

Evidencia de TDD y Clean Code

Diego Acuña – 222675, Felipe Brioso – 269851, Nicole Uhalde – 270303

Contenido

Feature 1: Promociones	2
Feature 2: Usuario.....	8
Feature 3: Administrador	12
Code Coverage:	15

Feature 1: Promociones

Las promociones nacen en el commit 4e3a83d cuando al crear la compra, queríamos incluir la promoción aplicada. Por este motivo, se escribió la siguiente prueba:

```
BackEnd/UnitTest/PurchaseTest.cs
@@ -67,5 +67,14 @@ public void GivenPurchaseReturnsItsDate()
67 67
68 68     Assert.AreEqual(now, purchaseSample.Date);
69 69 }
70 +
71 +     [TestMethod]
72 +     public void GivenPurchaseReturnsPromotionUsed()
73 +     {
74 +         Promotion p = new Promotion();
75 +         purchaseSample.Promotion = p;
76 +
77 +         Assert.AreEqual(p, purchaseSample.Promotion);
78 +     }
79 }
80 }
```

Y para que la prueba fuera correcta se escribió el menor código necesario. Observar que únicamente se definió la clase:

```
BackEnd/Backend/Promotion.cs
@@ -0,0 +1,6 @@
1 + namespace BackEnd
2 + {
3 +     public class Promotion
4 +     {
5 +     }
6 + }
```

Luego, con el fin de implementar las promociones, se creó una rama llamada *feature/promotionCreation*, cuyo historial de commits se encuentra a continuación:

Commits on Sep 6, 2023

[green] applicability of 1 item cart for total look promotion diegoacu1234 committed last month	448c2f2	<>
[refactor] inverted conditionals and removed magic numbers diegoacu1234 committed last month	18cf182	<>
[green] generalized calculation discount method diegoacu1234 committed last month	2591dc5	<>
[green] calculates discount for 2 products diegoacu1234 committed last month	024ea6d	<>
[green] throws exception at applying a non applicable promotion diegoacu1234 committed last month	4aeda08	<>
[green] tested 3 item cart applicability diegoacu1234 committed last month	e1624b0	<>
[refactor] extracted common attributes at test class diegoacu1234 committed last month	d8bccdc	<>
[refactor] changed class names diegoacu1234 committed last month	dff6bb7	<>
[green] returns 20% off promo is not applicable for 1 item diegoacu1234 committed last month	aaebb41	<>
[green] returns 20% off promotion is applicable if has 2 items diegoacu1234 committed last month	2196b0d	<>
[green] returns 20% off promotion is not applicable if the purchase i... ...s empty diegoacu1234 committed last month	4681c7d	<>

Commits on Sep 6, 2023

[green] calculation of 3x2 discount diegoacu1234 committed last month	55e6e0b	<>
[green] calculates discount for a fixed purchase diegoacu1234 committed last month	a564bda	<>
[green] throws exception at calculating discount diegoacu1234 committed last month	005cf88	<>
[refactor] simplified code diegoacu1234 committed last month	e390977	<>
[green] identifies whether there are 3 items with same category diegoacu1234 committed last month	97c8ed1	<>
[green] returns 3x2 is never applicable diegoacu1234 committed last month	db03f20	<>
[refactor] removed magic strings and numbers diegoacu1234 committed last month	34873c2	<>
[refactor] extracted common test diegoacu1234 committed last month	7b66f51	<>
[green] calculates the higher discount if 2 different combinations ca... diegoacu1234 committed last month	8ccf27a	<>
[refactor] renamed test initializer to follow a criterion diegoacu1234 committed last month	f4e7c7f	<>
[refactor] removed unnecessary dependencies and changed names diegoacu1234 committed last month	aa5ddd8	<>

[refactor] renamed functions and attributes diegoacu1234 committed last month	49ba1fa	<>
[green] calculates total look discount diegoacu1234 committed last month	1649fe4	<>
[green] throws exception at calculate discount diegoacu1234 committed last month	f20a6dd	<>
[refactor] remove magic numbers diegoacu1234 committed last month	7600800	<>
[refactor] extracted methods diegoacu1234 committed last month	8fc04ba	<>
[green] generalized algorithm for >= 3 items of same color diegoacu1234 committed last month	952f2f8	<>
[green] can distinguish if is applicable for 3 item cart diegoacu1234 committed last month	42bed07	<>
[green] returns promotion is applicable for 3 items diegoacu1234 committed last month	9fcf345	<>
[green] applicability for 2 item cart diegoacu1234 committed last month	0de3a13	<>

Commits on Sep 7, 2023

[refactor] used of linq at 3x2 FACULTADES\alumnoFI committed last month	0c5fbae	<>
[refactor] used linq to improve clarity FACULTADES\alumnoFI committed last month	5b45c26	<>
[green] calculates discount FACULTADES\alumnoFI committed last month	d11b44b	<>
[green] if promo applies returns fixed value FACULTADES\alumnoFI committed last month	5a6fc98	<>
[green] throws exception at trying to calculate discount FACULTADES\alumnoFI committed last month	badee32	<>
[green] solved mistakes made at refactoring FACULTADES\alumnoFI committed last month	6695cda	<>
[refactor] extracted constants at tests FACULTADES\alumnoFI committed last month	82ab545	<>
[green] further testing] FACULTADES\alumnoFI committed last month	ad846c0	<>
[green] can distinguish applicability for a 3-item cart FACULTADES\alumnoFI committed last month	bcf3822	<>
[green] returns discount is applicable if number of items is 3 FACULTADES\alumnoFI committed last month	10b945d	<>
[green] returns 3x1 fidelity is never applicable FACULTADES\alumnoFI committed last month	806a265	<>
[refactor] removed magic numbers from prices FACULTADES\alumnoFI committed last month	a03fdab	<>
[refactor] extracted repeated code from tests FACULTADES\alumnoFI committed last month	7edef35	<>
[green] created interface for promotions FACULTADES\alumnoFI committed last month	465d17a	<>
[green] discounts cheapest item if more than 3 items in same category FACULTADES\alumnoFI committed last month	3de1c53	<>

