AP PROJECT REPORT

Book a Buggy (Travel-App)

Akshay Kalidatta - 200953106
Siddhant Gupta - 200953092
Aditya Chandra - 200953032
Vinayak Mehrotra - 200953054

Index:

Abstract

Planning a proper trip with the best traveling option is very paramount for anyone. Therefore our aim in this project is to develop a user friendly platform where a person can book a buggy for any suitable time and date. It is a one-stop shop for buggy booking.
Students can't afford to lose their precious time just in the name of walking to various destinations. They can't even spend hefty amounts as well.
Not only the students face problems due to absence of the application but buggy managers also face the problem of not receiving payments as many people tend to skip the payment and go away.
Already there are buggy services inside the MIT campus but the problem faced by the students is that there isn't any particular application for booking the same. Our aim is not only to provide booking options within the campus but we have further aimed at providing options to travel from one MAHE campus to another as well, for eg: KMC to MIT or MIT to MIC, etc.
The project is developed using Tkinter and sqlite3.
The page encompasses details of the services as well as the contact information.User can also provide feedback based on their experience. There is also an option to see the booking history. This system is effective and saves time and cost of users. It also provides a secure, reliable and fast system.

Introduction

Going to class at 8 AM is a nightmare most students are familiar with. To make this journey more manageable, we have developed a buggy ticket booking system that students can use to make easy reservations for the buggies moving around campus. Using this app, students can be assured of their seats on the buggies and have the payment sorted before they get on.

The application uses a reservation system to help students so they don't have to wait long for transportation and end up getting late for places. This will also be the first application to have connectivity outside MIT campus and all over Manipal, including places like KMC, Tiger Circle, and Edu building.

Background Information

Back in 2020 when a startup launched E-Taxi services in Manipal, commonly known as ‘Buggy’, many students loved the idea of having an option to book a buggy on their way to classes instead of walking. The idea seemed very intriguing for everyone in the campus, but since it was a fresh idea, it lacked many important features which could really make this service a convenience.

Our product comes into the picture to provide a real seamless service and to address many issues. These issues include the uncertainty of the timings of the buggy which bothered the students in deciding whether to wait for a buggy or walk to their classes. Many times, students wait for the buggy a few minutes before the class starts hoping for a buggy to arrive but it either doesn’t arrive or arrives with all the seats filled. Another problem is the time crunch while paying when students are in a hurry. Our service provides a smooth booking service which includes easy payment, even availability of seats won’t be an issue because our service provides multiple buggies running in the interval of 5-10 minutes.

Methodology

Our initial plan was to provide a service that provides a seamless experience to book a buggy that helps travel across the campus and Manipal without the hassle of waiting long periods.
Our product consists of several pages which include a feedback, “About Us”, book and cancel page.

The main feature of our application revolves around booking and canceling a buggy.
The booking page includes the source and destination of the ride and the timing at which the rides are available, along with passenger details which include Name, Gender and Phone Number.
Our next important page is canceling a ride, which takes the phone number as the input and cancels the last booked ticket using that.
We also have an about us page that helps users get to know more about what the product does.

One of our most unique features includes a feedback page where the users can help us improve the application.
Our final page is the payments page which gives the users two options for payment which are Pay using Cash or Pay using UPI.
This is an overview of what the application is capable of.

Implementation

We have divided our project into 6 different pages (about-page, booking-page, cancel-page, feedback-page, show-current-bookings-page, services-page)

For saving the current bookings we have created an SQLite3 database, which has the following schema:

Tickets: :

```
    ORIGIN TEXT NOT NULL,
    DEST TEXT NOT NULL,
    NAME TEXT NOT NULL,
    DAY TEXT,
    MONTH TEXT,
    YEAR TEXT,
    GENDER TEXT,
    TIMING TEXT,
    MOBILE TEXT NOT NULL,
    PAIDTHROUGH TEXT NOT NULL,
    AMOUNTPAID TEXT NOT NULL
```

For storing user feedbacks we have created another table which will be storing all the feedbacks in the backend.

FeedBacks:

```
NAME TEXT NOT NULL,
GENDER TEXT,
MOBILE TEXT NOT NULL,
BOOKINGEXPERIENCE TEXT NOT NULL,
CUSTOMERSERVICE TEXT NOT NULL,
CALLSERVICE  TEXT NOT NULL,
PAYMENTEXPERIENCE TEXT NOT NULL
```

Main Function Structure:

1. About_page.py:
   This page gives a brief description about our company. It is completely informational and has no such important functionalities.

