

*Carleton University, Ottawa, Canada*  
*School of Computer Science*  
*Winter Term, 2020*

**COMP3008A Project 2**  
**Quantitative Usability Evaluation**

**TEAM “ROBOTS”**

- Tejal Darpan (101063975)
- Khushal Singh (101094697)
- Daniel Shifman (101040846)
- Jason Yang (101028952)

**Submission Date : April 10, 2020**

## Part 1 - Descriptive Statistics

### Part 1.1 Text Password Scheme

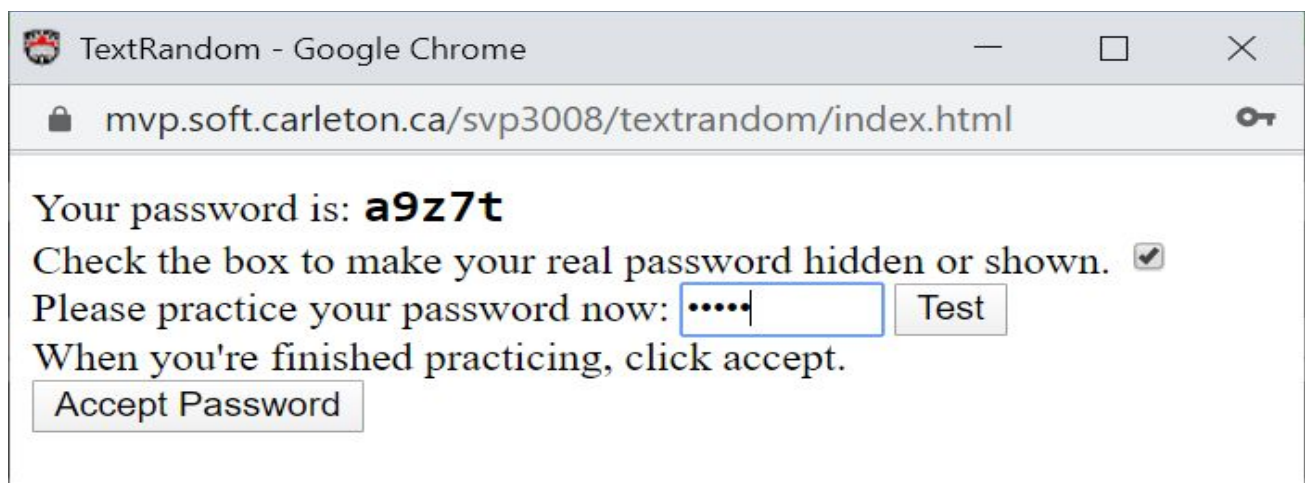
#### Advantages

- Easy to remember due to it consisting of only 5 strings
- Assigning random alphanumeric strings to passwords makes it strong and harder to bruteforce
- Text passwords can be stored in one's notes or can be screenshotted so one can save it
- Most laptops and mobile phones provide an option to "remember password", therefore one need not enter it again

#### Disadvantages

1. Due to the unfamiliarity associated with the randomness of a generated password, it may be difficult for the user to recall
2. Most text passwords consist of conditions (such as upper- and lower-case characters or one special character). The text scheme password does not enforce this, which weakens its security
3. For people of older age, this randomly text generated password can be hard to remember.
4. If the user stores the password, in case they were to forget it, this presents a security risk given that a third party may access the user's data with the stored password.

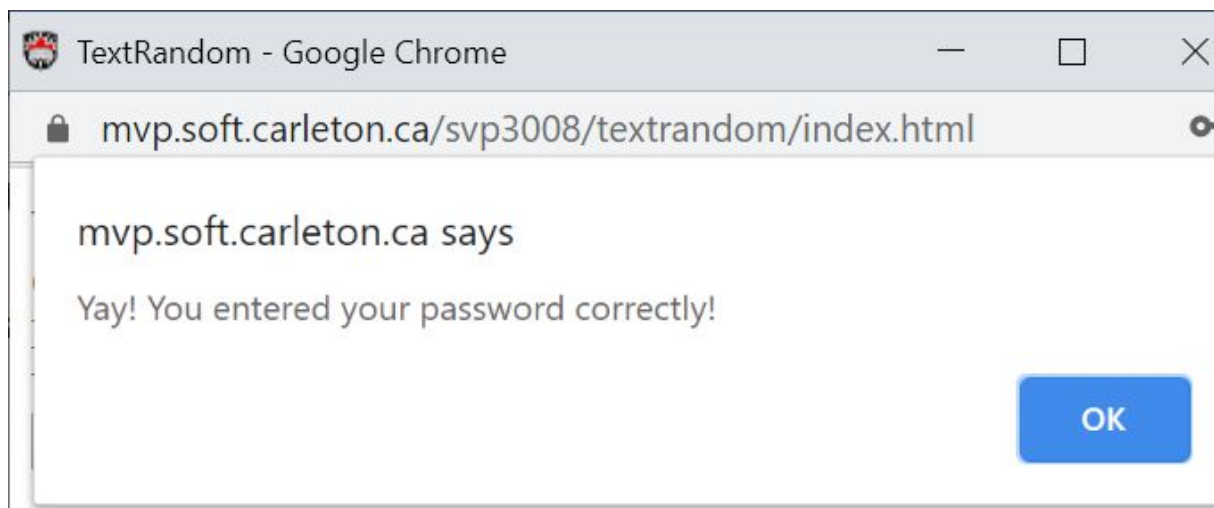
Figure (1) Accepting Password Page



The screenshot shows a web browser window titled "TextRandom - Google Chrome". The address bar displays the URL "mvp.soft.carleton.ca/svp3008/textrandom/index.html". The main content area of the page contains the following text and controls:

- "Your password is: **a9z7t**"
- "Check the box to make your real password hidden or shown. ☒
- "Please practice your password now:  Test"
- "When you're finished practicing, click accept."
- "Accept Password" button

**Figure (2) Password Acceptance page (Correct)**



**Figure (3) And you can keep on setting passwords**

**SVP Password Tester**

User: svp536315

Scheme: textrandom; Condition: az09-5

---

Create Password for: **Email**

Create Next

---

Create Password for: **Banking**

Create Next

---

Create Password for: **Shopping**

Create Next

---

Enter Password for: **Banking** (3 Attempts Allowed)

Enter Next

---

Enter Password for: **Shopping** (3 Attempts Allowed)

Enter Next

---

Enter Password for: **Email** (3 Attempts Allowed)

Enter Next

**Figure(4) Log data history**

**Log Data:**

- 2020-04-09T23:41:15.042Z svp536315 Email textrandom;az09-5 Create Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36
- 2020-04-09T23:41:23.165Z svp536315 Email textrandom;az09-5 pwtest good
- 2020-04-09T23:58:46.934Z svp536315 Email textrandom;az09-5 Create Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36
- 2020-04-09T23:59:00.493Z svp536315 Email textrandom;az09-5 Create Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36
- 2020-04-10T00:01:32.995Z svp536315 Email textrandom;az09-5 passwordSubmitted pw:a9z7t
- 2020-04-10T00:01:32.999Z svp536315 Email textrandom;az09-5 CreateDone Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36
- 2020-04-10T00:01:47.007Z svp536315 Banking textrandom;az09-5 Create Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36
- 2020-04-10T00:02:04.727Z svp536315 Banking textrandom;az09-5 pwtest good
- 2020-04-10T00:04:01.942Z svp536315 Banking textrandom;az09-5 Create Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36
- 2020-04-10T00:04:20.817Z svp536315 Banking textrandom;az09-5 pwtest bad

## Image Password Scheme

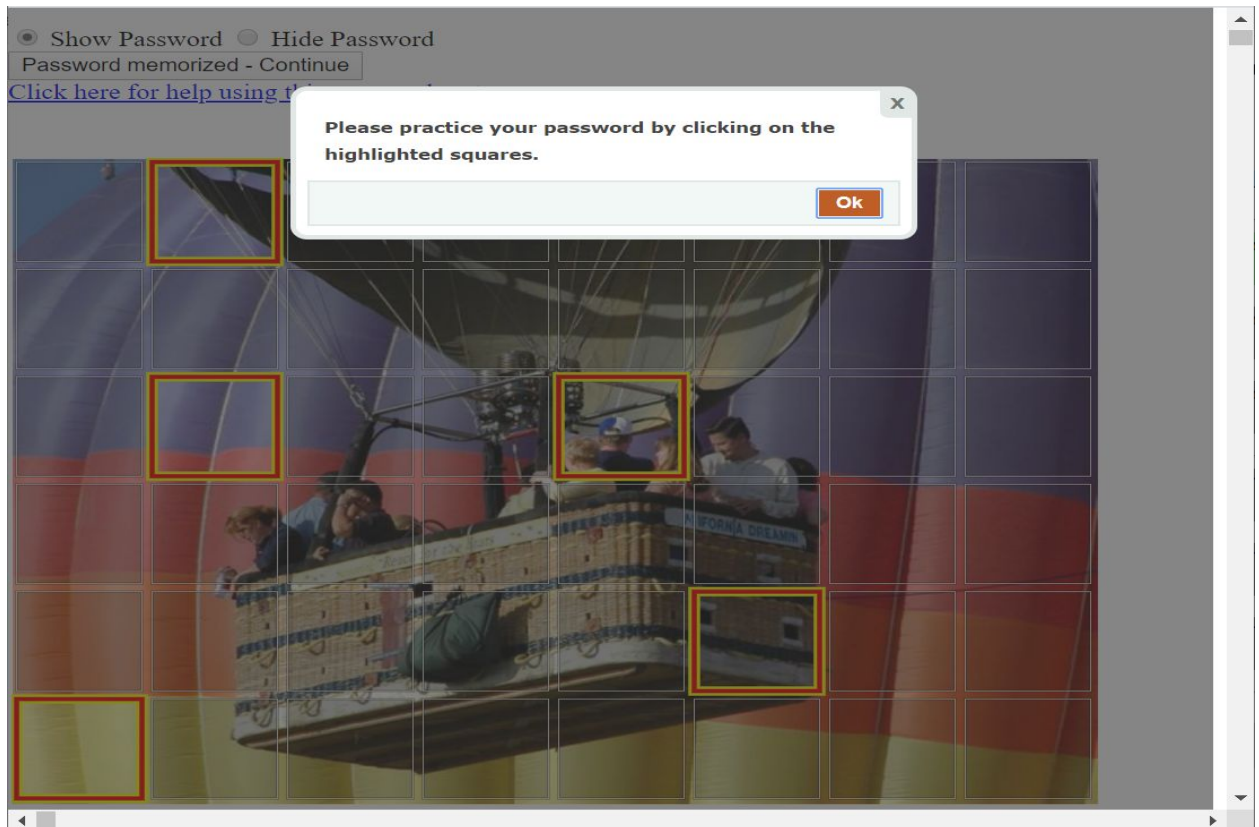
### Advantages

- It's easy to remember for visual learners
- Commutative key selection
- Image password scheme can be visually appealing to younger users

