Item	Priority	Value
Storage for data that is handled in rest requests	High	MVP - Decide our server implementation and it store our spreadsheets in JSON format assigned to the publisher who made them, additionally knowing who can add and request that spreadsheets JSON data
What rest API will we end up using	High	MVP - Deciding what REST api we will use and start implementation
Login/Create Account GUI	High	MVP - GUI implementation of create account or login
Homepage on startup, with options available on the GUI	High	MVP - Homepage for user to select to: log in/create an account, create new spreadsheets access past saved Spreadsheets
Build rest requests	High	MVP - Adjust the rest requests format so that it correctly communicates with the server
Rest protocol deals with the login/create account requests	High	MVP - API communicates with the server to request a login or register a new account to the server and fetches data of the user
Create account page	High	MVP - Having an account allows the user to securely store and access their spreadsheets across multiple sessions.
Create New Spreadsheet	High	MVP - Creates a spreadsheet filled with blank cells and is untitled - default sized column for each letter, 100 rows.
Delete data in individual cell	High	MVP - User has the option to select a cell and delete the contents

Serialize spreadsheet data into different data format for storage	High	MVP - Transform the data before using the API to send it to the server, changing it into a JSON format
Options on GUI - NonFunctional	High	MVP - Adding Save File, Return Home, and Download file options on GUI without functionality
Save File implementation	High	MVP - Necessary to allow the user to save their work to adjust later. Save the sheet with the publisher's name to the server with API.
Return home implementation	High	MVP - Users must have the option to return to the homepage while editing a spreadsheet, to open up another file, log out, etc.
Cell referencing operations	High	MVP- Allows user to reference cells by their address (eg: $A1) and update the values within them.
Formula evaluations	High	MVP - User will be able to apply arithmetic operations: '+' | '-' | '*' | '/' | '<' | '>' | '=' | '<>' | '&' | '|' | ':' to and within individual cells.
Download File implementation	High	MVP - Capability to download the files to the local system in different formats - .CSV, .pdf
Functional cell operations	Medium	Desirable - Users will be able to access and apply built in functions to individual cell values. This should include the functions `'IF' | 'SUM' | 'MIN' | 'AVG' | 'MAX' | 'CONCAT' | 'DEBUG'`
Modifiable number of rows and columns	Medium	Desirable - User can select amount of columns + rows to be used in the spreadsheet Specify number of rows/columns, max being 100x100

Able to add ASCII characters into cells	Medium	Desirable - Allows a person to add text to cells that is included in the ASCII ‘alphabet’
Cut, Copy, and Paste	Medium	Desirable - Ability to copy, paste, and cut multiple cells at once for efficient cell movement
Resizing columns and rows	Medium	Desirable - Allows the user to adjust the column and row size, allowing longer character sequences to be displayed without being cut-off.
Cell view line	Medium	Desirable - Allows users to view cell contents and manipulate individual cell data.
Updating Sheets (Subscriber<->Publisher)	Medium	Desirable - Requests the sheet data from the database and updates your own sheet with this.
Keyboard Shortcuts	Medium	Desirable - Shortcuts for Save, Print, Undo, Redo, and Delete
Concurrent Editing	Medium	Desirable - Allow multiple users to edit the same spreadsheet concurrently in a similar style to github, frontend GUI implementation of pushing to the server.
Concurrent Editing Server Side	Medium	Communicating with the database sending API requests to fetch and push these changes to the server before they get accepted to the host’s main file. Not allowing for conflict, taking requests the order they came in for approval.
Undo	Medium	Desirable - Allow the user to restore the previous spreadsheet state up to 5

		states back, cached on client-side
Redo	Medium	Desirable - After undoing, hold x previous states reverted, allowing the user to cancel their undo's (up to 5 as well), cached on client-side.
Auto-save	Low	Bonus - Periodically save the user's current spreadsheet state. Allows the user to restore a previous save. Every *5 min send API request to server with current state of spreadsheet.
Formatting Toolbar	Low	Bonus - Quickly adjust font size, color, undo, redo, print, symbols and centering
Styling/formatting options	Low	Bonus - Allow the user to change font size/color, highlight cells, change cell border colors.
Restoring Previous Version	Low	Bonus - Change all cells back to a saved version of the spreadsheet
Restoring Previous Version Request	Low	Bonus - Request previous save state from server with an API call
Dynamic Filling	Low	Bonus - Allow the user to drag and apply calculations to multiple columns and rows
Change UI Theme	Low	Bonus - Allow for UI theme adjustment to light, dark, or system mode, this is a local change

User Stories:

Storage for data that is handled in rest requests - Users want their data to be stored on a machine non locally so they can access it on a different device.