Con respecto a los commits de esta rama, dada la gran cantidad de los mismos, no vemos con sentido analizar cada uno de los commits, sin embargo, se realizarán comentarios generales y se elegirán algunos commits puntuales para analizar con mayor profundidad.

En primer lugar, como se ve en la figura anterior, el equipo decidió identificar cada commit con la fase del ciclo TDD. En la implementación de esta feature, se realizó un commit por cada test. A continuación, se muestra un ejemplo del primer commit de dicha rama:

```
Commit

[green] returns 20% off promotion is not applicable if the purchase i...
...s empty

Develop
diegoacu1234 committed last month
1 parent c5814cd commit 4681c7d

Showing 5 changed files with 33 additions and 7 deletions.

Filter changed files

Backend
  Backend
    20%OffPromotion.cs
    Backend.csproj
    Promotion.cs
  UnitTest
    20%OffPromotionTest.cs
    UnitTest.csproj

Backend/Backend/20%OffPromotion.cs
... @@ -0,0 +1,12 @@
1 + using System;
2 +
3 + namespace Backend
4 + {
5 +     public class Promotion
6 +     {
7 +         public bool IsApplicable(Purchase p)
8 +         {
9 +             return false;
10 +        }
11 +    }
12 + }

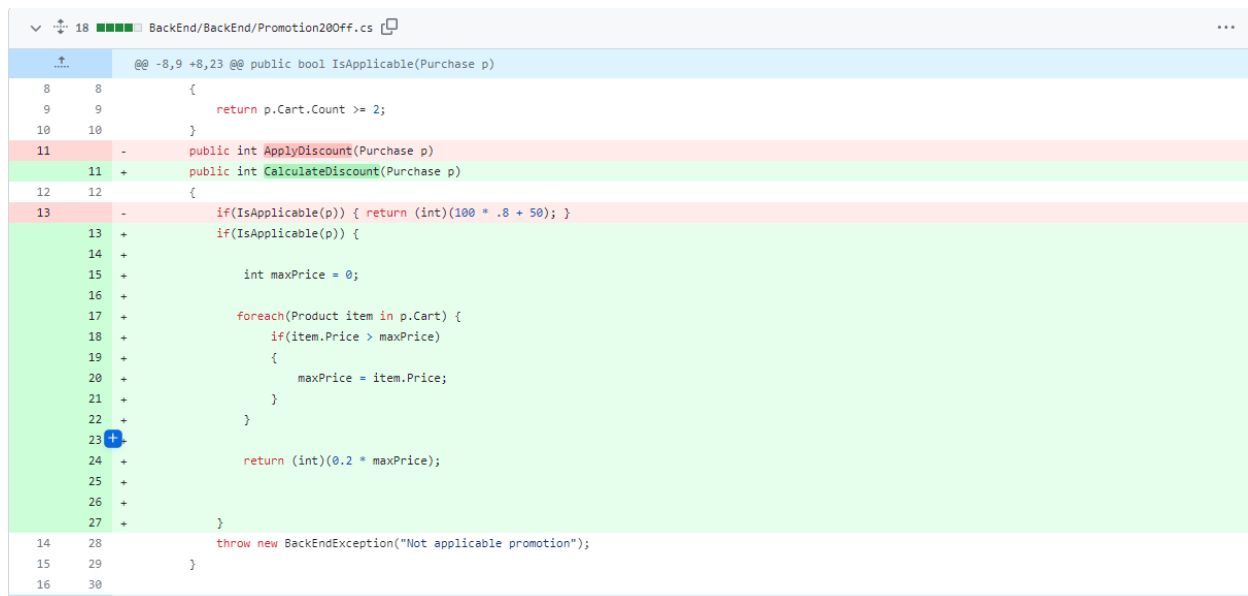
Backend/UnitTest/20%OffPromotionTest.cs
... @@ -0,0 +1,19 @@
1 + using Microsoft.VisualStudio.TestTools.UnitTesting;
2 + using System;
3 + using Backend;
4 +
5 + namespace UnitTest
6 + {
7 +     [TestClass]
8 +     public class PromotionTest
9 +     {
10 +         [TestMethod]
11 +         public void GivenEmptyPurchaseReturnsPromotionIsNotApplicable()
12 +         {
13 +             Purchase p = new Purchase();
14 +             Promotion promotion = new Promotion();
15 +             Assert.IsFalse(promotion.IsApplicable(p));
16 +         }
17 +     }
18 + }
19 + }
```

Observar que el código de producción es el más simple para que la prueba pase. A su vez, únicamente leyendo el nombre de la prueba podemos saber qué se está testeando.

Dado que se optó por un commit por test, los mensajes de commit no se diferenciarán mucho del nombre del test creado.

Si observamos los commits posteriores a este, vemos que de a poco va creciendo la complejidad del código y van desapareciendo los números mágicos.

