



**Gestor de Flota de Autobuses
FlotAPP**

Asignatura: Programación Orientada a Objetos

Integrantes:

Mateo Echavarria Sierra
Miguel Angel Fonseca Aldana
Haison Urrutia Manyoma
Juan Pablo Pineda Lopera

Universidad Nacional de Colombia Sede Medellín

2022-1

1. Descripción general de la solución:

Se pretende llevar a cabo la implementación de una aplicación de escritorio en Java bajo el paradigma de programación orientada a objetos que permita administrar diferentes funcionalidades a una flota de autobuses (Compra de tiquetes, recomendación de destinos, gestión de conductores, especialistas y viajes, visualizar estadísticas y calcular la rentabilidad de cada viaje).

- Visualizar estadísticas: Permite visualizar porcentaje de ocupación por viaje, número de visitas por ciudad y de acuerdo al porcentaje de ocupación de cada viaje se modifica la frecuencia (mayor al 85% aumenta la frecuencia en 1 hora, si está entre el 40%-60% disminuirá la frecuencia del viaje en 2 horas y si es menor al 10% brindará la opción de eliminar o continuar con el viaje).
- Gestión de conductores: Se puede visualizar los conductores existentes, asignar una tarea, despedirlo, visualizar el historial de viajes asignados.
- Gestión Especialistas: Se puede visualizar los especialistas existentes dependiendo de su especialidad, asignarles un vehículo a revisar de los que existen, ver historial de vehículos revisados, despedirlo.
- Gestionar viaje: Pide un número de cédula con el que se lograra acceder a los tiquetes que compró, se le permite gestionar sus tiquetes donde puede cambiar un tiquete para una fecha posterior o simplemente otra silla y cancelar un tiquete.
- Compra de tiquetes: Un usuario puede comprar un tiquete, (haciendo referencia a que está en la terminal de transporte ubicada en Medellín y saldrá de allí) para un viaje disponible de acuerdo al nombre de la ciudad que desea viajar y los tiquetes disponibles.
- Recomendación de destino: A un usuario se le recomendará un destino de acuerdo a: 1) Si el usuario ya tiene un historial de viajes existentes, recomendará la ciudad más visitada de acuerdo a si tiene una promoción asignada en la lista de promociones. Y 2) Si es usuario nuevo recomendará la ciudad más visitada con descuento existente en la lista de descuentos.
- Calcular rentabilidad de cada viaje: Permite calcular el porcentaje de ocupación del viaje seleccionado, las ganancias y utilidades por cada viaje y el promedio de ocupación y utilidades de la ruta.
- Aplicar Bono Empleado: Se visualizan todos los empleados con su especialidad o conductor, se pregunta por cual desea aplicar el bono y su sueldo aumenta.