2. Service_page.py:
   This is also a page which contains a brief description of the services we provide as a company. This is also a page with no important functionality.

3. Feedback_page.py:
   This is a page through which users provide us valuable feedback on our booking system using which we as a company strive to excel and improve our technology. We save all the feedback received in our database for further reference.

4. Show_booking_page.py:
   This page basically shows all the bookings made by a user in the past. We retrieve this information from our database which is constructed while the bookings are done.

5. Cancel_page.py:
   This page cancels any booking that is previously made by a user by taking their phone number as input (Primary key).The record of the booking is also deleted from the database.

6. Main_page.py:
   This is the main page that is displayed as soon as the application starts running. It consists of all the pages described above and also the necessary details required for the booking to be made are taken as an input from this page. When all the details are filled , it redirects us to the payment page.

7. Payment_page.py:
   This page is displayed after the main page and has 2 options which are pay using cash and pay using credit/debit card. The first option simply redirects us to the main page and the second option redirects us to the online payment portal.

Following are some snapshots from the implementation of the code.

```
swidth = root.winfo_screenwidth()
sheight = root.winfo_screenheight()

# fr = tk.Frame(root, image=background_image, width=swidth, height=sheight).pack()
bon = Label(root, text="Book-A-Buggy", font=("lucida calligraphy", 40, "bold"), bd=0).place(x=400, y=10)
label_me = tk.Label(root).place(x=0, y=80, width=swidth, height=70)

about_button = tk.Button(root, text="About", bd=0, fg='white', font=("Heveltica", 15), command=about)
about_button.place(x=10, y=100, width=150, height=40)

services_button = tk.Button(root, text="Services", bd=0, fg='white', font=("Heveltica", 15), command=service)
services_button.place(x=150, y=100, width=150, height=40)S

feedback_button = tk.Button(root, text="Feedback", bd=0, fg='white', font=("Heveltica", 15), command=feed)
feedback_button.place(x=290, y=100, width=150, height=40)

show_booking = tk.Button(root, text="Show Bookings", bd=0, fg='white', font=("Heveltica", 15), command=showbok)
show_booking.place(x=440, y=100, width=150, height=40)

cancel_button = tk.Button(root, text="Cancel", bd=0, fg='white', font=("Heveltica", 15), command=cancel)
cancel_button.place(x=590, y=100, width=150, height=40)

book_frame = tk.Frame(root, bg="grey", width="1100",height="700").place(x=50, y=200)
book_head_label = tk.Message(root, text="Book Your Tickets", width="500", bd=0, bg="grey", font=("Heveltica", 30)).place(x=420, y=220)

from_input = StringVar()
to_input = StringVar()
name_input = StringVar()
gender_input = StringVar()
mobile_input = StringVar()
year_input = StringVar()
month_input = StringVar()
```

Adding and Combining all the functions to the main page.

```
            label_to1 = tk.Label(rootT, text=b, font=payFont, bd=0, bg = "#161617", fg = "White").place(x=1100, y=250)
            label_tot = tk.Label(rootT, text="TRAVELLING DATE  : ", font=payFont, bd=0, bg = "#161617", fg = "White").place(x=160, y=350)
            label_tod = tk.Label(rootT, text=date, font=payFont, bd=0, bg = "#161617", fg = "White").place(x=600, y=350)
            label_to3 = tk.Label(rootT, text="AMOUNT PAID    :", font=payFont, bd=0, bg = "#161617", fg = "White").place(x=160, y=450)
            label_to2 = tk.Label(rootT, text=price, font=payFont, bd=0, bg = "#161617", fg = "White").place(x=600, y=450)
            label_stb = tk.Label(rootT, text="BOOKING STATUS :", font=payFont,bg = "#161617", fg = "White").place(x=150, y=530)
            label_s = tk.Label(rootT, text="CONFIRMED", font=payFont,bg = "#161617", fg = "White").place(x=500, y=530)

            def lahome():
                rootT.destroy()
                home()
            but_return = tk.Button(rootT, text="home ", font=(
                "Heveltica", 15), bg="Lightblue", command=lahome).place(x=500, y=600, width=350, height=40)

            rootT.mainloop()
        bau_paytm = tk.Button(roop, text="PAY WITH THE CASH", font=(
            "lucida calligraphy", 20), bg="lightblue", command=paywith).place(x=600, y=400)
        new = 1
        url = "https://p-y.tX/UC-jmiq"

        def callback(url):
            webbrowser.open_new(url)
        link1 = tk.Label(roop, text="PAY WITH DEBIT/CREDIT/INTERNET BANKING",
                         bg="lightblue", cursor="hand2", font=("lucida calligraphy", 20))
        link1.place(x=460, y=500)
        link1.bind(
            "<Button-1>", lambda e: callback("https://p-y.tm/UC-jmiq"))
        roop.mainloop()

submit_button = tk.Button(root, text="Proceed ", font=("Heveltica", 18, "bold"), bg="Lightgreen", command=pay).place(x=700, y=700, width=420, height=40)
root.mainloop()
```