Un ejemplo de esto es el commit 2591dc5. A continuación solo se incluyen los cambios en el código de producción (no los de tests):

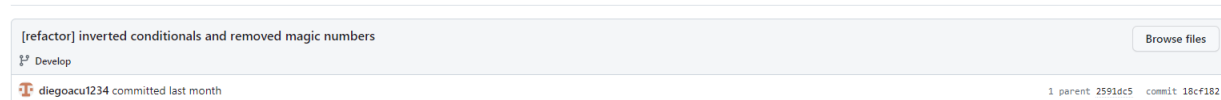


```
18 BackEnd/Backend/Promotion200Off.cs
@@ -8,9 +8,23 @@ public bool IsApplicable(Purchase p)
8      {
9          return p.Cart.Count >= 2;
10     }
11 - public int ApplyDiscount(Purchase p)
11 + public int CalculateDiscount(Purchase p)
12     {
13 -         if(IsApplicable(p)) { return (int)(100 * .8 * 50); }
13 +         if(IsApplicable(p)) {
14 +             int maxPrice = 0;
15 +             foreach(Product item in p.Cart) {
16 +                 if(item.Price > maxPrice)
17 +                 {
18 +                     maxPrice = item.Price;
19 +                 }
20 +             }
21 +             return (int)(0.2 * maxPrice);
22 +         }
23 +     }
24 +     throw new BackendException("Not applicable promotion");
25 + }
26 +
27 +
28 +
29 +
30 +
```

Si se observa la línea 13, vemos que hasta ese momento el resultado estaba “hardcoded”, en este commit se genera el primer intento de algoritmo (no hardcodeado) para calcular el descuento.

Después del commit anterior (green), se hizo un commit de refactorio. Mostrando una vez más que se cumplieron las etapas de TDD:

Commit



```
[refactor] inverted conditionals and removed magic numbers
Develop
diegoacu1234 committed last month
1 parent 2591dc5 commit 18cf182
```

El equipo hizo énfasis en tratar de ser claros y concisos en los mensajes de commit. Con el fin de evitar tener que abrir un commit para conocer los cambios.

