# Emergency Calculator

Developers: Dylan Engle, Eric Bunch, Weston Abner

## Requirements

1. Front-End:
   1. User Interface
      1. Implemented with HTML5 and CSS, scripting with typescript.
      2. Canvas for two dimensional graphing.
         1. Canvas will be a 2d grid with points for position graphing.
         2. Will display as a grid with grid lines
            1. Will have an option to disable gridlines
         3. Will be created from a given function.
         4. Will feature default range for graph
            1. Allow zoom for graph features. Allows users to adjust range.

Zoom into the graph relative to the mouse using mouse button 3. (Scroll Wheel)

Drag zoom with mouse based on mouse position

Buttons that allow the user to zoom in and out from the graph. (Plus and Minus)

* + 1. Upon hitting the exit button, confirmation box will appear
       1. Box will prompt the user asking them if they want to save
          1. If no exit the website without saving
          2. If yes, ask if they want to save graph data, equations, or both

Both button will call both the file saver and the image exporter (3, 4)

Individual buttons will call whichever application is selected (image exporter or file saver)

* + 1. CUI (Calculator user Interface)
       1. Will feature a collapsible menu in order to bring up calculator buttons
          1. The button will input into the calculation form
          2. When menu button is clicked the menu will expand showing many new buttons for user, and can be collapsed

Buttons will highlight on mouseover

Buttons will be input into calculation form (1.1.5.

* + - * 1. Menu will contain buttons for most math functions

Basic math operators: addition, subtraction, multiplication, division

All Numbers: [0-9]

Trigonometric functions: sin, cos, tan

Exponential functions and roots

Exponents: x, y, z

Logarithmic functions: log, ln, log base

Important variables: e, pi

Constant Physics variables: Newtonian constant of gravitation, speed of light in vacuum, planck constant

Moving back and forth in an equation (similar to hitting the left and right buttons on a keyboard), for moving the cursor left and right

Clear button to remove current calculations from screen

Changing number from positive to negative and vice versa

* + - 1. Button to pull up table menu
         1. User can input specific locations into the graph

Will display a table for x and y values

Will only have two columns, for x and y

Will feature unlimited rows so the user can have as many lines on the graph as they want

Will place a dot at specific x and y value on the graph

Delete all button that will remove all points from the table

* + 1. Calculator form
       1. The calculator form will take input from both the CUI buttons and the users keyboard
       2. Hitting enter in the calculator form or hitting the finished button beside the form will bring up 2 buttons

The calculator form will bring up an enter button to run the calculations in the form

Hitting this button will display the answer to the calculation in a text box

The calculator form will bring up a graph button

Hitting this button will bring up a menu with options for colors,styles, and an ok button

The styles will show a dropdown menu letting you choose from points, single line, or dotted line.

The colors will also be a drop down menu allowing the basic colors as options(yellow, green, blue, violet, red, orange, and black)

The default for the graph button will be black and single line

Hitting the ok button will graph on the canvas (1.1.2.)

Hitting this button with a graph already on the canvas will graph onto the same canvas that the graph is already on

The form will contain a clear graph button that will clear all graphs from the canvas

This will bring up a text box with a yes and no

Hitting yes will remove graphs hitting no will remove the buttons from the screen

* + - * 1. If there is an error that is entered into either the graph or enter button a text box will appear

The text box will contain the error along with 2 buttons

It will have a clear calculator form button

It will have a keep calculator form button

* + 1. Hamburger menu button for saving and loading
       1. Save Calculations
          1. Save user calculations as addressed in 3.1

Will be saved from given user login

* + - 1. Load Calculations
         1. Load user calculations as address in 3.2

Will be loaded from given user login

* + - 1. Save Graph
         1. Save user-made graphs as addressed in 3.3
      2. Load Graph
         1. Load user-made graph using functions from 3.3