Setting up Modes of Payments, Online mode redirects to Paytm.

```
    p_exp_enr = ttk.Combobox(roof, textvariable=p_exp, values=val).place(
        x=110, y=650, height=40, width=450)

    def feedsql():
        a = name_of.get()
        b = mobile_num.get()
        l = len(b)
        c = genderf.get()
        d = b_e.get()
        e = c_s.get()
        f = cll_s.get()
        g = p_exp.get()
        dfed = (a, c, b, d, e, f, g)
        conn.execute(('''INSERT INTO  FEEDBACKT('NAME','GENDER','MOBILE','BOOKINGEXPERIENCE','CUSTOMERSERVICE','CALLSERVICE','PAYMENTEXPERIENCE')VALUES(?,?,?,?,?,?,?)'''),
        dfed)
        conn.commit()
        messagebox.showinfo("SUCESS", "FEEDBACK SUBMITTED SUCESSFULLY")
        roof.destroy()
        # home()
    b_submit = tk.Button(roof, text="SUBMIT", font=("lucida calligraphy", 15), bg="Lightblue", command=feedsql).place(x=750, y=650, width=350, height=40)
    roof.mainloop()
```

Booking the ticket and inserting it into a database

```
    def dell():
        a = mob_can.get()
        print(type(a))
        print(a)
        conn = sqlite3.Connection("example.db")
        conn.execute('''DELETE FROM TEST3 WHERE MOBILE = ?''', (a,))
        conn.commit()

        messagebox.showinfo(
            "Congratulations!!", "YOUR TICKET WAS SUCESFULLY CANCELLED")
        rooc.destroy()
        # home()

    def home_cancel():
        rooc.destroy()
        # home()

    cancel_button = tk.Button(rooc, text="Cancel Ticket", font=("Heveltica", 18), command=dell, bg="Red", fg="white", width=25, height=3).place(x=200, y=550)
    bu_t_home = tk.Button(rooc, text="Home", font=("Heveltica", 18), command=home_cancel, bg="blue", fg="white", width=25, height=3).place(x=600, y=550)
```