Si miramos en detalle el commit mencionado anteriormente vemos que los refactorios fueron hechos tanto en el código de producción como en los tests.

```

28 BackEnd/Backend/Promotion200ff.cs
@@ -4,28 +4,30 @@ namespace BackEnd
4 4 {
5 5     public class Promotion200ff
6 6     {
7 7         private const float _twentyPercent = 0.2f;
8 8
9 9         public bool IsApplicable(Purchase p)
10 10         {
11 11             return p.Cart.Count >= 2;
12 12         }
13 13
14 14         public int CalculateDiscount(Purchase p)
15 15         {
16 16             if(IsApplicable(p)) {
17 17                 if (!IsApplicable(p))
18 18                 {
19 19                     throw new BackEndException("Not applicable promotion");
20 20                 }
21 21             }
22 22         }
23 23     }
24 24 }
25 25
26 26
27 27
28 28
29 29
30 30
31 31
32 32
33 33
34 34
35 35
36 36
37 37
38 38
39 39
40 40
41 41
42 42
43 43
44 44
45 45
46 46
47 47
48 48
49 49
50 50
51 51
52 52
53 53
54 54
55 55
56 56
57 57
58 58
59 59
60 60
61 61
62 62
63 63
64 64
65 65
66 66
67 67
68 68
69 69
70 70
71 71
72 72
73 73
74 74
75 75
76 76
77 77
78 78
79 79
80 80
81 81
82 82
83 83
84 84
85 85
86 86
87 87
88 88
89 89
90 90
91 91
92 92
93 93
94 94
95 95
96 96
97 97
98 98
99 99
100 100
101 101
102 102
103 103
104 104
105 105
106 106
107 107
108 108
109 109
110 110
111 111
112 112
113 113
114 114
115 115
116 116
117 117
118 118
119 119
120 120
121 121
122 122
123 123
124 124
125 125
126 126
127 127
128 128
129 129
130 130
131 131
132 132
133 133
134 134
135 135
136 136
137 137
138 138
139 139
140 140
141 141
142 142
143 143
144 144
145 145
146 146
147 147
148 148
149 149
150 150
151 151
152 152
153 153
154 154
155 155
156 156
157 157
158 158
159 159
160 160
161 161
162 162
163 163
164 164
165 165
166 166
167 167
168 168
169 169
170 170
171 171
172 172
173 173
174 174
175 175
176 176
177 177
178 178
179 179
180 180
181 181
182 182
183 183
184 184
185 185
186 186
187 187
188 188
189 189
190 190
191 191
192 192
193 193
194 194
195 195
196 196
197 197
198 198
199 199
200 200
201 201
202 202
203 203
204 204
205 205
206 206
207 207
208 208
209 209
210 210
211 211
212 212
213 213
214 214
215 215
216 216
217 217
218 218
219 219
220 220
221 221
222 222
223 223
224 224
225 225
226 226
227 227
228 228
229 229
230 230
231 231
232 232
233 233
234 234
235 235
236 236
237 237
238 238
239 239
240 240
241 241
242 242
243 243
244 244
245 245
246 246
247 247
248 248
249 249
250 250
251 251
252 252
253 253
254 254
255 255
256 256
257 257
258 258
259 259
260 260
261 261
262 262
263 263
264 264
265 265
266 266
267 267
268 268
269 269
270 270
271 271
272 272
273 273
274 274
275 275
276 276
277 277
278 278
279 279
280 280
281 281
282 282
283 283
284 284
285 285
286 286
287 287
288 288
289 289
290 290
291 291
292 292
293 293
294 294
295 295
296 296
297 297
298 298
299 299
300 300
301 301
302 302
303 303
304 304
305 305
306 306
307 307
308 308
309 309
310 310
311 311
312 312
313 313
314 314
315 315
316 316
317 317
318 318
319 319
320 320
321 321
322 322
323 323
324 324
325 325
326 326
327 327
328 328
329 329
330 330
331 331
332 332
333 333
334 334
335 335
336 336
337 337
338 338
339 339
340 340
341 341
342 342
343 343
344 344
345 345
346 346
347 347
348 348
349 349
350 350
351 351
352 352
353 353
354 354
355 355
356 356
357 357
358 358
359 359
360 360
361 361
362 362
363 363
364 364
365 365
366 366
367 367
368 368
369 369
370 370
371 371
372 372
373 373
374 374
375 375
376 376
377 377
378 378
379 379
380 380
381 381
382 382
383 383
384 384
385 385
386 386
387 387
388 388
389 389
390 390
391 391
392 392
393 393
394 394
395 395
396 396
397 397
398 398
399 399
400 400
401 401
402 402
403 403
404 404
405 405
406 406
407 407
408 408
409 409
410 410
411 411
412 412
413 413
414 414
415 415
416 416
417 417
418 418
419 419
420 420
421 421
422 422
423 423
424 424
425 425
426 426
427 427
428 428
429 429
430 430
431 431
432 432
433 433
434 434
435 435
436 436
437 437
438 438
439 439
440 440
441 441
442 442
443 443
444 444
445 445
446 446
447 447
448 448
449 449
450 450
451 451
452 452
453 453
454 454
455 455
456 456
457 457
458 458
459 459
460 460
461 461
462 462
463 463
464 464
465 465
466 466
467 467
468 468
469 469
470 470
471 471
472 472
473 473
474 474
475 475
476 476
477 477
478 478
479 479
480 480
481 481
482 482
483 483
484 484
485 485
486 486
487 487
488 488
489 489
490 490
491 491
492 492
493 493
494 494
495 495
496 496
497 497
498 498
499 499
500 500
501 501
502 502
503 503
504 504
505 505
506 506
507 507
508 508
509 509
510 510
511 511
512 512
513 513
514 514
515 515
516 516
517 517
518 518
519 519
520 520
521 521
522 522
523 523
524 524
525 525
526 526
527 527
528 528
529 529
530 530
531 531
532 532
533 533
534 534
535 535
536 536
537 537
538 538
539 539
540 540
541 541
542 542
543 543
544 544
545 545
546 546
547 547
548 548
549 549
550 550
551 551
552 552
553 553
554 554
555 555
556 556
557 557
558 558
559 559
560 560
561 561
562 562
563 563
564 564
565 565
566 566
567 567
568 568
569 569
570 570
571 571
572 572
573 573
574 574
575 575
576 576
577 577
578 578
579 579
580 580
581 581
582 582
583 583
584 584
585 585
586 586
587 587
588 588
589 589
590 590
591 591
592 592
593 593
594 594
595 595
596 596
597 597
598 598
599 599
600 600
601 601
602 602
603 603
604 604
605 605
606 606
607 607
608 608
609 609
610 610
611 611
612 612
613 613
614 614
615 615
616 616
617 617
618 618
619 619
620 620
621 621
622 622
623 623
624 624
625 625
626 626
627 627
628 628
629 629
630 630
631 631
632 632
633 633
634 634
635 635
636 636
637 637
638 638
639 639
640 640
641 641
642 642
643 643
644 644
645 645
646 646
647 647
648 648
649 649
650 650
651 651
652 652
653 653
654 654
655 655
656 656
657 657
658 658
659 659
660 660
661 661
662 662
663 663
664 664
665 665
666 666
667 667
668 668
669 669
670 670
671 671
672 672
673 673
674 674
675 675
676 676
677 677
678 678
679 679
680 680
681 681
682 682
683 683
684 684
685 685
686 686
687 687
688 688
689 689
690 690
691 691
692 692
693 693
694 694
695 695
696 696
697 697
698 698
699 699
700 700
701 701
702 702
703 703
704 704
705 705
706 706
707 707
708 708
709 709
710 710
711 711
712 712
713 713
714 714
715 715
716 716
717 717
718 718
719 719
720 720
721 721
722 722
723 723
724 724
725 725
726 726
727 727
728 728
729 729
730 730
731 731
732 732
733 733
734 734
735 735
736 736
737 737
738 738
739 739
740 740
741 741
742 742
743 743
744 744
745 745
746 746
747 747
748 748
749 749
750 750
751 751
752 752
753 753
754 754
755 755
756 756
757 757
758 758
759 759
760 760
761 761
762 762
763 763
764 764
765 765
766 766
767 767
768 768
769 769
770 770
771 771
772 772
773 773
774 774
775 775
776 776
777 777
778 778
779 779
780 780
781 781
782 782
783 783
784 784
785 785
786 786
787 787
788 788
789 789
790 790
791 791
792 792
793 793
794 794
795 795
796 796
797 797
798 798
799 799
800 800
801 801
802 802
803 803
804 804
805 805
806 806
807 807
808 808
809 809
810 810
811 811
812 812
813 813
814 814
815 815
816 816
817 817
818 818
819 819
820 820
821 821
822 822
823 823
824 824
825 825
826 826
827 827
828 828
829 829
830 830
831 831
832 832
833 833
834 834
835 835
836 836
837 837
838 838
839 839
840 840
841 841
842 842
843 843
844 844
845 845
846 846
847 847
848 848
849 849
850 850
851 851
852 852
853 853
854 854
855 855
856 856
857 857
858 858
859 859
860 860
861 861
862 862
863 863
864 864
865 865
866 866
867 867
868 868
869 869
870 870
871 871
872 872
873 873
874 874
875 875
876 876
877 877
878 878
879 879
880 880
881 881
882 882
883 883
884 884
885 885
886 886
887 887
888 888
889 889
890 890
891 891
892 892
893 893
894 894
895 895
896 896
897 897
898 898
899 899
900 900
901 901
902 902
903 903
904 904
905 905
906 906
907 907
908 908
909 909
910 910
911 911
912 912
913 913
914 914
915 915
916 916
917 917
918 918
919 919
920 920
921 921
922 922
923 923
924 924
925 925
926 926
927 927
928 928
929 929
930 930
931 931
932 932
933 933
934 934
935 935
936 936
937 937
938 938
939 939
940 940
941 941
942 942
943 943
944 944
945 945
946 946
947 947
948 948
949 949
950 950
951 951
952 952
953 953
954 954
955 955
956 956
957 957
958 958
959 959
960 960
961 961
962 962
963 963
964 964
965 965
966 966
967 967
968 968
969 969
970 970
971 971
972 972
973 973
974 974
975 975
976 976
977 977
978 978
979 979
980 980
981 981
982 982
983 983
984 984
985 985
986 986
987 987
988 988
989 989
990 990
991 991
992 992
993 993
994 994
995 995
996 996
997 997
998 998
999 999
1000 1000