2. Descripción del diseño estático del sistema en la especificación UML

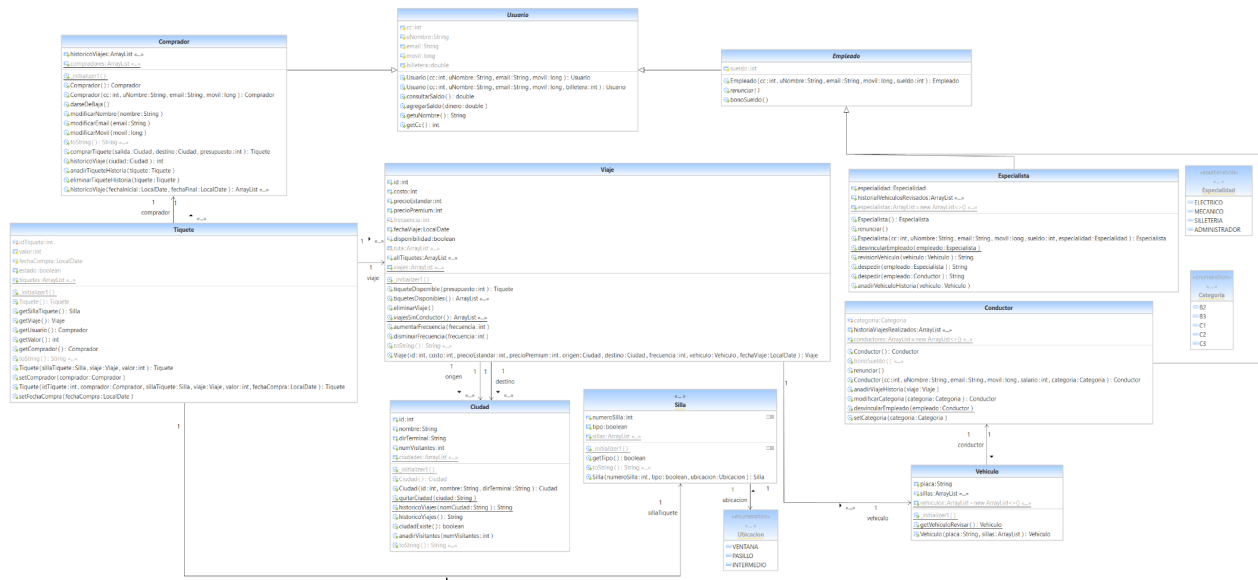


Imagen completa

3. Descripción de la Implementación de características de programación orientada a objetos en el proyecto

- **Clases Abstracta y Métodos Abstractos**
- **Herencia**

```

9
10 public abstract class Usuario implements Serializable {
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
104
```

Las clases **Usuario** y **Empleado** se implementan como abstractas Empleado heredando de Usuario. El Empleado define el método abstracto **renunciar** con el fin de obligar a las clases hijas **Conductor** y **Especialista** a implementar dicho método.

Las dos clases pretenden por medio de la herencia la reutilización de código por parte de sus subclases.

- Interfaces
- Ligadura dinámica (1)

```
16 public void bonoSueldo() {  
17     sueldo += sueldo*0.1; //Se b
```

```
46 @Override  
47 public void bonoSueldo() {  
48     sueldo += sueldo*0.15; //bono del 15% por ligadura dinamica  
49     Conductor superConductor = null;
```

```
@Override  
public String toString() {  
    return "Id: " + super.cc + "\n"+  
           "Nombre: " + super.uNombre+"\n"+  
           "Sueldo: " + super.sueldo+"\n"+  
           "Especialidad: " + especialidad+"\n";  
}
```

Bono sueldo de la clase **Empleado** aumenta el salario en un 10% a los empleados, pero si este es un **Conductor** aumenta en un 15% y si a su vez el conductor es quien ha realizado más viajes se le aplica el 30% de aumento en su salario.

- Ligadura Dinámica (2)

```
public static void desicionEspecialistas(Especialista especialista){
```

```
public static Viaje asignarVehiculo(Conductor conductor, Viaje viaje){
```

```
public static String asignarVehiculo(Especialista mecanico, Vehiculo vehiculo){
```

Asignar Vehículo le asigna a un viaje a un empleado - conductor y añade ese viaje a el historial de viajes realizados, si el empleado es un especialista se le asigna la revisión y se añade al historial de vehículos revisados

- Atributos de clase y métodos de clase

```
9 private static ArrayList<Especialista> especialistas = new ArrayList<>();
```

```
44 public static void visualizarEstadisticas(){
```

En las clases donde se generan instancias, estas son almacenadas en tiempo de ejecución del programa en listas de tipo static y del mismo modo los métodos para las funcionalidades fueron declarados de tipo estático para poder ser llamado sin necesidad de instanciar,

- **Uso de constante**
- **Implementación de un caso de enumeración**

```
public enum Ubicacion {  
    VENTANA, PASILLO, INTERMEDIO  
}
```

```
public enum Especialidad {  
    ELECTRICO, MECANICO, SILLETERIA, ADMINISTRADOR
```

```
3 public enum Categoria {  
4     B2, B3, C1, C2, C3
```

Se en la aplicación se implementaron 3 clases enumeradas, **Especialidad**, **Categoría** y **Ubicación** las cuales responden a llamado desde la clase **Especialista**, **Conductor** y **Silla** respectivamente.

