Chapter 4

IMPLEMENTATION PHASE

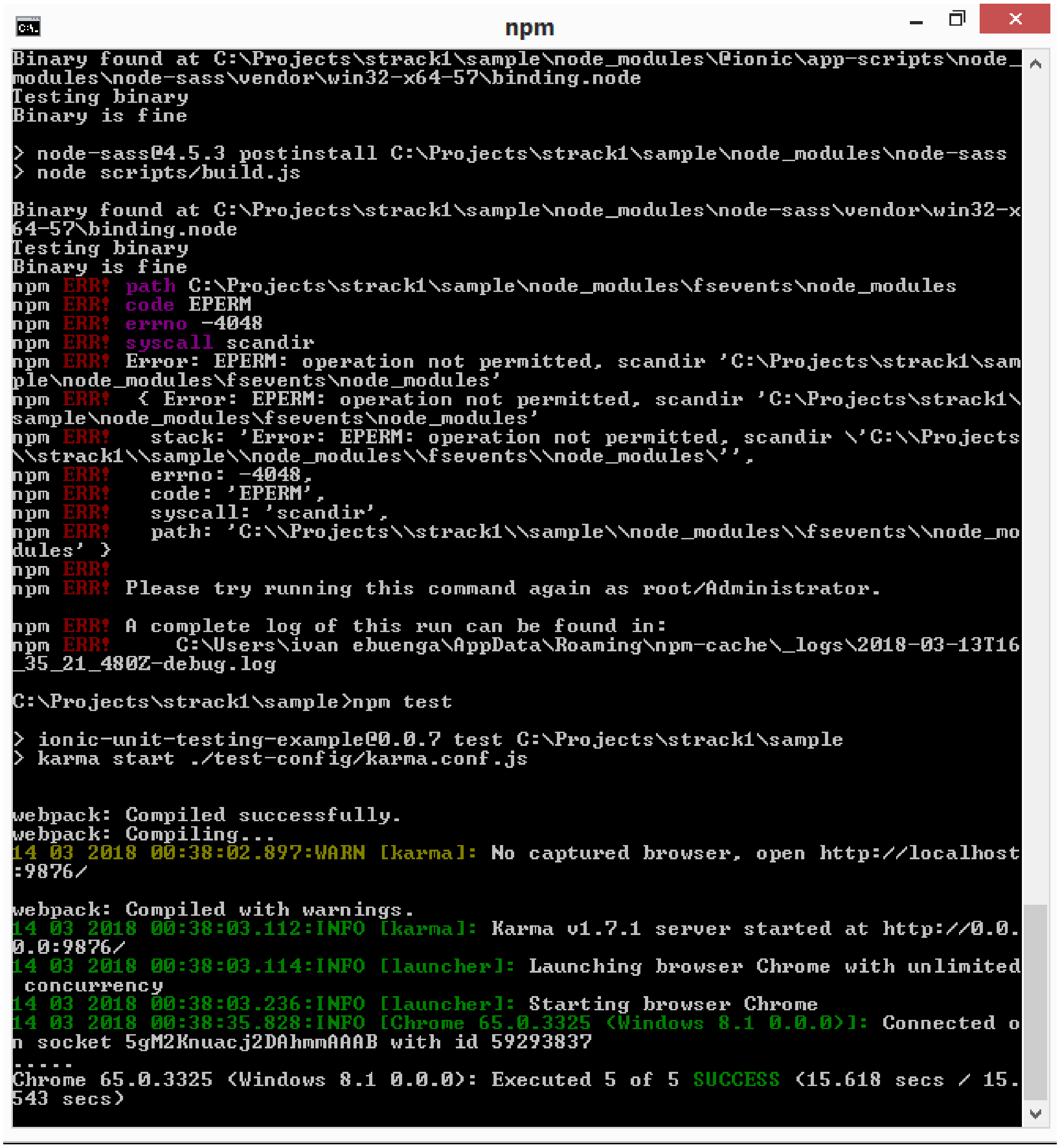
Testing the System

After the code is developed it is tested against the requirements to make sure that the product is solving the needs addressed and gathered during the requirements phase. During this phase unit testing, system testing and acceptance testing are done.

