# Calculator:

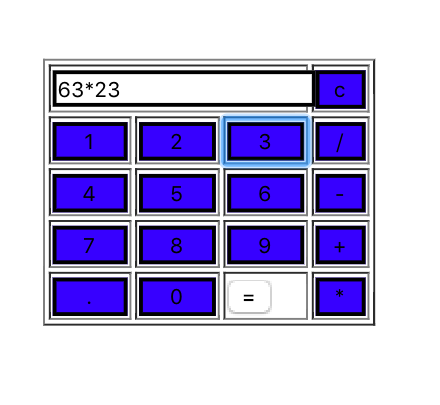
## Goal and Purpose:

Proposed system aims to develop client server-based web application to perform arithmetic operations like addition, subtraction, multiplication and division.

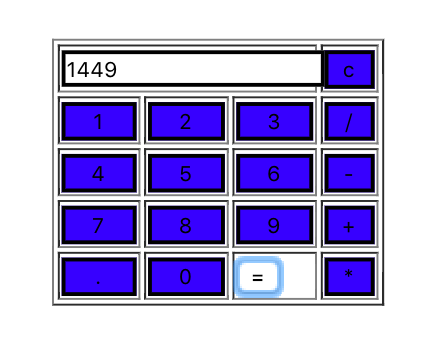
## Chosen System design:

## Result:

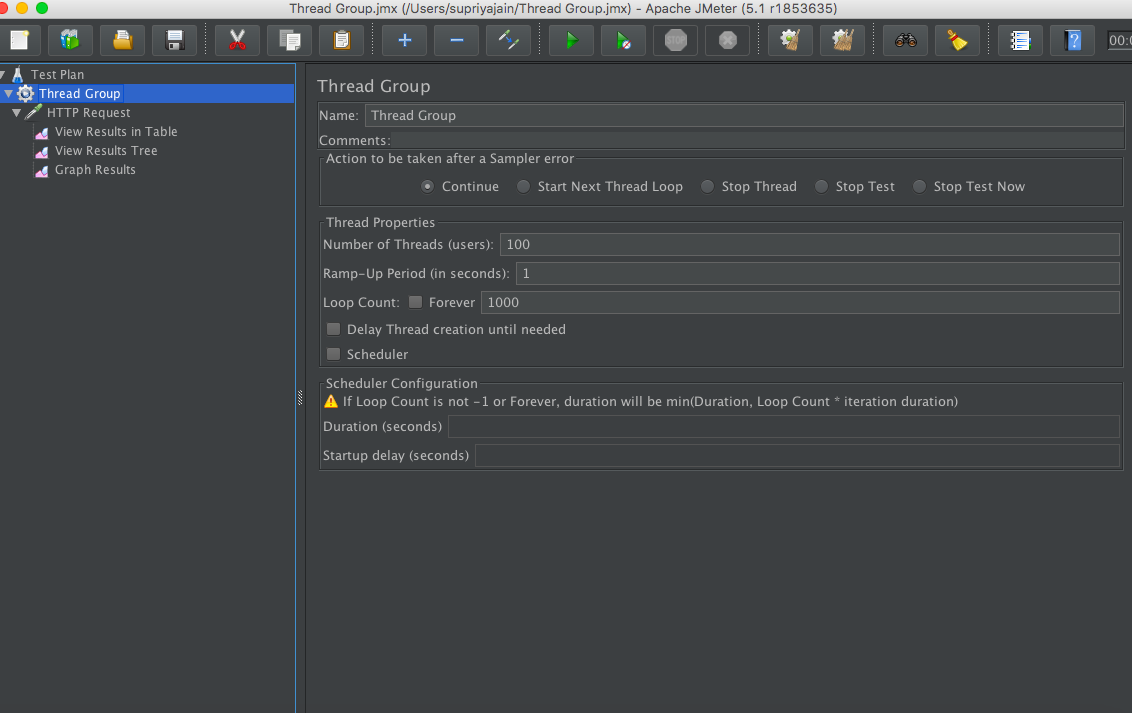
Enter expression:

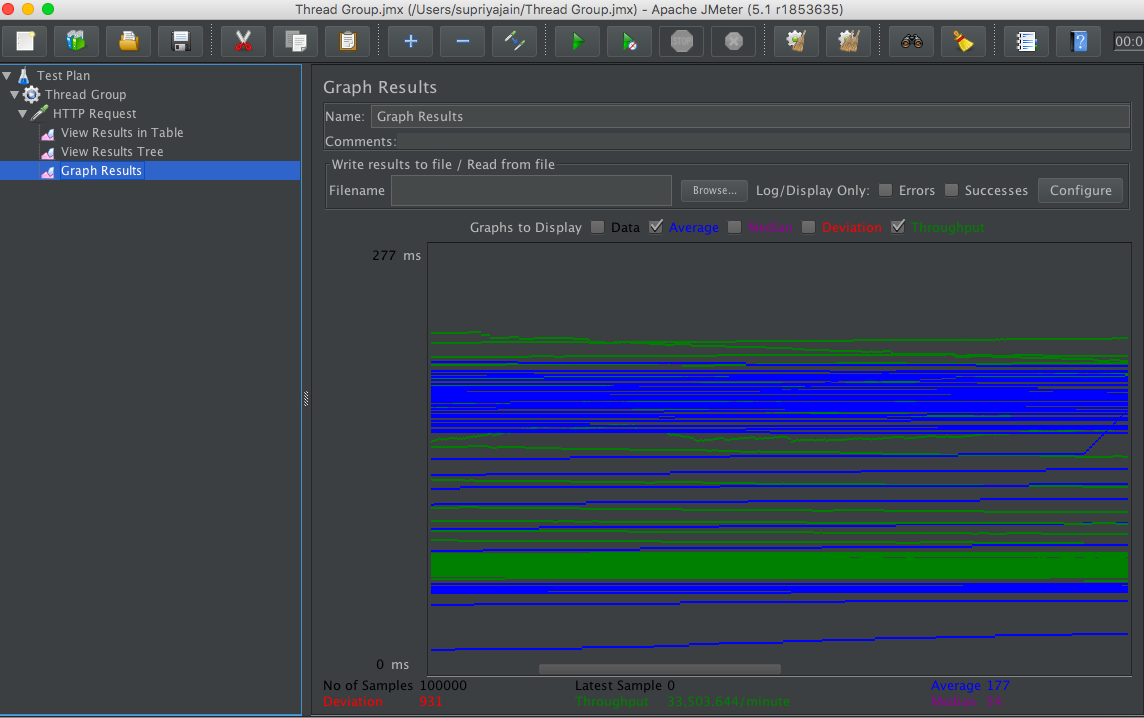


After clicking on “=”



Performance:





# Questions:

### Explain the encryption algorithm used in your application. Mention different encryption algorithms available and the reason for your selection of the algorithm used.

1. I used the Bcrypt for encryption of password which creates the hash of the password entered with some random text added to it and insert the hash of this text into the database. Whenever user logs in, password entered by the user will be converted into hash and then hashes of both passwords will be compared.
2. There are many algorithms are used for the password security of the account. MD5, SHA256 and BCrypt are the popular once.
3. MD5 provides basic hashing for generating secure password hash. It creates a 128-bit hash out of the given password.

To make it more secure and to avoid collision SHA128 (produces 128-bit hash), SHA256 (produces 256-bit hash), and SHA512 (produces 512-bit hash) are used.

The probability of a collision decreases as the number of bits increase. Either of the above two is sufficient to keep application secure.

1. However, SHA hashed secure passwords are able to be cracked with today’s fast hardware’s. For two texts there can be a similar hash possible. And It would be easy for hackers to predict the password. To beat this, we need algorithms like PBKDF2, BCrypt and SCrypt which can make the brute force attacks slower and minimize the impact from hackers.
2. When it comes to hashing and encryption slower is always better. The longer it takes to encode something, the longer it takes a computer to try and identify the input. We can make the algorithm work n times slower by using Bcrypt.
3. Bcrypt uses a random number while generating the hash, referred to as Salt, add salt to the password and generate the hash of salt added password then store that password in the database. You just need to generate a random number of a particular length and append it to the plain text password, and then hash it.

In this way, even if passwords for two accounts are same, their hashes will not be same because the salts used in both cases will be different(random).

1. In a nutshell, you should always hash your users’ passwords before storing them in your database. It’s an important security measure to follow. And use modern hashing algorithms like SHA256, or SHA512 Bcrypt, to do so. So even if an attacker gains access to your database, he won’t have the actual passwords of your users.

### 2.Compare the results of graphs with and without in-built mysql connection pooling of database. Explain the result in detail and describe the connection pooling algorithm if you need to implement connection pooling on your own.

### 3. What is SQL caching? What all types of SQL caching are available, and which suits your code the most. You don’t need to implement the caching, write pseudocode or explain in detail.

### Caching is storing the results of resource intensive process so that the resources won’t have to be spent again.

### The SQL caching is the optimization done in-order to increase the performance of running the SQL queries. It caches the select query along with the result set, so that identical selects to execute faster as the data will be fetched from the cache memory.

### Whenever SQL query is created and executed, a query plan will be generated and then the query will be executed. After the successful execution this query will be saved in the cache.

### So when you send the same query twice, the second time as the previous query is already in shared memory, the response time will be much faster.

### Every vendor has different choices for what they cache. The Query, Query Plan or the result of the query can be stored to process the same information later. Here hash code of the query plan will be generated and stored in the cache.

### The method of storing the query plans and resultsets of the queries in the buffer cache is known as SQL caching. The MySQL query cache is one of the important features in MySQL and a vital part of query optimization. Query cache is a global and shared among the sessions.

Types of SQL Caching:

* **Algebrizer trees:**
  + The Algebrizer’s job is to produce an algebrizer tree, which represents the logic structure of a query.
* **Compiled Plans:**
  + When the query optimizer finishes compiling a query plan, the principal output is compiled plan.
* **Cursors:**
  + Cursors track the execution state of server-side cursors, including the cursor’s current location within a resultset.
* **Execution contexts:**
  + While executing a compiled plan, SQL Server has to keep track of information about the state of execution.

### Is your session strategy horizontally scalable? If YES, explain your session handling strategy. If NO, then explain how you can achieve it.

1. Yes. Session handling strategy is horizontally scalable that means when multiple users tries to access the canvas application simultaneously, then session handling will be done properly without collision of one user with other.
2. I am using PassportJS for making session handling horizontally scalable.

Whenever user logs in to the application, passport will generate a jwt token and each time the user logs in, application will generate the signed jwt token and returns that token in the response. The client will saves the token in **localStorage** and for every subsequent request that needs authentication it will send that token in a request.

1. All requests needing authentication pass through a middleware that checks the provided token and allows the request only if the token is verified.