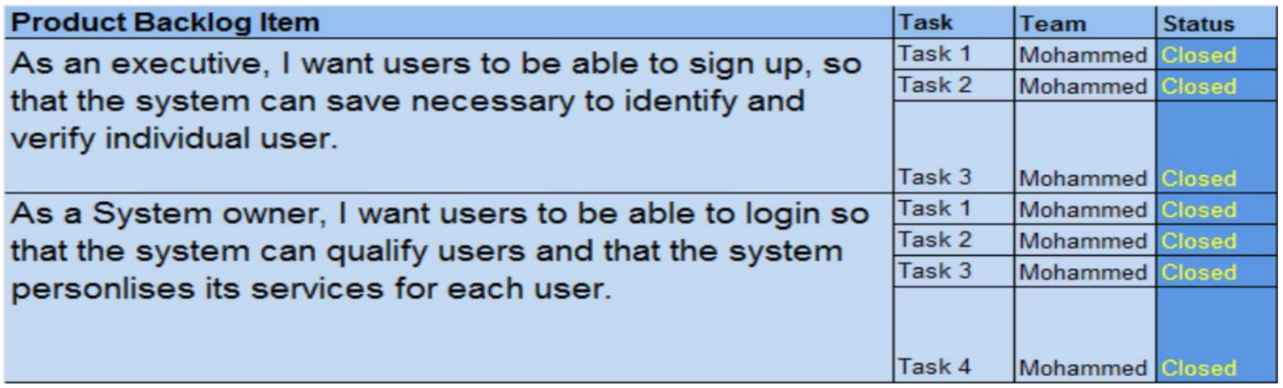
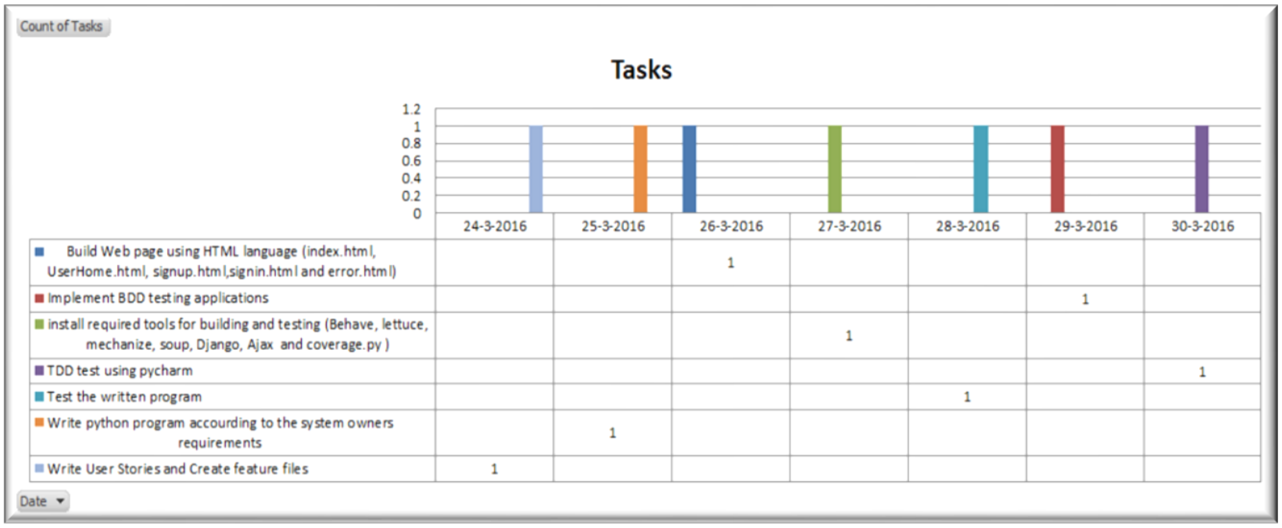
BlueGarden Project

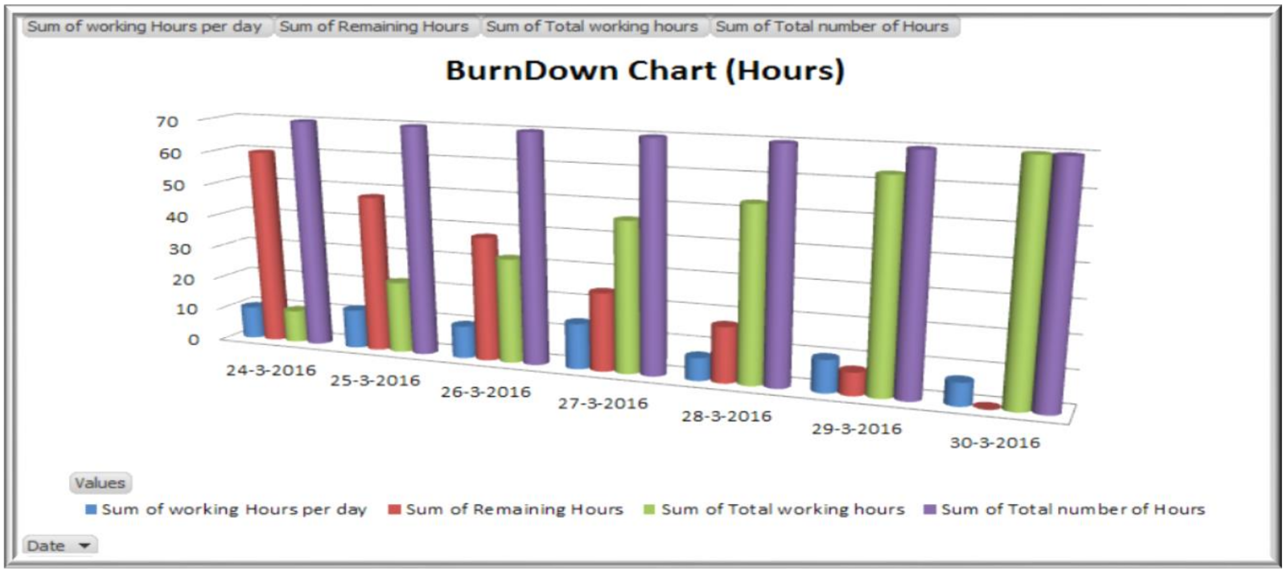
Sprint 1 (Backlog and Statistics):