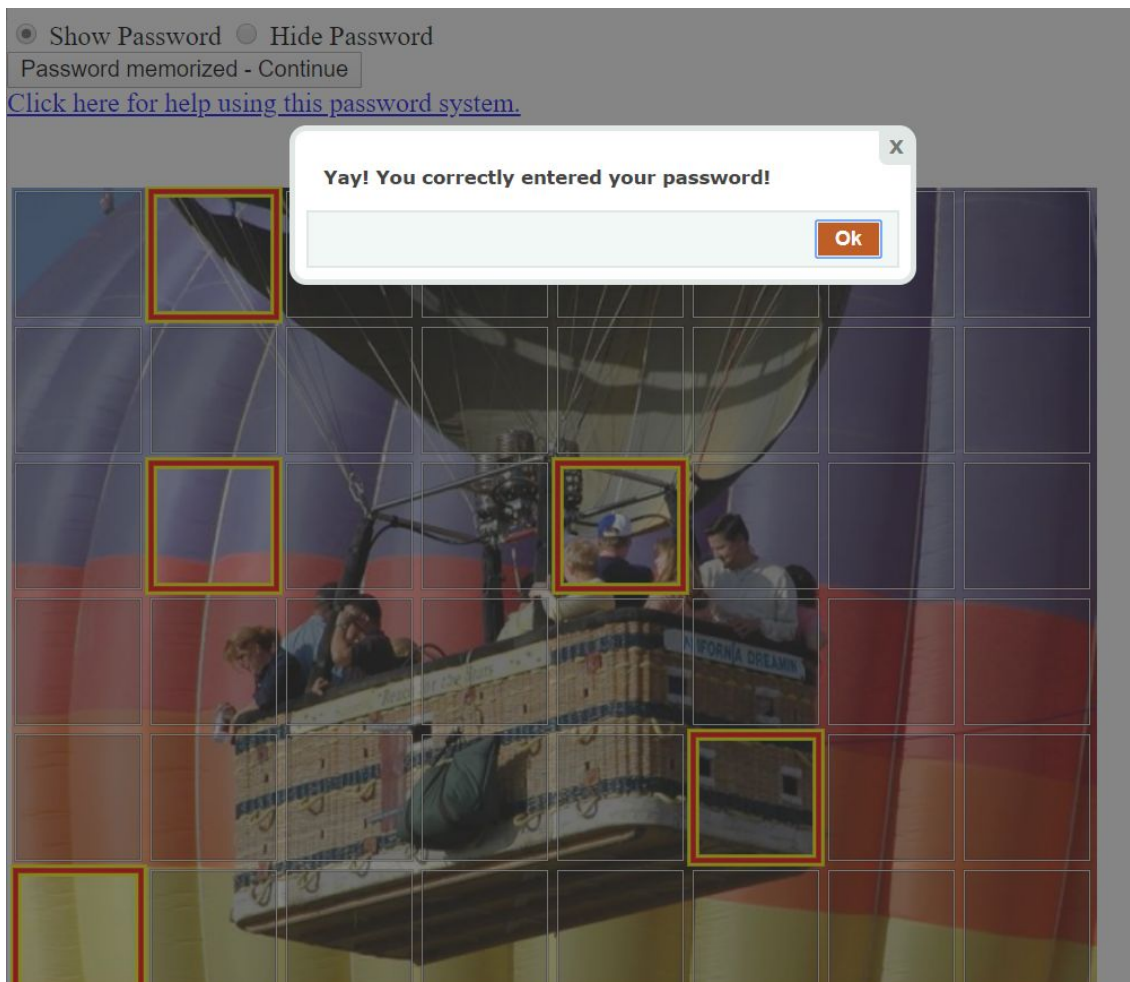
### Disadvantages

- People who use text passwords often can find this scheme difficult to remember
- Clicking 5 unique locations can be extremely confusing
- Easier for people around to see and remember the images
- More difficult to store, unlike a text-based approach

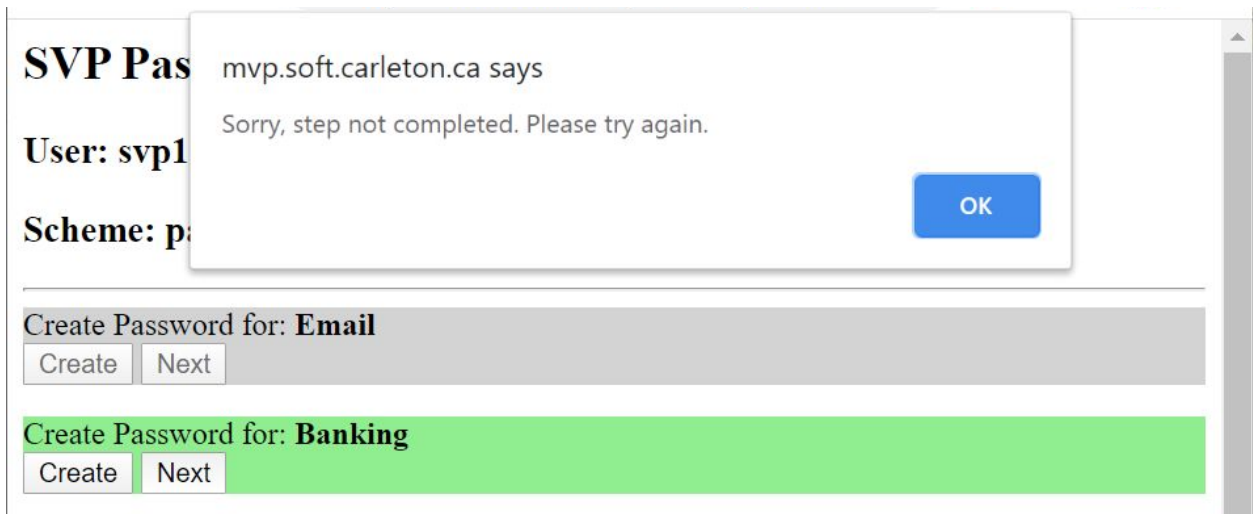
### Figure(1) Practice



**Figure(2) Successful Attempt**



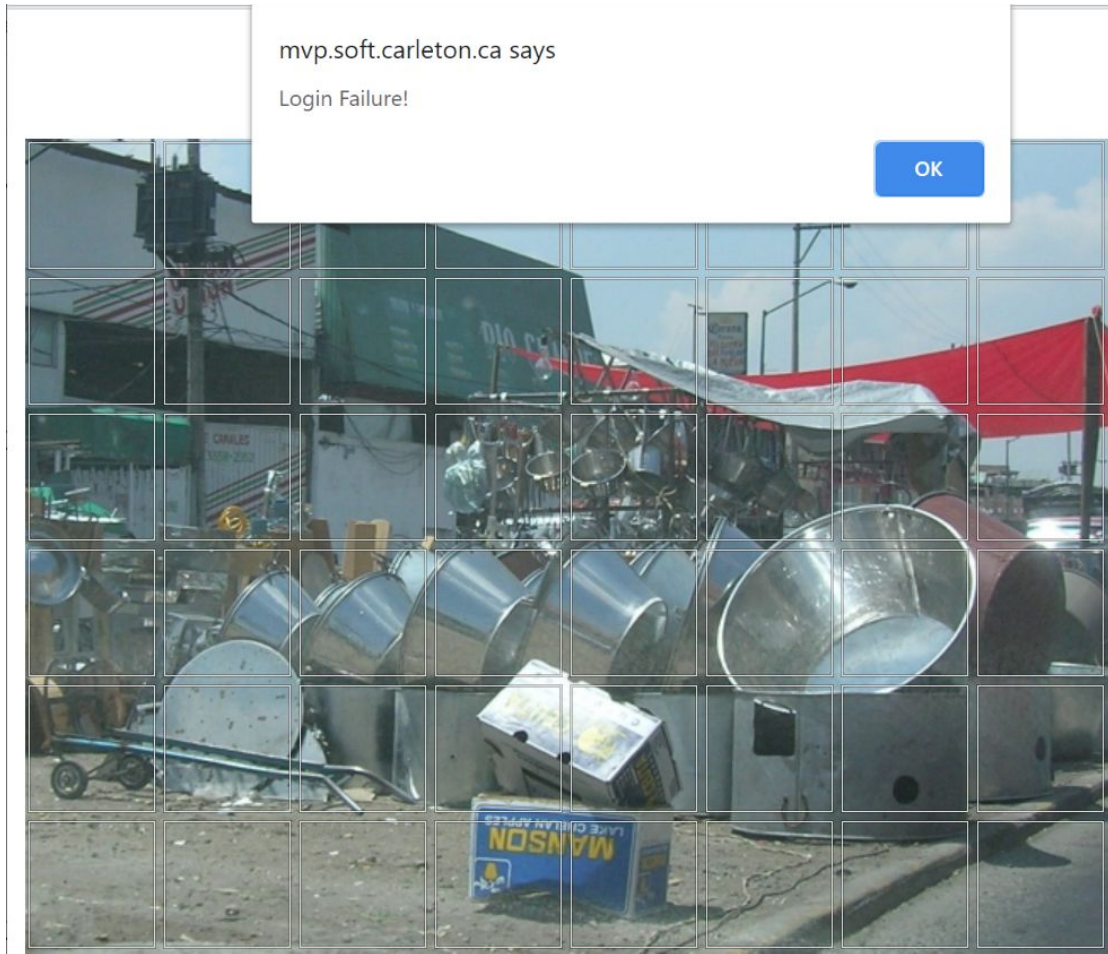
**Figure (3) Create and Next to be followed same way in order to chose next option**





## Incorrect login Training Page Images

### - Figure (4) Incorrect Image Password entered



### - Figure(5) Incorrect text password entered

#### SVP Password Tester

User: svp328224

Scheme: textrandom; Condition: az09-5

Create Password for: **Email**

Create Next

Create Password for: **Banking**

Create Next

Create Password for: **Shopping**

Create Next

Enter Password for: **Shopping** (3 Attempts Allowed)

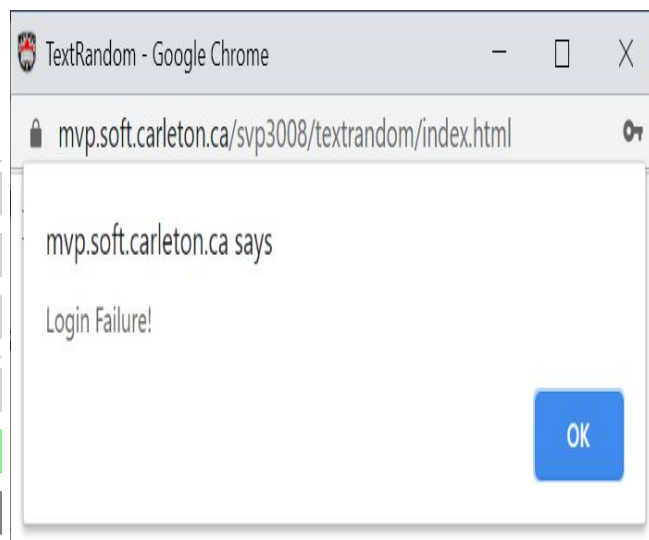
Enter Next

Enter Password for: **Email** (3 Attempts Allowed)

Enter Next

Enter Password for: **Banking** (3 Attempts Allowed)

Enter Next



### Part 1.3 High-level explanation:

The program takes the two supplied CSVs from the “Working Logs” directory, splices them together, converts the CSV format to the requested user-oriented format, and outputs the resulting CSV to the “Working Logs” directory as dataOut.csv.