```

Con respecto a clean code, con los refactorios se trataba de cumplir lo máximo posible con estos principios. Hasta que en el último commit de esta rama se llegó al siguiente código:

```

1 + namespace BackEnd
2 + {
3 +     public class Promotion200ff : IPromotionable
4 +     {
5 +         private const float _twentyPercent = 0.2f;
6 +
7 +         public bool IsApplicable(Purchase p)
8 +         {
9 +             return p.Cart.Count >= 2;
10 +        }
11 +
12 +         public int CalculateDiscount(Purchase p)
13 +         {
14 +             if (!IsApplicable(p))
15 +             {
16 +                 throw new BackEndException("Not applicable promotion");
17 +             }
18 +
19 +             int maxPrice = 0;
20 +             foreach (Product item in p.Cart)
21 +             {
22 +                 if (item.Price > maxPrice)
23 +                 {
24 +                     maxPrice = item.Price;
25 +                 }
26 +             }
27 +
28 +             return (int)(_twentyPercent * maxPrice);
29 +         }
30 +     }
31 + }
32 +
33 +

```

- Se implementó una interfaz, ya que se reconoció código repetido con las demás promociones.

- Se definió la constante 20% para facilitar la lectura del código.
- Se negaron los condicionales de los if, de modo de mejorar la claridad y de reducir los niveles de indentación.
- Se utilizaron nombres de variables y de funciones claras y concisas, un ejemplo de esto es el nombre de la función “IsApplicable”, que con solo leerlo ya nos imaginamos que el retorno va a ser de tipo bool.

Las demás promociones son análogas a esta, se aplicó la misma metodología que la promoción detallada anteriormente, pero debido a la acotada extensión de este documento, no serán incluidas.

Feature 2: Usuario

Si se buscan los primeros commits, vemos que estos refieren a la clase User. Se definió el User desde el dominio, con sus properties y sus validaciones correspondientes. A continuación, se muestra un ejemplo de un commit que implementa las validaciones en el email del usuario: <https://github.com/IngSoft-DA2-2023-2/222675-269851-270303/commit/ab023f05922489c95d46f6df64f9736b2c49707b>

Por motivos que serán explicados en el archivo de *descripción del diseño*, luego de tener la clase User se procedió a crear el UserController. A continuación, se presenta un link en caso de querer ver los commits de la rama utilizada: <https://github.com/IngSoft-DA2-2023-2/222675-269851-270303/commits/feature/user>

Con respecto a TDD, a continuación se muestra el primer commit de dicha rama:

<pre>[TestClass] public class UserControllerTest { [TestMethod] public void GetAllUsersOk() { List<User> usersSample = new() { new User {Email= "mail1@sample.com",Name="name1",Password="password1" }, new User {Email= "mail2@sample.com",Name="name2",Password="password2" }, new User {Email= "mail3@sample.com",Name="name3",Password="password3" }, }; Mock<IUserLogic> mock = new(); mock.Setup(u => u.GetUsers()).Returns(usersSample); UserController userController = new(mock.Object); var result = userController.GetAllUsers().Result as OkObjectResult; Assert.IsNotNull(result); Assert.AreEqual(usersSample, result.Value); } }</pre>	<pre>[Route("api/[controller]")] [ApiController] public class UserController : ControllerBase { private IUserLogic _userLogic; public UserController(IUserLogic logic) { _userLogic = logic; } [HttpGet] public ActionResult<List<User>> GetAllUsers() { try { return Ok(_userLogic.GetUsers()); } catch (Exception) { return StatusCode(500); } } }</pre>
---	---

Haciendo una retrospectiva, detectamos múltiples errores en este código. Desde el uso de try y catches en el controller, hasta no haber usado “MockBehavior.Strict”. Errores que con el transcurso de los commits fueron solucionados.