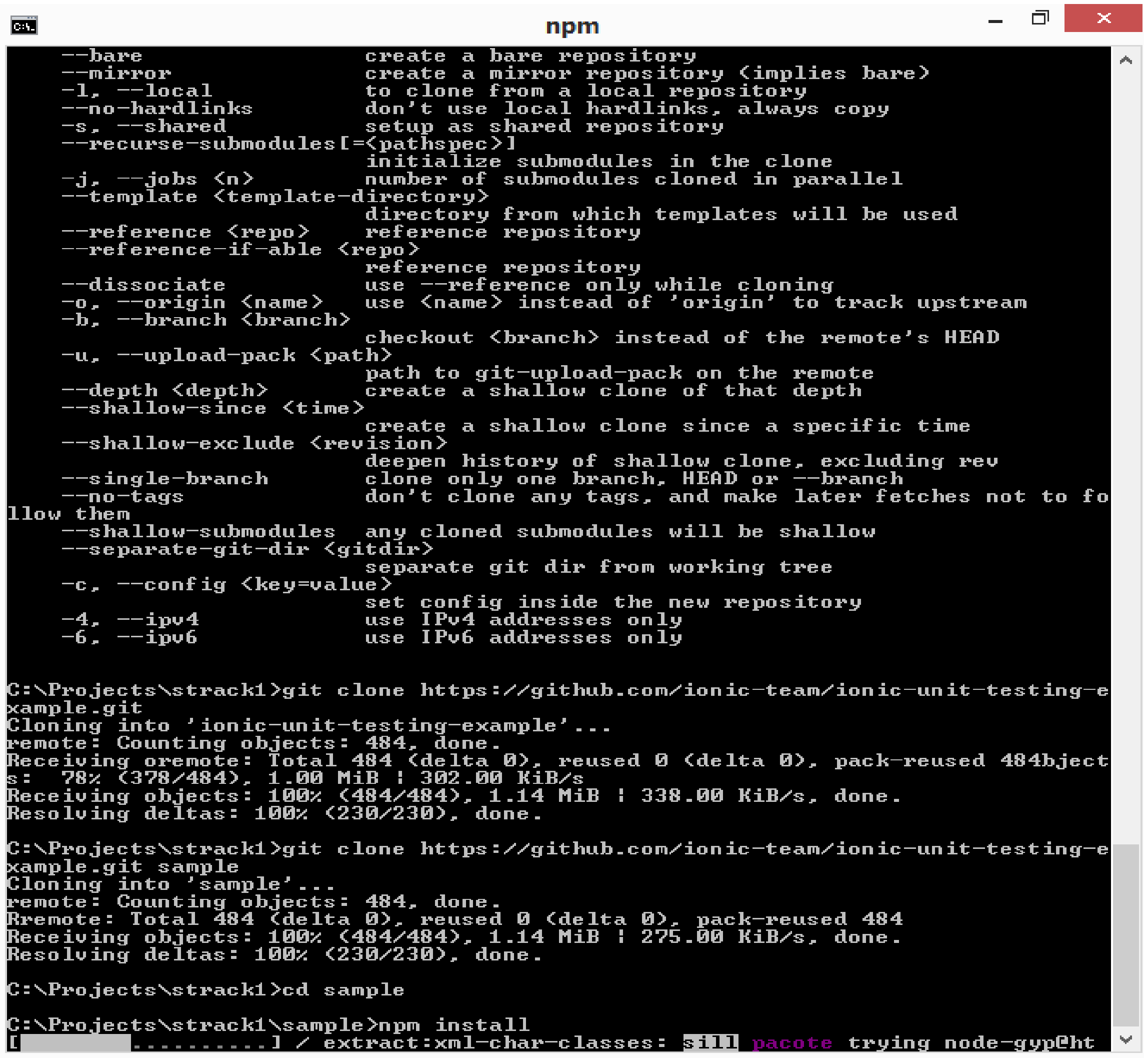
a. Unit Testing

According to ISTQB Exam Certification, a unit is the smallest testable part of an application like functions, classes, procedures, interfaces. Unit testing is a method by which individual units of source code are tested to determine if they are fit for use. The goal of unit testing is to segregate each part of the program and test that the individual parts are working correctly.