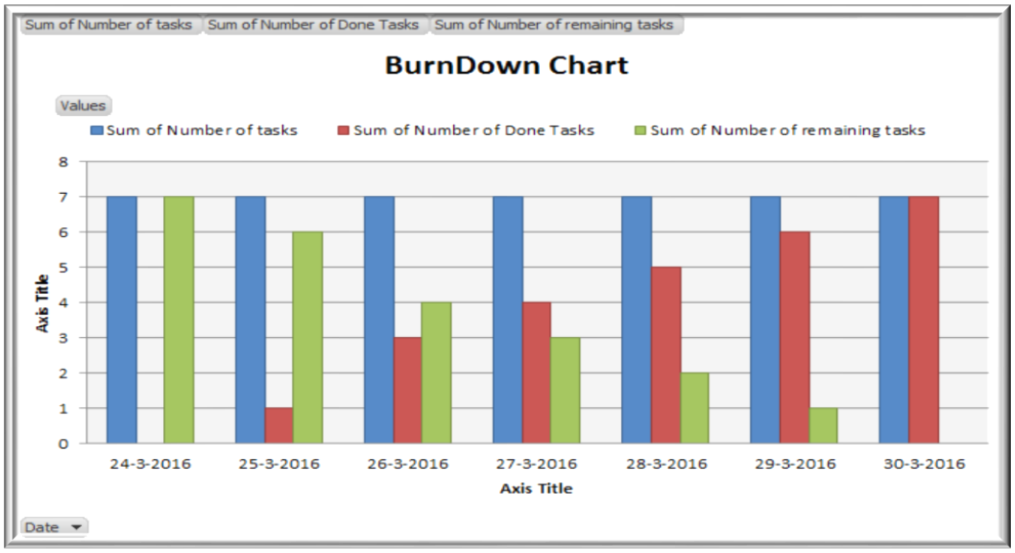


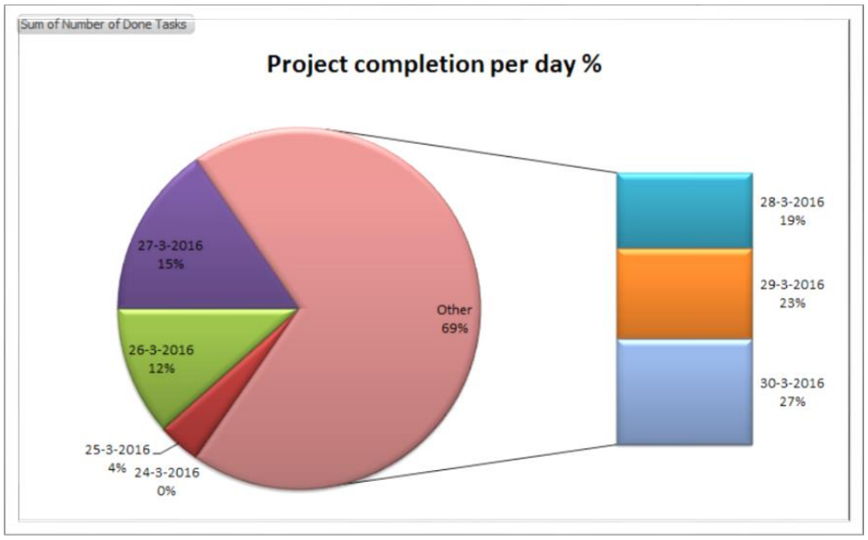
Product Backlog Item	Task	Team	Status
As an executive, I want users to be able to sign up, so that the system can save necessary to identify and verify individual user.	Task 1	Mohammed	Closed
	Task 2	Mohammed	Closed
	Task 3	Mohammed	Closed
As a System owner, I want users to be able to login so that the system can qualify users and that the system personlises its services for each user.	Task 1	Mohammed	Closed
	Task 2	Mohammed	Closed
	Task 3	Mohammed	Closed
	Task 4	Mohammed	Closed

