**TRIPTOPIA – YOUR OFFICIAL TRIP PLANNER**

**A MINI-PROJECT REPORT**

***Submitted by***

**Niranjana J 180701154**

**Padmashree J 180701159**

**Prathiksha M 180701171**

***in partial fulfillment of the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI**

**CHENNAI**

**APRIL 2019**

**BONAFIDE CERTIFICATE**

Certified that this project **“TRIPTOPIA-YOUR OFFICIAL TRIP PLANNER”** is the bonafide work of **“ Niranjana J, Padmashree J, Prathiksha M ”** who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **SIGNATURE**  **Dr.P.KUMAR M.E PhD.,**  **HEAD OF THE DEPARTMENT**  Dept. of Computer Science and Engg,  Rajalakshmi Engineering College,Chennai | **SIGNATURE**  **Mrs. JANANEE V**  **ASSISTANT PROFESSOR**  Dept. of Computer Scienceand Engg,  Rajalakshmi Engineering College,Chennai |

This mini project report is submitted for the viva vice examination to be held on \_\_\_\_\_\_\_\_\_\_

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ABSTRACT**

In this busy life, we find very little time for recreation or vacation. Nowadays more and more people would like to travel, explore and enjoy. A travel planning approach has recently been focus of interest for enhancing performance. Most traveler need to plan prior their traveling for energy saving and visiting maximum places of interest within the specific travel time.

Triptopia helps people around the world to organize their itinerary for their holidays. Whether you’re planning a vacation with a loved one abroad or inland, the app generates your personalized packing list based on destination, planned activities, transport, and other personal needs. Triptopia makes it easy for you to figure out what items you shouldn’t forget to pack into your backpack or suitcase for your weekend, business or holiday trip.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|  | **ABSTRACT** | **3** |
| **1** | **INTRODUCTION** | **5** |
| 1.1 | INTRODUCTION |  |
| 1.2 | SCOPE OF THE WORK |  |
| 1.3 | AIM AND OBJECTIVES OF THE PROJECT |  |
| **2** | **SYSTEM SPECIFICATIONS** | **6** |
| 2.1 | HARDWARE SPECIFICATIONS |  |
| 2.2 | SOFTWARE SPECIFICATIONS |  |
| **3** | **MODULE DESCRIPTION** | **7** |
| 3.1 | APP DESCRIPTION |  |
| 3.2 | REGISTRATION |  |
| 3.3 | LOGIN |  |
| 3.4 | TRIP ITINERARY |  |
| **4** | **SYSTEM DESIGN** | **8** |
| 4.1 | ARCHITECTURE DIAGRAM |  |
| **5** | **CODING** | **9** |
| **6** | **SCREENSHOTS** | **27** |
| **7** | **CONCLUSION** | **31** |
| **8** | **REFERENCES** | **32** |

**CHAPTER 1**

**INTRODUCTION**

**1.1 INTRODUCTION**

This project is an effort to help people plan their vacation, even at the last minute. This provides them an effective itinerary for their trip. This trip can either be a family vacation or a personal trip for the user, our application serves either way. This also provides the user a list of items they should pack for their trip and a rough expense for the day

**1.2 SCOPE OF THE WORK**

Our trip planner will make sure that our user has an hassle free itinerary and a vital packing list for for their trip. Enter the city and number of days, and it will tell you what it thinks you want to do in that city with an approximate budget for the day(hopefully you like what it chooses!).

**1.3 AIM AND OBJECTIVES OF THE PROJECT**

The objective of this application is to compile a travel itinerary for every day based on the length of the users trip and to provide a packing list and a rough estimate for the expenses for each day

**CHAPTER 2**

**SYSTEM SPECIFICATIONS**

**2.1 HARDWARE SPECIFICATIONS**

|  |  |  |
| --- | --- | --- |
| Processor | **:** | Intel i5 |
| Memory Size | **:** | 8GB (Minimum) |
| HDD | **:** | 1 TB (Minimum) |

**2.2 SOFTWARE SPECIFICATIONS**

|  |  |  |
| --- | --- | --- |
| Operating System | **:** | WINDOWS 10 |
| Front – End | **:** | Python |
| Back – End | **:** | Oracle |
| Language | **:** | Python |

**CHAPTER 3**

**MODULE DESCRIPTION**

**3.1 APP DESCRIPTION :**

The app Triptopia first directs to the Login or Register screen. If you are an already existing user,click on the Login button and proceed on with organizing your trip . Else,click on Register , enter your details and then log in, to organize your trip. Once you've logged in, you've to enter your destination and number of days of your trip. This lets the app to generate all the essentials needed for your trip,the nearby attractions and the approximate cost of your trip

**3.2 REGISTRATION MODULE :**

If a person needs to use this app,he first needs to register by entering his username,password,name,mobile number and email id . Once registered,he can log in anytime he wants to organize his trips.

