**Note**: The functions are first in the pseudo code, the calls to these functions in below them

**Pseudo code:**

//Variables

scheme\_type\_1 = ‘Image21’

scheme\_type2 = ‘Text21’

//Function to obtain a dataframe from a csv file

function csv\_to\_df(file){

//Obtain dataframe from csv

df = get\_df(file)

//Add headers

df = df.add\_header("time", "user", "site", "scheme", "mode", "event", "event details", "data")

//Drop unnecessary columns

df = df.drop\_column('site', 'data', 'scheme', 'mode')

//Keep rows that are login events or enter events with an event detail of start

df = df.keep\_rows(df[‘event’] == ‘login’ or (df[‘event’] == ‘enter’ and df['event details'] == 'start'))

return df

}

//Function to generate a new dataframe with the columns userid, login result (success or failure), and time (time taken to complete a login) using a given dataframe

function calculate\_time\_df (df){

//Variables

start = none //the start time

user = none // the user id of the user for the current login calculation

temp = [] //list to hold each row of the new dataframe

for each row in df{ //go through each row in the datafram

if(row[‘event] == “enter” and row[“event details”] == “start”{ //login has started in log data

user = row[‘user’] #store current userid

start = row[“time”] #store start time

}

else{ //is a login event

if start != none and user != none{ //ensure that login process has been started

if user == row[‘user’]{ //make sure that the current is the same as the one that started the login

//Add new row to temp

temp.append((user, row[‘event detail’], row[‘time’] - start))

}

}

#Reset the start time and current user

start = none

user = none

}

}

#Create a dataframe from the data in temp

Return dataframe(temp, headers = [“userid”, “login result”, “time”])

}

// Function to generate a dataframe with the number of logins for success, failure, and combined total for each user. Also mean of login time for success and failure for each user.

function calculate\_stats\_df(df){

//split the dataframe into two dataframes for success and failure

df\_success = df.keep\_rows("login result" == "success")

df\_fail = df.keep\_rows("login result" == "failure")

//drop the result columns

df = df.drop(['login result'])

df\_success = df\_success.drop(['login result'])

df\_fail = df\_fail.drop(['login result'])

//calculate number of logins (create dataframes) and name the new columns

df\_total\_count = df.groupby('userid').count("time").name(“total logins”) //total count

df\_success\_count = df\_success.groupby('userid').count("time").name("successful logins") //success count

df\_fail\_count = df\_fail.groupby('userid').count("time”).name("unsuccessful logins") //failed count

// mean of time (create dataframes) and name the new columns

df\_success\_mean = df\_success.groupby(‘userid’).mean(“time”).name("avg login time success (s)")

df\_fail\_mean = df\_fail.groupby('userid').mean("time").name("avg login time failed (s)")

//merge the frames

Resulting\_df = merge(df\_total\_count, df\_success\_count, df\_fail\_count, df\_success\_mean, df\_success\_mean)

//fill in any Nan values in the number of login columns

resulting\_df["successful logins", "unsuccessful logins"].fillna(value = 0)

return resulting\_df

}

// Function to join two dataframes and sort based on userid

Function merge\_df(df1, df2){

//Combine the two dataframes

result\_df = concat(df1, df2)

#sort by userid

Return result\_df.sortBy('userid')

}

//Function calls

//get dataframes from the two csv files

df\_image = csv\_to\_df('imagept21.csv') #image21

df\_text = csv\_to\_df('text21.csv') #text21

// Obtain a data frame with the time taken to complete logins

df\_image = calculate\_time\_df(df\_image)

df\_text = calculate\_time\_df(df\_text)

// Generate dataframe with mean value for success, and mean value for failure for each user

df\_image = calculate\_stats\_df(df\_image)

df\_text = calculate\_stats\_df(df\_text)

//add column pwd scheme to the dataframes

//also pass the password scheme as the value for each row

df\_image.insert("pwd scheme" = scheme\_type\_1)

df\_text.insert("pwd scheme" = scheme\_type\_2)

// join the two dataframes

df\_result = join\_and\_sort(df\_image, df\_text)

//Generate csv file

df\_result.to\_csv('combined.csv')