

University of Pretoria  
Software Engineering - COS 301

---

# **Amazon Dash Software Requirements Specification**

---

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.

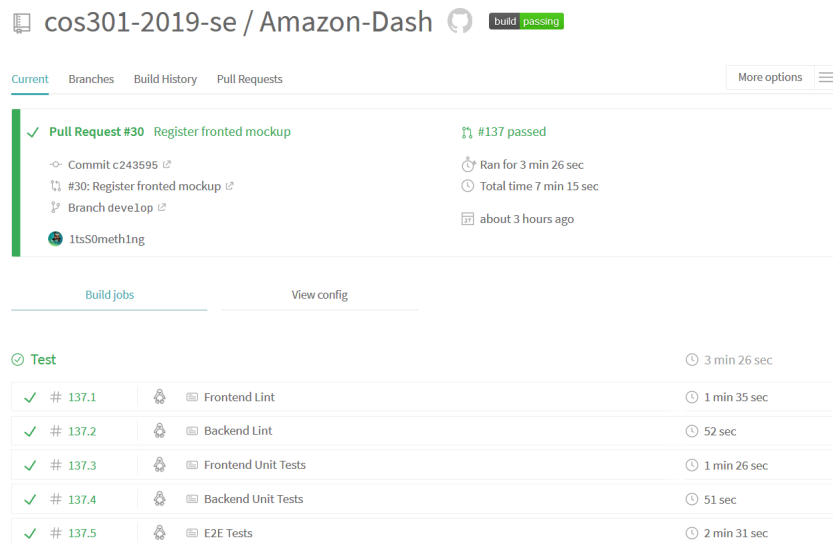
## 3 Test Evaluation

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	🕒 1 min 35 sec
✓ # 137.2	🔗 Backend Lint	🕒 52 sec
✓ # 137.3	🔗 Frontend Unit Tests	🕒 1 min 26 sec
✓ # 137.4	🔗 Backend Unit Tests	🕒 51 sec
✓ # 137.5	🔗 E2E Tests	🕒 2 min 31 sec

Figure 1: Travis CI repository

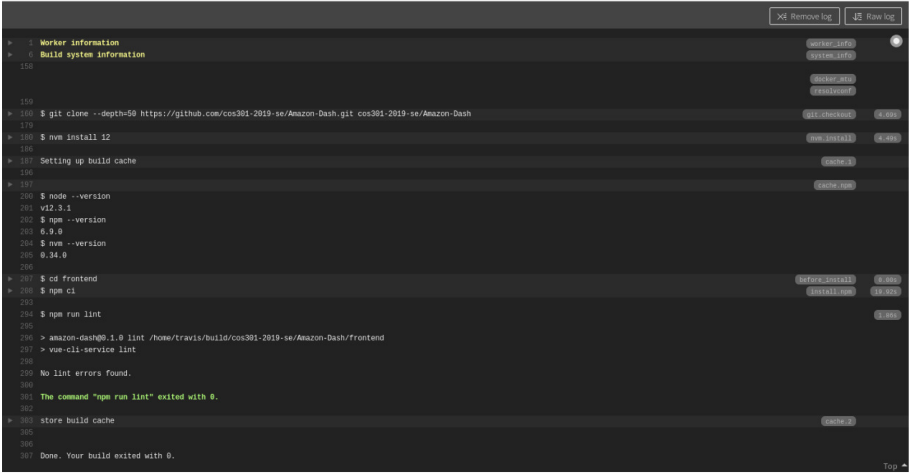
```
Worker information
Build system information
159
160 $ git clone --depth=50 https://github.com/cos301-2019-se/Amazon-Dash.git cos301-2019-se/Amazon-Dash
161
162 $ npm install 12
163
164 Setting up build cache
165
166 $ node --version
167 v12.3.1
168 $ npm --version
169 6.9.0
170 $ npm --version
171 6.9.0
172
173 $ cd frontend
174
175 $ npm ci
176
177 $ npm run test:unit
178
179 > amazon-dash@1.0 test:unit /home/travis/build/cos301-2019-se/Amazon-Dash/frontend
180 > vue-cli-service test:unit
181
182 PASS tests/unit/example.spec.ts
183   HelloWorld.vue
184     renders the heading (h1)
185
186 Test Suites: 1 passed, 1 total
187 Tests: 1 passed, 1 total
188 Snapshots: 0 total
189 Time: 0.251s
190 Ran all test suites.
191 The command "npm run test:unit" exited with 0.
192
193 store build cache
194
195 Done. Your build exited with 0.
```

