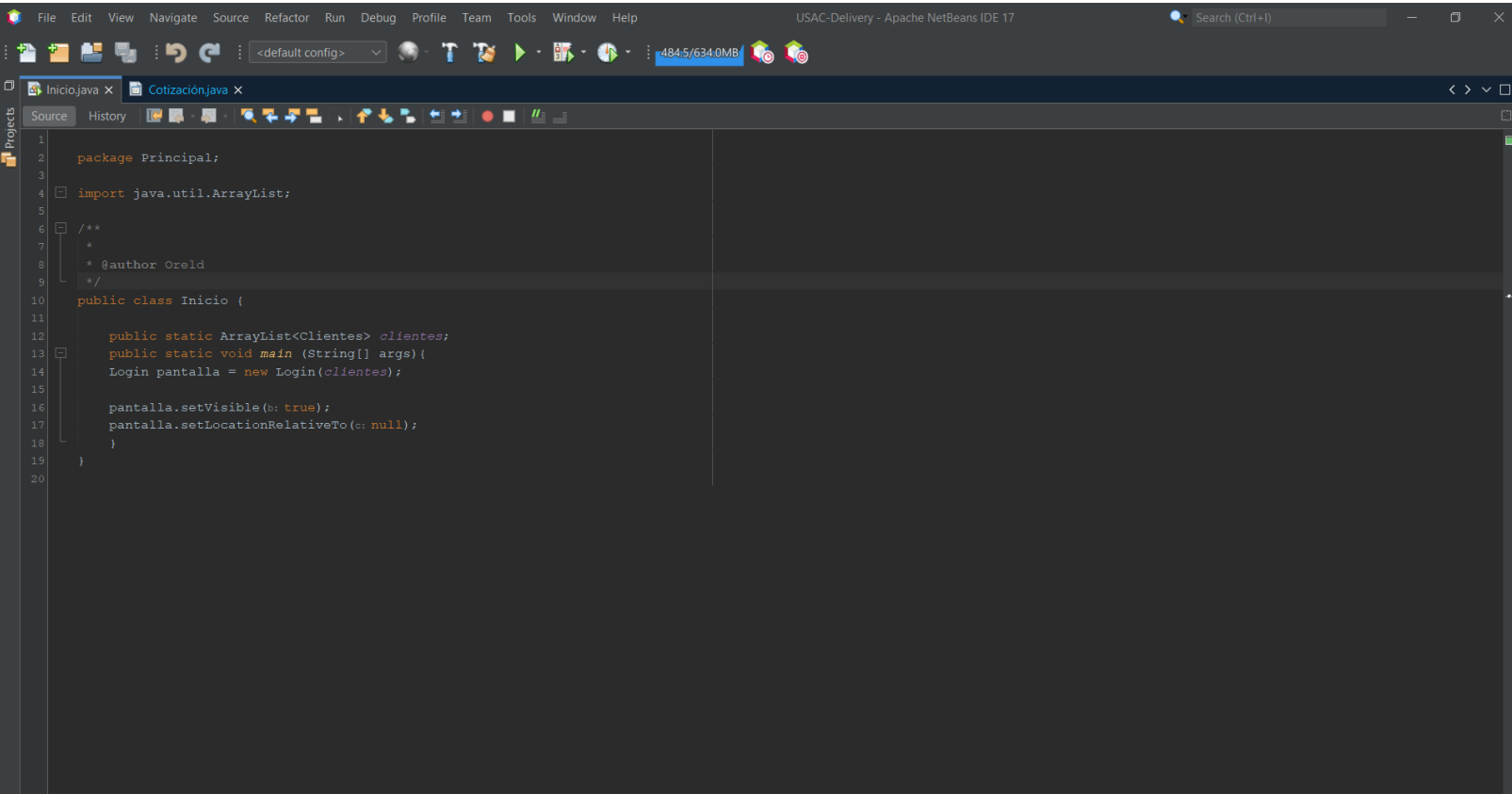


Manual técnico

Clase principal

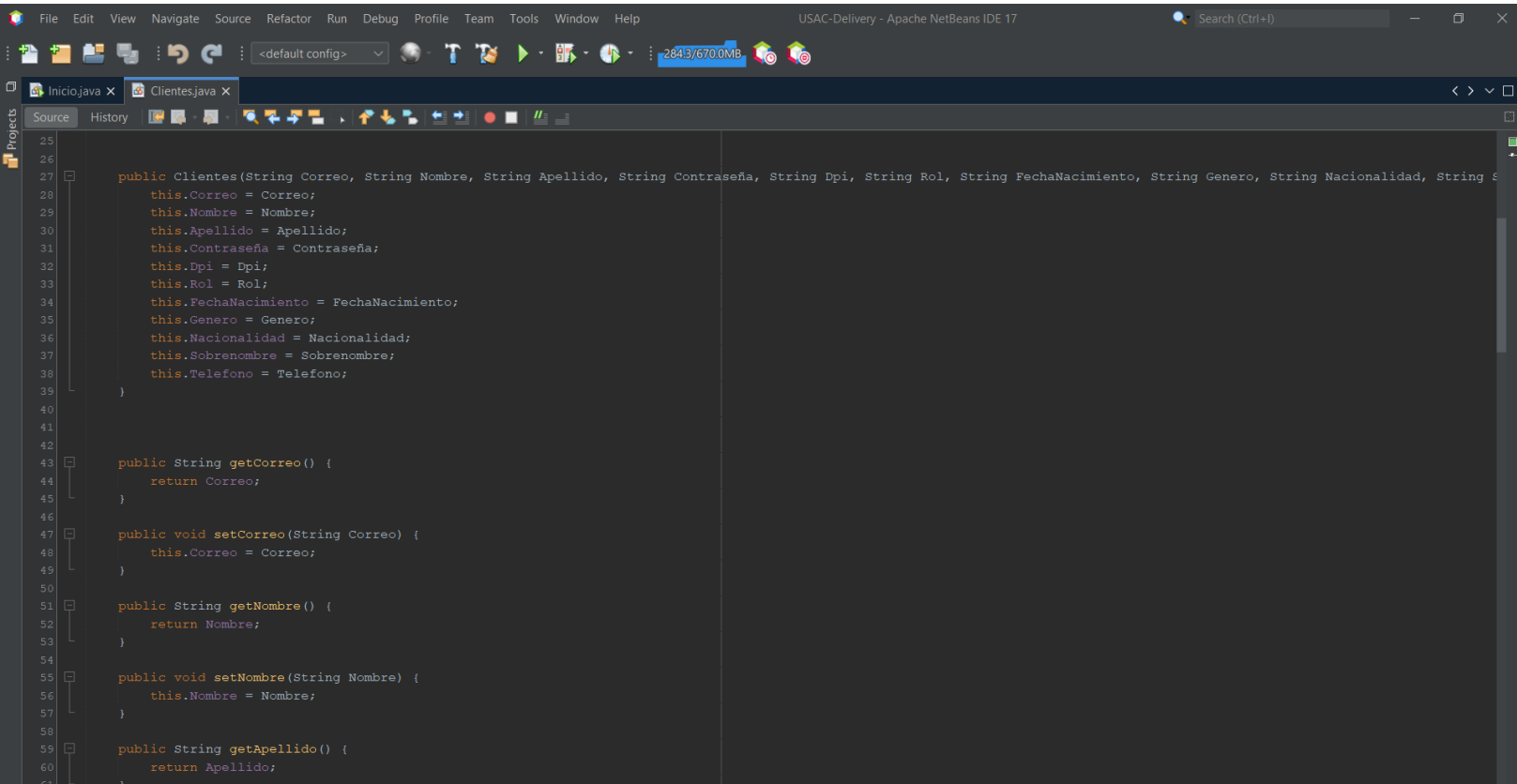
La clase principal es la que contiene al método main encargado de llamara a la ventana login y centrar su posición respecto a la pantalla.



```
1 package Principal;
2
3
4 import java.util.ArrayList;
5
6 /**
7  *
8  * @author Orelid
9  */
10 public class Inicio {
11
12     public static ArrayList<Clientes> clientes;
13     public static void main (String[] args){
14         Login pantalla = new Login(clientes);
15
16         pantalla.setVisible(true);
17         pantalla.setLocationRelativeTo(null);
18     }
19 }
20
```