Acceptance Criteria (conditions of satisfaction):

1. The application securely stores user data on a remote server accessible from different devices.
2. Users can access their stored data from any device.
3. Data integrity is maintained during storage and retrieval processes.

Formula evaluations - user wants to do math with cell b1 and b3 adding these two values together to get profit

Acceptance Criteria (conditions of satisfaction):

1. The user is able to input their desired formula.
2. If given a valid formula, evaluate and display the correct result.

What rest API will we end up using - As a developer, knowing the best API that is most up to date to use will help drastically reduce the difficulty of adjusting it for our spreadsheet.

Acceptance Criteria (conditions of satisfaction):

1. The application implements a rest API for handling data storage and retrieval.
2. The API offers strong documentation and support for easy integration and maintenance.
3. The API's performance and security features align with the requirements of the application.

Login/Create Account GUI - A user wants the login and creating account page to be seamless and easily visible so that they can get to working on the spreadsheet as quick as possible

Acceptance Criteria (conditions of satisfaction):

1. The login and account creation pages are designed well and accessible within the application.
2. Users can navigate between the login and account creation interfaces.
3. Clear instructions and error messages are provided to guide users through the login or account creation process.

Homepage on startup, with options available on the GUI - a user opens up the web app, they would like to be able to easily spot how to create access a previous spreadsheet and login quickly without having to jump through hoops, they came for the spreadsheet not the homepage

Acceptance Criteria (conditions of satisfaction):

1. After launching the app, users are directed to a homepage displaying relevant options and actions.
2. The homepage has options to create a new spreadsheet, access previous documents, and log in/create account.
3. Navigation elements allow users to quickly transition from the homepage to their desired tasks.

Build rest requests - as a User, a well developed rest request though out of sight will help the information get from the server to me quicker, and the closer to instantaneous and the less time it needs to load in the better.

Acceptance Criteria (conditions of satisfaction):

1. Efficient REST Requests: The system should send and receive REST requests with minimal latency.
2. Performance: Response times should be near-instantaneous, ideally under 200ms.

Rest protocol deals with the login/create account requests - As a user, having my information be stored in a server and can be pulled by a request is great because it would allow me to keep access to previous spreadsheets without saving and have my login information stored safely somewhere else.

Acceptance Criteria (conditions of satisfaction):

1. Secure Storage: User information should be securely stored on the server.
2. Accurate Retrieval: User information should be accurately retrieved upon login.
3. Data Consistency: The system should ensure data consistency across sessions.

Delete Data in individual cell - Users want to be able to delete data previously added to a cell if data has changed.

Acceptance Criteria (conditions of satisfaction):

1. Clear Deletion: Users should be able to delete cell data with a single action.
2. Immediate Update: The cell should reflect the deletion immediately.

Serialize spreadsheet data into different data format for storage - User wants a seamless experience when inputting and accessing the spreadsheet data

Acceptance Criteria (conditions of satisfaction):

1. Data Serialization: The system should support serialization of spreadsheet data into multiple formats.
2. Accurate Conversion: Data should be accurately converted without loss.
3. Easy Access: Users should be able to easily access and export the data in different formats.

Options on GUI - NonFunctional - User needs an intuitive display of sheet functionality to access

Acceptance Criteria (conditions of satisfaction):

1. Intuitive Design: The GUI should be user-friendly and easy to navigate.
2. Clear Labels: All options and functionalities should be clearly labeled.
3. Accessibility: The GUI should be accessible to users with disabilities.

Save file implementation - Users want to be able to save their spreadsheet data on the server so that they can access it in the future when re-opening the application. They also want the option to access it from different devices.

Acceptance Criteria (conditions of satisfaction):

1. Server Storage: The system should save spreadsheet data on the server.
2. Cross-Device Access: Users should be able to access their data from different devices.
3. Auto-Save: The system should periodically auto-save to prevent data loss.

Return home implementation - User wants to return to home screen to create a different spreadsheet for different datasets while still saving their current sheet

Acceptance Criteria (conditions of satisfaction):

1. Easy Navigation: Users should be able to return to the home screen with a single action.
2. Save Current Sheet: The system should save the current sheet before navigating to the home screen.
3. Multiple Sheets: Users should be able to create and manage multiple spreadsheets.

Functional cell operations - Users want to perform logical and arithmetic operations on their numerical inputs to help analyze and break down data.

Acceptance Criteria (conditions of satisfaction):

1. When user uses the symbol ‘=’ in the cell, any numerical input proceeding will result in a mathematical simplification.
2. Error handling: if user doesn’t use appropriate arithmetic expression format and symbols proceeding ‘=’ symbol, #NAME? error will be displayed.
3. Cell address reference: user can reference cell address (if it holds numerical value) to replace any manual numerical input in operation.