#### Pseudocode:

```
textData <- text CSV Data
imageData <- image CSV Data
combinedData <- empty CSV Data
users <- empty user list

For each row in textData {
    Add row to combinedData
}

For each row in imageData {
    Add row to combinedData
}

startTime <- empty time
duration <- empty int

For each row in combined data {
    exists <- false
    u <- new userData
    u.username <- row.username
    u.scheme <- row.scheme
    u.failedattempts, u.successfulattempts, u.totalattempts <- 0,0,0
    if users is empty {
        add u to users
    } else {
        exists <- false
        for each user in users {
            if row.username is user.username {
                u <- user
                exists <- true
            }
        }
    }
}
```

```

                                break
                            }
                        }
                    }
                }
            }
            if row.EventDetails = "start" and row.Event = "enter" {
                startTime <- row.Time
            }

            if row.Event is "enter" and (row.EventDetails is
"passwordSubmitted" or row.EventDetails is "order inputPwd") {
                u.totalattempts++
            }

            if row.EventDetails is "failure" and row.Event is "login" {
                u.FailedAttempts++
                failTime <- row.Time
                duration <- startTime - failTime
                add duration to u.FailedTime
            }

            if row.EventDetails is "success" and row.Event is "login" {
                u.SuccessfulAttempts++
                successTime <- row.Time
                duration <- startTime - successTime
                add duration to u.SuccessfulTime
            }

            if exists false {
;        add u to users
            }
        }

for each item in users {
    if item.scheme is "testpasstiles" {
        item.scheme <- "Image21"
    }
    if item.scheme is "testtextrandom" {
        Item.scheme <- "Text21"
    }
}

```



```
}
```

**Source:**

```
/*
COMP 3008 - Project 2 - Team: ROBOTS
CSV Processing Program
*/

package main

import (
    "encoding/csv"
    "fmt"
    "io"
    "os"
    "time"
)

// CSV location strings
var textLocation    string = "./Working Logs/text21.csv"
var imageLocation   string = "./Working Logs/imagept21.csv"
var finalLocation   string = "./Working Logs/dataOut.csv"

// Declaration of list variables
var textDataList    dataList
var imageDataList   dataList
var combinedData    dataList
var users            userList

// Declaration of CSV data objects
var imageData        CSVData
var textData         CSVData

// Declaration of list types
type userList []*userData
type dataList []CSVData
```

```

// Struct for storing original CSV values
type CSVData struct {
    Time          string
    User          string
    Site          string
    Scheme        string
    Mode          string
    Event         string
    EventDetails  string
    Data          string
}

// Struct for storing user data
type userData struct {
    User          string
    Scheme        string
    TotalAttempts int
    SuccessfulAttempts int
    FailedAttempts int
    SuccessfulTime []string
    FailedTime     []string
}

// ReadCSV reads the CSV at location.
// It returns the data as an array of lines and an error.
func ReadCsv(location string) ([][]string, error) {

    f, err := os.Open(location)
    if err != nil {
        return [][]string{}, err
    }
    defer f.Close()

    lines, err := csv.NewReader(f).ReadAll()
    if err != nil {
        return [][]string{}, err
    }

```

```

    }

    return lines, nil
}

// loadTextData loads the data from text21.csv into a CSVData object.
func loadTextData() {
    lines, err := ReadCsv(textLocation)
    if err != nil {
        panic(err)
    }

    for _, line := range lines {
        data := CSVData{
            Time:      line[0],
            User:      line[1],
            Site:      line[2],
            Scheme:    line[3],
            Mode:      line[4],
            Event:     line[5],
            EventDetails: line[6],
            Data:      line[7],
        }

        textDataList = append(textDataList, data)

        //fmt.Println(data.Time + " " + data.User + " " + data.Site + "
" + data.Scheme + " " + data.Mode + " " + data.Event + " " +
data.EventDetails + " " + data.Data)
    }
}

// loadImageData loads the data from imagept21.csv into a CSV data
object
func loadImageData() {
    lines, err := ReadCsv(imageLocation)
    if err != nil {
        panic(err)
    }

```

```

}

for _, line := range lines {
    data := CSVData{
        Time:      line[0],
        User:      line[1],
        Site:      line[2],
        Scheme:    line[3],
        Mode:      line[4],
        Event:     line[5],
        EventDetails: line[6],
        Data:      line[7],
    }

    imageDataList = append(imageDataList, data)
    //fmt.Println(data.Time + " " + data.User + " " + data.Site + "
" + data.Scheme + " " + data.Mode + " " + data.Event + " " +
data.EventDetails + " " + data.Data)
}
}

// encode encodes an object into CSV format
// The method takes the data from a *userList and writes it to an
io.writer object
// It returns an error
func (data *userList) encode(w io.Writer) error {
    n := csv.NewWriter(w)
    err := n.Write([]string{"UserID", "Scheme", "Total Attempts",
"Successful Attempts", "Failed Attempts", "Successful Times (s)",
"Failed Times (s)"})
    if err != nil {
        return err
    }
    for _, v := range *data {
        err := n.Write([]string{
            v.User,
            v.Scheme,
            fmt.Sprintf("%d", v.TotalAttempts),

```

```

        fmt.Sprintf("%d", v.SuccessfulAttempts),
        fmt.Sprintf("%d", v.FailedAttempts),
        fmt.Sprintf("%v", v.SuccessfulTime),
        fmt.Sprintf("%v", v.FailedTime)})
    if err != nil {
        return err
    }
}

n.Flush()
return n.Error()
}

// saveData saves the users to a file
func saveData() {
    f, err := os.Create(finalLocation)
    if err != nil {
        panic(err)
    }
    defer f.Close()

    users.encode(f)
}

// convertScheme converts the password scheme into the requested
// format
// It takes in a pointer to a string
func convertScheme(str* string) {
    if *str == "testpasstiles" {
        *str = "Image21"
    }

    if *str == "testtextrandom" {
        *str = "Text21"
    }
}

// getLoginData calculates and applies the login data to each user

```

```

add appends the user to the user list
func (data dataList) getLoginData(users* userList) {
    var startTime time.Time
    var duration int
    for _, v := range data {
        exists := false
        u := new(userData)
        u.User = v.User
        u.Scheme = v.Scheme
        u.FailedAttempts, u.SuccessfulAttempts, u.TotalAttempts = 0, 0,
0
        if users == nil {
            *users = append(*users, u)
        } else {
            exists = false
            for _, w := range *users {
                if v.User == w.User {
                    u = w
                    exists = true
                    break
                }
            }
        }

        fmt.Println(v.EventDetails)

        if v.EventDetails == "start" && v.Event == "enter" {
            startTime = convertTime(v.Time)
        }

        if v.Event == "enter" && (v.EventDetails == "passwordSubmitted"
|| v.EventDetails == "order inputPwd") {
            u.TotalAttempts++
        }

        if v.EventDetails == "failure" && v.Event == "login" {
            u.FailedAttempts++
            failTime := convertTime(v.Time)

```



```

        duration = calculateDuration(startTime, failTime)
        u.FailedTime = append(u.FailedTime, fmt.Sprintf("%d",
duration))
    }

    if v.EventDetails == "success" && v.Event == "login" {
        u.SuccessfulAttempts++
        successTime := convertTime(v.Time)
        duration = calculateDuration(startTime, successTime)
        u.SuccessfulTime = append(u.SuccessfulTime, fmt.Sprintf("%d",
duration))
    }

    if !exists {
        *users = append(*users, u)
    }
}

```

// convertTime is a helper function to convert login time strings to Time objects

// It takes in a time string and returns a Time object

```

func convertTime(aTime string) time.Time {
    t, _ := time.Parse("2006-01-02 15:04:05", aTime)
    return t
}

```

// calculateDuration calculates time difference

// It returns an int representing duration in seconds

```

func calculateDuration(t1 time.Time, t2 time.Time) int {
    dur := t2.Sub(t1)
    return int(dur.Seconds())
}

```

```

func main() {
    //Import data from files
    loadTextData()
    loadImageData()
}

```

```
//Merge data
for _, v := range textDataList {
    combinedData = append(combinedData, v)
}
for _, v := range imageDataList {
    combinedData = append(combinedData, v)
}

combinedData.getLoginData(&users)

//Convert scheme labels
for _, v := range users {
    convertScheme(&v.Scheme)
}

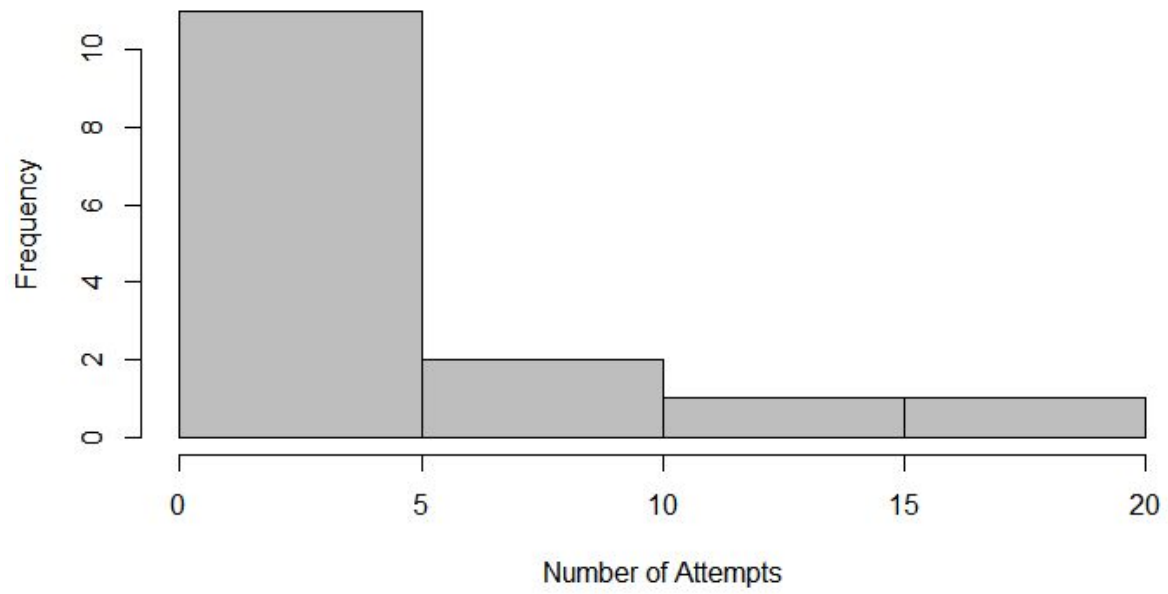
saveData()
}
```