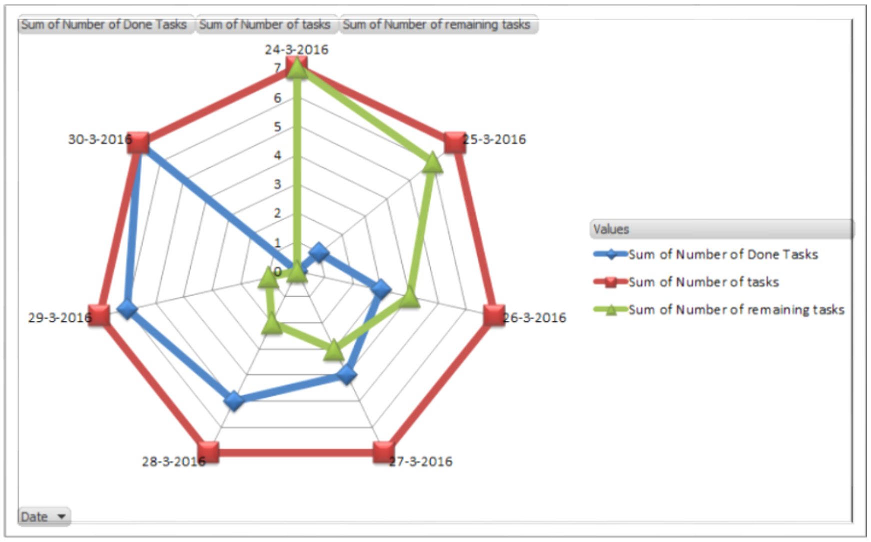


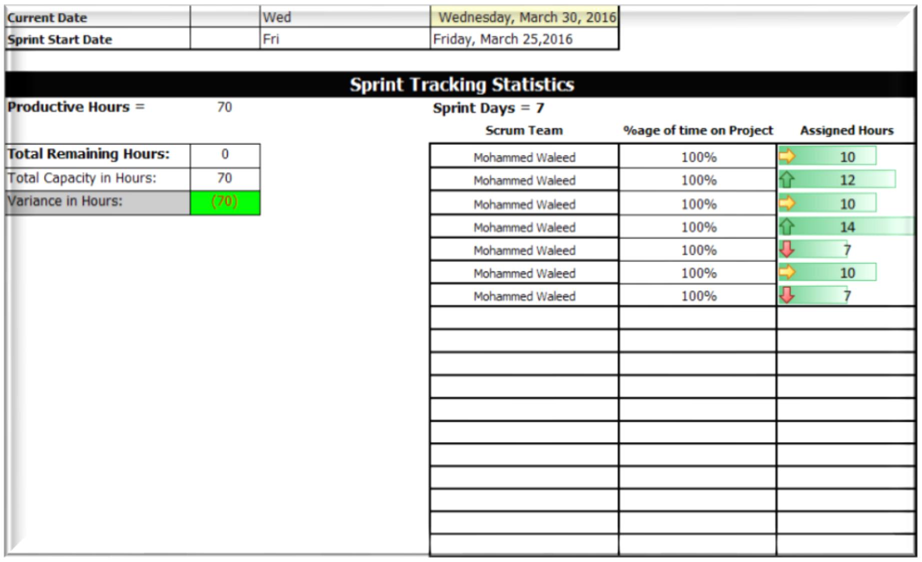
	24-3-2016	25-3-2016	26-3-2016	27-3-2016	28-3-2016	29-3-2016	30-3-2016
Build Web page using HTML language (index.html, UserHome.html, signup.html,signin.html and error.html)			1				
Implement BDD testing applications						1	
install required tools for building and testing (Behave, lettuce, mechanize, soup, Django, Ajax and coverage.py)				1			
TDD test using pycharm							1
Test the written program					1		
Write python program accourding to the system owners requirements		1					
Write User Stories and Create feature files	1						




Sum of Number of tasks
Sum of Number of Done Tasks
Sum of Number of remaining tasks
BurnDown Chart
Values
Sum of Number of tasks
Sum of Number of Done Tasks
Sum of Number of remaining tasks
Axis Title
24-3-2016
25-3-2016
26-3-2016
27-3-2016
28-3-2016
29-3-2016
30-3-2016
Axis Title
Date


Sum of Number of Done Tasks
Project completion per day %
27-3-2016 15%
26-3-2016 12%
25-3-2016 4%
24-3-2016 0%
Other 69%
28-3-2016 19%
29-3-2016 23%
30-3-2016 27%


Sum of Number of Done Tasks
Sum of Number of tasks
Sum of Number of remaining tasks
24-3-2016
25-3-2016
26-3-2016
27-3-2016
28-3-2016
29-3-2016
30-3-2016
Values
Sum of Number of Done Tasks
Sum of Number of tasks
Sum of Number of remaining tasks
Date



Current Date		Wed	Wednesday, March 30, 2016
Sprint Start Date		Fri	Friday, March 25,2016

Sprint Tracking Statistics

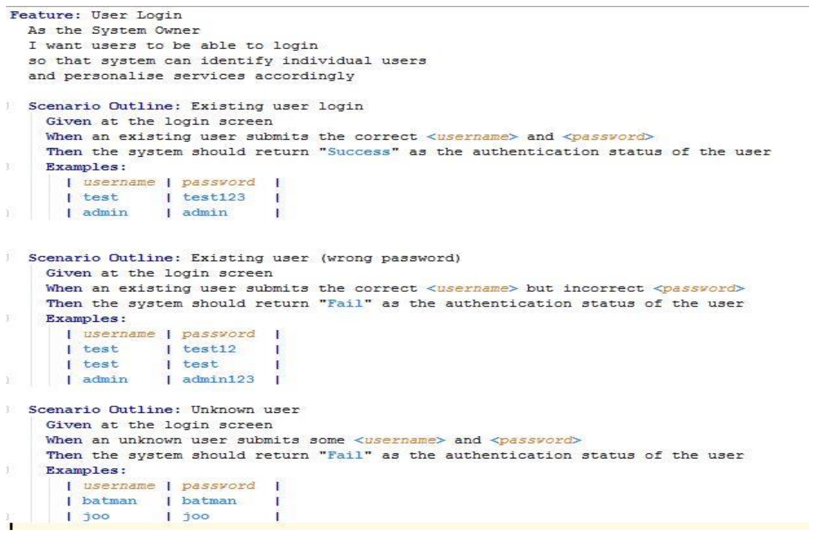
Productive Hours =	70
Total Remaining Hours:	0
Total Capacity in Hours:	70
Variance in Hours:	(70)