- **Encapsulamiento (private, protected y public)**

```
7 public class Ciudad implements Serializable {  
8     private int id;  
9     private String nombre;  
10    private String dirTerminal;  
11    private int numVisitantes;  
12    private static ArrayList<Ciudad> ciudades;  
13    static {  
14        ciudades = new ArrayList<Ciudad>();
```

```
public abstract class Usuario implements Serializable {  
  
    protected int cc;  
    protected String uNombre;  
    protected String email;  
    protected long movil;  
    protected double billetera;
```

```
    public static void desvincularEmpleado(Conductor empleado) {  
        Conductor.conductores.remove(empleado);  
    }
```

En la aplicación se implementó los atributos de tipo privado, los métodos públicos y para la clase **Usuario** se implementaron los atributos de tipo protected con el fin de que fueran accesibles para las clases hijas.

- **Sobrecarga de métodos y constructores**

Método

```
public static Viaje asignarVehiculo(Conductor conductor, Viaje viaje){
```

```
public static String asignarVehiculo(Especialista mecanico, Vehiculo vehiculo){
```

```
public String despedir(Especialista empleado){
```

```
public String despedir(Conductor empleado){
```

Constructores

```
11= public Especialista(){  
12     super(0, "ESPECIALISTA NO REGISTRADO", "noemail@error.exe", 666, 0); this.especialidad = Especialidad.ADMINISTRADOR;  
13 }  
14  
15= public Especialista(int cc, String uNombre, String email, long movil, int sueldo, Especialidad especialidad) {  
16     super(cc, uNombre, email, movil, sueldo);  
17     this.especialidad = especialidad;  
18     this.historialVehiculosRevisados = new ArrayList<Vehiculo>();  
19     Especialista.especialistas.add(this);  
20 }
```

```
public Conductor(){super(0, "CONDUCTOR NO REGISTRADO", "noemail@error.exe", 666, 0);}
```

```
public Conductor(int cc, String uNombre, String email, long movil, int salario, Categoria categoria) {
```

```
//CONSTRUCTOR  
public Comprador(){super(cc 0, uNombre: "USUARIO NO EXISTENTE", email: "us  
  
public Comprador(int cc, String uNombre, String email, long movil) {  
    super(cc, uNombre, email, movil, billetera: 0);  
    this.historicoViajes = new ArrayList<>();  
    Comprador.compradores.add(this);  
}
```

- Manejo de referencias this para desambiguar y this() entre otras

this para desambiguar

```
28= public Viaje(int id, int costo, int precioEstandar, int precioPremium, Ciudad origen, Ciudad destino, int frecuencia,  
29     Vehiculo vehiculo, LocalDate fechaViaje) {  
30  
31     this.id = id;  
32     this.costo = costo;  
33     this.precioEstandar = precioEstandar;  
34     this.precioPremium = precioPremium;  
35     this.origen = origen;  
36     this.destino = destino;  
37     this.frecuencia = frecuencia;  
38     this.fechaViaje = fechaViaje;  
39     this.vehiculo = vehiculo;  
40     this.allTiquetes = new ArrayList<>();  
41  
--  
23= public Ciudad(int id, String nombre, String dirTerminal) {  
24     this.id = id;  
25     this.nombre = nombre;  
26     this.setDirTerminal(dirTerminal);  
27     ciudades.add(this);  
--
```

otros usos del this

```
87= public boolean ciudadExiste() {  
88     if(ciudades.contains(this)) {  
89         return true;  
90     }else {  
91         return false;  
--
```

```

60 public String historicoViajes() {
61
62     String hV = "";
63     if (!ciudades.isEmpty()) {
64         for (Ciudad c: ciudades) {
65             if (c.nombre.equals(this.nombre)) {

```

4. Muestra de funcionalidad:

Usuario

Funcionalidades:

- 1) Visualizar Estadísticas
- 2) Gestionar Conductores
- 3) Gestionar Especialistas
- 4) Gestionar Viaje - (cc)
- 5) Compra de Tiquete
- 6) Recomendación
- 7) Rentabilidad Viajes
- 8) Aplicar Abono Empleado

1

VisualizarEstadisticas.Menu

id: 1 - Nombre: MEDELLÍN
Número de Visitante: 0

id: 12 - Nombre: BELLO
Número de Visitante: 0

id: 7 - Nombre: POPAYÁN
Número de Visitante: 0

id: 8 - Nombre: CALI
Número de Visitante: 2

Viaje: 1 - Origen: Ciudad: BELLO
id: 12
Con dirección del terminal: calle Y - 72
Total de visitantes: 0 - Destino: Ciudad: CALI
id: 8
Con dirección del terminal: calle F - 13
Total de visitantes: 2

id: 5 - Nombre: MONTERÍA
Número de Visitante: 0

id: 4 - Nombre: CARTAGENA
Número de Visitante: 1

Viaje: 2 - Origen: Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0 - Destino: Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 1

id: 6 - Nombre: PASTO

Número de Visitante: 3

Viaje: 4 - Origen: Ciudad: MONTERIA

id: 5

Con dirección del terminal: circular 4 # 1 - 44

Total de visitantes: 0 - Destino: Ciudad: PASTO

id: 6

Con dirección del terminal: Carrera 24 # 4 - 22

Total de visitantes: 3

id: 9 - Nombre: BARRANQUILLA

Número de Visitante: 0

id: 3 - Nombre: MANIZALES

Número de Visitante: 0

Viaje: 3 - Origen: Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0 - Destino: Ciudad: MANIZALES

id: 3

Con dirección del terminal: avenida 24 # 3-23

Total de visitantes: 0

Dime el ID del viaje que deseas gestionar :

1

Promedio de ocupación: 25.0%

Esta funcionalidad muestra el promedio de ocupación del viaje lo cual nos permite dependiendo la ocupación del mismo, eliminar el viaje o esperar (tener fé) hasta que se compran tiquetes por parte de los usuarios para ese viaje.

También puede darse el caso que el viaje no esté registrado, esto se debe a que fue eliminado debido a su poca ocupación.

2

GestionarConductores.Menu

CC: 28 - Nombre: Don Javie
Cantidad de Viajes asignados: 1

CC: 29 - Nombre: Don Hernan
Cantidad de Viajes asignados: 1

CC: 30 - Nombre: Dona Marta
Cantidad de Viajes asignados: 0

Dime la CC del conductor que deseas gestionar:

29

Don Hernán.Gestionar Conductores

- 4) Visualizar Historial de viajes Asignados
- 5) Asignar un Viaje
- 6) Despedir

4

VisuaizarHistorialdeViajesAisgnados.GestionarConductores

*Viaje{id=3, origen=Ciudad: Medellin
id: 1
Con dirección del terminal: calle x-95
Total de visitantes: 0, destino=Ciudad: MANIZALES
id: 3
Con dirección del terminal: avenida 24 # 3-23
Total de visitantes: 0, fecha Viaje=2022-06-18}*

Esta funcionalidad se encarga de hacer gestiones sobre los conductores que hay disponibles en el programa. Empezando por seleccionar al conductor deseado, luego tienes las opciones de visualizar el historial de viajes asignados al conductor, asignar un viaje o despedir.

3

GestionarEspecialistas.Menu

- 1) Eléctrico, 2) Mecánico, 3) Silleteria

1

Eléctrico.Gestionar Especialistas

ELÉCTRICO - CC: 78 - Nombre: Maria
Cantidad de vehículos revisados: 1

ELÉCTRICO - CC: 88 - Nombre: Maria

Cantidad de vehículos revisados: 0

ELÉCTRICO - CC: 98 - Nombre: Maria

Cantidad de vehículos revisados: 0

Dime la CC del especialista que deseas gestionar:

88

Maria.GestionarEspecialistas

4) Visualizar Historial de vehículos Asignados

5) Asignar un vehículo

6) Despedir

6

Despedir.GestionarEspecialistas

ESPECIALISTA: Maria DESPEDIDO

Esta funcionalidad se obtiene al ingresar a gestionar especialistas, a partir de ahí tenemos el menú con las diferentes ocupaciones de los especialistas, luego eliges el especialista que quieres gestionar y tenemos el menú para visualizar historial de vehículos asignados, asignar un vehículo o despedir al especialista.