**RESULTING DATA IS ATTACHED WITH ASSIGNMENT SUBMISSION FOLDER  
AS DATAOUT.CSV**

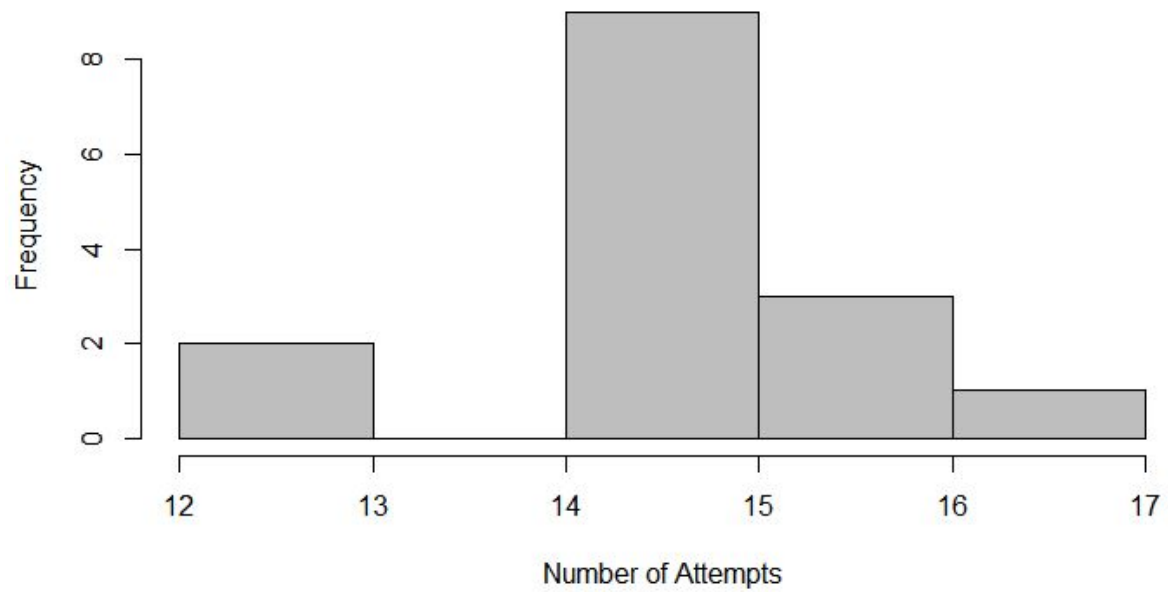
### **Descriptive Statistics of Data**

<b>Type of Data</b>	<b>Text21.csv</b>	<b>Image21.csv</b>
<b>Total login Attempts ( Mean)</b>	<b>21.77</b>	<b>26.4</b>
<b>Total login Attempts ( S.D )</b>	<b>6.40</b>	<b>8.33</b>
<b>Total login Attempts ( Median)</b>	<b>21.5</b>	<b>25</b>
<b>Successful logins (Mean)</b>	<b>14.05</b>	<b>14.933</b>
<b>Successful logins (S.D)</b>	<b>3.4</b>	<b>1.3</b>
<b>Successful logins (Median)</b>	<b>15</b>	<b>15</b>
<b>Unsuccessful logins (Mean)</b>	<b>2.55</b>	<b>4.46</b>
<b>Unsuccessful logins (S.D)</b>	<b>3.32</b>	<b>4.4</b>
<b>Unsuccessful login(Median)</b>	<b>1</b>	<b>3</b>
<b>Successful login time (in seconds) (mean)</b>	<b>9.82</b>	<b>3707.7</b>
<b>Successful login time (in seconds) (S.D)</b>	<b>11.5</b>	<b>31928.0</b>
<b>Successful login time (in seconds) (median)</b>	<b>8</b>	<b>14</b>
<b>Unsuccessful login time (in seconds) (mean)</b>	<b>11.54</b>	<b>22.2</b>
<b>Unsuccessful login time (in seconds) (S.D)</b>	<b>9.44</b>	<b>23.3</b>
<b>Unsuccessful login time (in seconds) (Median)</b>	<b>9</b>	<b>16</b>