Sprint Days = 7

Scrum Team	%age of time on Project	Assigned Hours
Mohammed Waleed	100%	➡ 10
Mohammed Waleed	100%	⇧ 12
Mohammed Waleed	100%	➡ 10
Mohammed Waleed	100%	⇧ 14
Mohammed Waleed	100%	⇩ 7
Mohammed Waleed	100%	➡ 10
Mohammed Waleed	100%	⇩ 7

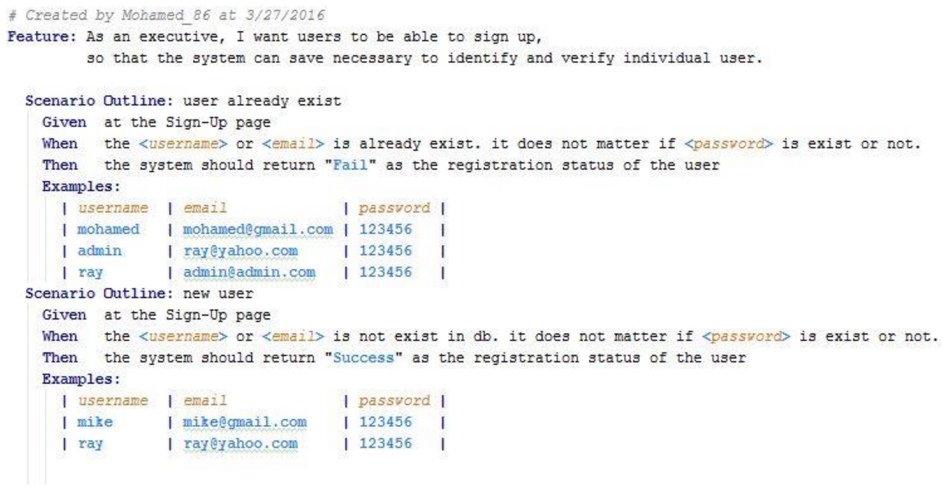
Write the users story using Gherkin format:

User story 1:



```
Feature: User Login
  As the System Owner
  I want users to be able to login
  so that system can identify individual users
  and personalise services accordingly

  Scenario Outline: Existing user login
    Given at the login screen
    When an existing user submits the correct <username> and <password>
    Then the system should return "Success" as the authentication status of the user
    Examples:
      | username | password  |
      | test     | test123   |
      | admin    | admin     |


  Scenario Outline: Existing user (wrong password)
    Given at the login screen
    When an existing user submits the correct <username> but incorrect <password>
    Then the system should return "Fail" as the authentication status of the user
    Examples:
      | username | password  |
      | test     | test12    |
      | test     | test      |
      | admin    | admin123  |

  Scenario Outline: Unknown user
    Given at the login screen
    When an unknown user submits some <username> and <password>
    Then the system should return "Fail" as the authentication status of the user
    Examples:
      | username | password  |
      | batman   | batman    |
      | joo      | joo       |
```

User story 2:

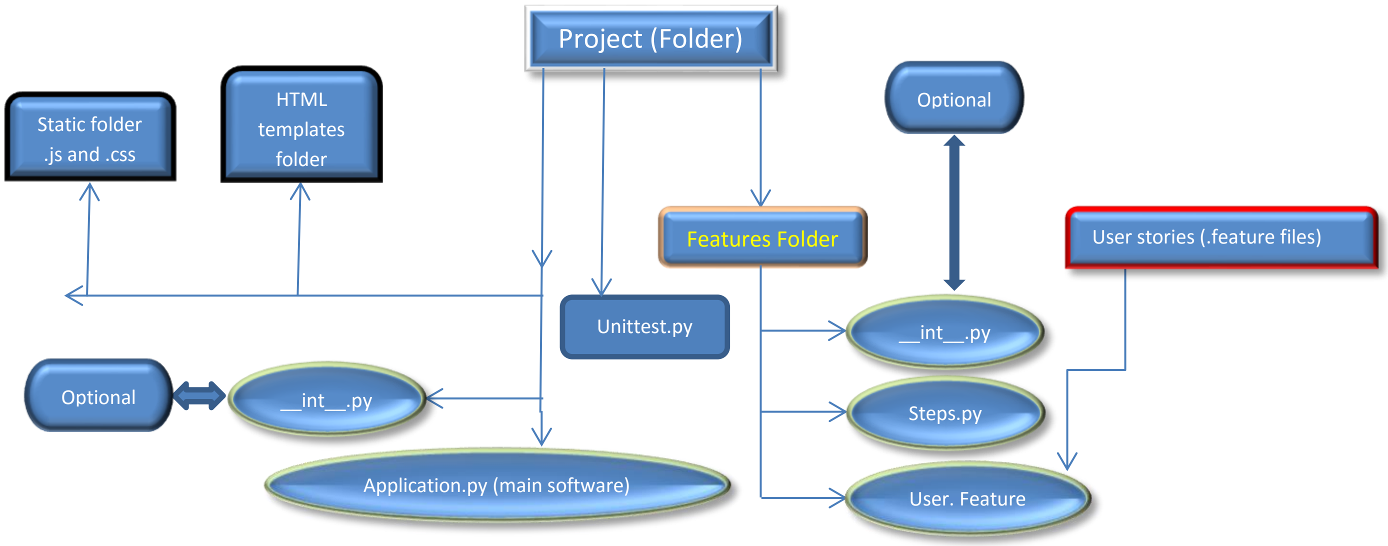


```
# Created by Mohamed_86 at 3/27/2016
Feature: As an executive, I want users to be able to sign up,
         so that the system can save necessary to identify and verify individual user.

  Scenario Outline: user already exist
    Given  at the Sign-Up page
    When   the <username> or <email> is already exist. it does not matter if <password> is exist or not.
    Then   the system should return "Fail" as the registration status of the user
    Examples:
      | username  | email               | password |
      | mohamed   | mohamed@gmail.com   | 123456   |
      | admin     | ray@yahoo.com       | 123456   |
      | ray       | admin@admin.com     | 123456   |
  Scenario Outline: new user
    Given  at the Sign-Up page
    When   the <username> or <email> is not exist in db. it does not matter if <password> is exist or not.
    Then   the system should return "Success" as the registration status of the user
    Examples:
      | username  | email               | password |
      | mike      | mike@gmail.com      | 123456   |
      | ray       | ray@yahoo.com       | 123456   |
```