A continuación se mostrará cómo fue el proceso de implementar el delete de usuarios. El primer paso fue crear el método en el controller, en el commit e186dbf se observan los siguientes cambios:

Tests:

```
163 +
164 +     [TestMethod]
165 +     public void DeleteUserReturnsOk()
166 +     {
167 +         Mock<UserLogic> mock = new();
168 +
169 +         var userId = Guid.NewGuid();
170 +         mock.Setup(logic => logic.DeleteUser(userId));
171 +
172 +         UserController userController = new(mock.Object);
173 +         var result = userController.DeleteUser(userId).Result as StatusCodeResult;
174 +
175 +         Assert.IsNotNull(result);
176 +         Assert.AreEqual(200, result.StatusCode);
177 +     }
178 +
179 +     [TestMethod]
180 +     public void DeleteUserThrowsLogicalException()
181 +     {
182 +         Mock<UserLogic> mock = new();
183 +
184 +         var userId = Guid.NewGuid();
185 +         mock.Setup(logic => logic.DeleteUser(userId)).Throws(new LogicException("error"));
186 +
187 +         UserController userController = new(mock.Object);
188 +         var result = userController.DeleteUser(userId).Result as StatusCodeResult;
189 +
190 +         Assert.IsNotNull(result);
191 +         Assert.AreEqual(400, result.StatusCode);
192 +     }
193 + }
```

UserController:

```
71 +     [HttpDelete]
72 +     public ActionResult<CreateUserResponse> DeleteUser([FromQuery] Guid userId)
73 +     {
74 +         try
75 +         {
76 +             _userLogic.DeleteUser(userId);
77 +             return Ok();
78 +         }
79 +         catch (Exception)
80 +         {
81 +             return BadRequest();
82 +         }
83 +     }
84 +
85 +
86 +
87 + }
```

IUserLogic:

```
1  Ecommerce/LogicInterface/IUserLogic.cs
@@ -6,6 +6,7 @@ public interface IUserLogic
6  {
7      public List<User> GetUsers();
8      public Guid AddUser(User user);
9 +     public void DeleteUser(Guid id);
10
11 }
12 }
```

Luego, más adelante se implementó la capa lógica. Específicamente en el commit bb3f869

Tests:

UserLogic:

123 + [TestMethod]
124 + public void DeleteUser()
125 + {
126 + User toDelete = new User() { Email = "a@a.com" };
127 +
128 + User deleted = new User()
129 + {
130 + Name = "Juan",
131 + Email = "a@a.com",
132 + Address = "aaa",
133 + Password = "12345",
134 + Roles = new List<string> { "buyer" },
135 + };
136 +
137 + Mock<IUserRepository> repo = new Mock<IUserRepository>(MockBehavior.Strict);
138 + repo.Setup(logic => logic.DeleteUser(It.IsAny<User>())).Returns(deleted);
139 + var userLogic = new UserLogic(repo.Object);
140 +
141 + var result = userLogic.DeleteUser(toDelete);
142 +
143 + repo.VerifyAll();
144 + Assert.AreEqual(result.Email, toDelete.Email);
145 + }
146 +

Ecommerce/BusinessLogic/UserLogic.cs
@@ -35,7 +35,7 @@ public User UpdateUser(User user)
35 35
36 36 public User DeleteUser(User user)
37 37 {
38 - throw new NotImplementedException();
38 + return _userRepository.DeleteUser(user);
39 39 }
40 40
41 41

Y por ende se modificó el IUserRepository:

@@ -5,6 +5,7 @@ namespace DataAccessInterface
5 public interface IUserRepository
6 {
7 User CreateUser(User user);
8 + User DeleteUser(User user);
9 bool Exist(Func<User, bool> predicate);
10 IEnumerable<User> GetAllUsers(Func<User, bool> predicate);
11 User UpdateUser(User user);

Por último, en el commit intitulado “[green] deletion of user from DB”, se modificó la capa de la BD.

El código y las pruebas correspondientes fueron:

Tests

UserRepository

```

[TestMethod]
public void DeleteUser()
{
    User newUser = new User
    {
        Name = "TestUser",
        Email = "test@example.com"
    };

    var userContext = new Mock<ECommerceContext>();
    userContext.Setup(c => c.Users).ReturnsDbSet(
        new List<User>
        {
            new User
            {
                Name = "TestUser",
                Email = "test@example.com"
            }
        }
    );
    userContext.Setup(c => c.Users.Remove(newUser));
    userContext.Setup(c => c.SaveChanges());

    IUserRepository userRepository = new UserRepository(userContext.Object);
    var expectedReturn = userRepository.DeleteUser(newUser);
    Assert.AreEqual(expectedReturn.Name, newUser.Name);
    Assert.AreEqual(expectedReturn.Email, newUser.Email);
}

[TestMethod]
[ExpectedException(typeof(DataAccessException))]
public void DeleteNonExistingUser()
{
    User newUser = new User
    {
        Name = "TestUser",
        Email = "test@example.com"
    };

    var userContext = new Mock<ECommerceContext>();
    userContext.Setup(c => c.Users).ReturnsDbSet(new List<User>());
    userContext.Setup(c => c.Users.Remove(newUser));
    userContext.Setup(c => c.SaveChanges());

    IUserRepository userRepository = new UserRepository(userContext.Object);
    var expectedReturn = userRepository.DeleteUser(newUser);
}