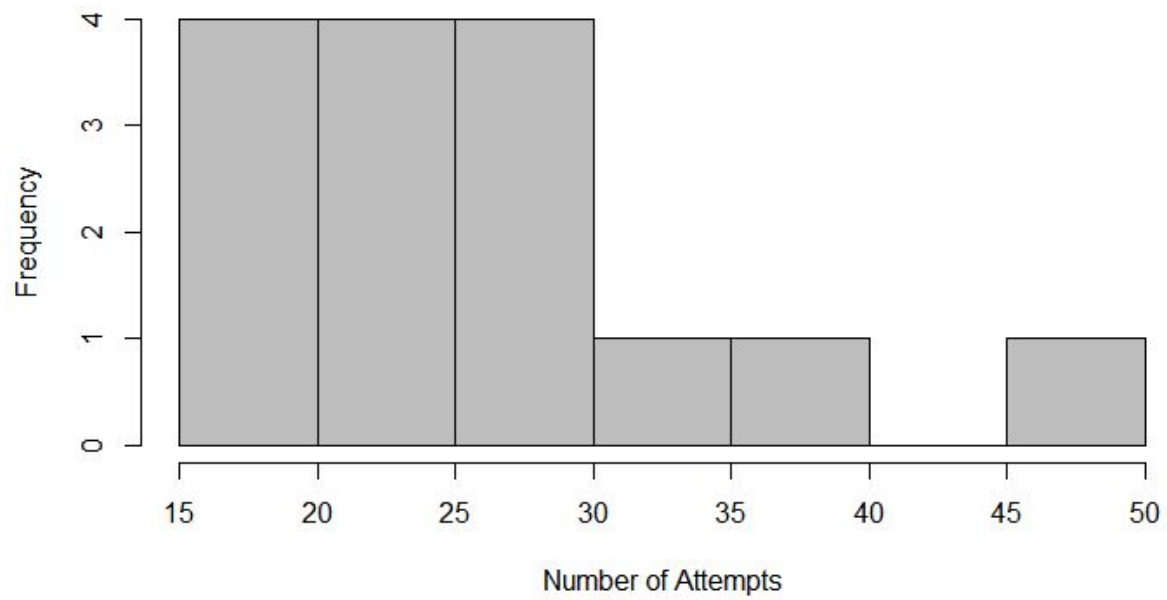
**Image Failed Attempts**



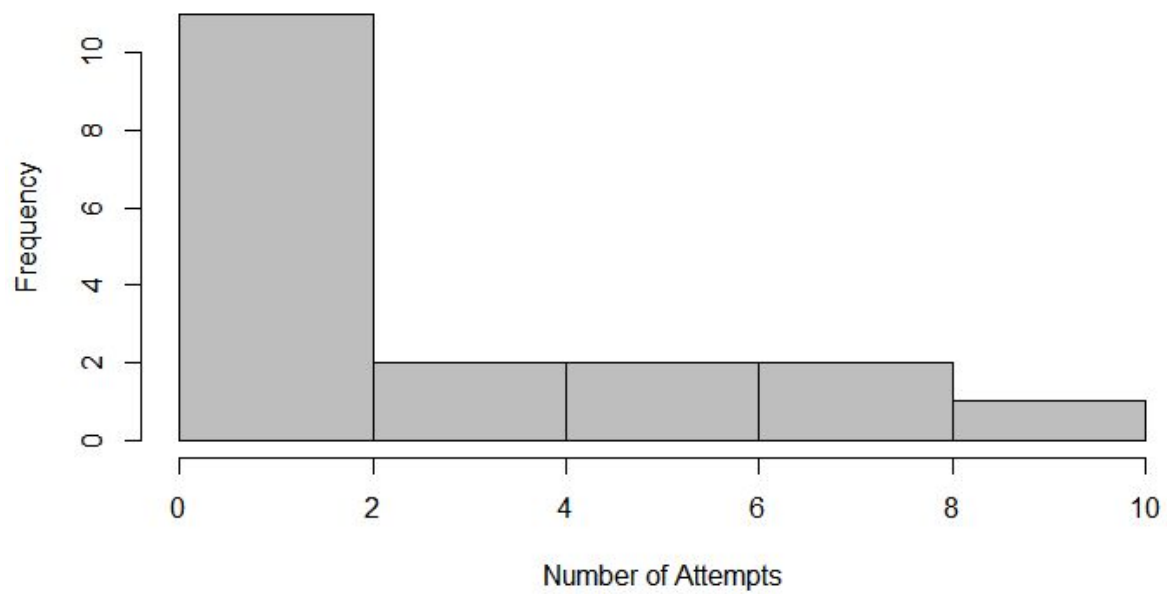
**Image Successful Attempts**



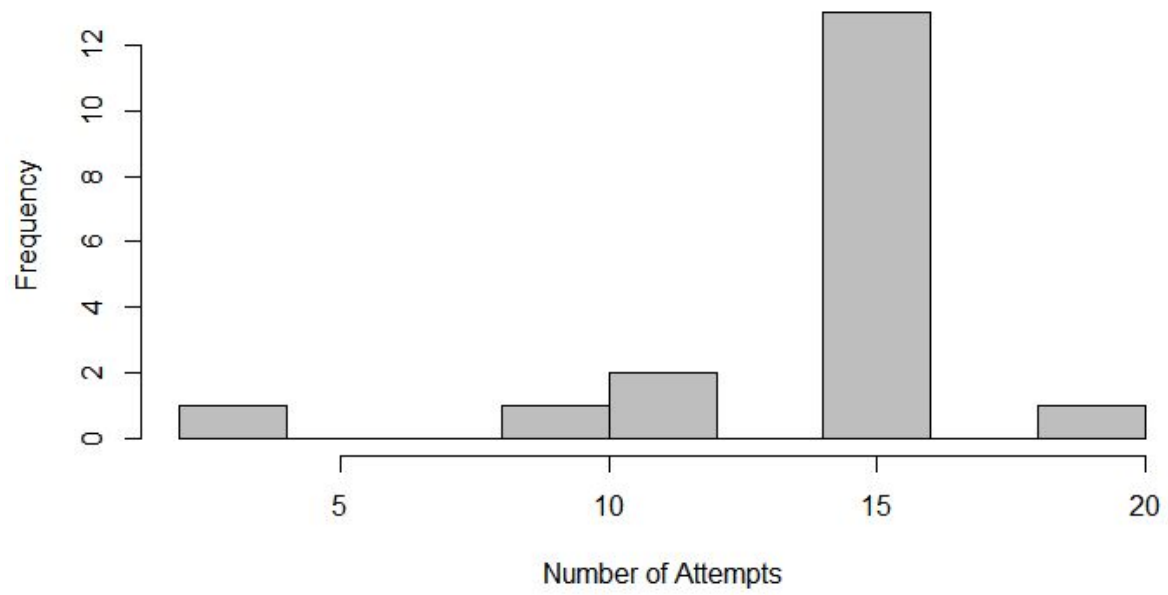
**Image Total Attempts**



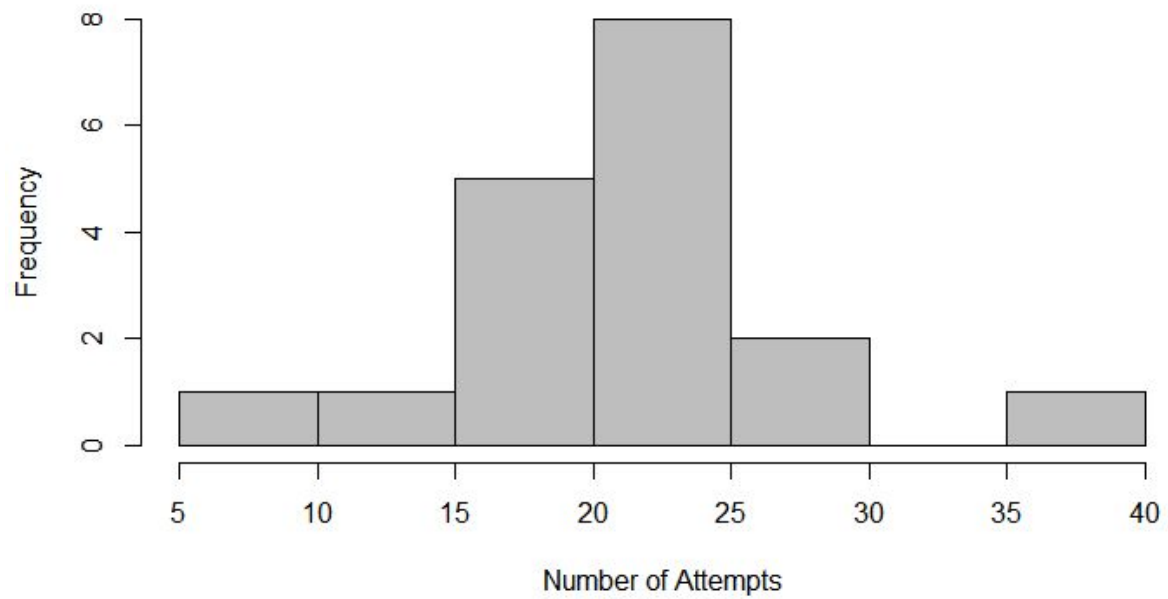
**Text Failed Attempts**



**Text Successful Attempts**

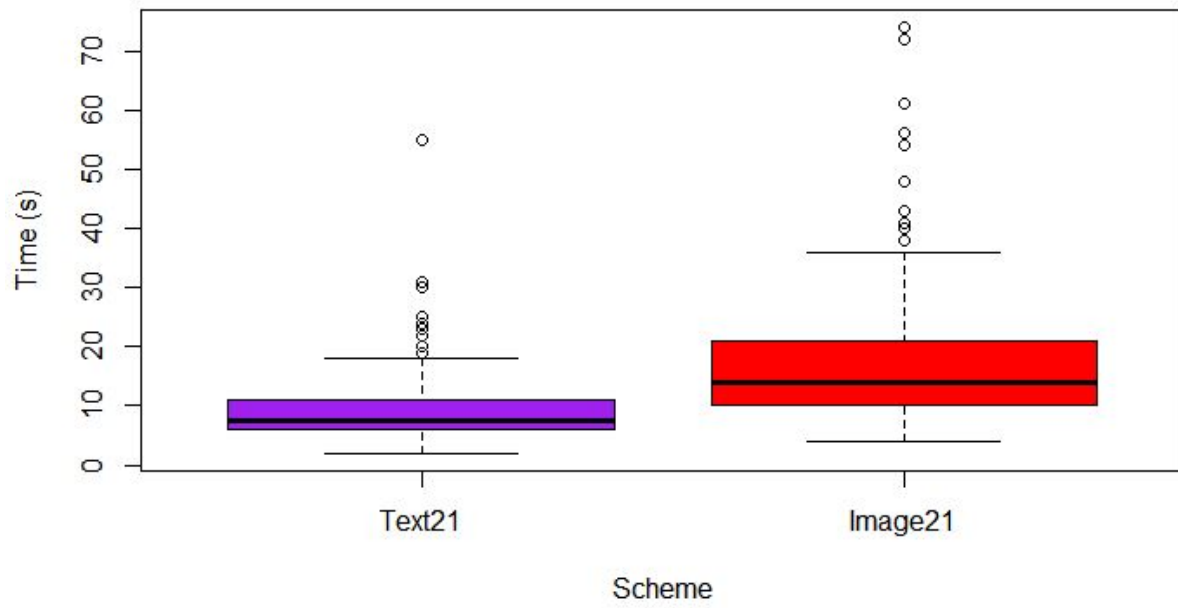


**Text Total Attempts**

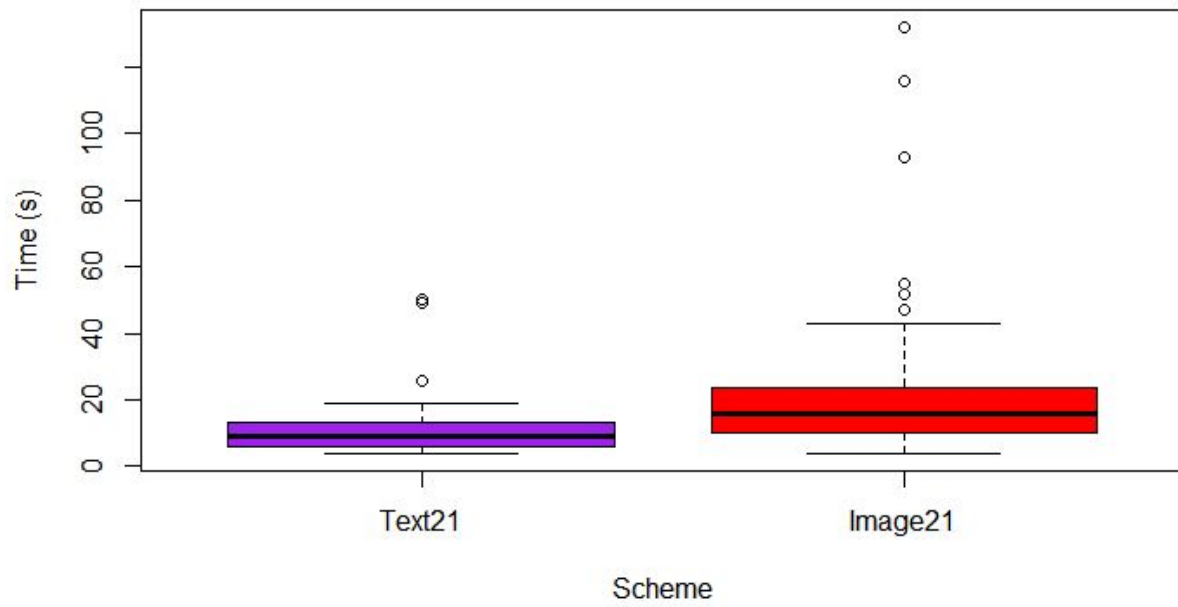




### Successful Login Times



### Failed Login Times



## Explanation of Statistical Analysis

When Analyzing the statistical data carefully we see variations in everything when analysing data of both the image and text schemes. The number of total logins is very high in the text scheme relative to the image scheme, which shows that users prefer to login using text-based passwords. The number of successful logins is again higher using the text scheme due to users being able to login more easily using this method. To confirm whether this is really the case we then plotted histograms of unsuccessful logins of both the schemes and it was concluded that failed attempts were higher in image scheme, hence people intend to forget image based passwords more as compared to text passwords.

For the number of both successful and failed logins, we have created boxplots for individual and comparative data. The data has been again proven that the text scheme is being used more successfully and less unsuccessfully when compared to the image based scheme.

The reason for showing this comparison with a boxplot:

“A histogram is highly useful when wide variances exist among the observed frequencies for a particular data set. As seen in the two graphs to the left, the histogram shows that there are three peaks within the data, indicating it is tri-modal (three commonly recurring groups of numbers). This is important because to improve processes, it is critical to understand what is causing these three modes. Had this data simply been graphed using a box plot, the values would average one another out, causing the distribution to look roughly normal”

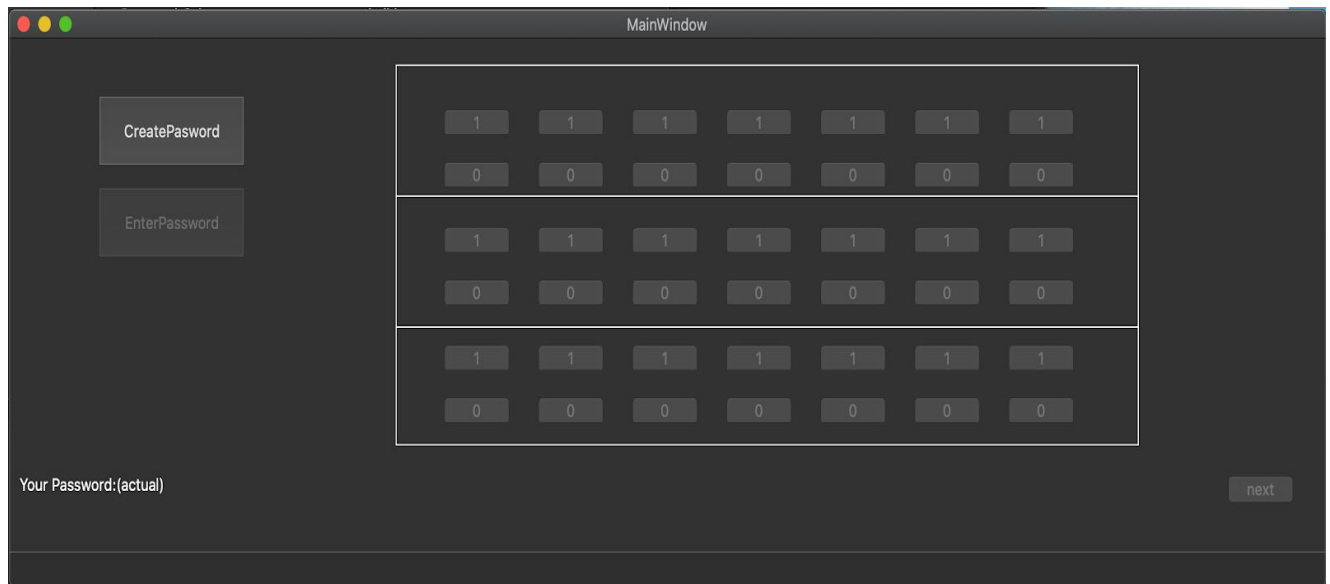
(Plowman, N. (2018, November 18). Comparing Box Plots and Histograms – Which Is the Better Tool? Retrieved April 10, 2020, from

<https://www.brighthubpm.com/six-sigma/58254-box-plots-vs-histograms-in-project-management>

To conclude, the presented statistics demonstrate a higher proficiency of use toward the password scheme “Text21”. These results could imply either a higher user password-retention or higher ease of use with respect to the “Text21” password scheme compared to the “Image21” password scheme.

## PART 2: DESIGN, IMPLEMENTATION, STATISTICAL INFERENCE

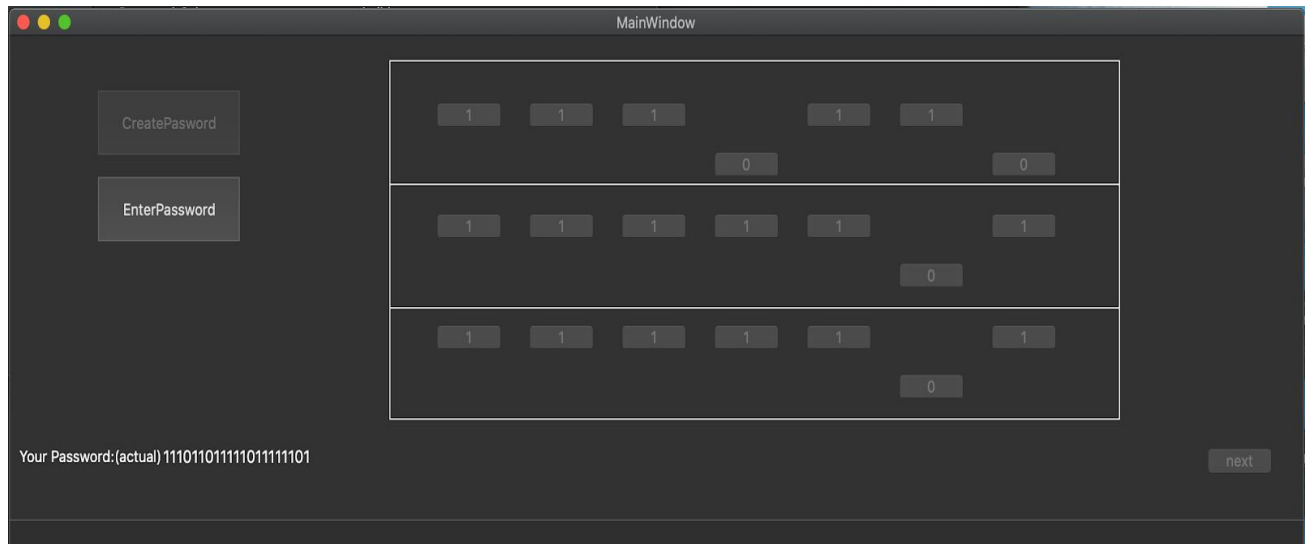
### 2.1: DESIGN



The conventional approach towards password schemes is somewhat difficult. Usually, when people make their passwords, it consists of words and special characters, and they tend to forget it over time. It is because in their mind it is just another random word. They usually don't have some interesting memories attached to them. The password scheme that we have built is easier to understand and easy to remember. Our approach is to provide a simpler way to remember the password. When the program runs, there is a button to create a password for the user. When the user clicks on it, the program generates a binary code of 21 bits randomly. It seems like 21 bits of code is hard to remember but here is the additional approach. It's not just a password but a pattern which is divided into three parts(rows) and each row has seven bits which is easy to understand and remember even by looking at it. So each row has seven bits consisting of 14 buttons which is seven 1's and 7 0's. Only 7 bits are going to be there. So it will be either 0 or 1 from each pair. It will show the pattern by removing the untaken bits in the generated password (see below figure). Usually, it can make patterns (ex. circle, S-shape). This 21-bit string of 1s

and 0s makes the password strong and it is hard to decode. It will make close to  $2^{21}$  combinations.