1. Back-end:
   1. Google Chrome will be the supported browser
   2. Using Docker to run the server
      1. Server language will be javaScript
      2. Files from user will be saved to database
         1. Calculations saved in .csv files
      3. Files from user will be loaded from database
   3. Database using Microsoft SQL server
      1. 3 Tables
         1. User information
         2. Calculation saving information
         3. Image saving information
   4. Typescript Application for buttons
      1. Each button mentioned in 1.1.3 will be implemented
   5. Application for saving files
      1. Will feature parts that will allow the user to save different types of files (continued in 3)
   6. Application for Exporting images
      1. Will implement simple image exporter (Continued in 4)
   7. Application for sharing
      1. User can send calculations using a given username
         1. Once another valid username is added, the other user will be given access to the first users save file
         2. Will allow loading from the receiving user's calculator, not a file.
            1. Can be accessed in their load file menu.
2. File Saver:
   1. Will allow user to save calculations to a .txt
      1. When the user chooses to save, a box will appear to ask for a save location. This will occur for every save instance for all functions that deal with saving.
   2. Will allow user to load calculations from a .txt
   3. Will save graphing functions to a .txt
      1. Allow user to quickly pull up an already made graph
3. Image Exporter:
   1. Export user created graph to PNG, JPG
      1. Give user option to save as PNG or JPG
         1. User interface to ask where to save image
         2. Message notifying user graph has been saved
         3. Returns to User interface noted in 1.1

## Design description

Emergency calc is a web-based calculator that will run off the HTML, JavaScript, TypeScript, and CSS programming languages for the website. This will also use a server running on node.js and using docker. It will also use a database written in SQL using a Microsoft SQL server. This program is intended for students as a calculator is a very useful tool for students taking STEM classes. This calculator will be able to solve basic and advanced math equations, have inbuilt formulas for important equations, and be able to create two dimensional graphs. It is web based to allow it to run off multiple operating systems and, in the future, would be adjusted to be mobile friendly so that more people could use it. This program is meant to help students get through their stem classes and be available to as many students as possible through the internet.

The system itself is simple. As shown in Appendix 4A The emergency calc website itself communicates directly to the server which then communicates with the database. This uses the server as a go between for the website and the database. This will mostly be used to store user generated data and user account info. It will also be used to share images and equations between other instances of the website allowing for more collaboration between users. Inside the website itself it has 2 main components: the UI and the image exporter. The UI contains everything the user will interact with which also contains 2 main components The CUI or calculator user interface, and the graph canvas. The CUI contains everything the user needs to use the calculator. This includes the calculator form where data is inputted, to the button menu that contains all of the buttons required to input data, to a set of tables for graphing on the graph canvas. The graph canvas is much simpler, containing a canvas for graphing and the code that allows the canvas to be edited and printed upon by the user. This includes what color the points are, if the graph is just points or a line, and zoom in and out capabilities. The other part of the emergency calc website is the image exporter. This will take the canvas used to create a graph and then turn it into either a png or jpeg file for use outside of the website. It will then allow the user to save this generated image file to their local computer.

