**Mini Excel**



**Session 2023-2027**

**Supervised By:**

Sir Nazeef ul Haq

**Submitted By:**

Muhammad Tayyab Amir (2023-CS-101)

Muhammad Harris Amin (2023-CS-160)

**University of Engineering and Technology, Lahore**

1. **Project Overview:**
   1. **Description**

We all have used excel for some data work and it’s a very versatile and useful application. The Mini Excel project is a web-based spreadsheet application that simulates the core functionalities of popular spreadsheet software like Microsoft Excel or Google Sheets. This project aims to provide users with an interactive platform for data manipulation, computation, and organization. The development process mainly focuses on using different data structures and algorithms to enhance the functionalities of the project. By implementing these concepts, the project performs efficient computations and ensures dynamic updates in the interconnected cells.

* 1. **Business case**

In today’s data-driven world, spreadsheets have become an essential tool for organizing, analyzing, and presenting information across various domains such as finance, education, research, and business management. While commercial spreadsheet applications like Microsoft Excel and Google Sheets dominate the market, they often come with limitations such as high costs, excessive complexity. The mini excel is a good solution for these limitations.

* 1. **Motivation**

The Mini Excel project was assigned by our instructor as an opportunity to apply theoretical knowledge of data structures and algorithms in a practical setting. The goal is to understand how real-world applications depends on computational efficiency and logical structuring to handle data.

* 1. **Features:**

1. Interactive user interface
2. Basic arithmetic operations
3. Dynamic updates
4. Error handling
5. Data structures implementation
6. Cell formatting (bold, italic, underline, alignment)
7. Functional dependencies
8. Prevents circular dependencies
9. Efficient data searching in grid
10. Undo/Redo
    1. **Technology Stack:**

Table 1. Technology Stack

|  |  |
| --- | --- |
| Front End | HTML, CSS, Javascript |
| Version control | Git |
| Development Environment | Visual Studio Code |
| Testing Tools | Console.log() and Chrome |

1. **Use Cases**
   1. **Main Screen**

Table 2. Main Screen

|  |  |
| --- | --- |
| Use Case Id | U01 |
| Name | Main Screen |
| Actor | User |
| Description | It displays the spreadsheet on the screen |
| UI | Fig1. Main Screen |

* 1. **Writing data**

Table 3. Writing data

|  |  |  |
| --- | --- | --- |
| Use Case Id | | U02 |
| Name | | Writing data |
| Action | | User |
| Description | | It provides access to write data in cells |
| UI | Fig. 2. Writing data in cells | |

* 1. **Undo Redo**

Table 4. Undo Redo

|  |  |
| --- | --- |
| Use Case Id | U03 |
| Name | Undo Redo |
| Actor | User |
| Description | The user can perform undo and redo by using ctrl+z and ctrl + y respectively |
| UI | Fig 3. Simply data writing    Fig 4. After undo    Fig 5. After Redo |

* 1. **Functionalities**

Table 5. Functionalities

|  |  |
| --- | --- |
| Use Case Id | U04 |
| Name | Functionalities |
| Actor | User |
| Description | The user can perform several mathematical calculations on the data |
| UI | Fig 6. Applying formula    Fig 7. SUM result    Fig 8. By operators    Fig. 9. Arithmetic result    Fig 11. Preventing circular dependencies    Fig. 10. Reevaluation of dependencies |

* 1. **Find functionality**

Table 6. Find functionality

|  |  |
| --- | --- |
| Use Case Id | U05 |
| Name | Find functionality |
| Actor | User |
| Description | Find data in the spreadsheet |
| UI | Fig. 12. Searching |

1. **Implementation**
   1. **Data Structures**
      1. **Linked Lists**

Linked lists are utilized to represent the grid, providing a dynamic and flexible way to manage the cells. Each cell in the grid is represented as a node in the linked list, where the node contains the cell’s data and links to its adjacent cells. Specifically, every cell is connected to the cell on its left, right, above, and below, enabling easy manipulation and traversal of the grid. This structure allows for efficient row and column navigation, making it easy to access and modify the value of any cell based on its row and column number. Using a linked list helps in maintaining the grid's data efficiently, ensuring quick updates and smooth interaction between cells.

* + 1. **Graphs**

A Directed Acyclic Graph (DAG) is employed in the project to model and manage cell dependencies, which is crucial for ensuring accurate calculations and preventing errors in formulas. When a cell’s value depends on other cells, such as in a formula like =A1+B1, a directed edge is created from each of the dependent cells (A1 and B1) to the calculated cell. This graph structure helps track and manage these dependencies, ensuring that the necessary calculations occur in the correct order. The use of a directed acyclic graph is particularly effective because it prevents the creation of circular dependencies, where a cell might depend on itself either directly or indirectly.

* + 1. **Stacks**

Stacks are used extensively throughout the project, particularly for implementing undo and redo functionalities, which enhance the user experience by providing the ability to revert changes and restore previous states. Every time a user performs an action—such as editing a cell's value, changing its formatting, or making other modifications—the action is pushed onto an undo stack. This stack operates based on the Last In First Out (LIFO) principle, meaning that the most recent action can be undone first. When an action is undone, it is moved to the redo stack, where it can be reapplied if the user decides to restore it. This dual stack structure makes it easy to navigate backward and forward through changes, allowing users to manage their edits effectively. Additionally, stacks are also used in methods that involve calculations, such as those using postfix notation, to simplify and organize the sequence of operations, ensuring that the calculations are performed in the correct order.

