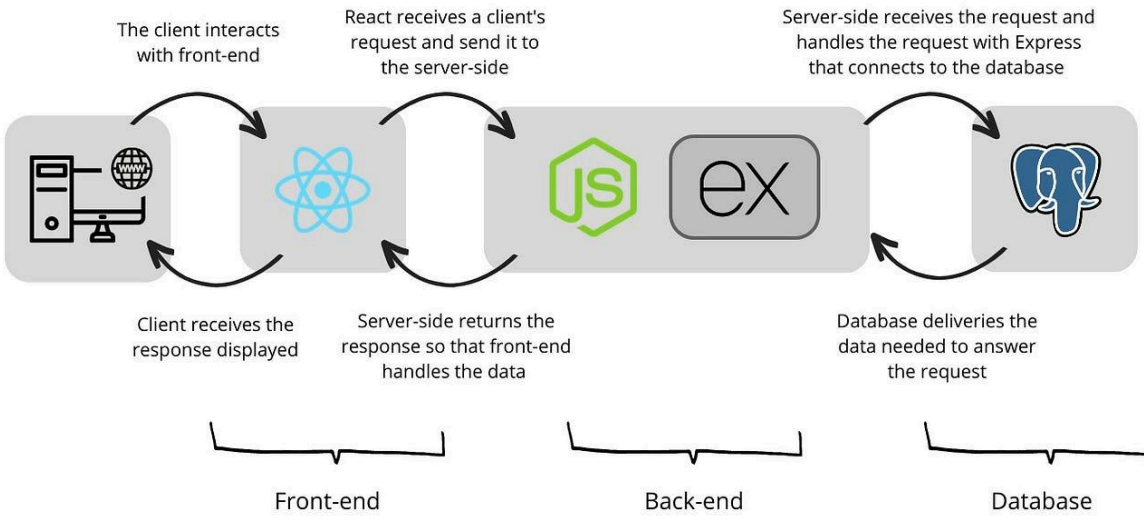
Team Members:
Brandon Petersen, Emily Fink, Shaan Patel, Shubham Yadav

Statement of Work:

We will be developing Husksheets, a distributive collaborative spreadsheet application that allows users to store and modify data in an intuitive environment. To protect spreadsheet data and server security, users will be required to login before viewing/editing their sheets. At the most basic level, users will have access to spreadsheet functionality such as creating, opening, editing, saving, and sharing documents. Within the spreadsheet, data will be stored in 'cells' which are labeled rows (by numerical value 1,2,3, etc..) and columns (labeled alphabetically A,B,C,etc…). Basic arithmetic operations and boolean logic will be implemented into the spreadsheet functionality. Spreadsheets will be stored in a database on the server for persistent changes when clients edit a document. Users will be able to share their work with others to collaborate on projects.

Husksheet Stack:
Everything will be coded in Typescript with an Express.JS backend, React front end, and PostgreSQL database



#	Item	Priority	Value
Minimum Viable Product			
1	Spreadsheet user interface of	VERY	Without the basic spreadsheet UI

	cells labeled alphabetically by column and numbered by rows	HIGH	we have no project to build on, so it should be the highest of priorities
1a	Cell combining. Bring elements together	VERY HIGH	Combining Cells to a 2d array, and
1b	Spreadsheet entity and cell entity for backend.	VERY HIGH	Entities are used in API's to communicate data between a database and a user
2	Server register client functionality	VERY HIGH	We must be able to register users to track on the server and connect with spreadsheets.
2a	Register endpoint connecting user entity with service methods	HIGH	The register endpoint will be used by the client for API
2b	getPublishers endpoint and service methods	HIGH	This endpoint returns a list of all registered publishers
3	Ability to create, open, edit, and delete spreadsheets	HIGH	Managing multiple spreadsheets will be an essential part of this application
3a	Front end file taskbar that implements create, edit, and delete	HIGH	Users interaction with the backend functionality
3b	Data structure to store representations of a whole spreadsheet	HIGH	This data will be stored in the SQL database and will be the backend
3c	createSheet, deleteSheet, and getSheets endpoints	VERY HIGH	These backend endpoints are the API access points the front end will use
3d	Front end navigation menu to see which sheets are available to edit on the server	HIGH	A user may choose to work on any of their previous sheets and
4	Updating and saving cells on an open sheet	VERY HIGH	This database allows their server to store persistent instances of the spreadsheet
4a	Create Reference and Term data type for edit request	VERY HIGH	A reference and term are the two pieces of necessary information to update a cell on a server.
4b	Ability to see spreadsheet updates, backend endpoints: getUpdatesForSubscription,	HIGH	These API endpoints will be how the frontend accesses the changes on database