```

```

public User DeleteUser(User user)
{
    throw new NotImplementedException();
    var existingUser = _eCommerceContext.Users.FirstOrDefault(u => u.Email == user.Email);

    if (existingUser != null)
    {
        _eCommerceContext.Users.Remove(existingUser);
        _eCommerceContext.SaveChanges();
        return existingUser;
    }
    throw new DataAccessException($"No users found");
}

```

Con respecto a los filtros, a continuación, se presenta el AnnotatedCustomExceptionFilter:

```

namespace WebApi.Filters
{
    public class AnnotatedCustomExceptionFilter : ExceptionFilterAttribute
    {
        public override void OnException(ExceptionContext context)
        {
            if (context.Exception is LogicException)
            {
                context.Result = new ObjectResult(new { ErrorMessage = context.Exception.Message })
                {
                    StatusCode = StatusCodes.Status400BadRequest,
                };
            }
            else if (context.Exception is UnauthorizedAccessException)
            {
                context.Result = new ObjectResult(new { ErrorMessage = context.Exception.Message })
                {
                    StatusCode = StatusCodes.Status403Forbidden,
                };
            }
        }
    }
}

```

Reconocemos que no es lo más fiel a los principios de clean code preguntar por tipo(rtti), y que lo óptimo hubiera sido usar polimorfismo. No obstante, dada la baja cantidad de excepciones no nos pareció que fuera grave.

Por último, para el login y logout se utilizaron dos ramas nuevas: sessionCreation y sessionDeletion. A continuación se muestran los commits de esta última:

[refactor] decorators at controller diegoacu1234 committed last week
[green] data access diegoacu1234 committed last week
[green] logic diegoacu1234 committed last week
[green/refactor] controller diegoacu1234 committed last week
[green] model in diegoacu1234 committed last week

A modo de ejemplo se muestra el caso del commit “[green/refactor] controller”

SessionControllerTest

```
[TestMethod]
public void DeleteSession()
{
    Guid guid = Guid.NewGuid();

    DeleteSessionRequest received = new DeleteSessionRequest()
    {
        Token = guid,
    };

    Session session = new Session() { SessionToken = guid };
    var expectedMappedResult = new SessionResponse(session);

    Mock<ISessionLogic> logic = new Mock<ISessionLogic>(MockBehavior.Strict);
    logic.Setup(logic => logic.LogOut(It.IsAny<Guid>())).Returns(session);
    var sessionController = new SessionController(logic.Object);
    var expectedActionResult = new CreatedActionResult("CreateSession", "Session", new { id = 5 }, expectedMappedResult);

    var result = sessionController.LogOut(received);

    logic.VerifyAll();
    OkObjectResult resultObject = result as OkObjectResult;
    SessionResponse resultValue = resultObject.Value as SessionResponse;
    Assert.AreEqual(resultObject.StatusCode, StatusCodes.Status200OK);
    Assert.AreEqual(guid, resultValue.Token);
}
```

SessionController

```
+
+ [HttpPost]
+ public IActionResult Logout([FromBody] DeleteSessionRequest request)
+ {
+     Session session = _sessionLogic.LogOut(request.Token);
+     var result = new SessionResponse(session);
+
+     return Ok(result);
+ }
```

Se observa que los tests siguen el formato AAA y que los nombres de las variables son simples y autoexplicativos.

Feature 3: Administrador

Para implementar las funcionalidades del administrador se tuvo que modificar el controlador del usuario y su lógica. A continuación se muestra los commits hechos en refactor/splitUserAndAdmin

[refactor] continue separating client and admin diegoacul234 committed last week	720d755
[refactor] changed so that if updating a user no data is put, then no... ... changes are made there diegoacul234 committed last week	8f32019
[refactor] changed name of update by admin and allow an admin to modi... ...fy user email diegoacul234 committed last week	3e6b9d7
[refactor] splitted updation in 2 cases (own and by admin) diegoacul234 committed last week	0fb3a23
[refactor] further testing diegoacul234 committed last week	eac998e
[green/refactor] splitted user creation in 2 cases(self and by admin) diegoacul234 committed last week	ebd480c
Merge branch 'quickfix/userGeneralRefactors' into Develop diegoacul234 committed last week	61a3b5
[refactor] split user request in 2 cases and removed logic from data ... diegoacul234 committed last week	15ff53f
[refactor] corrected name notation for endpoints diegoacul234 committed last week	c4afcd0

Por ejemplo, en “[green/refactor] splitted user creation in 2 cases(self and by admin)”, se observa que la función que antes se llamaba CreateUser del Controller pasó a llamarse RegistrationByAdmin, y a su vez se creó el método SelfRegistration, ambos respaldados por tests, como se observa a continuación:

UserControllerTest:

```
[TestMethod]
public void CreateUser()
public void CreateUserByAdmin()

(continúa el test)

[TestMethod]
public void CreateUserByThemself()

(continúa el test)
```