Clase clientes

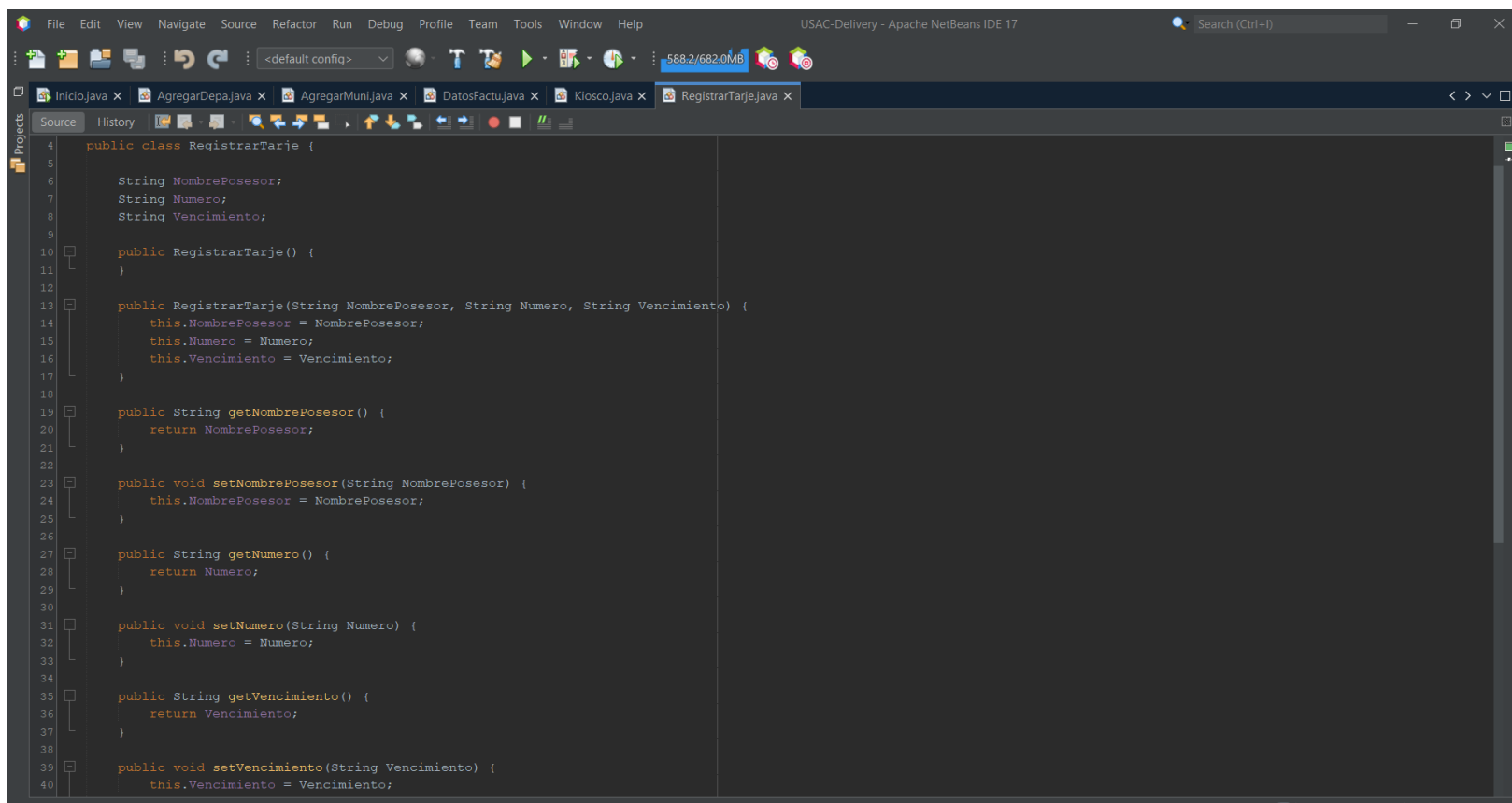
La clase clientes contiene un constructor vacío y otro con parámetros junto con sus métodos correspondientes get y set.



```
25
26
27 public Clientes(String Correo, String Nombre, String Apellido, String Contraseña, String Dpi, String Rol, String FechaNacimiento, String Genero, String Nacionalidad, String S
28     this.Corrreo = Correo;
29     this.Nombre = Nombre;
30     this.Apellido = Apellido;
31     this.Contraseña = Contraseña;
32     this.Dpi = Dpi;
33     this.Rol = Rol;
34     this.FechaNacimiento = FechaNacimiento;
35     this.Genero = Genero;
36     this.Nacionalidad = Nacionalidad;
37     this.Sobrenombre = Sobrenombre;
38     this.Telefono = Telefono;
39 }
40
41
42
43 public String getCorreo() {
44     return Correo;
45 }
46
47 public void setCorreo(String Correo) {
48     this.Corrreo = Correo;
49 }
50
51 public String getNombre() {
52     return Nombre;
53 }
54
55 public void setNombre(String Nombre) {
56     this.Nombre = Nombre;
57 }
58
59 public String getApellido() {
60     return Apellido;
61 }
```

AgregarDepa, AgregarMuni, DatosFac, Kiosco, cotiza y RegistrarTarje

Las clases mencionadas tiene la similitud de que tiene un constructor con parámetros y otro vacío y cuentas con sus respectivos métodos get y set