A malicious third-party won't be able to remember it easily just by looking at the 21 digits. Although, anything can be made better. It can be more efficient if we include images. For example, emojis are more lively than digits. Just as colors have psychological effects, certain emojis can be remembered better.

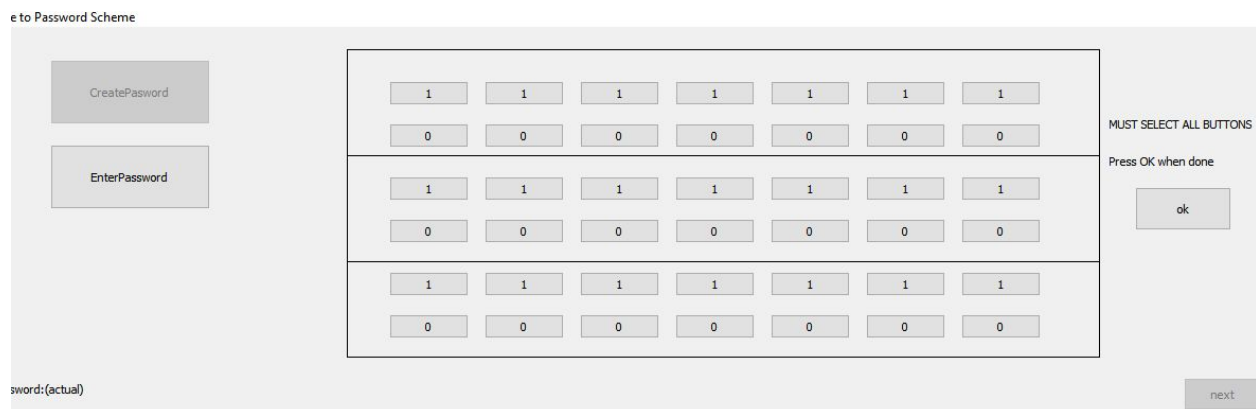


## 2.2:IMPLEMENTATION



**Fig 1- creating a random password**

The password which is generated randomly is shown in visual and numerical forms as a S



**Fig2-Entering password**

After creating password and the “enter password” button is clicked the user can start entering the combo and pressing ok after. The user can press the enter password again if they have messed up an input by accident and haven't pressed the ok button yet and won't count as a try.

Welcome to Password Scheme

CreatePassword

EnterPassword

0	0	0	0	1	1	1
0	0	0	1	1	1	1
0	1	1	1	0	0	0

**FAILURE!**

Must Select All Buttons

Press OK when done

ok

Your Password:(actual) 101001000101001001101

next

**Fig3-Failure**

If the user has entered the wrong combo and pressed the ok button it will show his actual combo and display failure.

Welcome to Password Scheme

CreatePassword

EnterPassword

0	0	1	1	1	1	0
0	1	0	0	1	0	1
1	1	1	1	0	0	0

**SUCCESS!**

Must Select All Buttons

Press OK when done

ok

Your Password:(actual) 001111001001011010110

next

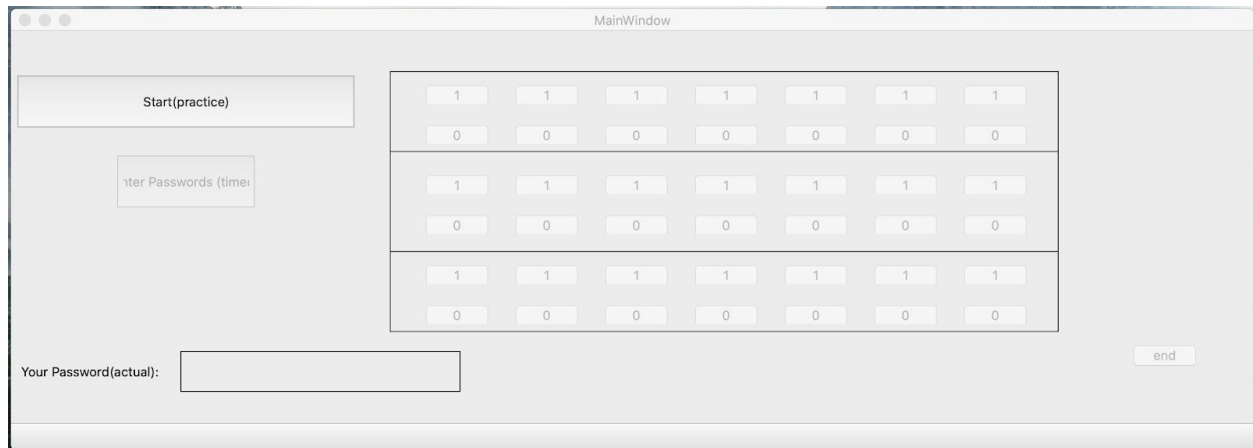
**Fig4-Success!**

If the user's entered combo was correct then a success message will appear. After success or failure the next button will reset the form for the next user.

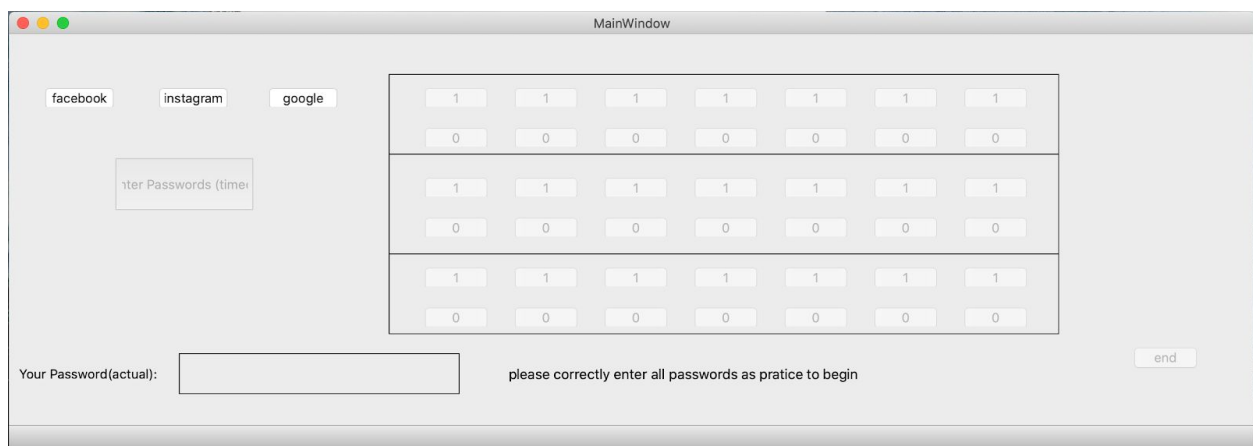


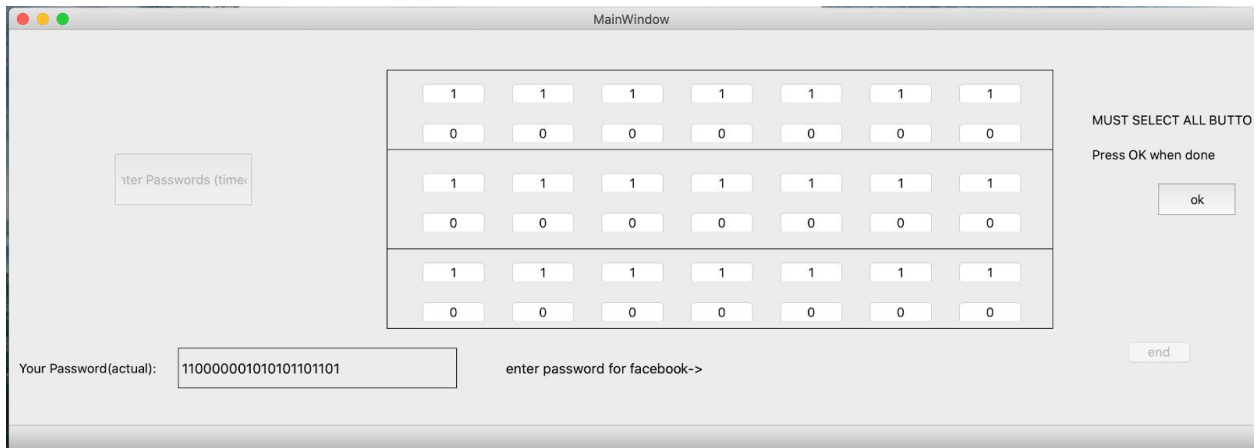
## 2.3 Framework:

This password scheme is built in the QT framework(4.12 version). When we click on the start button(build), the program starts up. It is a window consisting of 42 input buttons(i.e 21 - 1's and 21 - 0's). Along with that, there is start practice which is enabled at the start of the program. Moreover, there are two disabled buttons: enter button and end button. There's a given space where the actual password is shown (figure shown below).

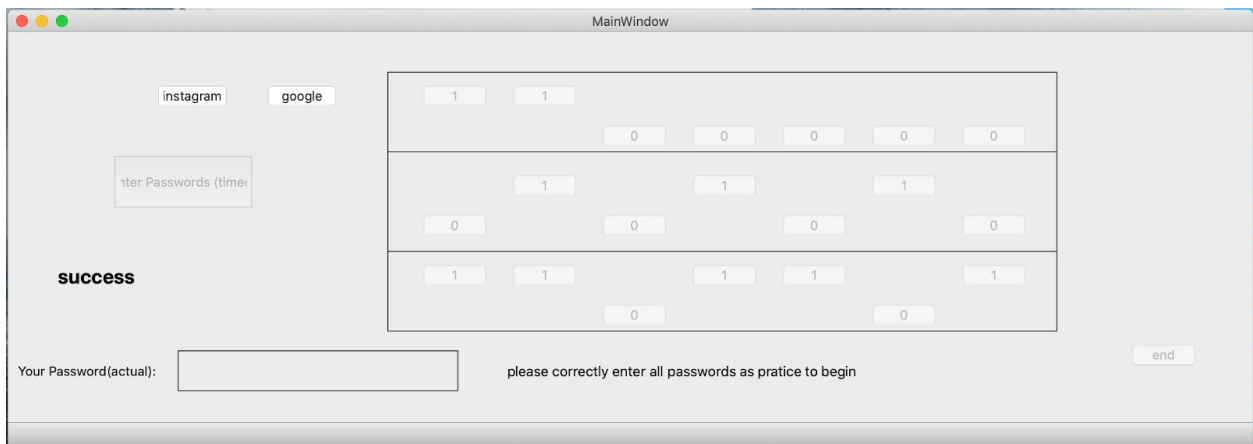


So, when we click on the start practice button, thereafter, there are 3 choices of passwords to choose from: Facebook, shopping, and banking. If the user clicks on either one of these they will get a random password. That'll be shown in the box. (see below figure)





Suppose the user clicks on the Facebook option. The program will give the set password, which the user will have to try to remember and get it right before moving onto the next to the momozation part. If we type it correctly in one of the passwords and press OK, then success will be shown and that facebook button will not be visible. The remaining two random passwords for Instagram and Google will be left. (see below figure)



When the user clicks on Instagram next, it will show the set password for that. Which the user will do the same as the step above.