- Write the program to fulfill the system owner needs.
  - A- Python programming language to implement main feature of the system (Sign-In, Sign-Up). (.py files)
  - B- HTML language to create: Home, Login and Sign-up web pages.(.html files)
  - C- Flask framework: to make a web based application.
  - D- MySQL (database server), create a database and connect it with the Flask
  - E- Ajax: is the method of exchanging data with a server, and updating parts of a web page - without reloading the entire page (.js file).
  - F- Behave, lettuce, mechanize, soup and Django tools for Acceptance test.
  - G- Coverge.py
  - H- Pycharm code inspection feature for unit test

- Prepare BDD (Behavior driven development) files and directories to perform the acceptance test:

  BDD puts the user at the center of the tests. It is an increasingly popular method that helps to validate the code from an end-user point of view. Tests are expressed as scenarios (use cases).



- __init__.py: mark directory as a Python package.
- steps.py: The Python code which is executed by the .feature files.
- User. Feature: The behavior test which describes the functionality of the user endpoint in our application.
- application.py: The entry point where our Flask application is created and the server started.

Initializing the Test Environment

We'll start by initializing the test environment and determining which test browser to use, since we will rely on a Web browser to execute our tests. We should append the following python program in features/environment.py.

Behave tests are made of 2 types of files:

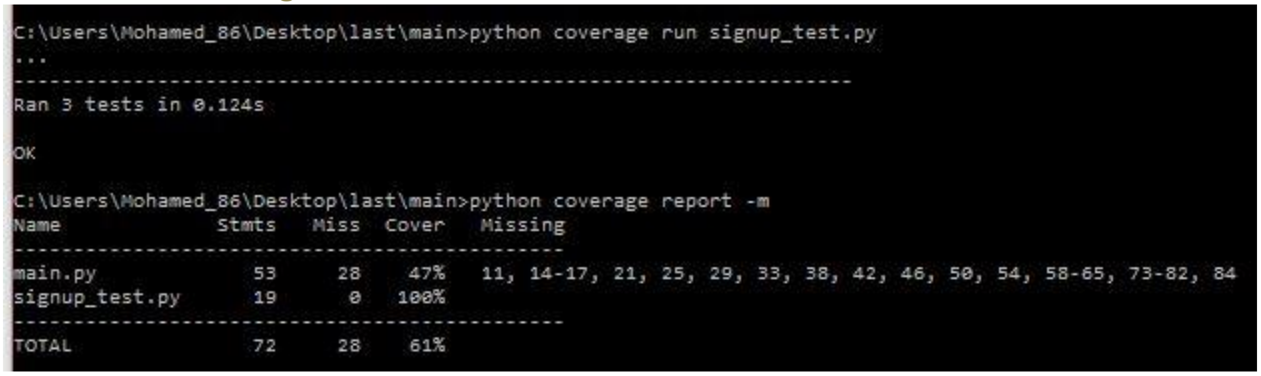
- **Feature description**: contains the human-readable form of your tests, written as scenarios (user. Feature)
- **Test script**: contains the machine-executable form of your tests, written in Python (steps.py)
-

We have to write a test program to check the functionality of our application depending on the user story. **Save the tests files as Steps.py and run the tests file using behave_django. Get the behave report analyze it to check the application functionality.**

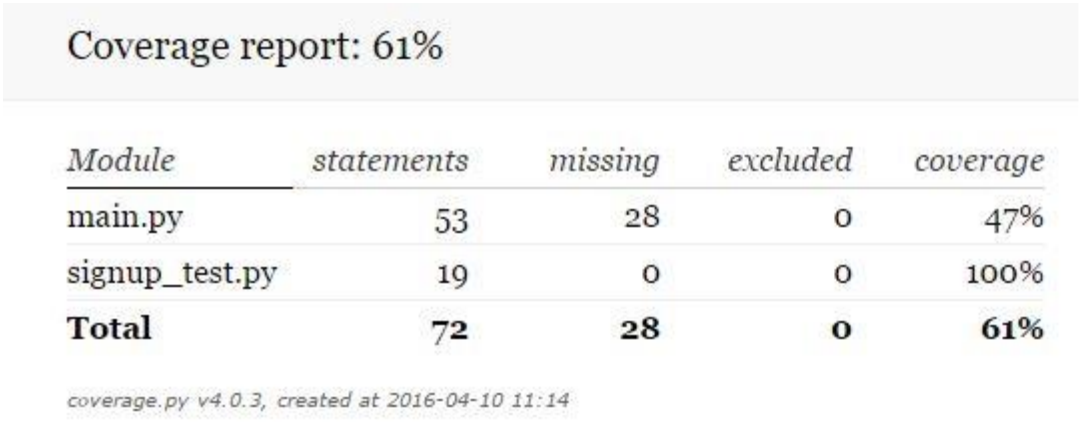
- **Test Driven development (TDD):** This is the process of building integrated tests into all the code that you create, and running those tests every time you do a build. It's as if you are extending the compiler, telling it more about what your program is supposed to do. That way, the build process can check for more than just syntax errors, since you teach it how to check for semantic errors as well.