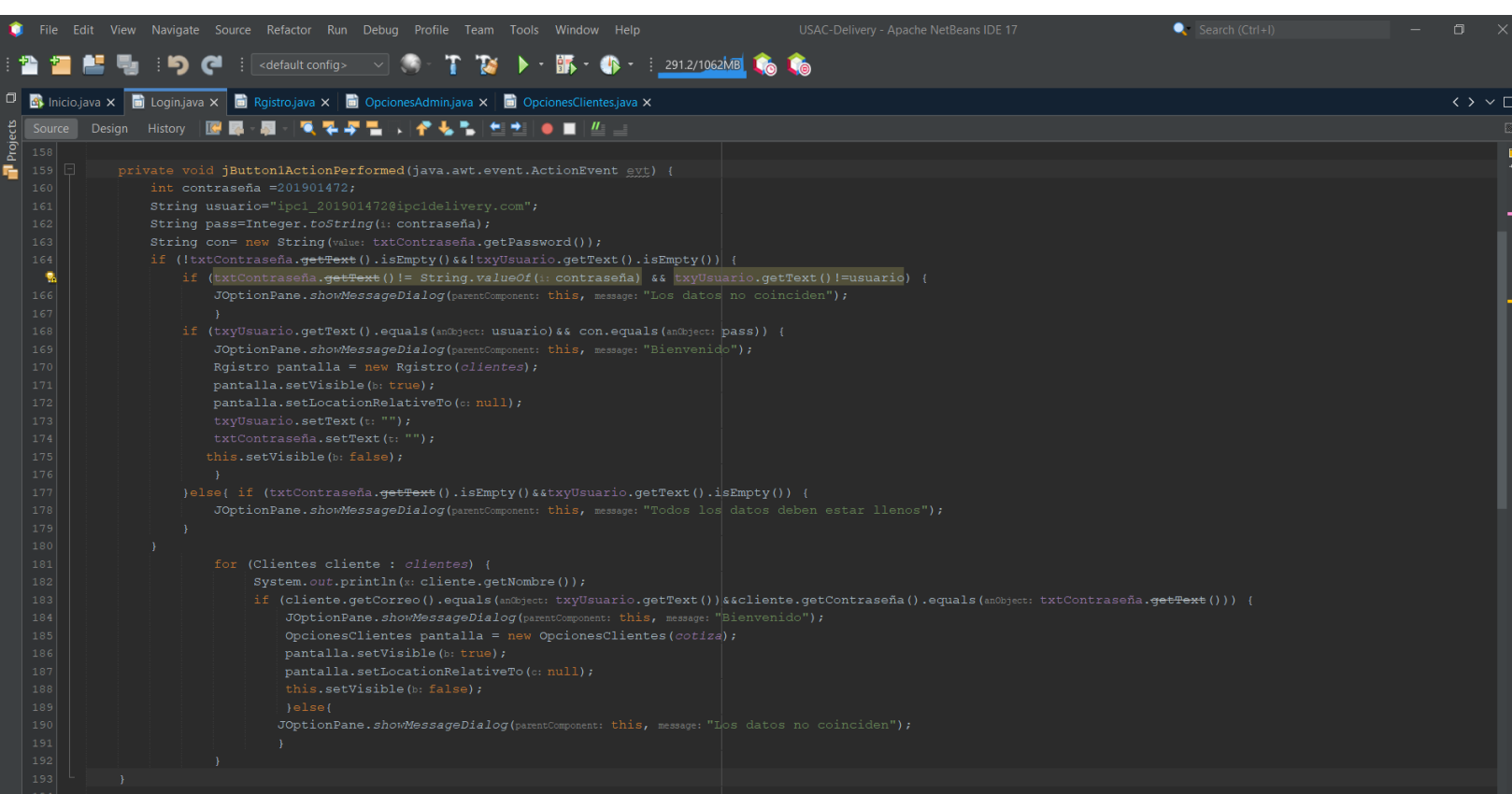


The screenshot displays the Apache NetBeans IDE 17 interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar indicates the project is 'USAC-Delivery'. The toolbar shows various development tools. The project explorer on the left lists several Java files: Inicio.java, AgregarDepa.java, AgregarMuni.java, DatosFactu.java, Kiosco.java, and RegistrarTarje.java. The main editor window shows the source code of the RegistrarTarje class, which is a public class with three private String attributes: NombrePoseedor, Numero, and Vencimiento. It includes a no-argument constructor, a three-parameter constructor, and getter and setter methods for each attribute.

```
4 public class RegistrarTarje {
5
6     String NombrePoseedor;
7     String Numero;
8     String Vencimiento;
9
10    public RegistrarTarje() {
11    }
12
13    public RegistrarTarje(String NombrePoseedor, String Numero, String Vencimiento) {
14        this.NombrePoseedor = NombrePoseedor;
15        this.Numero = Numero;
16        this.Vencimiento = Vencimiento;
17    }
18
19    public String getNombrePoseedor() {
20        return NombrePoseedor;
21    }
22
23    public void setNombrePoseedor(String NombrePoseedor) {
24        this.NombrePoseedor = NombrePoseedor;
25    }
26
27    public String getNumero() {
28        return Numero;
29    }
30
31    public void setNumero(String Numero) {
32        this.Numero = Numero;
33    }
34
35    public String getVencimiento() {
36        return Vencimiento;
37    }
38
39    public void setVencimiento(String Vencimiento) {
40        this.Vencimiento = Vencimiento;
41    }
42 }
```