4. Cell selection: user can select cells holding numerical value to perform operations on and replace manual numerical input.

Cell referencing operations - User wants to reference a cell by its address for ease of data access and entering formulas, especially if the cell is not in the direct view in the GUI.

Acceptance Criteria (conditions of satisfaction):

1. Precondition: can only be referenced within cells and cell view line
2. Reference is obtained using a three part string reference, $. A . 1
3. Error handling: if user does not use appropriate reference, message #ERROR! Will be displayed in cell

---

Download File implementation: As a user, I would like to be able to easily download my spreadsheet into a shareable file, such as a CSV.

Acceptance Criteria (conditions of satisfaction):

1. User should have the option to voluntarily save the spreadsheet locally.
2. User should have options to save as different file types
3. User should be able to choose save location
4. Error handling: user should be presented with an error message if save is unsuccessful.

Modifiable number of rows and columns: As a user, I would like to be able to alter the number of rows and columns in my spreadsheet, to compensate for having a larger data set.

Acceptance Criteria (conditions of satisfaction):

1. Pop-up: user should be able to enter preferred spreadsheet size into allocated spaces on GUI
2. Limit to maximum size is 26 columns (A-Z) and 100 rows
3. If user exceeds maximum bounds, error message will be displayed, stating the limits of the spreadsheet size.

Able to add ASCII characters into cells: As a user, I would like to be able to enter a standardized text format into cells in the spreadsheet.

Acceptance Criteria (conditions of satisfaction):

1. Format: unless ‘=’ symbol is added at the beginning of user input, ascii characters will not be considered
2. Specific function should be called to derive ascii value
3. Error message: If invalid input value is entered and used in function, #VALUE! message will be displayed in cell

---

Cut, Copy, and Paste: As a user, I would like to be able to perform cut, copy, and paste

operations on multiple cells, columns, and/or rows.

Acceptance Criteria (conditions of satisfaction):

1. Preconditions: cut/copy/paste is implemented on UI
2. When user selects cell/ highlights cell value, they should be able to either cut or copy the value.
3. When user selects a cell, they should be able to paste a copied value
4. If user tries to paste and no value has been copied, no action should be taken

---

Resizing columns and rows: As a user, when I enter longer strings or numbers into my cells, I need the ability to resize the column/row to be larger so the text does not get cut off.

Acceptance Criteria (conditions of satisfaction):

1. Cell length and width should be customizable if user drags cell margins
2. Any long text that previously couldnt be displayed in a smaller version of a cell should be correspondingly more visible if cell is made larger and vice versa

---

Cell view line: As a user, when selecting a cell I need a clear text box to perform functional operations on it.

Acceptance Criteria (conditions of satisfaction):

1. Clear Input Box: The selected cell should display a text box for easy editing.
2. Functional Operations: Users should be able to perform operations directly in the text box.
3. Real-Time Update: Changes should be reflected in the cell immediately.
4. Error Handling: The text box should indicate if an operation is invalid.

Updating Sheets (Subscriber<->Publisher): As a user, when working with other collaborators on a spreadsheet, I need access to the most recently updated version from my team members to avoid merge conflicts when working concurrently.

Acceptance Criteria (conditions of satisfaction):

1. Real-Time Updates: The spreadsheet should display the latest version from all collaborators.
2. Conflict Avoidance: The application should handle and resolve merge conflicts automatically.
3. Notification: Users should be notified of updates from collaborators.
4. Error Handling: An error message should appear if updates fail.

Concurrent Editing: As a user that works within a team, I would like to collaborate with my team members on a single spreadsheet and have synchronized updates.

Acceptance Criteria (conditions of satisfaction):

1. Real-Time Collaboration: Multiple users should be able to edit the spreadsheet simultaneously.
2. Synchronized Updates: Changes made by any user should be immediately visible to others.
3. Conflict Resolution: The application should manage concurrent edits without data loss.
4. User Notification: Users should be notified of any concurrent editing issues.

Auto-save: As a user, I would like to have my edits be saved automatically to avoid losing my work if I forget to manually save before leaving the application.

Acceptance Criteria (conditions of satisfaction):

1. Periodic Saving: The application should automatically save changes at regular intervals.
2. Frequency Settings: Users should be able to adjust the auto-save frequency.
3. Save Confirmation: A visual indicator should confirm that changes have been saved.
4. Error Handling: Users should be notified if auto-save fails.

Undo: As a user, I would like to be able to easily undo my last edit with the click of a button.