Figure 2: Frontend Unit Testing

```
Worker information
Build system information
159
160 $ git clone --depth=50 https://github.com/cos301-2019-se/Amazon-Dash.git cos301-2019-se/Amazon-Dash
161
162 $ source ~/.virtualenv/python3.6/bin/activate
163
164 Setting up build cache
165
166 $ python --version
167 Python 3.6.7
168 $ pip --version
169 pip 19.0.3 from /home/travis/virtualenv/python3.6.7/lib/python3.6/site-packages/pip (python 3.6)
170
171 $ cd backend
172
173 $ pip install -r requirements.txt
174
175 $ make unit
176
177 python3 -m tests
178 ..
179 .....
180 Ran 2 tests in 0.022s
181
182 OK
183 The command "make unit" exited with 0.
184
185 store build cache
186
187 Done. Your build exited with 0.
```

Figure 3: Backend Unit Testing

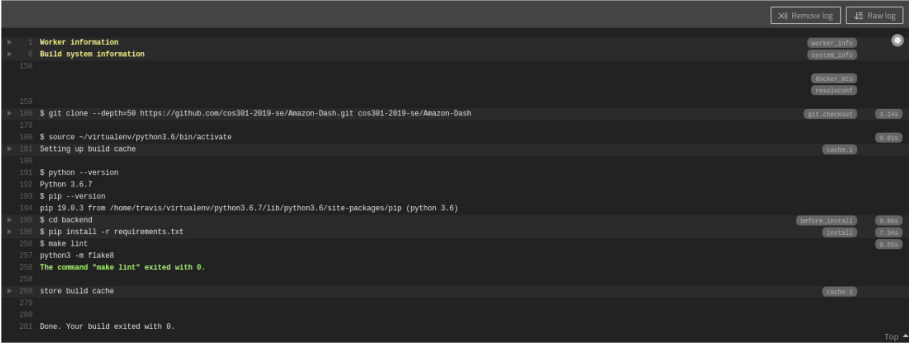
<https://www.overleaf.com/6667523733rrcczmrfhxqq>



The screenshot shows a Travis CI build log for a project. The log is displayed in a dark-themed interface with a sidebar on the left showing the build steps. The main area shows the execution of commands. The steps are: 1. Worker information, 2. Build system information, 3. git clone, 4. npm install, 5. Setting up build cache, 6. node --version, 7. npm --version, 8. npm --version, 9. npm --version, 10. cd frontend, 11. npm ci, 12. npm run lint, 13. store build cache, 14. Done. The linting step shows the command 'npm run lint' and the output 'No lint errors found.'.

```
158 $ git clone --depth=50 https://github.com/cos301-2019-se/Amazon-Dash.git cos301-2019-se/Amazon-Dash
159
160 $ npm install 12
161
162 Setting up build cache
163
164 $ node --version
165 v12.1.1
166 $ npm --version
167 6.9.0
168 $ npm --version
169 6.9.0
170 $ npm --version
171 6.9.0
172
173 $ cd frontend
174 $ npm ci
175
176 $ npm run lint
177
178 > amazon-dash@1.0 lint /home/travis/build/cos301-2019-se/Amazon-Dash/frontend
179 > vue-cli-service lint
180
181 No lint errors found.
182
183 The command "npm run lint" exited with 0.
184
185 store build cache
186
187 Done. Your build exited with 0.
```

Figure 4: Frontend Linting



The screenshot shows a Travis CI build log for a project. The log is displayed in a dark-themed interface with a sidebar on the left showing the build steps. The main area shows the execution of commands. The steps are: 1. Worker information, 2. Build system information, 3. git clone, 4. source ~/.virtualenv/python3.6/bin/activate, 5. Setting up build cache, 6. python --version, 7. python 3.6.7, 8. pip --version, 9. pip 19.0.3 from /home/travis/virtualenv/python3.6.7/lib/python3.6/site-packages/pip (python 3.6), 10. cd backend, 11. pip install -r requirements.txt, 12. make lint, 13. python3 -m flake8, 14. store build cache, 15. Done. The linting step shows the command 'make lint' and the output 'The command "make lint" exited with 0.'.

```
158 $ git clone --depth=50 https://github.com/cos301-2019-se/Amazon-Dash.git cos301-2019-se/Amazon-Dash
159
160 $ source ~/.virtualenv/python3.6/bin/activate
161
162 Setting up build cache
163
164 $ python --version
165 Python 3.6.7
166 $ pip --version
167 pip 19.0.3 from /home/travis/virtualenv/python3.6.7/lib/python3.6/site-packages/pip (python 3.6)
168
169 $ cd backend
170 $ pip install -r requirements.txt
171
172 $ make lint
173
174 python3 -m flake8
175
176 The command "make lint" exited with 0.
177
178 store build cache
179
180 Done. Your build exited with 0.
```

Figure 5: Backend Linting