The website will be made of several distinct layers of design. The first layer, the front end, will handle everything that the user can interact with in the server. The first thing the user will be able to interact with is the UI. When first entering the website, it will display a login screen. This will contain a box titled username and one titled password. At the bottom of this screen there will be a create new account button. Entering the username and password boxes will have them checked against the database while making a new user will save the username and password to the database. The UI includes the parts discussed for it in the system overview of the CUI and the graph canvas along with a file menu for saving graph functions and equations, sharing graph function and equations with other users, and saving graph images to the local user’s computer. This part of the website will be mostly written in HTML5 and CSS and will use typescript for scripting. The graph canvas will be a 2d grid with points for positioning and will display like a sheet of graph paper with points and grid lines. The grid lines can be disabled leaving just the points to be displayed on the canvas. This graph will also have a default range of 10. This default can be adjusted through the graph menu. It can also be adjusted using mouse button 3 (the mouse wheel) or the zoom in and zoom out buttons displayed underneath the graph canvas. You can also move what part of the graph you are looking at by holding the left mouse button and moving the mouse. The second part of the UI is the GUI.(1B) This will contain a form for user calculation and a hamburger style menu that can be opened to access all the buttons for the calculator. This menu will contain buttons corresponding to the buttons that would be found on a more traditional graphing calculator along with those added for the useability of the system for STEM students. Using these buttons will place the variables directly into the calculator form. These buttons include: all numbers (0-9), basic math functions (+,-,/,\*), The trigonometric functions and their inverses (cos, sin, tan, and cos^-1,sin^-1, tan^-1), exponential functions and roots, exponents (x,y,z), logarithmic functions (log, ln, log base), important mathematical variables such as e and pi, important physics constants (planck's constant, the speed of light in a vacuum and the Newtonian constant of gravitation), a left and right button that will correspond to the arrow keys on the users keyboard, a clear button to clear the calculator form, a delete button that will act like the delete button on the users keyboard, a negative sign (-), and a table button. This table button will bring up the table menu. This menu will act like the graphing table on a graphing calculator. It will have an unlimited number of rows so that the user can plot as many points as they wish. It will also have a submit and a clear button. The submit button will graph all the points on the table onto the graph canvas and clear the table. The clear button will just clear the table. There will also be a calculator form for all of this user input to be displayed and edited. The calculator form can be edited by the user’s keyboard or by the calculator form. Hitting the enter key in the form or hitting the finished button will bring up two buttons. There will be an enter button to run the calculations or a graph button to graph the equation onto the graph canvas. Hitting the graph button while there is already a graph on the graph canvas will add the graph to the canvas. This allows for multiple graphs to be on the canvas simultaneously. The graph button will also bring up your graphing options of colors (yellow, green, blue, violet, red, orange, and black) and graph styles (points, lines, and dotted lines). This graph menu will also have a clear graphs function that will make a new canvas for future graphing. There will also be a save menu. The save menu will have options to load or save graphs and equations from the database, and options to export the graphs created as either a png or a jpeg.

The second part of the emergency calc website is the backend. This is where the website will interact with the server and the database. This will only work on google chrome currently and will not be mobile friendly. The server will be written in javascript using node js. The database will be written in SQL and use Microsoft SQL. The database will contain 2 tables one for user information, one for saving calculations and saving graph equations. These tables can be seen in Appendix 5A. The server will be used to share equations or graph functions between users. The user will be able to input a person’s username and if it’s a valid username whatever data they input will be sent to the other user. The server will also be used to save data into the user’s history in the database through the save menu in the UI. This data can then be accessed by the user later through the history function in the save menu in the UI. The server will use multiple messages to communicate effectively between the website and the database. All messages sent using the server will be in the JSON format. There will be a message sent from the client to the server to create an account that will contain a char(15) for the username and a char(50) for the password. There will be a message sent from the client to the server for login information that will contain a char(15) for the username and a char(50) for the password. There will be a message sent from the database to the server if the username the user tries to send is incorrect and it will contain a string containing the message "Error": "Incorrect Username or Password". If a client is logged in then they can send a message to the server with a string for the calculations they would like to save and an automatically generated string for the date. If the client wishes to see their previously saved information, they will send a message to the server and the server will respond with a message from the database containing all of their previously saved functions and the dates on which these calculations were saved. There will be a message for the user to send to the server if they want to save a graph equation they will send a string containing the graphs equation. There will be a message from the database to the server when the user requests the graph from history that will send a string containing the graph's equation. There will be a message for when the user wishes to share a calculation with another user that will contain a string for the username, a string for the date the calculation was saved, and a string for the calculation being shared. There will be a message for when the user wishes to share a graph with another user that will contain a string for the username, a string for the date the graph was saved, and a string for the graph equation being shared. If a user attempts to share a graph or an equation with a user that doesn’t exist the database will send a message to the server containing a string with the error message {"Error" : "Username does not exist"}.

