

University of Pretoria  
Software Engineering - COS 301

---

## Testing Policy

---

Contrapositives  
May 2019

**Authors:**

Brendan Bath	<b>u16023359</b>
Musa Mathe	<b>u15048030</b>
Jessica da Silva	<b>u16045816</b>
Natasha Draper	<b>u16081758</b>

## Contents

<b>1</b>	<b>Definition</b>	<b>2</b>
<b>2</b>	<b>Description of Tests</b>	<b>2</b>
<b>3</b>	<b>Test Evaluation</b>	<b>2</b>
<b>4</b>	<b>Link to Travis CI</b>	<b>2</b>
<b>5</b>	<b>Screenshots</b>	<b>1</b>

## 1 Definition

Testing is a process aimed to detect any errors, defects, or undesirable results in the software being developed. The other purpose of testing is to determine if the software and the results comply with the requirements that have been specified. By testing, the project can continuously be improved upon and chance for risks and errors to occur will be reduced.

## 2 Description of Tests

There are 3 types of testing that are being utilized:

- Unit testing
- Linting
- End-to-End

Unit testing tests individual units of code. The purpose of unit testing is to verify that the individual components are working properly, so that the system as a whole is unaffected. For the backend, each API endpoint is tested, and for the frontend each individual screen is tested.

Linting tests if code meets the coding standards and style standards, as well as checking for syntax errors. Linting is done to make sure that the code is more readable and understandable, which in turn will make it easier to find errors in code and improve the code overall. It also ensures that the code is uniform; different styles used in multiple places reduces the readability of the code. Both linting and unit testing are done on the frontend and the backend.

End-to-End (E2E) or Integration testing tests the whole application as if a user were using it. It can simulate mouse presses and typing; things that a user can do when using the application. E2E can also tests if the front and back are working together.

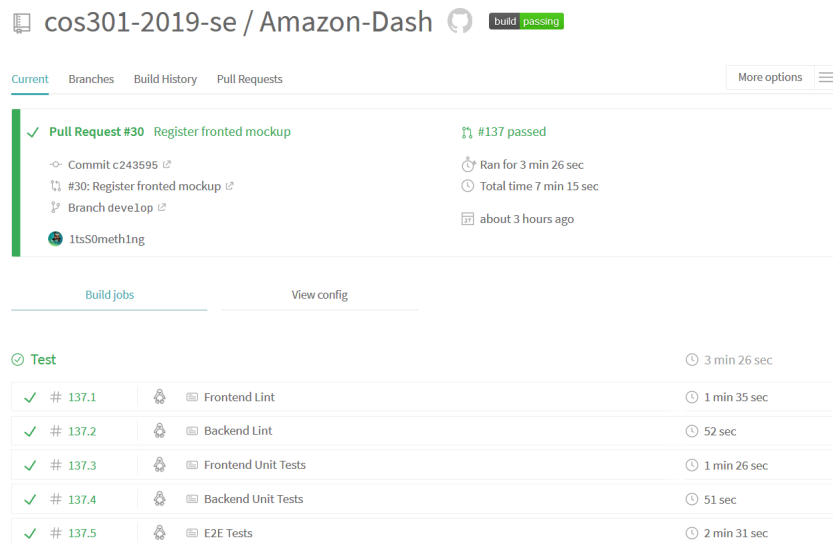
## 3 Test Evaluation

We test the return value of functions and make assertions. This is to ensure that they match the desired result or that they throw an appropriate exception. If an exception is thrown, the cause of the error can be easily identified and resolved.

## 4 Link to Travis CI

[Travis-CI](#)


## 5 Screenshots

cos301-2019-se / Amazon-Dash  build: passing

Current Branches Build History Pull Requests More options

✓ Pull Request #30 Register fronted mockup #137 passed

→ Commit c243595 [↗](#) Ran for 3 min 26 sec  
🔗 #30: Register fronted mockup [↗](#) Total time 7 min 15 sec  
🔗 Branch develop [↗](#) about 3 hours ago

 1ts0meth1ng

Build jobs View config

🟢 Test 3 min 26 sec






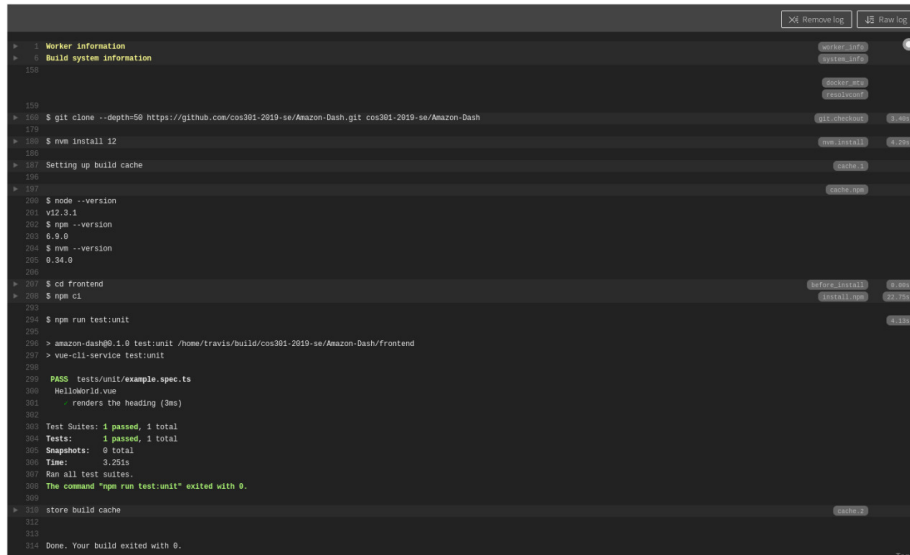
✓ # 137.1	 Frontend Lint	<span>1 min 35 sec</span>
✓ # 137.2	 Backend Lint	<span>52 sec</span>
✓ # 137.3	 Frontend Unit Tests	<span>1 min 26 sec</span>
✓ # 137.4	 Backend Unit Tests	<span>51 sec</span>
✓ # 137.5	 E2E Tests	<span>2 min 31 sec</span>