The unit testing is done using NPM Package Manager and it was successful. It is the default package manager for the JavaScript runtime environment Node.js. The following are the screenshots of the results in testing the code of the system.



```
Binary found at C:\Projects\strack1\sample\node_modules\@ionic\app-scripts\node_
modules\node-sass\vendor\win32-x64-57\binding.node
Testing binary
Binary is fine

> node-sass@4.5.3 postinstall C:\Projects\strack1\sample\node_modules\node-sass
> node scripts/build.js

Binary found at C:\Projects\strack1\sample\node_modules\node-sass\vendor\win32-x
64-57\binding.node
Testing binary
Binary is fine
npm ERR! path C:\Projects\strack1\sample\node_modules\fsevents\node_modules
npm ERR! code EPERM
npm ERR! errno -4048
npm ERR! syscall scandir
npm ERR! Error: EPERM: operation not permitted, scandir 'C:\Projects\strack1\sam
ple\node_modules\fsevents\node_modules'
npm ERR!  { Error: EPERM: operation not permitted, scandir 'C:\Projects\strack1\
sample\node_modules\fsevents\node_modules'
npm ERR!   stack: 'Error: EPERM: operation not permitted, scandir \'C:\\Projects
\\strack1\\sample\\node_modules\\fsevents\\node_modules\'',
npm ERR!   errno: -4048,
npm ERR!   code: 'EPERM',
npm ERR!   syscall: 'scandir',
npm ERR!   path: 'C:\\Projects\\strack1\\sample\\node_modules\\fsevents\\node_mo
dules' }
npm ERR!
npm ERR! Please try running this command again as root/Administrator.

npm ERR! A complete log of this run can be found in:
npm ERR!     C:\Users\ivan ebuenga\AppData\Roaming\npm-cache\_logs\2018-03-13T16
_35_21_480Z-debug.log

C:\Projects\strack1\sample>npm test

> ionic-unit-testing-example@0.0.7 test C:\Projects\strack1\sample
> karma start ./test-config/karma.conf.js


webpack: Compiled successfully.
webpack: Compiling...
14 03 2018 00:38:02.897:WARN [karma]: No captured browser, open http://localhost
:9876/

webpack: Compiled with warnings.
14 03 2018 00:38:03.112:INFO [karma]: Karma v1.7.1 server started at http://0.0.
0.0:9876/
14 03 2018 00:38:03.114:INFO [launcher]: Launching browser Chrome with unlimited
 concurrency
14 03 2018 00:38:03.236:INFO [launcher]: Starting browser Chrome
14 03 2018 00:38:35.828:INFO [Chrome 65.0.3325 (Windows 8.1 0.0.0)]: Connected o
n socket 5gM2Knuacj2DAhmmAAAB with id 59293837
.....
Chrome 65.0.3325 (Windows 8.1 0.0.0): Executed 5 of 5 SUCCESS (15.618 secs / 15.
543 secs)
```