The Third part of the application is an image exporter. This importer can be accessed in the UI through the save menu. This will have 2 buttons one for jpeg and one for png. The png one will convert the graph canvas into a png. The jpeg one will turn it into a jpeg instead. It will then ask the user where to save the image file that is created.

Emergency calc is a web-based calculator application. It is meant to provide a useful tool for both doing calculation and creating graphs. It will be doing this using a multitude of technologies separated mostly into three parts: the front end, the back end, and the image exporter. It will use a database to store user info and equations, a website for the user to interact with and use, and a server to communicate between the two along with allowing the sharing of information between users. This project is designed to help students with their classes along with scientists and mathematicians in need of an easy to access calculator.

-Notes:

Images as .png - data from the images for the graph: Will hold a photo of a given graph

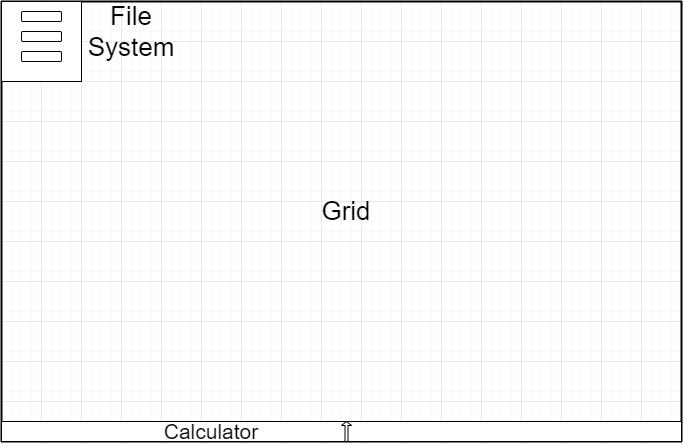
Calculation files as .csv: Will be read out of the .csv into the database and read back onto webpage

We know it should be stored on https but we do not have access to that so it will be on http