* + 1. **Queues**

Queues follow the first in first out and this feature of queues is proved useful in techniques like topological sort which is used as detecting cycles in the Directed Acyclic Graph to prevent the happenings of circular dependencies.

* + 1. **Binary Search Trees**

Binary Search Trees (BST) are used in the project to optimize the process of searching for data within the grid. BSTs are a highly efficient data structure for searching, insertion, and deletion operations, offering much better time complexity of O(log N) in best case than linear data structures like arrays or linked lists. In the context of the mini Excel application, the data entered into cells is stored within binary search trees, which allows for quick searching of specific values within the grid. By organizing the data in a hierarchical structure, BSTs enable faster access and manipulation, especially when dealing with large datasets.

* 1. **Initialization:**
     1. **Creation of grid**

The grid structure in the mini Excel application is initialized by creating nodes one at a time and systematically inserting them into the linked list. Each node represents a cell in the grid and is connected to its neighboring nodes to establish a well-defined structure. The process involves iterating through all rows and columns, and for each cell located at position (i,j), a new node is dynamically created and inserted into the ternary linked list using the insertNode(i, j) method. This method ensures that the nodes are appropriately linked, maintaining connections to their left, right, upper, and lower neighbors wherever applicable.

* + 1. **Bidirectional link**

Each node in a linked list is connected to its dom element (td element) and each td element (cell) is connected back to the node of the linked list. This ensures easy access to the cells and nodes and makes updates in the cells easy. The appropriate event listeners are also added for the cells to enable the data to be written inside the cells.

* + 1. **Ribbon**

The ribbon is a crucial component of the mini Excel application, designed to provide users with a convenient and accessible toolbar for managing and formatting their data. Located at the top of the window, the ribbon offers a wide range of functionalities that simplify the process of organizing and decorating data within the spreadsheet. Users can easily apply formatting options like bold, italic, and underline to text, change font sizes to improve readability, and manage the layout by inserting new rows and columns as needed.

* + 1. **Initialization of objects**

At the start of the application, several critical objects are initialized to provide the foundational functionality of the mini Excel application. These include objects for the dependency graph, binary search trees, and undo and redo stacks. The dependency graph, represented as a Directed Acyclic Graph (DAG), is created to manage cell dependencies, ensuring that formulas referencing other cells are calculated correctly and without circular dependencies. Binary search trees are initialized to store and retrieve data efficiently, particularly for search operations within the grid. The undo and redo stacks are set up to record user actions, allowing for the implementation of reversible operations.

* 1. **Working**
     1. **Cell Selection and navigation**

Each cell in the grid can be selected by clicking on it. The selected cell is highlighted, making it easy for users to identify which cell they are working on. Keyboard navigation is enabled through the use of arrow keys (Arrow Up, Arrow Down, Arrow Left, Arrow Right), allowing users to move between cells efficiently without requiring a mouse. The interaction is supported by a ternary linked list where each node represents a cell. This linked structure ensures that neighboring cells are directly accessible, enabling smooth traversal and updates.

* + 1. **Cell Editing**

Upon selecting a cell, users can input data or formulas directly. If the cell does not already contain an input element, one is dynamically created to allow for inline editing. This approach prevents unnecessary DOM elements and ensures a responsive interface. When a cell is clicked or navigated to via the keyboard, it becomes highlighted. This visual representation of the cell helps having understanding of whatever is happening within the grid. The focused cell can also be edited immediately, ensuring smooth interaction without extra clicks.

* + 1. **Undo and Redo**

Each state of the cell is being saved in the undo stack as the focus on the cell is removed. The undo and redo functionality is triggered by Ctrl+z and Ctrl+y respectively. Whenever an undoAction() is triggered. The state of the cell is popped from the undo stack and pushed to the redo stack and the cell then represents the state that was popped from the undo stack. Similarly, whenever redo event is triggered the state of the cell is popped from the redo stack and pushed to the undo stack and the cell then represents the state that was popped from the redo stack.

* + 1. **Formula functionalities**

The user can perform formula functionalities by simply using functions like SUM, MUL, MAX, MIN followed by an equal (=) sign and the two cells should be passed as arguments to these functions. If both cells are separated by colon (:) in the argument, then it will perform calculations on the range of cells between these two cells including both of those cells. If both cells are separated by a comma (,) then the calculation will be performed only those 2 cells. It can also perform simple mathematical operations like =23\*31 or =A1/9. If a cell is dependent on other cells then updating the dependent cell will also change the value present in the cell.

* + 1. **Find Functionality**

Whenever the focus on the cell is removed. The binary search tree is updated. You can find in the grid by using CTRL+F and then typing in the input box. The cell with specified input will be highlighted.

* + 1. **Insert rows and columns**

In the insert tab, new rows and columns can be inserted by simply clicking on the insert row and insert column respectively.

* + 1. **Other Functionalities**

In the toolbar, there are functions like bold, italic, underline, left align, center align and right align. You can use them either by clicking on them or by using shortcut keys for bold, italic and underline. (CTRL+I) for italic, (CTRL+B) for bold, (CTRL+Q) for underline.