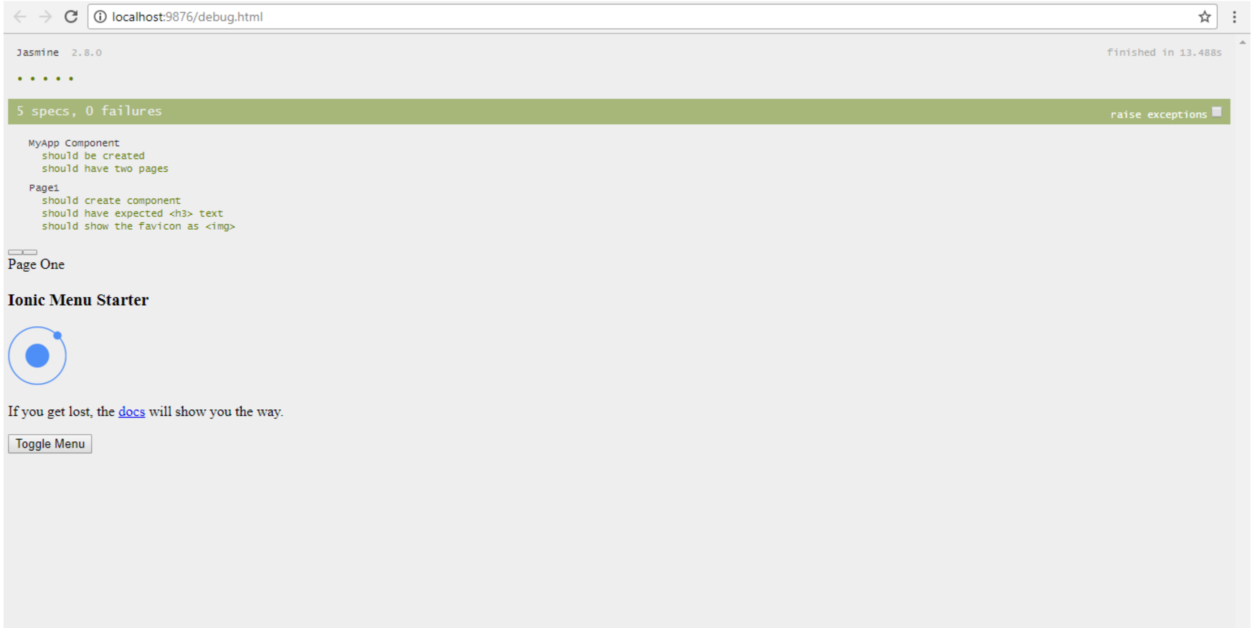
```
    --bare                create a bare repository
    --mirror              create a mirror repository (implies bare)
    -l, --local           to clone from a local repository
    --no-hardlinks        don't use local hardlinks, always copy
    -s, --shared          setup as shared repository
    --recurse-submodules[=<pathspec>]
                          initialize submodules in the clone
    -j, --jobs <n>        number of submodules cloned in parallel
    --template <template-directory>
                          directory from which templates will be used
    --reference <repo>    reference repository
    --reference-if-able <repo>
                          reference repository
    --dissociate          use --reference only while cloning
    -o, --origin <name>   use <name> instead of 'origin' to track upstream
    -b, --branch <branch>
                          checkout <branch> instead of the remote's HEAD
    -u, --upload-pack <path>
                          path to git-upload-pack on the remote
    --depth <depth>       create a shallow clone of that depth
    --shallow-since <time>
                          create a shallow clone since a specific time
    --shallow-exclude <revision>
                          deepen history of shallow clone, excluding rev
    --single-branch       clone only one branch, HEAD or --branch
    --no-tags             don't clone any tags, and make later fetches not to fo
llow them
    --shallow-submodules  any cloned submodules will be shallow
    --separate-git-dir <gitdir>
                          separate git dir from working tree
    -c, --config <key=value>
                          set config inside the new repository
    -4, --ipv4            use IPv4 addresses only
    -6, --ipv6            use IPv6 addresses only


C:\Projects\strack1>git clone https://github.com/ionic-team/ionic-unit-testing-e
xample.git
Cloning into 'ionic-unit-testing-example'...
remote: Counting objects: 484, done.
Receiving oremote: Total 484 (delta 0), reused 0 (delta 0), pack-reused 484bject
s:  78% (378/484), 1.00 MiB | 302.00 KiB/s
Receiving objects: 100% (484/484), 1.14 MiB | 338.00 KiB/s, done.
Resolving deltas: 100% (230/230), done.

C:\Projects\strack1>git clone https://github.com/ionic-team/ionic-unit-testing-e
xample.git sample
Cloning into 'sample'...
remote: Counting objects: 484, done.
Rremote: Total 484 (delta 0), reused 0 (delta 0), pack-reused 484
Receiving objects: 100% (484/484), 1.14 MiB | 275.00 KiB/s, done.
Resolving deltas: 100% (230/230), done.

C:\Projects\strack1>cd sample

C:\Projects\strack1\sample>npm install
[████████..........] / extract:xml-char-classes: sill pacote trying node-gyp@ht
```

Figure 4.1
Unit Testing

b. System Testing

According to ISTQB Exam Certification, in system testing the behavior of whole system/product is tested as defined by the scope of the development project or product. It may include tests based on risks and/or requirement specifications, business process, use cases, or other high-level descriptions of system behavior, interactions with the operating systems, and system resources.

The proponents held the system testing using Karma, Jasmine testing framework. The proponents did verify that the system met the specifications and purpose.



```
Jasmine  2.8.0                                        finished in 13.488s
•••••
5 specs, 0 failures                                    raise exceptions □

  MyApp Component
    should be created
    should have two pages
  Page1
    should create component
    should have expected <h3> text
    should show the favicon as <img>
```

Page One

Ionic Menu Starter

If you get lost, the docs will show you the way.

Figure 4.2
System Testing

c. User Acceptance Testing

According to TechoPedi, user acceptance testing (UAT) is the last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications.

The respondents test the High School Principal’s Department Office. Overall the system met the user requirements given by the principal and has been evaluated based on the overall functionality, reliability, usability, efficiency, maintainability and portability.

The proponents use the ISO 9126 Characteristics to test the user Acceptance.

ISO 9126 Characteristics	Standard Quality	User's Percentage Rating (100% highest – 0% lowest)
Functionality	A set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs.	
Reliability	A set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time.	
Usability	A set of attributes that bear on the effort needed for use, and on the individual assessment of such use by a stated or implied set of users.	
Efficiency	A set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions.	
Maintainability	A set of attributes that bear on the effort needed to make specified modified modifications.	
Portability	A set of attributes that bear on the ability of software to be transferred from on environment to another.	

Table 43
ISO 9126 Characteristics