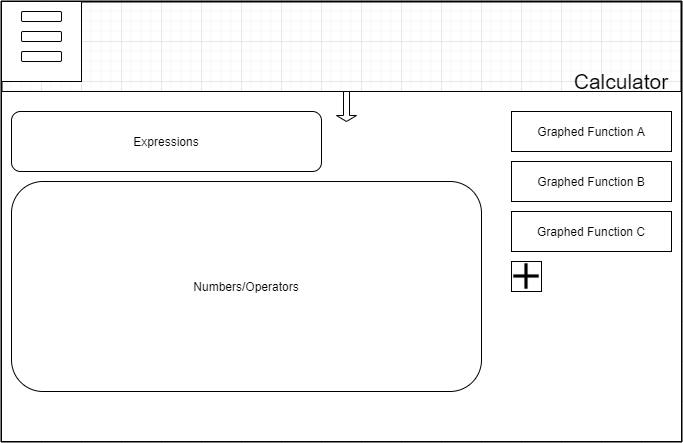
## Appendix

## 

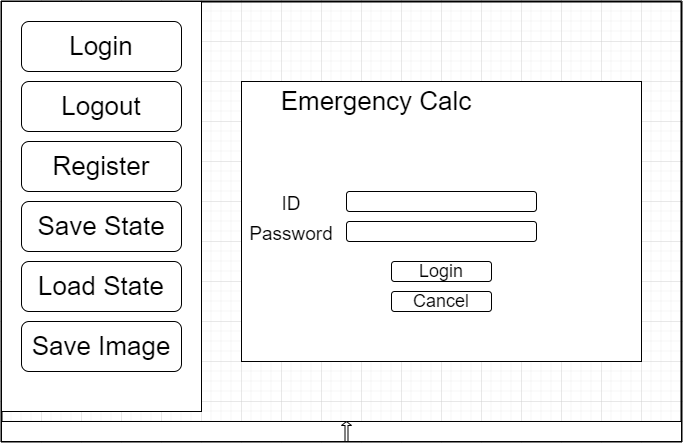
1A. Base UI



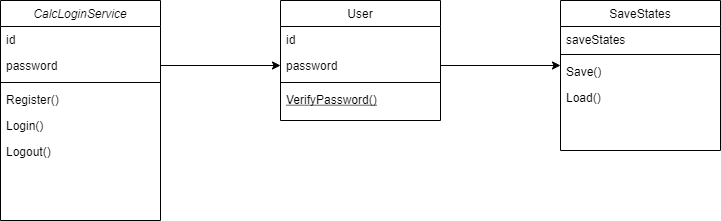
1B. Calculator UI



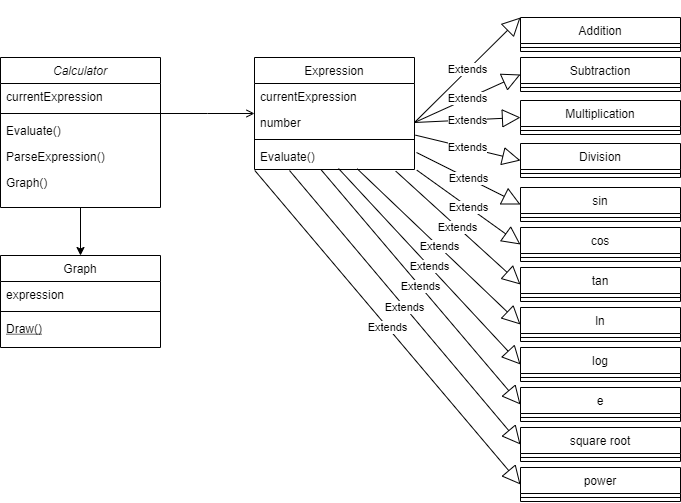
1C. User UI



2A. UserService Class Diagram



3A. Calculator Class Diagram



4A. Messaging Documentation

Client to server User creates an account: Username:char(15), Password: char(50)

JSON Sample: {"Username": "ShadowGear", "Password":"AbCd1846"}

Client to server on login: Username: char(15), Password: char(50):

JSON Sample: {"Username" : "ShadowGear", "Password" : "AbCd1846" }

Login information is not inside database: Incorrect Username(String)

JSON Sample: {"Error": "Incorrect Username or Password"}

Upon successful login CLIENT TO SERVE Rand client requests to save calculations: Calculation (string), Calculation\_Date(string)

JSON Sample: {"Calculation" : "a^2 + b^2 = c^2", "Calculation\_Date": "09/20/2021"}

Upon successful login SERVER TO CLIENT and client requests to load calculations: Calculation (string), Calculation\_Date(string)

JSON Sample: {"Calculation" : "a^2 + b^2 = c^2", "Calculation\_Date": "09/20/2021"}

Upon successful login CLIENT TO SERVER and client requests to save Graph functions: Graph\_Function (string)

JSON Sample: {"Graph\_Function" : "x^2"}

Upon successful login server to client and client requests to load Graph functions: Graph\_Function (string)

JSON Sample: {"Graph\_Function" : "x^2"}

Upon successful login Client to server and client requests to share calculations: Calculation (string), Username(string), Calculation\_Date(date)

JSON Sample: {"Username": "Defiance", "Calculation" : "x^2", "Calculation\_Date": "12/12/2021"}

Upon successful login Client to server and client requests to share Graph functions: Graph\_Function (string), Username(string)

JSON Sample: {"Username": "Defiance", "Graph\_Function" : "x^2"}

Requested username to share calculations with does not exist (server to client): Error(string)

JSON Sample: {"Error" : "Username does not exist"}

5A.