Login

El panel login realiza varias validaciones con funciones if, valida que los campos estén llenos, coincidan con las credenciales del administrador y si hay algún cliente creado que valide sus credenciales para ingresar al apartado correspondiente.



```
158
159 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
160     int contraseña = 201901472;
161     String usuario = "ipcl_201901472@ipcldelivery.com";
162     String pass = Integer.toString(1: contraseña);
163     String con = new String(value: txtContraseña.getPassword());
164     if (!txtContraseña.getText().isEmpty() && !txyUsuario.getText().isEmpty()) {
165         if (txtContraseña.getText() != String.valueOf(1: contraseña) && txyUsuario.getText() != usuario) {
166             JOptionPane.showMessageDialog(parentComponent: this, message: "Los datos no coinciden");
167         }
168         if (txyUsuario.getText().equals(anObject: usuario) && con.equals(anObject: pass)) {
169             JOptionPane.showMessageDialog(parentComponent: this, message: "Bienvenido");
170             Registro pantalla = new Registro(clientes);
171             pantalla.setVisible(b: true);
172             pantalla.setLocationRelativeTo(c: null);
173             txyUsuario.setText(t: "");
174             txtContraseña.setText(t: "");
175             this.setVisible(b: false);
176         }
177     } else { if (txtContraseña.getText().isEmpty() && txyUsuario.getText().isEmpty()) {
178         JOptionPane.showMessageDialog(parentComponent: this, message: "Todos los datos deben estar llenos");
179     }
180 }
181
182 for (Clientes cliente : clientes) {
183     System.out.println(x: cliente.getNombre());
184     if (cliente.getCorreo().equals(anObject: txyUsuario.getText()) && cliente.getContraseña().equals(anObject: txtContraseña.getText())) {
185         JOptionPane.showMessageDialog(parentComponent: this, message: "Bienvenido");
186         OpcionesClientes pantalla = new OpcionesClientes(cotiza);
187         pantalla.setVisible(b: true);
188         pantalla.setLocationRelativeTo(c: null);
189         this.setVisible(b: false);
190     } else {
191         JOptionPane.showMessageDialog(parentComponent: this, message: "Los datos no coinciden");
192     }
193 }
194 }
```

Registro

En el código de registro igualmente hacemos varias validaciones, tanto que todos los campos estén llenos como que la contraseña sea segura y aquí mismo inicializamos un arraylist de tipo cliente.

```
16 public class Registro extends javax.swing.JFrame {
17     public static ArrayList<Clientes> clientes = new ArrayList<Clientes>();
18
19     /**
20      * Creates new form Registro
21      */
22     public Registro(ArrayList<Clientes> clientes) {
23         this.clientes=clientes;
24         initComponents();
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
```