Coverage.py: Coverage.py is a tool for measuring code coverage of Python programs. It monitors your program, noting which parts of the code have been executed, then analyzes the source to identify code that could have been executed but was not. Coverage measurement is typically used to gauge the Effectiveness of tests. It can show which parts of your code are being exercised by tests, and which are not.

TDD test & coverage:



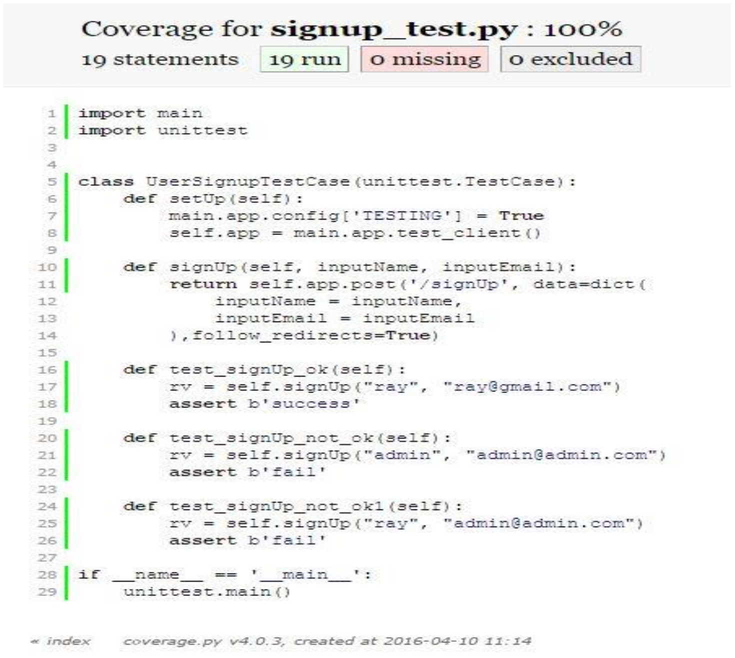
```
C:\Users\Mohamed_86\Desktop\last\main>python coverage run signup_test.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.124s

OK

C:\Users\Mohamed_86\Desktop\last\main>python coverage report -m
Name              Stmts   Miss  Cover   Missing
-------------------------------------------------
main.py              53     28    47%   11, 14-17, 21, 25, 29, 33, 38, 42, 46, 50, 54, 58-65, 73-82, 84
signup_test.py       19      0   100%
-------------------------------------------------
TOTAL                72     28    61%
```



Coverage report: 61%

Module	statements	missing	excluded	coverage
main.py	53	28	0	47%
signup_test.py	19	0	0	100%
Total	**72**	**28**	**0**	**61%**

