimeColumn.Header = "";
            PointsColumn.Header = "";

            LoadDataFromFile(@"..\..\Files\Completed.txt");
        }

        private void tabCancel_Click(object sender, RoutedEventArgs e)
        {
            tabIP.Height = 30;
            tabDue.Height = 30;
            tabComplete.Height = 30;
            tabCancel.Height = 35;

            btnComplete.IsEnabled = false;
            btnCancel.IsEnabled = false;

            DateColumn.Header = "Canceled";
            TimeColumn.Header = "";
            PointsColumn.Header = "";

            LoadDataFromFile(@"..\..\Files\Canceled.txt");
        }

        private void btnComplete_Click(object sender, RoutedEventArgs e)
        {
            StackComplete();
            LoadDataFromFile(@"..\..\Files\IP.txt");
        }

        private void btnCancel_Click(object sender, RoutedEventArgs e)
        {
            StackCancel();
            LoadDataFromFile(@"..\..\Files\IP.txt");
        }

        private void btnAdd_Click(object sender, RoutedEventArgs e)
        {
            EnqueueTask();
            ReorderFile(@"..\..\Files\IP.txt");
            LoadDataFromFile(@"..\..\Files\IP.txt");
        }
    }
}
```

APPENDIX

A. References

[1] Dhaliwal, H., Ahluwalia, K., Zada, D. K., Qin, D., Tanveer, R., Botros, J., & Xu, J. (2021, August 24). *An analysis of productivity app strengths: An environmental scan*. McMaster University, Hamilton, ON, Canada. Lakehead University, Thunder Bay, ON, Canada

[2] *Mobile App Download & Usage Report 2019: Stats You Must Know.* (n.d.) Retrieved August 26, 2021, from https://www.goodfirms.co/resources/app-download-usage-statistics-to-know

[3] Delmonte, M. M. (1984). Meditation Practice as Related to Occupational Stress, Health and Productivity. *Perceptual and Motor Skills*, 59(2), 581–582. https://doi.org/10.2466/pms.1984.59.2.58

[4] Lindsay, E. K., Young, S., Smyth, J. M., Brown, K. W., & Creswell, J. D. (2018). Acceptance lowers stress reactivity: Dismantling mindfulness training in a randomized controlled trial. *Psychoneuroendocrinology*, 87, 63–73. https://doi.org/10.1016/j.psyneuen.2017.09.015

B. Curriculum Vitae

AZRIEL PETER DEDUYO

About Me

I am a highly motivated and versatile professional with a passion for technology and a diverse skill set. With a solid foundation in SEO, programming languages, and graphic design, I seamlessly bridge the technical and creative realms.

Education

Mapúa Malayan Colleges Mindanao

- Bachelor of Science in Computer Science
- Expected graduation date 2026

Mapúa Malayan Colleges Mindanao

- Senior High School, TVL Strand (ICT-Programming)
- Completed in 2020

Contact

+63 920-683-7065

azriel.deduyo@gmail.com

Block 3 Lot 9 Buttercup St., Granville II, Catalunan Pequeno, Davao City

Expert Skills

- Graphic Design
- Programming Skills
- Off-page SEO

Work Experience

Graphic Designer/Photo Editor

2018-2023 | Freelance

- Has edited designs for T-shirts, jerseys, business logos and PowerPoint presentations for clients.
- Enhanced and restored photos from old pictures and has edited photos for posters.

SEO Analyst

April 2020-June 2023 | Agile Services Group

- Hosting/Joining weekly meetings for team updates and collaboration for ideas and news.
- Daily use of SEO tools such as Google Analytics, Moz, Ahrefs and Buzzstream. and stay updated with the SEO trends and E-mail marketing.

Content Publisher

October 2023 | Agile Services Group

- Published content to blogs and review pages using WordPress with Elementor extension.

Education Background

Mapúa Malayan Colleges Mindanao (2020 - present)

- Graduated Senior High School (ICT - Programming) with First Honors

Experience

Mapua Malayan Colleges Mindanao
Student Intern, May 2022

MMG IT Hub & Supplies
Part-time clerk, 2014-2018

Awards & Honors

- ·Computer Assisted Learning (CAL) Excellence in ICT Education Award
- ·National Movement of Young Legislations Academic Excellence Leadership Award

Hobbies & Interests

- Animation
- Reading
- UX Design

Character References

Martzel Baste
College Professor, Mapúa MCM
mbaste@mcm.edu.ph

Stacey Andrew GONZAGA

Personal Profile

A driven professional—I am a natural problem-solver and strategic thinker who is committed to achieving success through diligence, resilience, and thoroughness.

Skills

Transferable

- An ability to apply learnt theories in developing more efficient solutions.

Specialized

- Intermediate mastery of Java, C#, and Python in developing software systems.
- Proficiency with the use of Microsoft Office tools.

Artistic

- Intermediate mastery of Adobe Photoshop & Illustrator.
- Proficiency in designing and prototyping in Figma.

You can reach me at

Catalunan Grande, Davao City,
Davao del sur, PHL
TM / WhatsApp
(+63) 9979548196
staceyandrewgonzaga@gmail.com

C. Photo of the Data Gathering