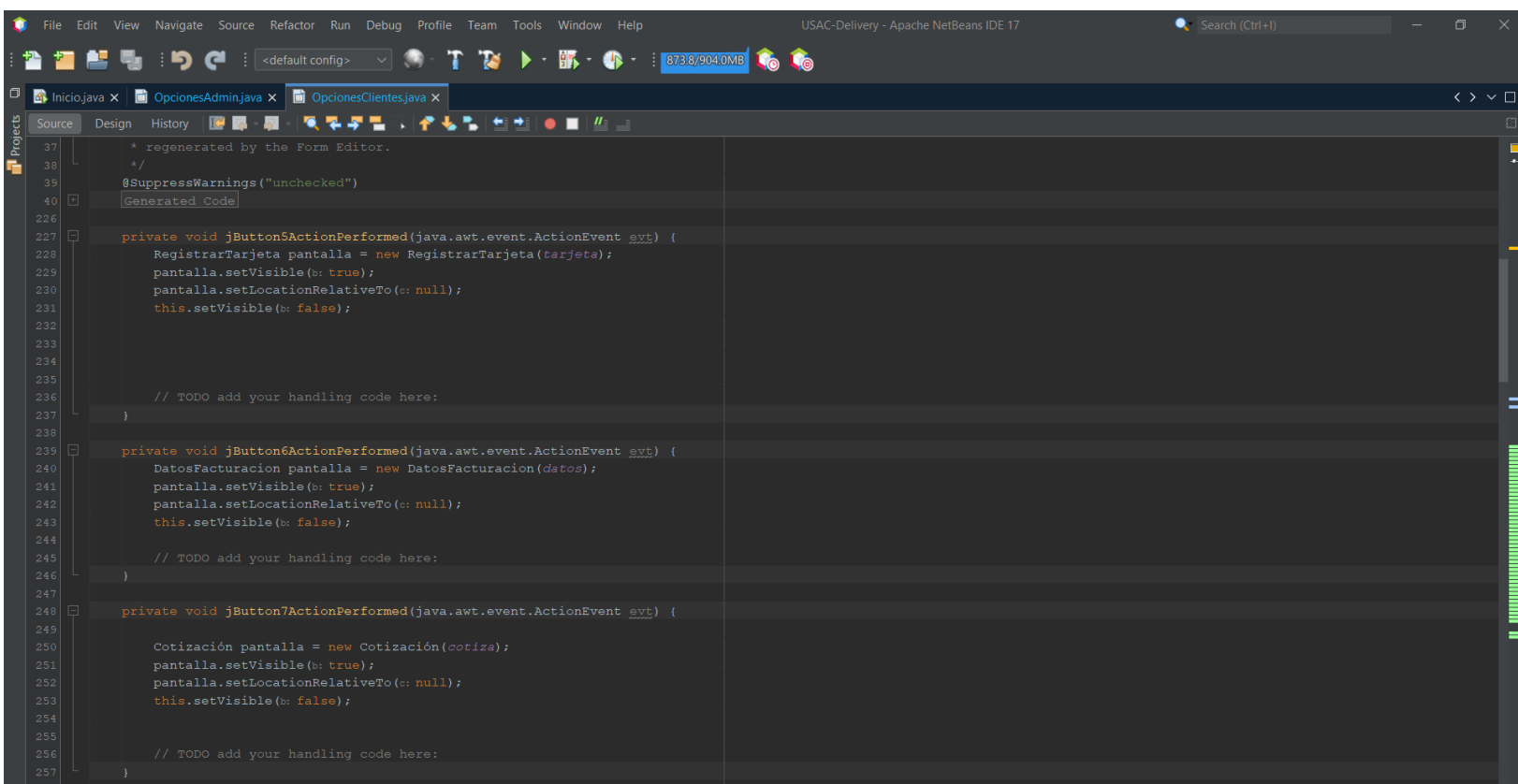
```
16 public class Registro extends javax.swing.JFrame {
17     public static ArrayList<Clientes> clientes = new ArrayList<Clientes>();
18
19     /**
20      * Creates new form Registro
21      */
22     public Registro(ArrayList<Clientes> clientes) {
23         this.clientes=clientes;
24         initComponents();
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
```

```
477 private void btnRegistrarActionPerformed(java.awt.event.ActionEvent evt) {
478
479     if (!CorreoElectronico.getText().isEmpty() && !Nombre.getText().isEmpty() && !Apellido.getText().isEmpty() && !Contraseña.getText().isEmpty() && !Dpi.getText().isEmpty() && !FechaNacimiento.getText().isEmpty() && !Sobrenombre.getText().isEmpty() && !Telefono.getText().isEmpty()) {
480         String password= Contraseña.getText();
481         if (esSegura(password)) {
482             Clientes cliente0=new Clientes(Correo:CorreoElectronico.getText(), Nombre:Nombre.getText(), Apellido: Apellido.getText(), Contraseña: Contraseña.getText(), Dpi:Dpi.getText(), FechaNacimiento: FechaNacimiento.getText(), Sobrenombre: Sobrenombre.getText(), Telefono: Telefono.getText());
483             clientes.add(cliente0);
484             JOptionPane.showMessageDialog(parentComponent: this,message: "Usuario creado con exito");
485             CorreoElectronico.setText("");
486             Nombre.setText("");
487             Apellido.setText("");
488             Contraseña.setText("");
489             Dpi.setText("");
490             FechaNacimiento.setText("");
491             Sobrenombre.setText("");
492             Telefono.setText("");
493         }else{
494             JOptionPane.showMessageDialog(parentComponent: this,message: "La contraseña debe cumplir los requisitos");
495         }
496     }else{
497         JOptionPane.showMessageDialog(parentComponent: this,message: "Debe llenar todos los campos");
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
```

```
public static boolean esSegura(String password){
    boolean mayuscula=false;
    boolean numero = false;
    boolean caracterEspecial=false;
    boolean minuscula=false;
    char c;
    int valorASCII;
    for (int i = 0; i < password.length(); i++) {
        c= password.charAt(i);
        valorASCII = (int) c;
        if (valorASCII>=33&&valorASCII<=47||valorASCII>=58&&valorASCII<=64||valorASCII>=91&&valorASCII<=96||valorASCII>=123&&valorASCII<=226) {
            caracterEspecial=true;
        }
        if (Character.isDigit(ch: c)) {
            numero=true;
        }
        if (Character.isUpperCase(ch: c)) {
            mayuscula=true;
        }
        if (Character.isLowerCase(ch: c)) {
            minuscula=true;
        }
    }
    if (numero && caracterEspecial && mayuscula && minuscula) {
        return true;
    }else{
        return false;
    }
}
```

Opciones de administrador y opciones de clientes.

En las opciones tanto de administrador y clientes solo llamamos a los demás paneles.



```
37  * regenerated by the Form Editor.
38  */
39  @SuppressWarnings("unchecked")
40  Generated Code
226
227  private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
228      RegistrarTarjeta pantalla = new RegistrarTarjeta(tarjeta);
229      pantalla.setVisible(true);
230      pantalla.setLocationRelativeTo(null);
231      this.setVisible(false);
232
233
234
235
236      // TODO add your handling code here:
237  }
238
239  private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
240      DatosFacturacion pantalla = new DatosFacturacion(datos);
241      pantalla.setVisible(true);
242      pantalla.setLocationRelativeTo(null);
243      this.setVisible(false);
244
245      // TODO add your handling code here:
246  }
247
248  private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {
249      Cotización pantalla = new Cotización(cotiza);
250      pantalla.setVisible(true);
251      pantalla.setLocationRelativeTo(null);
252      this.setVisible(false);
253
254
255
256      // TODO add your handling code here:
257  }
```

Datos de facturación, registrar tarjetas, Agregar departamentos, Agregar municipios, Kioscos.