coverage.py v4.0.3, created at 2016-04-10 11:14

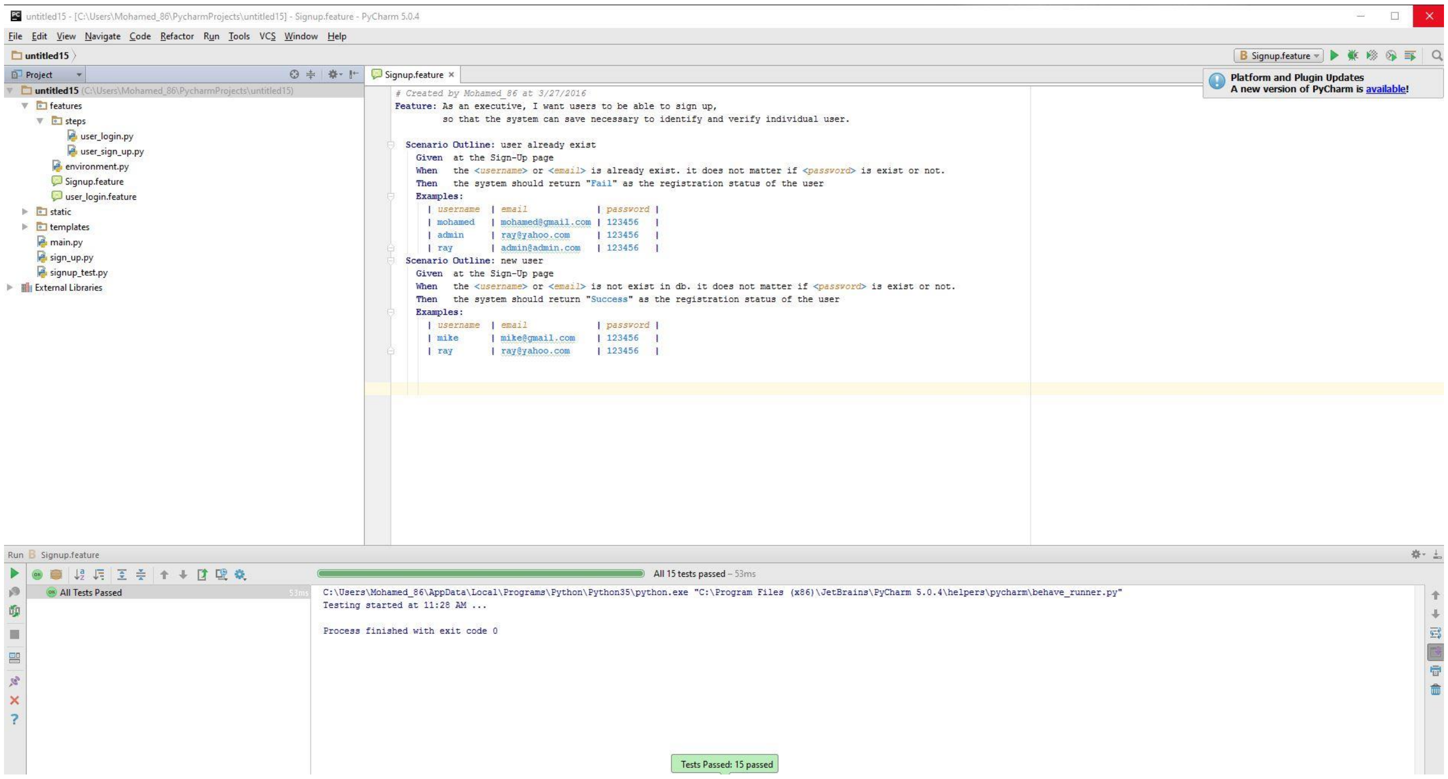


Coverage for **signup_test.py** : 100%
19 statements | 19 run | 0 missing | 0 excluded

```
import main
import unittest


class UserSignupTestCase(unittest.TestCase):
    def setUp(self):
        main.app.config['TESTING'] = True
        self.app = main.app.test_client()

    def signUp(self, inputName, inputEmail):
        return self.app.post('/signUp', data=dict(
            inputName = inputName,
            inputEmail = inputEmail
        ),follow_redirects=True)

    def test_signUp_ok(self):
        rv = self.signUp("ray", "ray@gmail.com")
        assert b'success'

    def test_signUp_not_ok(self):
        rv = self.signUp("admin", "admin@admin.com")
        assert b'fail'

    def test_signUp_not_ok1(self):
        rv = self.signUp("ray", "admin@admin.com")
        assert b'fail'

if __name__ == '__main__':
    unittest.main()
```

« index   coverage.py v4.0.3, created at 2016-04-10 11:14

BDD test:

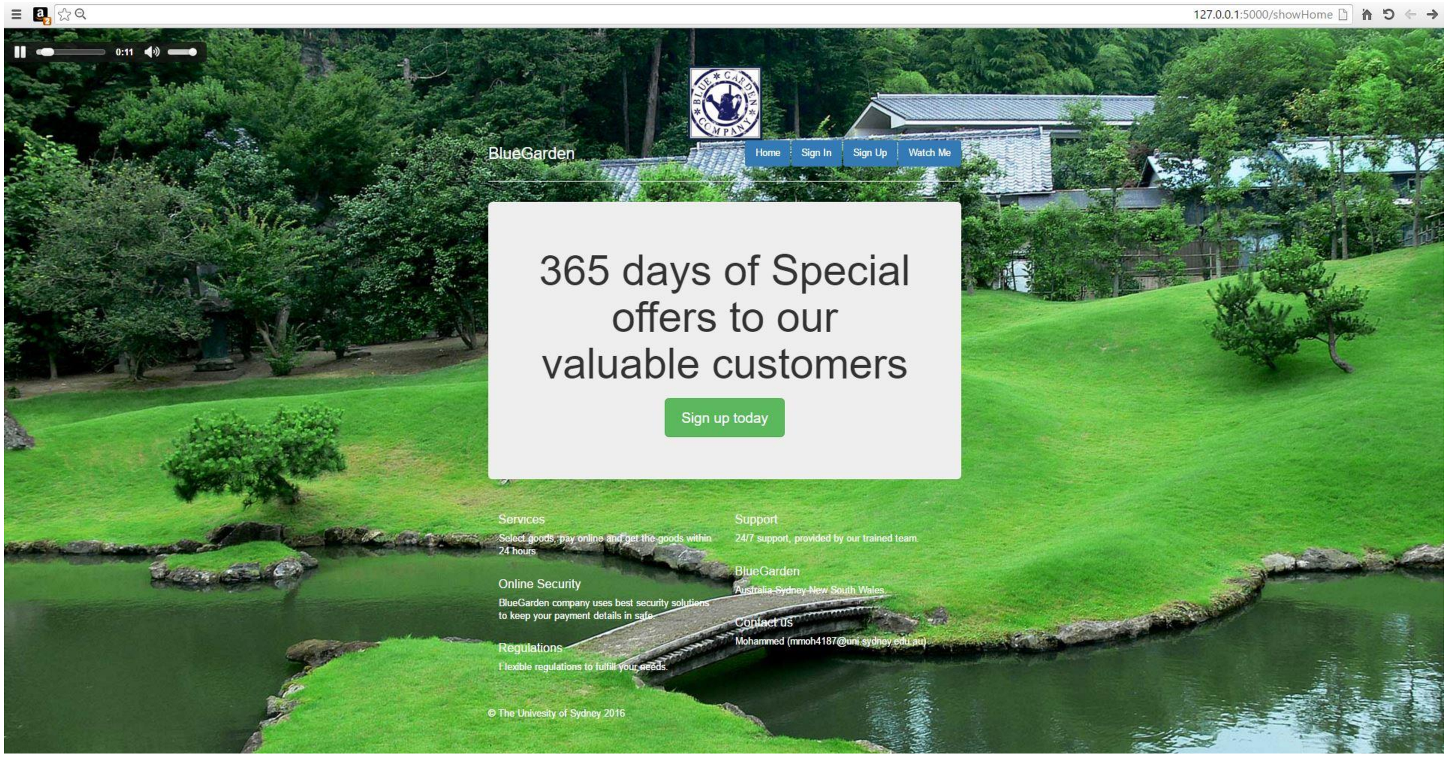


```
# Created by Mohamed_86 at 3/27/2016
Feature: As an executive, I want users to be able to sign up,
         so that the system can save necessary to identify and verify individual user.

  Scenario Outline: user already exist
    Given  at the Sign-Up page
    When   the <username> or <email> is already exist. it does not matter if <password> is exist or not.
    Then   the system should return "Fail" as the registration status of the user
    Examples:
      | username  | email              | password |
      | mohamed   | mohamed@gmail.com | 123456   |
      | admin     | ray@yahoo.com      | 123456   |
      | ray       | admin@admin.com    | 123456   |
  Scenario Outline: new user
    Given  at the Sign-Up page
    When   the <username> or <email> is not exist in db. it does not matter if <password> is exist or not.
    Then   the system should return "Success" as the registration status of the user
    Examples:
      | username  | email              | password |
      | mike      | mike@gmail.com     | 123456   |
      | ray       | ray@yahoo.com      | 123456   |
```

```
C:\Users\Mohamed_86\AppData\Local\Programs\Python\Python35\python.exe "C:\Program Files (x86)\JetBrains\PyCharm 5.0.4\helpers\pycharm\behave_runner.py"
Testing started at 11:28 AM ...

Process finished with exit code 0
```