**3.3 LOGIN MODULE :**

If you are an already existing user,just click on to the Login button to proceed on with organizing your trip. Once you log in,the app will redirect you to the trip itinerary page.

**3.4 TRIP ITINERARY MODULE :**

Once the user has logged in successfully,he has to choose his destination among the options . After choosing the destination,the user has to enter the number of days he will be staying . After the user has entered these two,the items which are required for the trip are displayed according to different categories followed by the nearby attractions for each day of the trip. The approximate cost for each day is also displayed towards the end.

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 ARCHITECTURE DIAGRAM**

Introduction

Login

Register

Registration Success

Login Failed

Login Success

Destination and Length of Stay

Cost for the Trip

Itinerary

Packing List

Linked List

Stack

(To display the items and link them)

(To add the cost and display it for each day)

**CHAPTER 5**

**CODING**

**File name: project**

from tkinter import \*

import os

def register():

global register\_screen

register\_screen = Toplevel(main\_screen)

register\_screen.title("Register")

register\_screen.geometry("800x450")

global username

global password

global name

global phno

global mail

global username\_entry

global password\_entry

global name\_entry

global phno\_entry

global mail\_entry

username = StringVar()

password = StringVar()

name = StringVar()

phno = StringVar()

mail = StringVar()

Label(register\_screen, text="Please enter details below").pack()

Label(register\_screen, text="").pack()

username\_lable = Label(register\_screen, text="Username \* ")

username\_lable.pack()

username\_entry = Entry(register\_screen, textvariable=username)

username\_entry.pack()

password\_lable = Label(register\_screen, text="Password \* ")

password\_lable.pack()

password\_entry = Entry(register\_screen, textvariable=password, show='\*')

password\_entry.pack()

name\_label=Label(register\_screen, text="Name \* ")

name\_label.pack()

name\_entry=Entry(register\_screen, textvariable=name)

name\_entry.pack()

phno\_label = Label(register\_screen, text="Mobile No. \* ")

phno\_label.pack()

phno\_entry=Entry(register\_screen,textvariable=phno)

phno\_entry.pack()

mail\_label=Label(register\_screen,text="E-Mail Id \* ")

mail\_label.pack()

mail\_entry=Entry(register\_screen,textvariable=mail)

mail\_entry.pack()

Button(register\_screen, text="Register", width=10, height=1, command = register\_user).pack()

def login():

global login\_screen

login\_screen = Toplevel(main\_screen)

login\_screen.title("Login")

login\_screen.geometry("300x250")

Label(login\_screen, text="Please enter details below to login").pack()

Label(login\_screen, text="").pack()

global username\_verify

global password\_verify

username\_verify = StringVar()

password\_verify = StringVar()

global username\_login\_entry

global password\_login\_entry

Label(login\_screen, text="Username \* ").pack()

username\_login\_entry = Entry(login\_screen, textvariable=username\_verify)

username\_login\_entry.pack()

Label(login\_screen, text="").pack()

Label(login\_screen, text="Password \* ").pack()

password\_login\_entry = Entry(login\_screen, textvariable=password\_verify, show= '\*')

password\_login\_entry.pack()

Label(login\_screen, text="").pack()

Button(login\_screen, text="Login", width=10, height=1, command = login\_verify,bg="blue").pack()

def register\_user():

username\_info = username.get()

password\_info = password.get()

name\_info = name.get()

phno\_info=phno.get()

mail\_info=mail.get()

file = open(username\_info, "w")

file.write(username\_info + "\n")

file.write(password\_info + "\n")

file.write(name\_info + "\n")

file.write(phno\_info + "\n")

file.write(mail\_info + "\n")

file.close()

username\_entry.delete(0, END)

password\_entry.delete(0, END)

name\_entry.delete(0,END)

phno\_entry.delete(0,END)

mail\_entry.delete(0,END)

Label(register\_screen, text="Registration Success", fg="green", font=("calibri", 11)).pack()

def login\_verify():

username1 = username\_verify.get()

password1 = password\_verify.get()

username\_login\_entry.delete(0, END)

password\_login\_entry.delete(0, END)

list\_of\_files = os.listdir()

if username1 in list\_of\_files:

file1 = open(username1, "r")

verify = file1.read().splitlines()

if password1 in verify:

login\_sucess()

else:

password\_not\_recognized()

else:

user\_not\_found()

def login\_sucess():

global login\_success\_screen

login\_success\_screen = Toplevel(login\_screen)

login\_success\_screen.title("Success")

login\_success\_screen.geometry("150x100")

Label(login\_success\_screen, text="Login Success").pack()