4

GestionarViaje.Menu

Gestionar Viaje - CC:

1

CC:1 - Mateo

id: [0]= Tiquete = ID : 7, SILLA{numero Silla=6, tipo=true, ubicación=PASILLO}

VIAJE = Viaje{id=1, origen=Ciudad: BELLO

id:12

Con dirección del terminal: calle Y - 72

Total de visitantes: 0, destino=Ciudad: CALI

id: 8

Con dirección del terminal: calle F - 13

Total de visitantes: 2, fecha Viaje=2022-06-15}, valor : 22000, fecha Compra : 2022-06-03

Dime el ID del viaje que deseas gestionar;

12

VIAJE NO REGISTRADO

La funcionalidad gestionar viaje se encarga de visualizar los viajes registrados por los usuarios según el Id del viaje, si se ingresa un id que no pertenece al viaje esto nos dirá que el viaje no está registrado.

5

CompraTiquete.Menu

Ciudad a la que desea viajar:

CARTAGENA

id : [0] = Tiquete = ID : 0, SILLA :Silla{numeroSilla=7, tipo=false, ubicación=VENTANA}

VIAJE =Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 1, fecha Viaje=2022-06-18}, valor : 1100, fecha Compra : 2022-06-18

id : [1] = Tiquete = ID : 1, SILLA :Silla{numeroSilla=8, tipo=false, ubicación=PASILLO}

VIAJE =Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 1, fecha Viaje=2022-06-18}, valor : 1100, fecha Compra : 2022-06-18

id : [2] = Tiquete = ID : 2, SILLA :Silla{numeroSilla=1, tipo=true, ubicación=VENTANA}

VIAJE =Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 1, fecha Viaje=2022-06-18}, valor : 17000, fecha Compra : 2022-06-18

id : [3] = Tiquete = ID : 3, SILLA :Silla{numeroSilla=2, tipo=true, ubicación=PASILLO}

VIAJE =Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 1, fecha Viaje=2022-06-18}, valor : 17000, fecha Compra : 2022-06-18

id : [4] = Tiquete = ID : 4, SILLA :Silla{numeroSilla=3, tipo=true, ubicación=VENTANA}

VIAJE =Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 1, fecha Viaje=2022-06-18}, valor : 17000, fecha Compra : 2022-06-18

id : [5] = Tiquete = ID : 5, SILLA : Silla{numeroSilla=4, tipo=true, ubicación=PASILLO}
VIAJE =Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 1, fecha Viaje=2022-06-18}, valor : 17000, fecha Compra : 2022-06-18

id : [6] = Tiquete = ID : 6, SILLA : Silla{numeroSilla=5, tipo=true, ubicación=VENTANA}

VIAJE =Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 1, fecha Viaje=2022-06-18}, valor : 17000, fecha Compra : 2022-06-18

1

Tiquete = ID : 1, SILLA : Silla{numeroSilla=8, tipo=false, ubicación=PASILLO}

VIAJE =Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 2, fecha Viaje=2022-06-18}, valor : 1100, fecha Compra : 2022-06-03

La funcionalidad compra tiquete se filtra según la ciudad de destino, esto mostrará los tiquetes disponibles para dicha ciudad acompañado de datos como dirección de la terminal, precio del viaje, sillas disponibles, fechas de compra, fecha del viaje y total visitantes.

Los tiquetes son elegidos por su ID y este muestra el valor del viaje, fecha de compra, fecha del viaje y total de visitantes.