Cancel ticket implementation

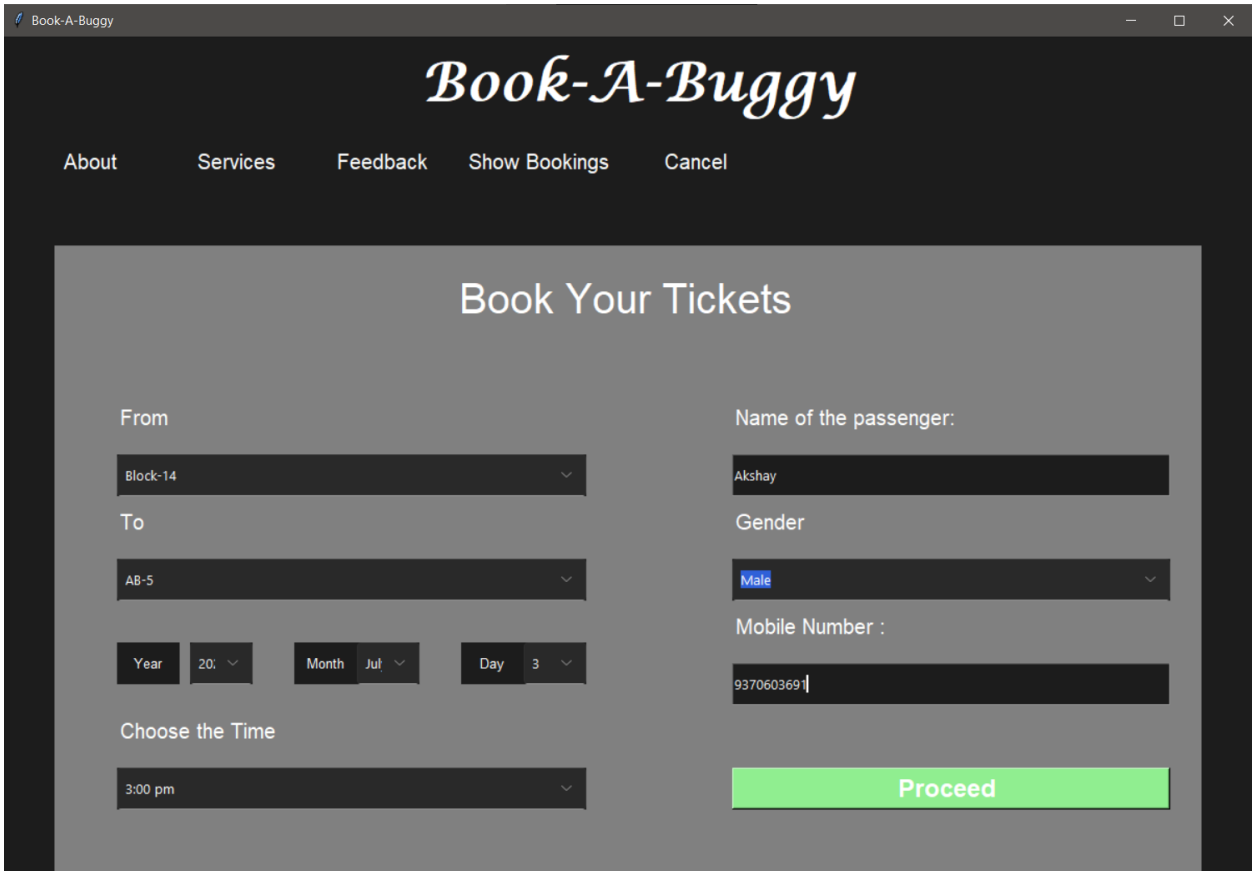
```python
def showbok():
        # root.destroy()
        ros = tk.Tk()
        sv_ttk.set_theme("dark")
        ros.configure()
        ros.geometry("1200x1200")
        conn = sqlite3.Connection("example.db")
        cu = conn.cursor()
        cu.execute("""SELECT * FROM TEST3""")
        Label(ros, text="From")
        for i in cu.fetchall():
            lab = tk.Label(ros, text=i, font=('Heveltica', 14)).pack()

        def homeb():
            ros.destroy()
            # home()
        bu_t = tk.Button(ros, text="home", command=homeb, bg="blue", font=(
            "lucida calligraphy", 15), fg="white", width=15, height=3).place(x=600, y=600)
```