	getUpdatesForPublished		
4c	Save a spreadsheet with updated changes, backend endpoints: updateSubscription, updatePublished	HIGH	Save functionality on the backend is essential for operation of changing documents on the server
Desirables			
5a	Login UI	HIGH	Interactive login functionality for user
5d	User password hashed in database	VERY HIGH	Hashing user passwords protects their profile's security
6	Functions for neighboring cells	HIGH	Functions are used in a number of user stories where a user may want to calculate the values of cells interacting with each other or format
6a	Ability for spreadsheet to support and distinguish between different cell types (numerical, alphabetical, boolean, or mixed)	HIGH	For functions support the spreadsheet must be able to know what functions can be applied given the type of the cell
6b	Support for basic numerical functions like addition, subtraction, multiplication, division (+, - , *, / , <, >, =, <>, &, |, :)	HIGH	Mathematical functions allow for a range of user cases like balancing a cost sheet or calculating taxes.
6c	Support for additional boolean operations: IF, NOT	HIGH	Boolean logic allows for more dynamic interactions between cells
7	Copy, Cut, and Paste functionality for single cells	Fair	Quickens the workflow of a user that has to move information from cells to others
8a	Multiple users can edit the same spreadsheet	HIGH	This is supposed to be a collaborative app so multiple users should be able to work on the same sheet
9	Support for expression grammar E ::= E Op E | '(' E ')' | Fun '(' [E [',' E]*] ')' | ['+'|'-'] <int> | <string> | Ref	HIGH	Formulas are written with expression grammar
Bonus			
5c	Logout Button	HIGH	To Logout of the application

14	Download the sheet	LOW	
18	Add/Delete Rows and Cols	FAIR	
Non-functional Requirements			
21	Opening a new spreadsheet takes under 1.5 seconds	HIGH	Poor performance when opening new sheets could frustrate potential users
22	Support for Spreadsheets up to 100x100 Cells, users can choose the number of cells	FAIR	Larger spreadsheets can support users with larger amounts of data requirements
23	Up to 5 users can be added as editors on a spreadsheet	FAIR	A normal team in a work environment could reasonably consists of around 5 people

Architecture

Backend:

Typescript: Core language of our application
Express: Connects backend to PostgreSQL database and facilitates server setup. Manages the life cycle of the program.

Frontend:

React: Web based and easy to work with. Need typescript implement.

Database:

SQL/PostgreSQL: Tables for Users, Sheets, Cells, and **Changes**.
EXAMPLE SCHEMA (Not fleshed out):

- Users: stores user_id username, password_hash
- Sheets: stores sheet_id, owner_id, title
- Cells: cell_id, sheet_id, cell_ref, value, formula
- Changes: change_id, sheet_id, user_id, timestamp, new_value

Testing:

Jest: Unit and integration tests.
React Testing Library: Front end testing

server side persistence

We store data in a database, our backend server handles messages to and from the database. Our front end clients talk to the server to append entries to a table and the server notifies the backend database and updates users that are currently also on that sheet.

User Stories

MVP:

1. **Spreadsheet user interface of cells labeled alphabetically by column and numbered by rows**

User Story:
As a user, I want the basic spreadsheet UI set up with correctly labeled columns and rows, so that when I want to edit the spreadsheets, I can, and the spreadsheet will be organized.

Acceptance Criteria:

- User will see a spreadsheet of size up to 25x25 cells, where at the top of each column, the spreadsheet will be labeled with a letter (“A” for the first column, “B” for the second column, and so on), and to the left side of each row, the spreadsheet will be labeled with the number corresponding to which numbered row it would be (“1” for first column, “2” for second column, and so on)

2. **Spreadsheet front end GUI**

User Story:
As a user, I want the spreadsheet front end GUI to be set up, so I will be able to interact with the spreadsheet when the different functionalities of the spreadsheet are implemented.