Welcome to Password Scheme

Enter Passwords (timed)

**failure**

Your Password(actual):

1	1	1	1	1	1	1
0	0	0	0	0	0	0
1	1	1	1	1	1	1
0	0	0	0	0	0	0
1	1	1	1	1	1	1
0	0	0	0	0	0	0

MUST SELECT ALL BUTTONS

Press OK when done

ok

end

pls try again for instagram, you have 1 tries left

After the initial check for the users knowledge check, the actual fun will begin. The order of the logins will be randomized and the user will no longer be able to see the numerical password. The user will have three tries to finish each password. After all three passwords are entered the can press the end button (failure or success), and the data for the time elapsed, tries left, and success/fail for each password will be saved to a csv file.

=====source code of the program=====

**To run this code you need to download QT latest version and thereafter you need to click on file to import the project.**

**To compile and run the project you need to add all the kits and packages.**

**Apply to all and run.**

**Source code of part 2.2 is in folder program scheme 2.2**

**Source code of part 2.3 is in folder program scheme 2.3**

=====end of source code of the program information=====

## Part 2.4 Survey Questions

This is the link of the survey we used to test it with the participants

<https://hotsoft.carleton.ca/comp3008limesurvey/index.php/931842?lang=en>

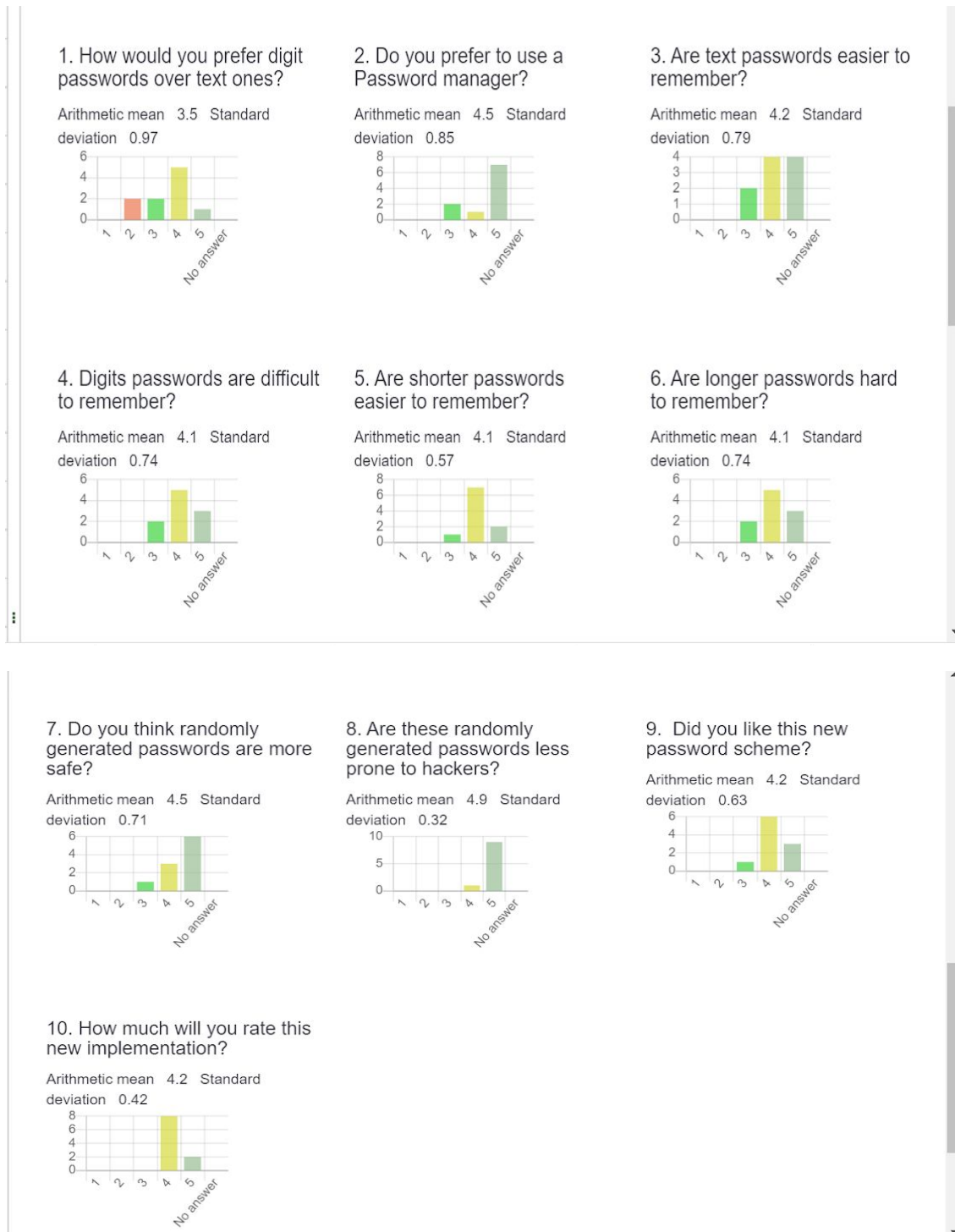
We recruited 10 students of Carleton University (including our own team members for the implementation of this password scheme).

Below are the questions we asked the users, we used Likert scale questions ( 1-5 ) for our survey questionnaires :

1. Would you prefer numeric passwords over text ones?
2. Do you prefer to use a password manager?
3. Are text passwords easier to remember?
4. Are numeric passwords difficult to remember?
5. Are shorter passwords easier to remember?
6. Are longer passwords difficult to remember?
7. Do you think randomly generated passwords are more safe?
8. Are these randomly generated passwords less prone to malicious attacks?
9. Did you like this new password scheme?
10. How would you rate this new implementation?

## Part 2.5 Survey Statistics ( Consent forms are attached in the appendix )

I have attached the statistics data we got from the responses from 10 users of the survey :



### **Summary of the data observed from responses :**

After carefully reading the responses from the 10 users, we know who the 10 users are but their responses were kept anonymous so they can choose answers freely without having doubts that we can see who submitted which one.

It was shown that as most of the participants belonged to the computer science field so they knew that how digits one passwords are more difficult to hack therefore most of them chose digit password over text ones ( mean =3.5).Almost everyone preferred password managers so they don't have to remember the password whenever required for logging in.( Mean = 4.5),this shows whether its a text password so some other scheme they will most prefer password managers. From other responses it has been observed that the majority of users prefer text passwords as their top preferred scheme ( mean = 4.2 ), users found longer passwords difficult to remember, and shorter passwords easier to remember as compared to longer ones.For security reasons , most of people rated digit passwords 4 and 5 out of scale of 5, another reason of this might be that long digits passwords seem more reliable and hard to be hacked by hackers therefore almost everyone found this reason quite reasonable advantage for this password scheme ( mean = 4.5 and 4.9 ).

To conclude , most of the people were neutral towards this new password scheme, most of them found it a little difficult because of having a long string of digits which they might have never used before. But overall no body disliked it as well for security reasons because if someone is using password managers then type and length of string shouldnt be an issue ( Mean= 4.2)



## Part 2.6 Usability Testing Data , Descriptive Statistics and Inferential Statistics.

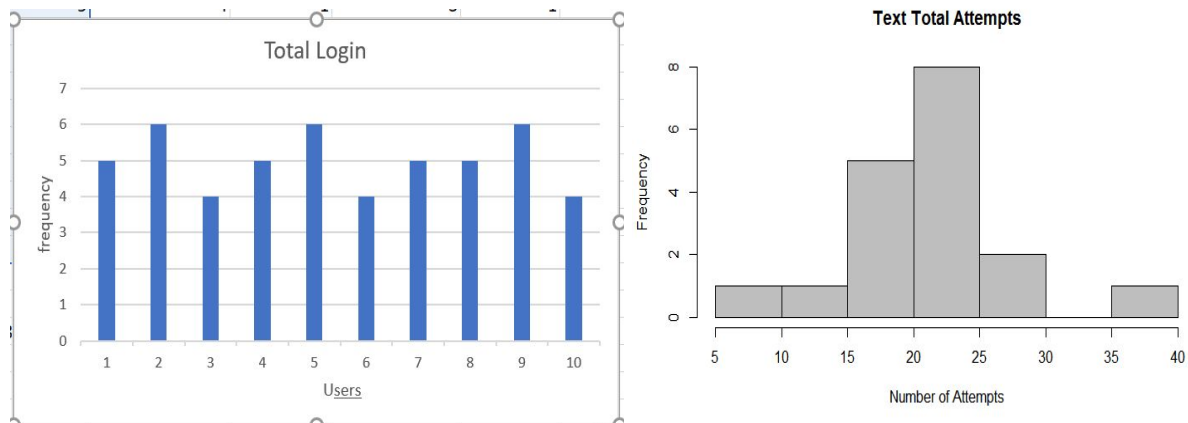
### Usability Testing Data ( Descriptive statistics of Digit Password Scheme )

*Digit-pwd.xlsx file submitted in zip folder is the data file for statistics purpose*

	Digits Password Scheme
Total Logins ( Mean )	5
Total Logins ( S.D )	0.8164
Total Logins ( Median )	5
Successful Logins ( Mean )	4.6
Successful Logins ( S.D )	0.51
Successful Logins ( Median )	5
Failed Logins ( Mean )	0.4
Failed Logins ( S.D )	0.51
Failed Logins ( Median )	0
Successful Times Login ( Mean )	3.2
Successful Times Login ( S.D )	1.03
Successful Times Login ( Median )	3
Failed Times Login ( Mean )	1.1
Failed Times Login ( S.D )	0.99
Failed Times Login ( Median )	1

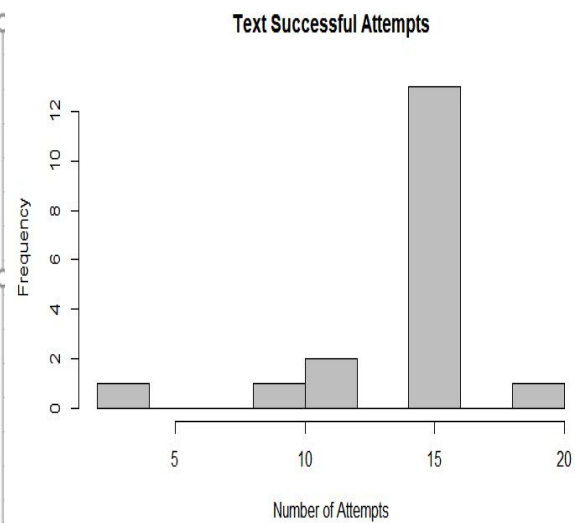
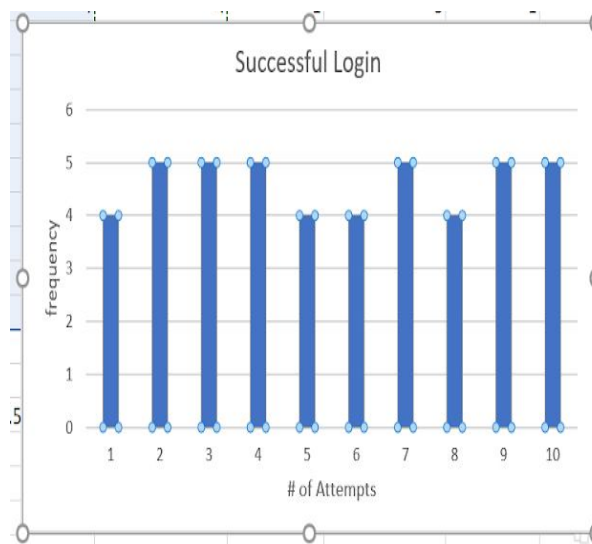
**-Blue chart represents the new password scheme data ( Total logins )**

This comparison has been conducted between the new password scheme we implemented and text password scheme. Both charts are different due to having limited users logged in for the new password scheme. One is histogram and other is a comparison chart of total login along with users and frequency of attempt.