En estos apartados verificamos que no haya campos vacíos y luego guardamos todos los datos en un arraylist y luego con este mismo mostramos en el historial los datos almacenados.

```
40  [Generated Code]
224
225 private void txtNitActionPerformed(java.awt.event.ActionEvent evt) {
226     // TODO add your handling code here:
227 }
228
229 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
230
231     OpcionesClientes pantalla = new OpcionesClientes(cotiza);
232     pantalla.setVisible(true);
233     pantalla.setLocationRelativeTo(null);
234     this.setVisible(false);
235     // TODO add your handling code here:
236 }
237
238 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
239
240     if (!txtNombre.getText().isEmpty() && !txtDireccion.getText().isEmpty() && !txtNit.getText().isEmpty()) {
241         DatosFactu tarjeta0 = new DatosFactu(nombre:txtNombre.getText(), direccion:txtDireccion.getText(), nit:txtNit.getText());
242         datos.add(e: tarjeta0);
243         JOptionPane.showMessageDialog(parentComponent: this, message: "Usuario creado con éxito");
244         txtNit.setText("");
245         txtDireccion.setText("");
246         txtNombre.setText("");
247     } else {
248         JOptionPane.showMessageDialog(parentComponent: this, message: "Debe llenar todos los campos");
249     }
250
251     // TODO add your handling code here:
252 }
253
254 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
255     if (datos.get(index: 0) != null) {
256         String [] Cabezera = {"Nombre Completo", "Direccion", "Nit"};
257         String [][] Datos = {
258             {datos.get(index: 0).getNombre(), datos.get(index: 0).getDireccion(), datos.get(index: 0).getNit()}
259         };
260
261         DefaultTableModel mod = new DefaultTableModel(data: Datos, columnNames: Cabezera);
262         JTable tabla = new JTable(dm: mod);
263         JScrollPane scroll = new JScrollPane(view: tabla);
264         scroll.setBounds(x: 40, y: 40, width: 400, height: 200);
265         JFrame ventana = new JFrame();
266         ventana.setLayout(manager: null);
267         ventana.setSize(width: 500, height: 500);
268         ventana.setLocationRelativeTo(c: null);
269         ventana.add(comp: scroll);
270         ventana.setVisible(v: true);
271     }
272     if (datos.get(index: 1) != null) {
273         String [] Cabezera = {"Nombre Completo", "Direccion", "Nit"};
274         String [][] Datos = {
275             {datos.get(index: 0).getNombre(), datos.get(index: 0).getDireccion(), datos.get(index: 0).getNit()},
276             {datos.get(index: 1).getNombre(), datos.get(index: 1).getDireccion(), datos.get(index: 1).getNit()}
277         };
278         DefaultTableModel mod = new DefaultTableModel(data: Datos, columnNames: Cabezera);
279         JTable tabla = new JTable(dm: mod);
280         JScrollPane scroll = new JScrollPane(view: tabla);
281         scroll.setBounds(x: 40, y: 40, width: 400, height: 200);
282         JFrame ventana = new JFrame();
283         ventana.setLayout(manager: null);
284         ventana.setSize(width: 500, height: 500);
285         ventana.setLocationRelativeTo(c: null);
286         ventana.add(comp: scroll);
287         ventana.setVisible(v: true);
288     }
289     if (datos.get(index: 2) != null) {
290         String [] Cabezera = {"Nombre Completo", "Direccion", "Nit"};
291         String [][] Datos = {
```

Precios de regiones

Para este apartado utilizamos un switch para según sea la región seleccionada se mande la información a los textfield.

```
223 private void OpcionItemStateChanged(java.awt.event.ItemEvent evt) {
224
225     String item = Opcion.getSelectedItem().toString();
226     System.out.println("item: " + item);
227     switch (item) {
228         case "Metropolitana":
229             Estandar.setText("Q 35.00");
230             Especial.setText("Q 25.00");
231             break;
232
233         case "Norte":
234             Estandar.setText("Q 68.50");
235             Especial.setText("Q 45.55");
236             break;
237
238         case "Nororiente":
239             Estandar.setText("Q 58.68");
240             Especial.setText("Q 35.48");
241             break;
242
243         case "Suroccidente":
244             Estandar.setText("Q 38.68");
245             Especial.setText("Q 32.48");
246             break;
247
248         case "Suroccidente":
249             Estandar.setText("Q 34.00");
250             Especial.setText("Q 29.00");
251             break;
252
253         case "Noroccidente":
254             Estandar.setText("Q 44.50");
255             Especial.setText("Q 40.00");
256             break;
257
258         default:
259             throw new AssertionError();
260     }
261 }
```