Button(login\_success\_screen, text="OK", command=delete\_login\_success).pack()

def password\_not\_recognized():

global password\_not\_recog\_screen

password\_not\_recog\_screen = Toplevel(login\_screen)

password\_not\_recog\_screen.title("Success")

password\_not\_recog\_screen.geometry("150x100")

Label(password\_not\_recog\_screen, text="Invalid Password ").pack()

Button(password\_not\_recog\_screen, text="OK", command=delete\_password\_not\_recognized).pack()

def user\_not\_found():

global user\_not\_found\_screen

user\_not\_found\_screen = Toplevel(login\_screen)

user\_not\_found\_screen.title("Success")

user\_not\_found\_screen.geometry("150x100")

Label(user\_not\_found\_screen, text="User Not Found").pack()

Button(user\_not\_found\_screen, text="OK", command=delete\_user\_not\_found\_screen).pack()

def delete\_login\_success():

login\_success\_screen.destroy()

def delete\_password\_not\_recognized():

password\_not\_recog\_screen.destroy()

def delete\_user\_not\_found\_screen():

user\_not\_found\_screen.destroy()

def main\_account\_screen():

global main\_screen

main\_screen = Tk()

main\_screen.geometry("800x450")

main\_screen.title("Account Login")

Label(text="Welcome to Triptopia", bg="blue", width="300", height="2", font=("Calibri", 15)).pack()

Label(text="Select Your Choice", bg="blue", width="300", height="2", font=("Calibri", 13)).pack()

Label(text="").pack()

Button(text="Login", height="2", width="30", command = login).pack()

Label(text="").pack()

Button(text="Register", height="2", width="30", command=register).pack()

main\_screen.mainloop()

main\_account\_screen()

import triptopia

**File name: Triptopia**

import destinations

import paris

import newyork

import london

import cost

import packinglist

def main():

print("Welcome to the Trip Planner\n")

destinations.print\_options()

choice = destinations.get\_choice()

destination = destinations.get\_info(choice)

while True:

try:

print("How many days will you be staying in",destination,"?")

global length\_of\_stay

length\_of\_stay = int(input())

if (length\_of\_stay < 0):

print("Please enter a positive number of days.")

continue

except ValueError:

print("Only numerical values are valid.")

else:

break

itnry(destination, length\_of\_stay)

def itnry(destination, length\_of\_stay):

print("Triptopia\n-----------------------------")

print("\nHere's your itinerary for", length\_of\_stay,"days")

get\_plans(destination)

print("Have a pleasant stay at", destination)

def get\_plans(destination):

while destination=="London":

packinglist.pl.printnode()

if length\_of\_stay==1:

london.day1()

print("cost = ",cost.n.item[0],"£")

break

elif length\_of\_stay==2:

london.day2()

print("cost = ",cost.n.item[1],"£")

break

elif length\_of\_stay==3:

london.day3()

print("cost = ",cost.n.item[2],"£")

break

while destination=="Paris":

packinglist.pl.printnode()

if length\_of\_stay==1:

paris.day1()

print("cost = ",cost.n.item[0],"$")

break

elif length\_of\_stay==2:

paris.day2()

print("cost = ",cost.n.item[1],"$")

break

elif length\_of\_stay==3:

paris.day3()

print("cost = ",cost.n.item[2],"$")

break

while destination=="New York":

packinglist.pl.printnode()

if length\_of\_stay==1:

newyork.day1()

print("cost = ",cost.n.item[0],"$")

break

elif length\_of\_stay==2:

newyork.day2()

print("cost = ",cost.n.item[1],"$")

break

elif length\_of\_stay==3:

newyork.day3()

print("cost = ",cost.n.item[2],"$")

break

# Call main

main()

**File name: packinglist**

class Node:

def \_\_init\_\_(self,data,nextnode=None):

self.data=data

self.nextnode=nextnode

def getdata(self):

return self.data

def setdata(self,data):

self.data=data

def getnextnode(self):

return self.nextnode

def setnextnode(self,val):

self.nextnode=val

class packing\_list:

def \_\_init\_\_(self,head=None):

self.head=head

self.tail=None

self.size=0

def addnodel(self,data):

if self.head==None:

newnode=Node(data,self.head)

self.head=newnode

self.size+=1

else:

c=self.head

newnode=Node(data,self.tail)

while(self.size>1 and c.getnextnode()!=None):

c=c.getnextnode()

c.setnextnode(newnode)

self.size+=1

return True

def printnode(self):

c=self.head

while c:

print(c.data)

c=c.getnextnode()

pl=packing\_list()

pl.addnodel("Essentials and Clothing:")

pl.addnodel("\tShirts")

pl.addnodel("\tPants")

pl.addnodel("Electronics:")

pl.addnodel("\tMobile phone")

pl.addnodel("\tChargers")

pl.addnodel("Hygiene:")

pl.addnodel("\tToothbrush")

pl.addnodel("\tToothpaste")

pl.addnodel("Documents:")

pl.addnodel("\tPassport")