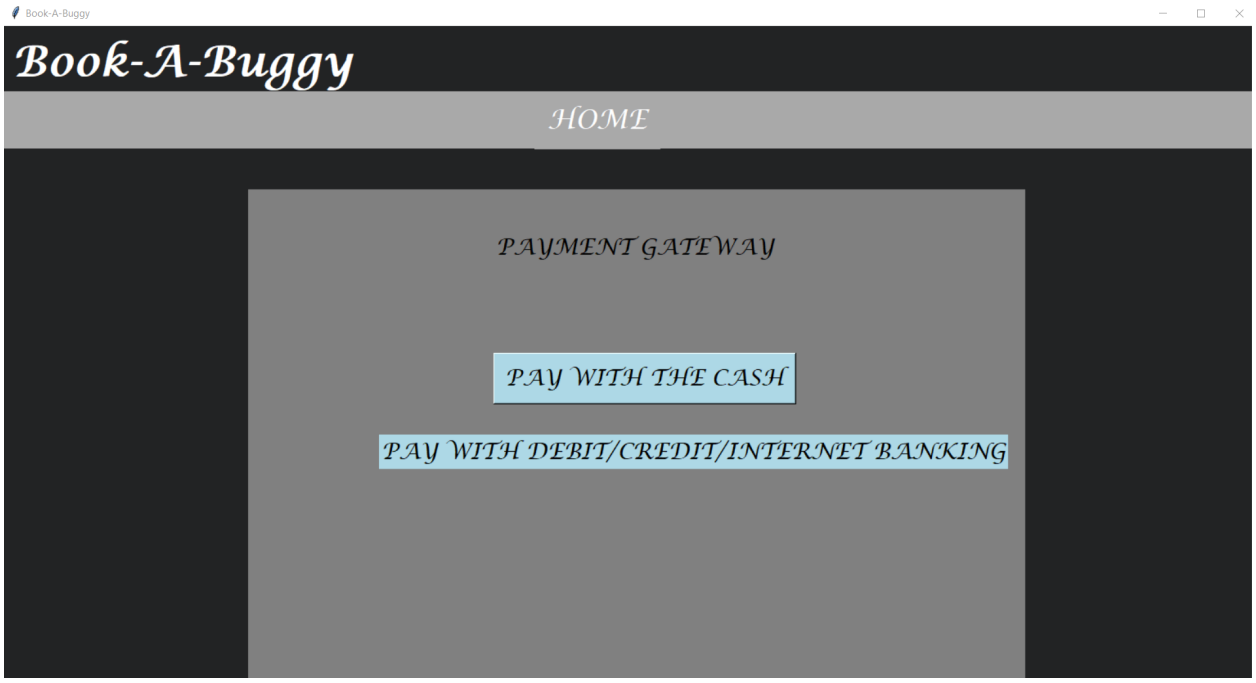
Implementation to see all bookings

Results

Main Page:

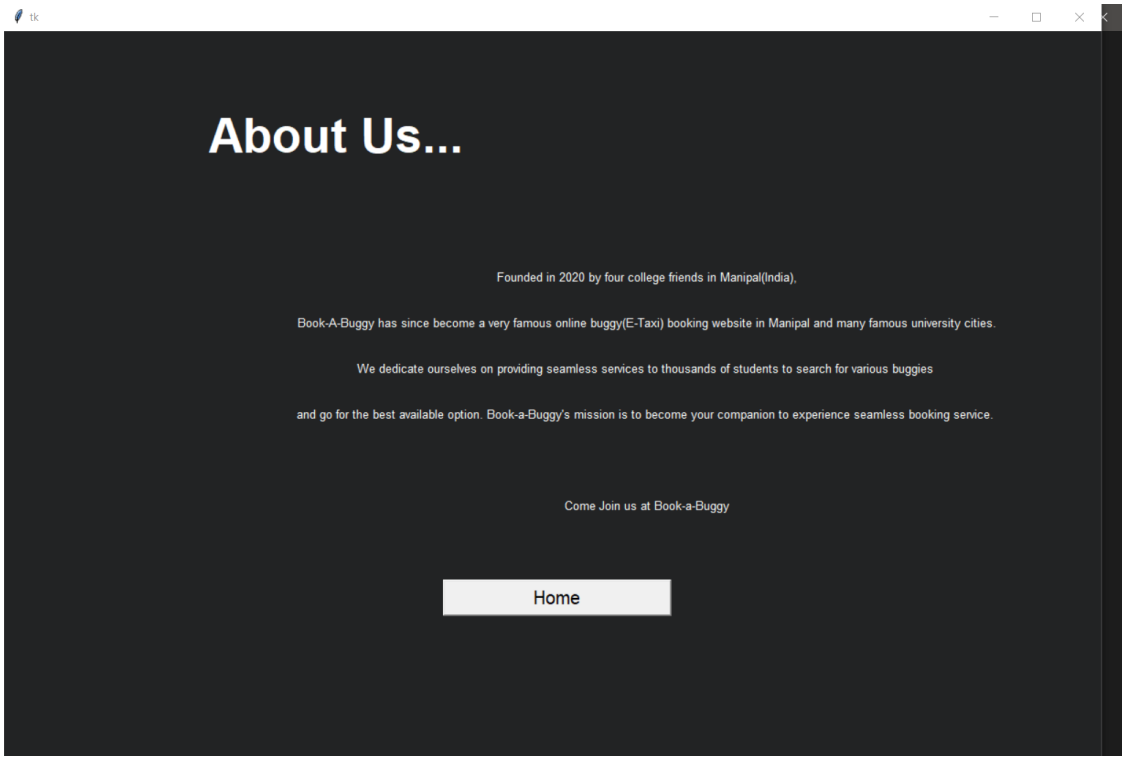


Payment page:

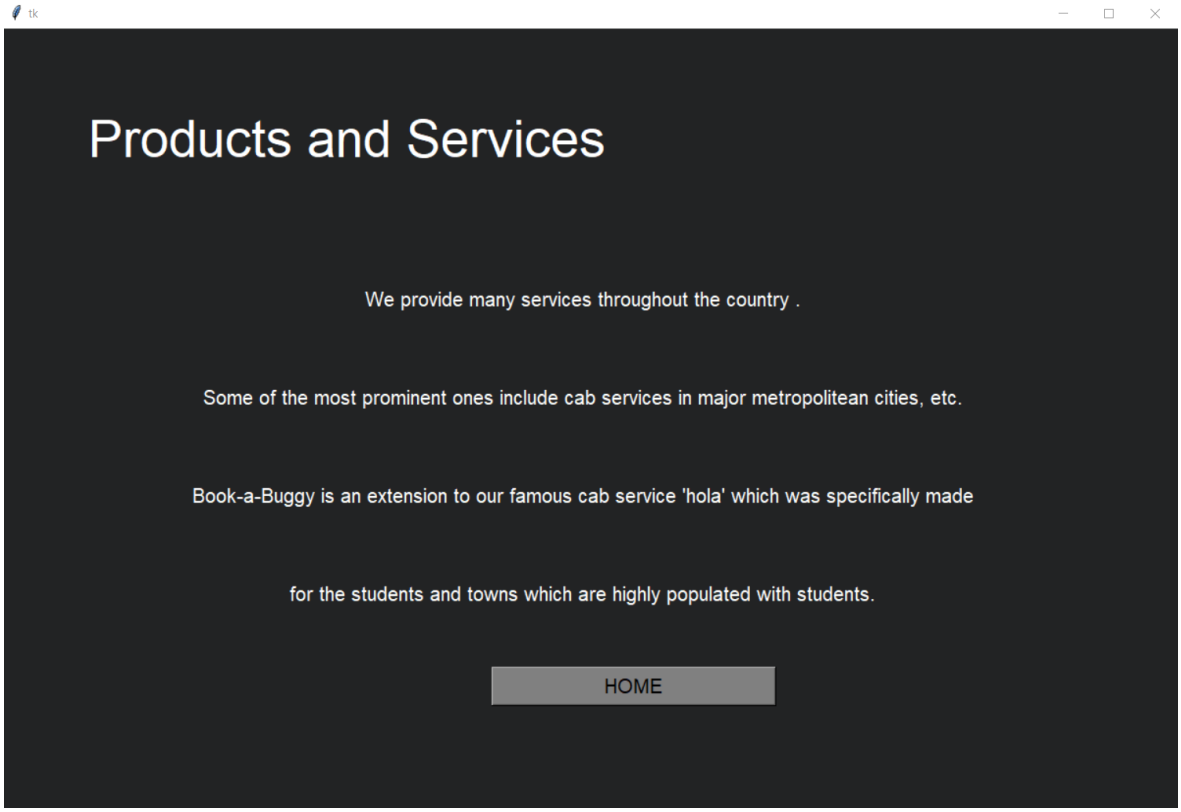


Booking details:

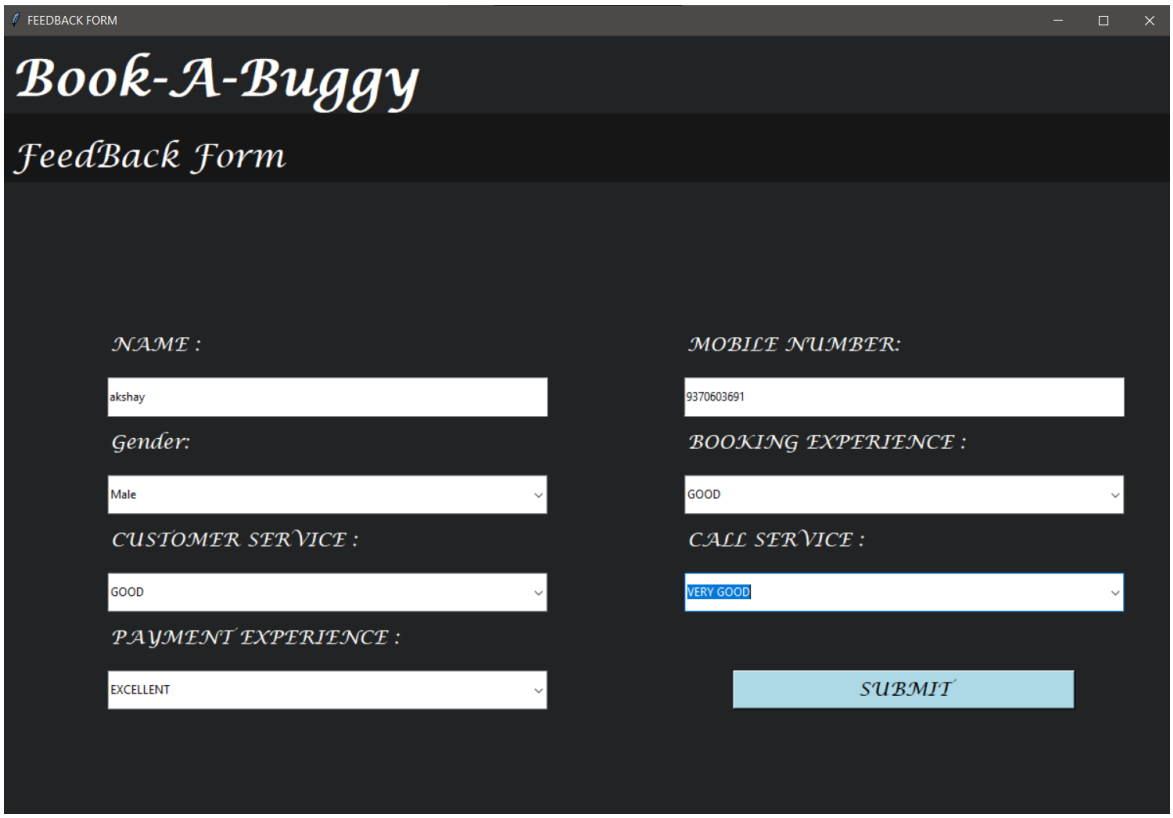
About us Page:



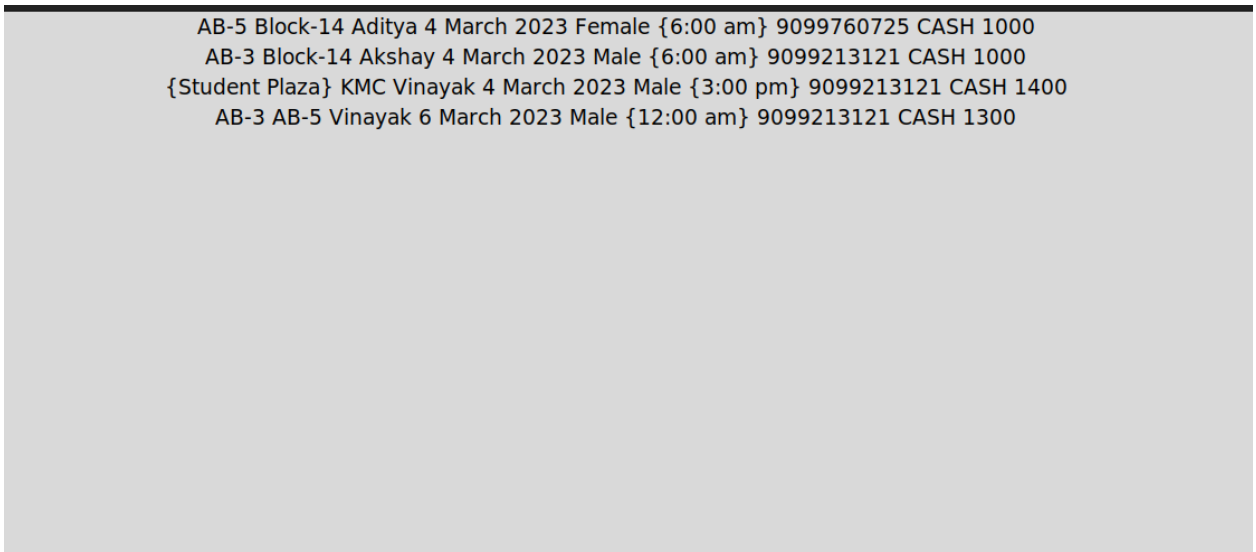
Services Page:



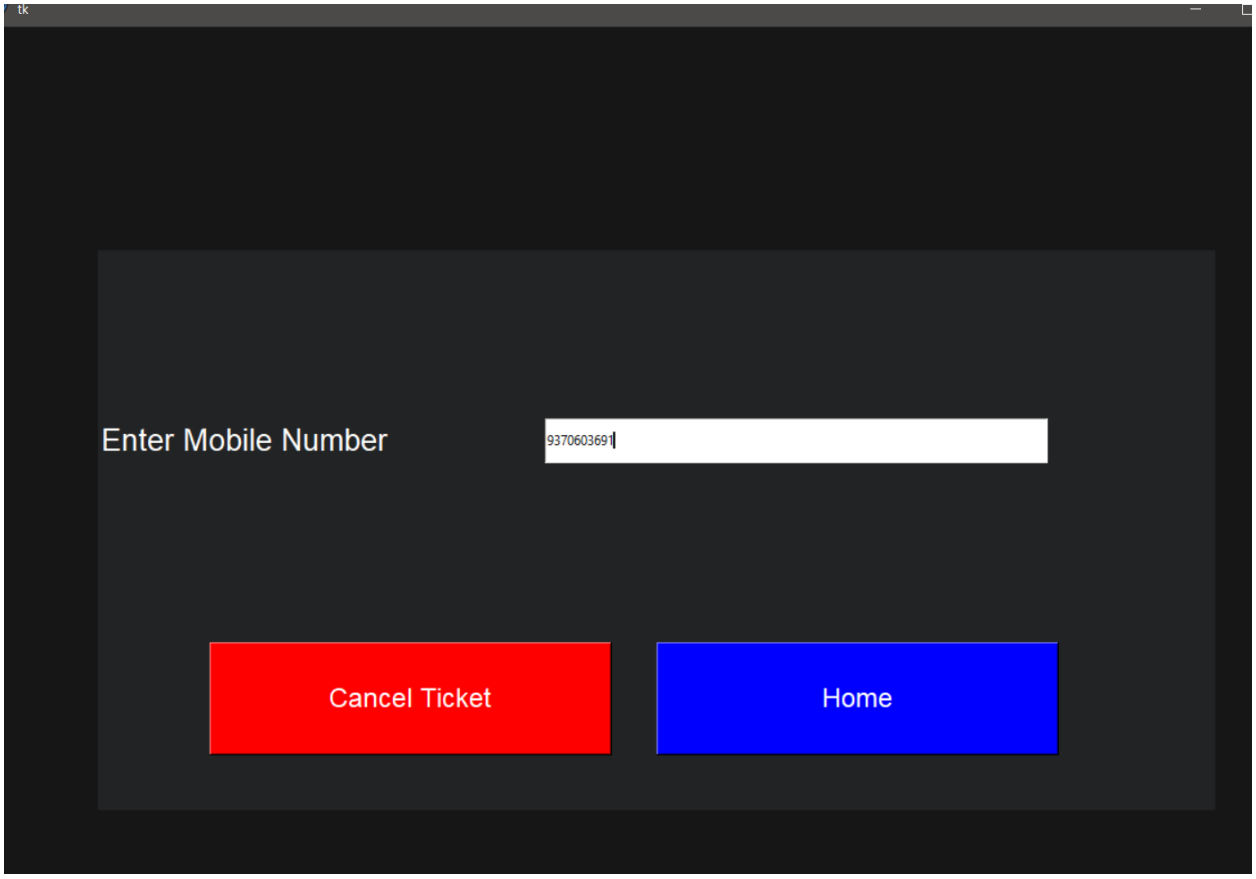
Feedback Page:



Show Bookings page:



Cancel page:



Conclusion

With the help of this application, we achieved a very smooth booking process for students and brought about certainty regarding the availability of buggies and their timings. We strive to provide and maintain this hassle free process to all our customers. They can now book any buggy according to their desired timing and pickup points.

Although our application solves many issues, it still has a lot of scope for improvement. One of the most prominent features we would like to include is to improve the online payment portal. Other improvements might include live tracking of the buggies which will help the students estimate the arrival of buggies and plan accordingly. We can also include the feature of chatting with the drivers to convey important messages.

References

1. https://docs.python.org/3/library/tk.html

2. https://docs.python.org/3/library/sqlite3.html