Acceptance Criteria:

-
- User will be able to click on Cells and buttons(add/delete row/col) icons and move their mouse around on the spreadsheet

3. **Spreadsheet cell functionality that allows users to select the contents of the cell**

User Story:
As a user, I want to be able to select the cells in the spreadsheet, so that I will be able to choose which cells I want to edit.

Acceptance Criteria:

- User will be able to click into a cell, whether it’s blank or already has data in it
- Selected cell is highlighted in the sheet on user’s screen

4. **Spreadsheet cell functionality that allows users to edit the contents of the cell**

User Story:
As a user, I want to be able to edit the contents of the cells in the spreadsheet, so that I will be able to perform actions on my data.

Acceptance Criteria:
- User can add, delete, or modify the data in that cell if they have write access
- When user edits contents of a cell, the resulting change is shown to the user in the cell that was edited
- When user edits contents of a cell, the data for that cell gets updated in the server and database to match what kind of edit user performed

5. **Ability to create, edit, and delete spreadsheets in front end**

User Story:
As a user, I want to be able to create, open, edit, and delete spreadsheets, so I can manage my data in multiple spreadsheets.

Acceptance Criteria:
- Users can open any sheet they have read/write access to. The selected sheet is displayed and edits can be made if write access is present.
- Sheets are tabular and you can move from one sheet to another by simply clicking on it on the top bar.
- Users can select to delete a sheet they created
- Users can select to create a new sheet and are prompted to name that new sheet

6. **Ability to create spreadsheets in the back end**

User Story:
As a user, I want to be able to create spreadsheets, so I can work on multiple spreadsheets.

Acceptance Criteria:
- When user decides to create a new sheet, a sheet is created with the publisher set to the person who created the sheet
- Name of the sheet is set to the name the user provides

7. **Ability to edit spreadsheets in the back end**

User Story:
As a user, I want to be able to edit my spreadsheets, so that I can add my data and manipulate it.

Acceptance Criteria:
- When user edits a sheet, the data for the cells that have been edited get updated in the server and database to match what kind of edit user performed

8. **Ability to delete spreadsheets in the back end**

User Story:
As a user, I want to be able to delete spreadsheets, so that I can get rid of spreadsheets that are no longer being used.

Acceptance Criteria:
- When users choose to delete a spreadsheet, the data for the sheet that matches the sheet name and publisher is cleared from server

9. **Front end file taskbar that implements create, and delete spreadsheets**

User Story:
As a user, I want there to be a bar of tabs, so that I can click between my sheets, be able to create more sheets and remove ones I don't want.
Acceptance Criteria:
- Users will see tabs that will have the options to switch between sheets. A plus button to create a sheet and a red "x" next to a sheet to delete a spreadsheet,

10. **All changes are synced**

User Story:
As a user, I don't want to have to remember to save the changes I made to the spreadsheet, so if I lose power or my laptop dies my progress can be saved and all collaborators have the latest version available.
Acceptance Criteria:
- User edits will be sent the to the server
- Edits will be handled as they arrive
- The current data for the spreadsheet on the server is overwritten with the new changes
- A visual confirmation is provided, confirming the sheet was saved
- When opening a sheet, the latest version is displayed

11. **Adding Collaborators**

User Story:
As a user, I want to be able to add collaborators to my spreadsheet, so I can work with peers on shared tasks.

Acceptance Criteria:
- User has an option to add a collaborator by username to their spreadsheet
- System checks if the username entered is a registered user, and adds the collaborator
- User receives feedback if adding the collaborator was successful, or if the user does not exist.

- User has view spreadsheets they have access to collaborate, can select a spreadsheet, and can implement savable changes

12. User Profiles

User Story:
As a user, I want to have my own user profile and other people to have their own profiles, so that we can distinguish between people so we know, and can specify, who has access to different spreadsheets.

Acceptance Criteria:
- A different user profile is made for each unique username used to login

13. Login UI

User Story:
As a user, I want to be able to login to my personal account, so I can securely access my spreadsheets.

Acceptance Criteria:
- User should be prompted with a login page with text fields to enter username and password or an option to register
- After entered credentials are matched to an existing account, user can see any existing spreadsheets related to the account
- If the username or password is incorrect it will prompt the user that the username and password did not match any of the existing user credentials.