6

Recomendacion.Menu

Recomendar Viaje - CC:

3

Te recomendamos viajar a:

Ciudad: PASTO

id: 6

Con dirección del terminal: Carrera 24 # 4 - 22

Total de visitantes: 3

La funcionalidad recomendación se encarga de mostrarle al usuario una recomendación de viaje basándose en la ciudad con más visitas.

7

RentabilidadViajes.Menu

Viaje{id=4, origen=Ciudad: MONTERIA

id: 5

Con dirección del terminal: circular 4 # 1 - 44

Total de visitantes: 0, destino=Ciudad: PASTO

id: 6

Con dirección del terminal: Carrera 24 # 4 - 22

Total de visitantes: 3, fecha Viaje=2022-06-15}

Viaje{id=3, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: MANIZALES

id: 3

Con dirección del terminal: avenida 24 # 3-23

Total de visitantes: 0, fecha Viaje=2022-06-18}

Viaje{id=1, origen=Ciudad: BELLO

id: 12

Con dirección del terminal: calle Y - 72

Total de visitantes: 0, destino=Ciudad: CALI

id: 8

Con dirección del terminal: calle F - 13

Total de visitantes: 2, fecha Viaje=2022-06-15}

Viaje{id=2, origen=Ciudad: MEDELLIN

id: 1

Con dirección del terminal: calle X - 95

Total de visitantes: 0, destino=Ciudad: CARTAGENA

id: 4

Con dirección del terminal: calle G - 14

Total de visitantes: 2, fecha Viaje=2022-06-18}

Digite el id del viaje al cual le quiere calcular la rentabilidad

1

Viaje{id=1, origen=Ciudad: BELLO

id: 12

Con dirección del terminal: calle Y - 72

Total de visitantes: 2, fecha Viaje=2022-06-16}

id: 8

Con dirección del terminal: calle F - 13

Total de visitantes: 2, fecha Viaje=2022-06-15}

La ocupación del vehículo fue del 24%, con 8 sillas disponibles en total.

Para este viaje se generó por tickets \$44000 y su costo de operación fue de 30000 con una del 14000

La ocupación promedio para la ruta de BELLO-CALI es del 24% y su utilidad promedio de 14000

La funcionalidad rentabilidad viajes muestra la rentabilidad de un viaje según el valor del ticket en ese momento, calcula el costo de operación por viaje y le resta el valor de ticket mostrando así la rentabilidad.

8

AplicarBono.Menu

Aplicar Bono Empleado - CC:

Id: 27

Nombre: Jose

Sueldo: 3500

Especialidad: MECÁNICO

Id: 78

Nombre: Maria

Sueldo: 4000

Especialidad: ELÉCTRICO

Id: 88

Nombre: Maria

Sueldo: 4000

Especialidad: ELÉCTRICO

Id: 98

Nombre: Maria

Sueldo: 4000

Especialidad: ELÉCTRICO

Id: 32

Nombre: Pablo

Sueldo: 3700

Especialidad: MECÁNICO

Id: 12

Nombre: Edgar

Sueldo: 3000

Especialidad: SILLETERIA

Id: 102

Nombre: Edgar

Sueldo: 3000

Especialidad: SILLETERIA

Id: 120

Nombre: Edgar
Sueldo: 3000
Especialidad: SILLETERIA

Id: 28
Nombre: Don Javie
Sueldo: 4000
Categoría: B3
Viajes realizados : 1

Id: 29
Nombre: Don Hernan
Sueldo: 4100
Categoría: C1
Viajes realizados : 1

Id: 30
Nombre: Dona Marta
Sueldo: 4200
Categoría: C2
Viajes realizados : 0

Digite la cc del Empleado

27

Id: 27
Nombre: Jose
Sueldo: 3850
Especialidad: MECÁNICO

La funcionalidad Aplicar bono le sube el sueldo al empleado seleccionado en un 10%, basándose en la historia de viajes realizados, pero por ligadura dinámica se le sube un 15% al conductor con viajes más realizados.