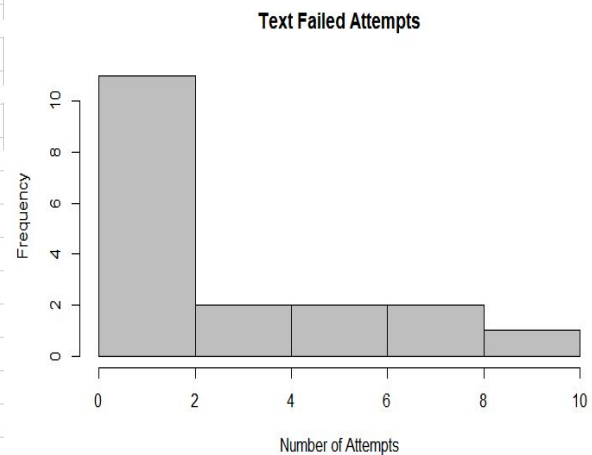
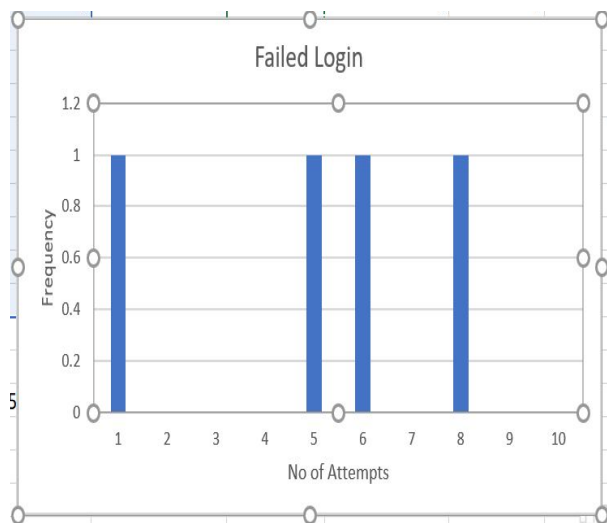
**Successful logins:**

The new password scheme and text password both show a high level of successful login due to having user friendly schemes. However some drop is being shown in the newer password scheme due to the fact having digit based password which is comparatively little hard to remember without having a password manager.



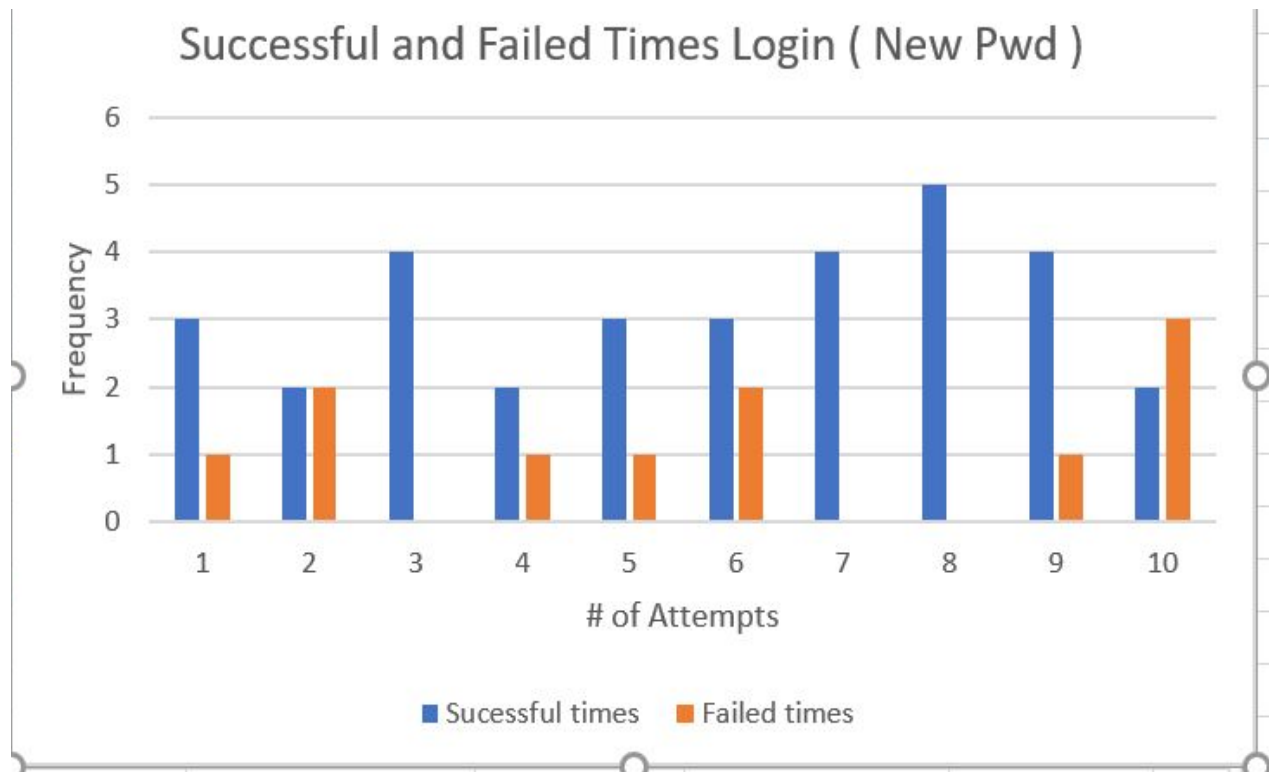
### Failed Logins:

This comparison shows that newer password schemes have more failed logins as compared to text failed attempts. This data was obvious due to the reason we saw before that as text based passwords had higher successful login, therefore another scheme which is digits one should be having a little high failed attempts.



### **Successful and Failed times login**

Below I have shown a comparative chart of successful and failed times login for the newer password scheme we implemented. Having tested it with 10 users the data shows that due to having randomly generated digit based it's still secure and when logging in it wouldn't outnumber successful times login with failed attempts. The pattern familiarity and the generating of random binary digits is giving users reliability and when they use this scheme they do not face greater numbers of failed attempts as they thought they would be. But it's being seen in the 10th number of attempts the failed time login increases which could be the glitch by the system of this scheme as most of the password schemes after too many failed attempts lock you out of the system and prefer to send you password link on your email due to security purposes.



### **Inferential Statistics: ( New password scheme Vs Text based scheme )**

**Hypothesis :** Variation of successful times and failed times login along with difference in time of such attempts.

$t = 1.567$ ,  $df = 17$ , p-value is 0.45

**Alternate Hypothesis :** The difference of means is not 0 in this data

**99 percent confidence interval**

-1.267      6.234

**X mean** = 9.098

**Y mean** = 6.434

*Explanation with comparative charts above has been written to show the pros and cons of New password scheme and text scheme passwords*

### **Workload Distribution**

<b>Group Member Names</b>	<b>Topics accomplished</b>
1. Tejal	<ul style="list-style-type: none"><li>- Part 1.1</li><li>- Part 1.4</li><li>- Part 2.4</li><li>- Part 2.5</li><li>- Part 2.6</li></ul>
2. Daniel	<ul style="list-style-type: none"><li>- Part 1.3</li><li>- Part 1.4</li><li>- ProofRead whole report</li></ul>
3. Khushal	<ul style="list-style-type: none"><li>- Part 2.1,2.3</li></ul>
4. Jason	<ul style="list-style-type: none"><li>- Part 2.2</li><li>- Part 2.3</li></ul>

### **Appendix ( Consent Forms )**

**Researchers' contact information:**

**Tejal Darpan**

Computer Science

Carleton University Tel:

Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood

School of Computer Science

Carleton University

Tel: 613-520-2600 x 8753

Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_Yes \_\_\_No

I agree to participate in this user study:

*Tejal*

Signature of participant

4/10/2020

Date

*Tejal*

Signature of researcher

4/10/2020

Date

**Researchers' contact information:**

**Tejal Darpan**

Computer Science

Carleton University Tel:

Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood

School of Computer Science

Carleton University

Tel: 613-520-2600 x 8753

Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_Yes \_\_\_No

I agree to participate in this user study:

*Khushal Singh*

Signature of participant

4/10/2020

Date

*Tejal*

Signature of researcher

4/10/2020

Date

**Researchers' contact information:****Tejal Darpan**

Computer Science  
Carleton University Tel:  
Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood  
School of Computer Science  
Carleton University  
Tel: 613-520-2600 x 8753  
Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_\_Yes \_\_\_\_No

I agree to participate in this user study:



Signature of participant

4/10/2020

Date



Signature of researcher

4/10/2020

Date

**Researchers' contact information:****Tejal Darpan**

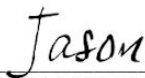
Computer Science  
Carleton University Tel:  
Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood  
School of Computer Science  
Carleton University  
Tel: 613-520-2600 x 8753  
Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_\_Yes \_\_\_\_No

I agree to participate in this user study:



Signature of participant

4/10/2020

Date



Signature of researcher

4/10/2020

Date



**Researchers' contact information:**

**Tejal Darpan**

Computer Science

Carleton University Tel:

Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood

School of Computer Science

Carleton University

Tel: 613-520-2600 x 8753

Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_ Yes \_\_\_ No

I agree to participate in this user study:



Signature of participant

4/10/2020

Date



Signature of researcher

4/10/2020

Date

**Researchers' contact information:**

**Tejal Darpan**

Computer Science

Carleton University Tel:

Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood

School of Computer Science

Carleton University

Tel: 613-520-2600 x 8753

Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_ Yes \_\_\_ No

I agree to participate in this user study:



Signature of participant

4/10/2020

Date



Signature of researcher

4/10/2020

Date

**Researchers' contact information:****Tejal Darpan**

Computer Science  
Carleton University Tel:  
Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood  
School of Computer Science  
Carleton University  
Tel: 613-520-2600 x 8753  
Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_ Yes \_\_\_ No

I agree to participate in this user study:



4/10/2020

\_\_\_\_\_  
Signature of participant

\_\_\_\_\_  
Date



4/10/2020

\_\_\_\_\_  
Signature of researcher

\_\_\_\_\_  
Date

**Researchers' contact information:****Tejal Darpan**

Computer Science  
Carleton University Tel:  
Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood  
School of Computer Science  
Carleton University  
Tel: 613-520-2600 x 8753  
Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_ Yes \_\_\_ No

I agree to participate in this user study:



4/10/2020

\_\_\_\_\_  
Signature of participant

\_\_\_\_\_  
Date



4/10/2020

\_\_\_\_\_  
Signature of researcher

\_\_\_\_\_  
Date

**Researchers' contact information:****Tejal Darpan**

Computer Science  
Carleton University Tel:  
Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood  
School of Computer Science  
Carleton University  
Tel: 613-520-2600 x 8753  
Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_ Yes \_\_\_ No

I agree to participate in this user study:



Signature of participant

4/10/2020

Date



Signature of researcher

4/10/2020

Date

**Researchers' contact information:****Tejal Darpan**

Computer Science  
Carleton University Tel:  
Email: [tejaldarpan@cmail.carleton.ca](mailto:tejaldarpan@cmail.carleton.ca)

**Supervisor contact information:**

Sana Maqsood  
School of Computer Science  
Carleton University  
Tel: 613-520-2600 x 8753  
Email: [sana.maqsood@carleton.ca](mailto:sana.maqsood@carleton.ca)

Do you agree to have your computer screen recorded: \_\_\_ Yes \_\_\_ No

I agree to participate in this user study:



Signature of participant

4/10/2020

Date



Signature of researcher

4/10/2020

Date