Cotización

Para el apartado de cotización guardamos todos los datos seleccionados en los combobox en el array correspondiente a la clase de cotización, validamos las regiones de destino y el tipo de servicio para mostrar los datos de cotización con un JOptionPane, cuando se selecciona la opción de cobro a mi cuenta se busca con un for se busca la existencia de una tarjeta creada en el array correspondiente para ser mostrada, de igual manera los datos de facturación.

```
480 private void txtEnviarPagoActionPerformed(java.awt.event.ActionEvent evt) {
481
482     Cotiza factu0 = new Cotiza(DepOrigen:txtDepOrigen.getSelectedItem().toString(), codigoPaquete+contador, DepDestino:txtDepDestino.getSelectedItem().toString(), MunDestino:String.valueOf(txtMunDestino.getSelectedItem().toString()));
483     cotiza.add(= factu0);
484     contador=contador+1;
485     JOptionPane.showMessageDialog(parentComponent: this,"Compra realizada con exito, su codigo de paquete es"+codigoPaquete+contador);
486 }
487
488 private void txtCotizarActionPerformed(java.awt.event.ActionEvent evt) {
489
490     String Destino= txtDepDestino.getSelectedItem().toString();
491     String Tamaño =txtNumeroPaquete.getSelectedItem().toString();
492     int Paquetes= Integer.parseInt(txtDatosFacturacion.getText());
493     String Servicio = txtServicio.getSelectedItem().toString();
494     if (Destino.equals(anObject: "Metropolitana")) {
495         if (Servicio.equals(anObject: "Servicio Estandar")) {
496             if (Tamaño.equals(anObject: "Pequeño")) {
497                 JOptionPane.showMessageDialog(parentComponent: this,"El total de es servicio es: " +Paquetes*25*35);
498                 Precio=Paquetes*25*35;
499             }
500             if (Tamaño.equals(anObject: "Mediano")) {
501                 JOptionPane.showMessageDialog(parentComponent: this,"El total de es servicio es: " +Paquetes*50*35);
502                 Precio=Paquetes*50*35;
503             }
504             if (Tamaño.equals(anObject: "Grande")) {
505                 JOptionPane.showMessageDialog(parentComponent: this,"El total de es servicio es: " +Paquetes*100*35);
506                 Precio=Paquetes*100*35;
507             }
508         }
509     }
510     if (Servicio.equals(anObject: "Servicio Especial")) {
511         if (Tamaño.equals(anObject: "Pequeño")) {
512             JOptionPane.showMessageDialog(parentComponent: this,"El total de es servicio es: " +Paquetes*25*25);
513             Precio=Paquetes*25*25;
514         }
515         if (Tamaño.equals(anObject: "Mediano")) {
516
517         }
518     }
519 }
520
521 private void txtCobroActionPerformed(java.awt.event.ActionEvent evt) {
522     // TODO add your handling code here:
523 }
524
525 private void txtCobroItemStateChanged(java.awt.event.ItemEvent evt) {
526
527     String pago = txtCobro.getSelectedItem().toString();
528     if (pago=="Cobro a mi cuenta") {
529         for (RegistrarTarje tarjeta : tarjeta) {
530             cboTarjetas.removeAllItems();
531             cboTarjetas.addItem(item: tarjeta.getNumero());
532         }
533     }else{
534         cboTarjetas.removeAllItems();
535     }
536
537     // TODO add your handling code here:
538 }
539 }
```

para la factura y la guía se creara un archivo en el cual se mostrara la información correspondiente

```
697 private void txtFacturaActionPerformed(java.awt.event.ActionEvent evt) {  
698  
699     File archivo;  
700     String Factura = "Factura";  
701     try {  
702         archivo = new File ("C:\\Users\\Oreld\\Downloads\\"+Factura+".txt");  
703         if (archivo.createNewFile()) {  
704             JOptionPane.showMessageDialog(parentComponent: this,message: "La factura se a descargado y esta en la tuta correspondiente");  
705         }else{  
706             JOptionPane.showMessageDialog(parentComponent: this,message: "La factura ya se encuentra en su sistema");  
707         }  
708         FileWriter escritor = new FileWriter(file: archivo, append:true);  
709         escritor.write(": contador);  
710         escritor.write(str: " ");  
711         escritor.write(str: codigoPaquete);  
712         escritor.write(str: " ");  
713         escritor.write(str: txtDepOrigen.getSelectedItem().toString());  
714         escritor.write(str: " ");  
715         escritor.write(str: txtDepDestino.getSelectedItem().toString());  
716         escritor.write(str: " ");  
717         escritor.write(str: txtCobro.getSelectedItem().toString());  
718         escritor.write(str: " ");  
719         escritor.write(str: txtNumeroPaquete.getSelectedItem().toString());  
720         escritor.write(str: " ");  
721         escritor.write(str: txtDatosFacturacion.getText());  
722         escritor.write(str: " ");  
723         escritor.write(str: String.valueOf(z: Precio));  
724         escritor.write(str: " ");  
725         escritor.close();  
726     } catch (Exception e) {  
727     }
```