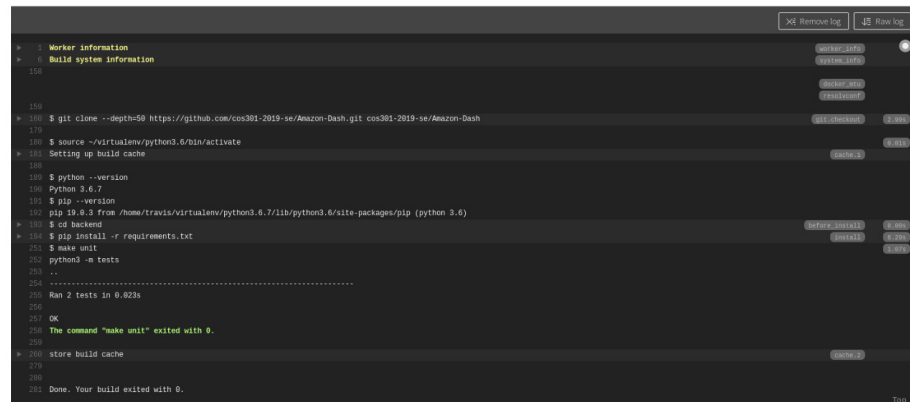
Figure 1: Travis CI repository



This screenshot shows a Travis CI build log for a frontend unit testing process. The log is displayed in a dark-themed interface with a sidebar on the left showing build stages: Worker information, Build system information, and a list of steps. The main area shows the execution of commands and their output. The steps include cloning the repository, installing Node.js, setting up the build cache, installing dependencies, running unit tests, and storing the build cache. The test output shows a single test passing.

```
159 $ git clone --depth=50 https://github.com/cos301-2019-se/Amazon-Dash.git cos301-2019-se/Amazon-Dash
160 $ npm install 12
161 Setting up build cache
162 $ node --version
163 v12.3.1
164 $ npm --version
165 6.9.0
166 $ npm --version
167 6.9.0
168 $ cd frontend
169 $ npm ci
170 $ npm run test:unit
171 > amazon-dash@1.0 test:unit /home/travis/build/cos301-2019-se/Amazon-Dash/frontend
172 > vue-cli-service test:unit
173
174 PASS tests/unit/example.spec.ts
175   HelloWorld.vue
176     renders the heading (h1)
177
178   Test Suites: 1 passed, 1 total
179   Tests: 1 passed, 1 total
180   Snapshots: 0 total
181   Time: 0.251s
182   Ran all test suites.
183 The command "npm run test:unit" exited with 0.
184 store build cache
185 Done. Your build exited with 0.
```

Figure 2: Frontend Unit Testing



This screenshot shows a Travis CI build log for a backend unit testing process. The log is displayed in a dark-themed interface with a sidebar on the left showing build stages: Worker information, Build system information, and a list of steps. The main area shows the execution of commands and their output. The steps include cloning the repository, installing Python and pip, setting up the build cache, installing dependencies, running unit tests, and storing the build cache. The test output shows two tests passing.

```
159 $ git clone --depth=50 https://github.com/cos301-2019-se/Amazon-Dash.git cos301-2019-se/Amazon-Dash
160 $ source ~/virtualenv/python3.6/bin/activate
161 Setting up build cache
162 $ python --version
163 Python 3.6.7
164 $ pip --version
165 pip 19.0.3 from /home/travis/virtualenv/python3.6.7/lib/python3.6/site-packages/pip (python 3.6)
166 $ cd backend
167 $ pip install -r requirements.txt
168 $ make unit
169 python3 -m tests
170 ..
171 .....
172 Ran 2 tests in 0.022s
173
174 OK
175 The command "make unit" exited with 0.
176 store build cache
177 Done. Your build exited with 0.
```

Figure 3: Backend Unit Testing

<https://www.overleaf.com/6667523733rrcczmrhxxqq>

✕ Remove log

📊 Run log

```
179 # 5 Worker information
180 # 6 Build system information
181
182
183
184
185
186 $ git clone --depth=50 https://github.com/cos301-2019-se/Amazon-Dash.git cos301-2019-se/Amazon-Dash
187
188 $ npm install 12
189
190 Setting up build cache
191
192
193 $ node --version
194 v12.13.1
195 $ npm --version
196 6.9.0
197 $ npm --version
198 6.9.0
199
200 $ cd frontend
201 $ npm ci
202
203 $ npm run lint
204
205 > amazon-dash@0.1.0 lint /home/travis/build/cos301-2019-se/Amazon-Dash/frontend
206 > vue-cli-service lint
207
208 No lint errors found.
209
210 The command "npm run lint" exited with 0.
211
212
213 store build cache
214
215 Done. Your build exited with 0.
```

worker\_info

system\_info

docker\_logs

cache\_logs

git\_checked

0.00s

npm\_install

4.40s

cache\_1

cache\_logs

before\_install

0.00s

install\_run

19.00s

cache\_2

0.00s

Top

Figure 4: Frontend Linting

XX Remote log

42 Run log

```
1 Worker information
2 Build system information
3
4
5
6
7
8
9
10
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
10
```

Figure 5: Backend Linting