UserController:

```
31 + [HttpPost]
32 + public IActionResult SelfRegistration([FromBody] CreateUserByThemselfRequest received)
33 + {
34 +     var user = received.ToEntity();
35 +     var resultLogic = _userLogic.AddUserByThemself(user);
36 +     var result = new CreateUserResponse(resultLogic);
37 +
38 +     return CreatedAtAction(nameof(RegistrationByAdmin), result);
39 + }
40
41 [HttpPost]
42 + [Route("admin")]
43 [AnnotatedCustomExceptionHandler]
44 [AuthenticationFilter]
45 - public IActionResult CreateUser([FromBody] CreateUserRequest received)
46 + public IActionResult RegistrationByAdmin([FromBody] CreateUserByAdminRequest received)
47 {
48     var user = received.ToEntity();
49     var resultLogic = _userLogic.AddUser(user);
50     var result = new UserResponse(resultLogic);
51     return CreatedAtAction(nameof(CreateUser), result);
52 +     var resultLogic = _userLogic.AddUserByAdmin(user);
53     var result = new CreateUserResponse(resultLogic);
54
55     return CreatedAtAction(nameof(RegistrationByAdmin), result);
56 }
```

Y para el update fue el mismo proceso, por lo que se incluye únicamente los cambios hechos en UserController en el commit “[refactor] splitted updation in 2cases”:

```

[HttpPut("admin/{id}")]
[AnnotatedCustomExceptionHandler]
[AuthenticationFilter]
public IActionResult UpdateUserByAdmin([FromBody] UpdateUserRequestByAdmin received, Guid id)
{
    var user = received.ToEntity();
    user.Guid = id;

    var resultLogic = _userLogic.UpdateUser(user);
    var result = new UserResponse(resultLogic);

    return Ok(result);
}

[HttpPut("{id}")]
[AnnotatedCustomExceptionHandler]
[AuthenticationFilter]
public IActionResult UpdateUser([FromBody] UpdateUserRequest received, Guid id)
public IActionResult UpdateUserByThemselves([FromBody] UpdateUserRequestByAdmin received, Guid id)
{
    var user = received.ToEntity();
    user.Guid = id;
}

```

Por último, para que los admins puedan ver las compras realizadas, se trabajó en una rama nueva llamada “feature/fixControllers”. Algunos de sus commits son:

Commits on Sep 29, 2023

- [GREEN] CreateProductUnauthorized test done
nuhalde committed 5 days ago
- [GREEN] UpdateProductUnauthorized test done
nuhalde committed 5 days ago
- [green] get instead of post in PurchaseController
nuhalde committed 5 days ago
- Merge pull request #6 from IngSoft-DA2-2023-2/feature/controllersAndF...
nuhalde committed 5 days ago
- [REFACTOR] refactor of ProductController and ProductControllerTest
nuhalde committed 5 days ago
- [REFACTOR] refactor of PurchaseController and PurchaseControllerTest
nuhalde committed 5 days ago
- [REFACTOR] refactor of SessionController and SessionControllerTest, a...
nuhalde committed 5 days ago

En particular, haremos énfasis en “[REFACTOR] refactor of PurchaseController and PurchaseControllerTest”, que ya se encontraba implementado, pero en este caso se le agregaron los filtros correspondientes para que solo el admin pueda ver todas las compras.

Code Coverage:

Hierarchy	Covered (Lines)	Partially Covered (Lines)	Not Covered (Lines) ▲	Covered (%Lines)
<ul style="list-style-type: none"> Felipe_DESKTOP-4PRQ0JJ_2023-10-05.18_09_04.coverage <ul style="list-style-type: none"> dataaccessinterface.dll <ul style="list-style-type: none"> DataAccessInterface.Exceptions utilities.dll <ul style="list-style-type: none"> Utilities logicinterface.dll <ul style="list-style-type: none"> LogicInterface.Exceptions apimodels.dll <ul style="list-style-type: none"> ApiModels.In ApiModels.Out domain.dll <ul style="list-style-type: none"> Domain.Exceptions Domain.ProductParts Domain webapi.dll <ul style="list-style-type: none"> WebApi.Controllers dataaccess.dll <ul style="list-style-type: none"> DataAccess.Repository businesslogic.dll <ul style="list-style-type: none"> BusinessLogic.Promotions BusinessLogic 	848 2 2 6 6 2 2 221 117 104 99 2 19 78 103 103 158 158 257 100 157	13 0 0 0 0 0 0 0 0 0 7 0 0 7 1 1 1 1 4 0 4	38 0 0 0 0 0 1 0 1 3 0 0 1 6 6 7 7 21 8 13	94.33% 100.00% 100.00% 100.00% 100.00% 100.00% 99.55% 100.00% 99.05% 90.83% 100.00% 95.00% 89.66% 93.64% 93.64% 95.18% 95.18% 91.13% 92.59% 90.23%

El code Coverage nos muestra que el equipo logro llegar a casi el 95% de cobertura. Al observar detenidamente que líneas no fueron testeadas, encontramos que estas principalmente son Funcs. Al realizar las pruebas con Moq, el equipo utilizo `It.IsAny<Func<...,bool>` para probar los predicados. Lo que trajo como consecuencia que dichas líneas no pudieran ser testeadas con mayor profundidad. No obstante, se intentara solucionar dicho problema para la segunda entrega.

<ul style="list-style-type: none"> <ul style="list-style-type: none"> SessionLogic.<>c__DisplayClass3_0 SessionLogic.<>c__DisplayClass4_0 UserLogic.<>c UserLogic.<>c__DisplayClass10_0 UserLogic.<>c__DisplayClass11_0 UserLogic.<>c__DisplayClass3_0 UserLogic.<>c__DisplayClass4_0 UserLogic.<>c__DisplayClass6_0 UserLogic.<>c__DisplayClass7_0 UserLogic.<>c__DisplayClass9_0 	0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1 1 1	0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00% 0.00%
---	--	--	--	--