Acceptance Criteria (conditions of satisfaction):

1. Undo Functionality: The application should provide an undo button to revert the last change.
2. Multiple Undo: Users should be able to undo multiple steps sequentially.
3. Visual Feedback: The application should visually confirm when an undo is successful.
4. Error Handling: Users should be notified if the undo action fails.

Redo: As a user, I would like to be able to overwrite an undo with the click of a button, returning to a more recent version of my spreadsheet.

Acceptance Criteria (conditions of satisfaction):

1. Redo Functionality: The application should provide a redo button to restore the last undone change.
2. Multiple Redo: Users should be able to redo multiple steps sequentially.
3. Visual Feedback: The application should visually confirm when a redo is successful.
4. Error Handling: Users should be notified if the redo action fails.

Dynamic Filling: As a user, I would like to be recommended auto completion options when editing my spreadsheet to save time with redundant tasks and improve productivity.

Acceptance Criteria (conditions of satisfaction):

1. Auto-Completion: The application should suggest auto-completion options based on existing data.
2. User Selection: Users should be able to accept or reject auto-completion suggestions.
3. Context Awareness: Auto-completion should be contextually relevant to the data being entered.
4. Error Handling: Users should be notified if auto-completion suggestions are unavailable.

Change UI Theme: As a user, I would like to have the option to switch to a dark mode as opposed to being restricted to a single user interface theme.

Acceptance Criteria (conditions of satisfaction):

1. Theme Selection: The application should provide an option to switch between light and dark mode.
2. User Preference: The selected theme should be saved and applied automatically on subsequent logins.
3. Visual Consistency: All UI elements should be consistently themed according to the selected mode.
4. Accessibility: The application should ensure readability and accessibility in both light and dark modes.

Keyboard Shortcuts - As a user, I want to use keyboard shortcuts for Save, Undo, and Redo, so I can perform these actions quickly and efficiently without needing to use the mouse or navigate through menus.

Acceptance Criteria (conditions of satisfaction):

1. Functional shortcuts: the application should allow users to execute standard keyboard shortcuts, namely save, undo and redo

2. Feedback: the application should display a notification indicating that executed shortcut has been completed successfully
3. Error handling: if the action fails, the application should display a notification indicating that the shortcut action failed

---

Concurrent Editing Server Side - As a user, I want the server to handle concurrent edits by sending API requests to fetch and push changes to the server before they get accepted into the host's main file. This should be done in the order the requests are received to prevent conflicts and ensure data consistency.

Acceptance Criteria (conditions of satisfaction):

1. Push spreadsheet updates: the application allows users to submit their edits to the spreadsheet for review to the owner
2. Review requested updates: the owner of the spreadsheet should be able to review requested changes from other members collaborating on the same spreadsheet
3. Implement changes: the owner of the spreadsheet can approve changes and changes should be seen across all accounts
4. Error Handling: if the process of requesting and approving updates fails, the application should display a notification detailing the failed step to the user

---

Formatting Toolbar (Visual/FrontEnd) - As a user, I want a visually apparent toolbar that I can easily spot to adjust my cell formatting; something clean and apparent but not completely in my way.

Acceptance Criteria (conditions of satisfaction):

1. Toolbar options: the formatting toolbar has options to adjust font size, color, undo, redo, print, insert symbols, and center text
2. Toolbar positioning: the application displays the toolbar at the top of the window for easy access, above where the user will be able to edit the spreadsheet

---

Styling/formatting Options - As a user, I want to change the font size and color, highlight cells, and change cell border colors, so I can customize the appearance of my spreadsheet to make it more readable and visually appealing.

Acceptance Criteria (conditions of satisfaction):

1. Functionality: the application shows the changes happening in real time when formatting settings in the toolbar are adjusted
2. Error handling: the application displays a notification detailing the step that failed to the user

---

Restoring Previous Versions (Visual/FrontEnd) - As a user, I want to restore all cells to a previously saved version of the spreadsheet, so I can recover from mistakes or changes that I don't want to keep.

Acceptance Criteria (conditions of satisfaction):

1. Working history: previous saved versions of the spreadsheet (3) are stored on the server to be restored if needed
2. Error handling: if spreadsheet version fails to save on the server, display a notification for the user to indicate that the previous version will not be available for restoration

---

Restoring Previous Version Request - As a user, I want to request a previous save state from the server with an API call, so I can easily revert to an earlier version of my spreadsheet.

Acceptance Criteria (conditions of satisfaction):

1. Restore option: the application displays a restoration option in the toolbar
2. Display working history: the application displays a dropdown menu consisting of previous versions for the user to pick from
3. Error handling: if there is an issue with selecting a previous version, the application displays a notification that details the issue to the user

---