**File name:cost**

class amount:

def \_\_init\_\_(self):

self.item=[]

def push(self,a):

self.item.append(a)

n=amount()

for i in range(10,1500,50):

n.push(i)

print(n.item)

**File name:london**

def day1():

print("\nDay 1:\n10:00am-10:30am : The Buckingham Palace\n11:00am-12:30pm : Coca-Cola London Eye\n01:00pm-03:30pm : The Big Ben\n04:00pm-06:00pm : The Shard")

def day2():

day1()

print("\nDay 2:\n10:00am-01:30pm :Madame Tussauds London\n02:00pm-06:00pm : Tower Bridge")

def day3():

day2()

print("\nDay 3:\n10:30am-02:30pm : Westminster abbey\n03:30pm-06:00pm : St.Paul's Cathedral")

**File name: Paris**

def day1():

print("\nDay 1:\n10:00am-10:30am : Place de la Bastille\n11:00am-12:30pm : Coulee Verte Rene-Dumont\n01:00pm-03:30pm : Lac des Minimes\n04:00pm-06:00pm : La Tour Eiffel")

def day2():

day1()

print("\nDay 2:\n10:00am-01:30pm : Musee d'Orsay\n02:00pm-06:00pm : Louvre Musee")

def day3():

day2()

print("\nDay 3:\n10:00am-12:30pm : Le Marais\n01:00pm-03:00pm : Canal Saint Martin\n03:30pm-05:00pm : 2nd Arrondissement\n05:30pm-06:00pm : Les Drapeaux de France")

**File name: Newyork**

def day1():

print("\nDay 1:\n11:00am-11:30am : Hawk’s Nest Highway\n12:30pm-1:30pm : Roebling Aqueduct Suspension Bridge\n2:00pm-4:30pm : Upper Delaware River")

def day2():

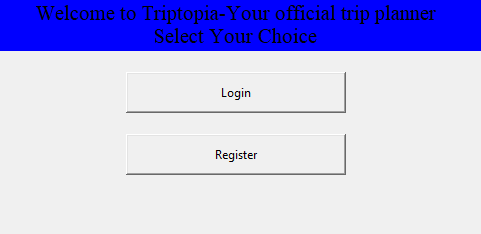
day1()

print("\nDay 2:\n10:00am-11:30am : 9/11 Memorial\n12:00pm-60:00pm : Central Park")

def day3():

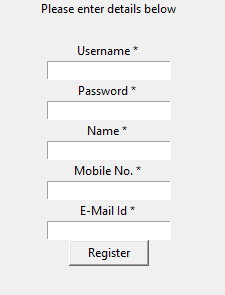
day2()

print("\nDay 3:\n11:30am-4:30pm : Ski Big Bear")

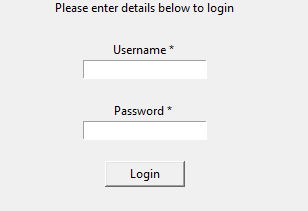
**CHAPTER 6**

**SCREEN SHOTS**

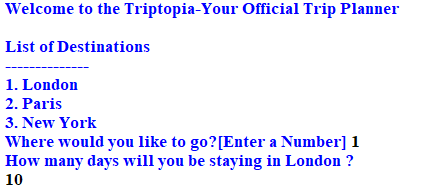
**Fig 6.1 Introduction Page**

****

**Fig 6.2 User Registration**

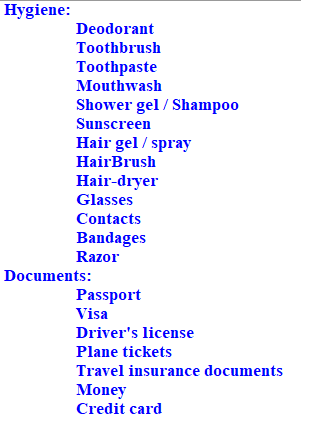
****

**Fig 6.3 User Login**

****

**Fig 6.4 Output for User After Login**

****

****

**Fig 6.5Packing list for the user**

**Fig 6.7 Itinerary for the user**

****

**CHAPTER 7**

**CONCLUSION AND FUTURE ENHANCEMENT**

The project is titled “Triptopia- Your Official Trip Planner”. It provides the user a convenient itinerary for their desired destination and duration of stay. This project includes all the concepts of DataBase Management Systems.

However, this project can be improved in such a way that it turns into a real time application. It can be created into an App that could be helpful and beneficial to many people during their travel.

**REFERENCES**

1. <https://www.w3schools.com/>
2. <https://www.tutorialspoint.com/>
3. <https://www.geeksforgeeks.org/python-gui-tkinter/>
4. <https://www.learnpython.org/>
5. <https://www.python-course.eu/python_tkinter.php>