14. Register UI

User Story:
As a user, I want to be able to register myself for an account, so that I'll be able to access the spreadsheets.

Acceptance Criteria:
-
- User should be prompted with a login page with text fields to enter username and password or an option to register
- When register option is chosen on login menu, user will be prompted to type their chosen username and password to be registered in two different text fields

15. Logout Button

User Story:

As a user, I want to be able to logout of my personal account, so that I don't constantly have it open on my device.

Acceptance Criteria:

- When user is logged in, there is a logout button in the top right corner, which when clicked, will log the user out of their account

16. Hashing of User Passwords

User Story:
As a system administrator, I want user passwords to be hashed before being stored in the database to protect client data and ensure they cannot be read by those accessing the database.

Acceptance Criteria:

-The implementation of a hashing algorithm(SHA-256) that ensures a plaintext password is never stored in the database.

-The user is prompted after account registration/creation that their account is successfully created. In order for the account to be created the hashed password must be stored.

17. Support for alphabetical operations: LOWER, UPPER, CONCAT

User Story:
As a user, I want to execute alphabetical operations on the data, so that I can easily correctly format the data inside a cell.

Acceptance Criteria:

- Users can apply alphabetical operations between cells. After changes to cells with operations are entered, the result is displayed

Desirables:

1. Copy, Cut, and Paste functionality for single cells

User Story:
As a user, I want to be able to copy the contents of a cell within a spreadsheet and paste it elsewhere within the same spreadsheet.

Acceptance Criteria:

-When a cell is selected via click, it is highlighted, and the option for the user to "copy" "cut" and "paste" is presented.

-If copy is selected, the contents of the cell are stored internally and a visual indication of the cell being copied is provided

-If cut is selected, the contents of the cell are stored internally and the cell is replaced with a blank cell

-If paste is selected, the contents of the cell are replaced with that stored in the internal clipboard. If the internal clipboard is empty, the cell is not changed.

2. **Error message if login failure**

User Story:
As a user, I want to be notified if my credentials are invalid or if there is any failure during user authentication

Acceptance Criteria:

- If either username or password fields are empty, the user is prompted to fill in all fields.

-Users entering invalid credentials that do not match with an existing account are prompted to retry with different credentials

-In the case of an error with backend “network or server issue” the user is notified and prompted to retry login.

-The option to register is always available on login page

3. **Additional range on sharing/permission levels among users**

User Story:
As a user, I want to be able to add collaborators to my spreadsheet with set permissions, so I can choose who can edit my spreadsheet and just view

Acceptance Criteria:

-Users allowing other user(s) to collaborate can set permissions (read only/edit). These permissions are made evident to respective collaborators

-Users with read only access cannot edit the sheet

4. **Error handling for incorrect data types for functions and formula inputs**

User Story:
As a user, I want to be notified if I enter a formula that is not supported, or I input incorrect data types into functions/formulas so I can avoid calculation errors and/or data corruption.

Acceptance Criteria:

-After a user has finished editing a cell, if a formula or function is deemed invalid(due to the formula/function not existing or incorrect data types), it is not executed, and the cell is placed in a visible error state.

-If the cell is edited to be valid, or if the cell is deleted, the error state is removed.

5. **Registration failure if username is a repeat**

User Story:
As a user, I want to have a unique username so that there is no confusion between me and another user when giving collaborator permissions.

Acceptance Criteria:
- If user tries to register with an already existing username, registration fails and they're prompted to register with a different username

6. **Support for Formulas (+, - , *, / , <, >, =, <>, &, |, :)**

User Story:
As a user, I want to execute formulas with arithmetic operations to compute data.

Acceptance Criteria:
- Users can apply arithmetic formulas within cells. After changes to cells with formulas are entered, the result is displayed
- If a user enters a operation that is not supported or a invalid formula notation the sell will say "NULL"

7. **Support for additional boolean operations: IF, SUM, MIN, MAX, NOT, AND, OR**

User Story:
As a user, I want to execute additional boolean operations on the data, so that I can interact with and summarize data between multiple cells.

Acceptance Criteria:
- Users can apply boolean operations between cells. After changes to cells with operations are entered, the result is displayed