untitled15 - [C:\Users\Mohamed_86\PycharmProjects\untitled15] - Signup.feature - PyCharm 5.0.4
File Edit View Navigate Code Refactor Run Tools VCS Window Help
untitled15
Project
untitled15 (C:\Users\Mohamed_86\PycharmProjects\untitled15)
features
steps
user_login.py
user_sign_up.py
environment.py
Signup.feature
user_login.feature
static
templates
main.py
sign_up.py
signup_test.py
External Libraries
Signup.feature
Platform and Plugin Updates
A new version of PyCharm is available!
Run Signup.feature
All 15 tests passed – 53ms
All Tests Passed 53ms
Tests Passed: 15 passed

Software output:Home page:



365 days of Special offers to our valuable customers

Services

Select goods, pay online and get the goods within 24 hours.

Online Security

BlueGarden company uses best security solutions to keep your payment details in safe.

Regulations

Flexible regulations to fulfill your needs.

Support

24/7 support, provided by our trained team.

BlueGarden

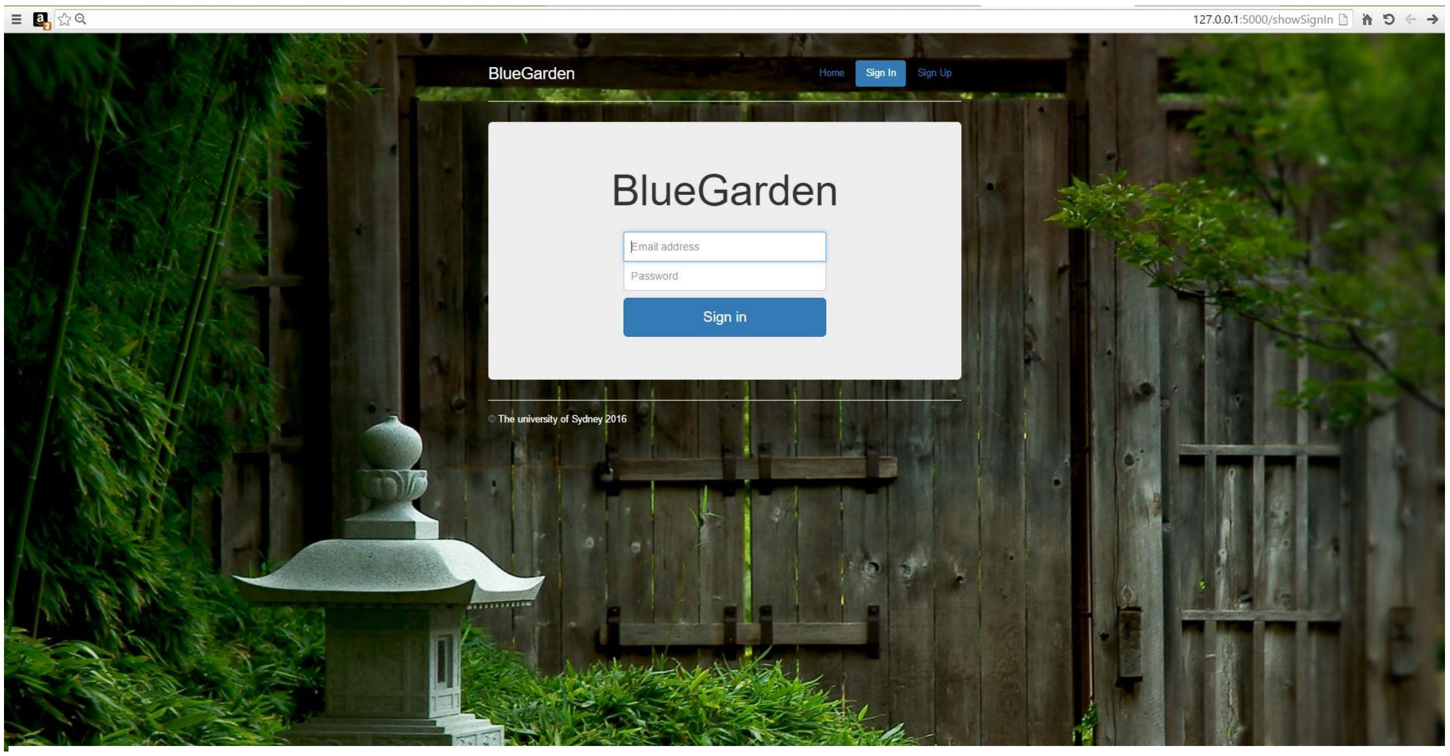
Australia-Sydney-New South Wales.

Contact us

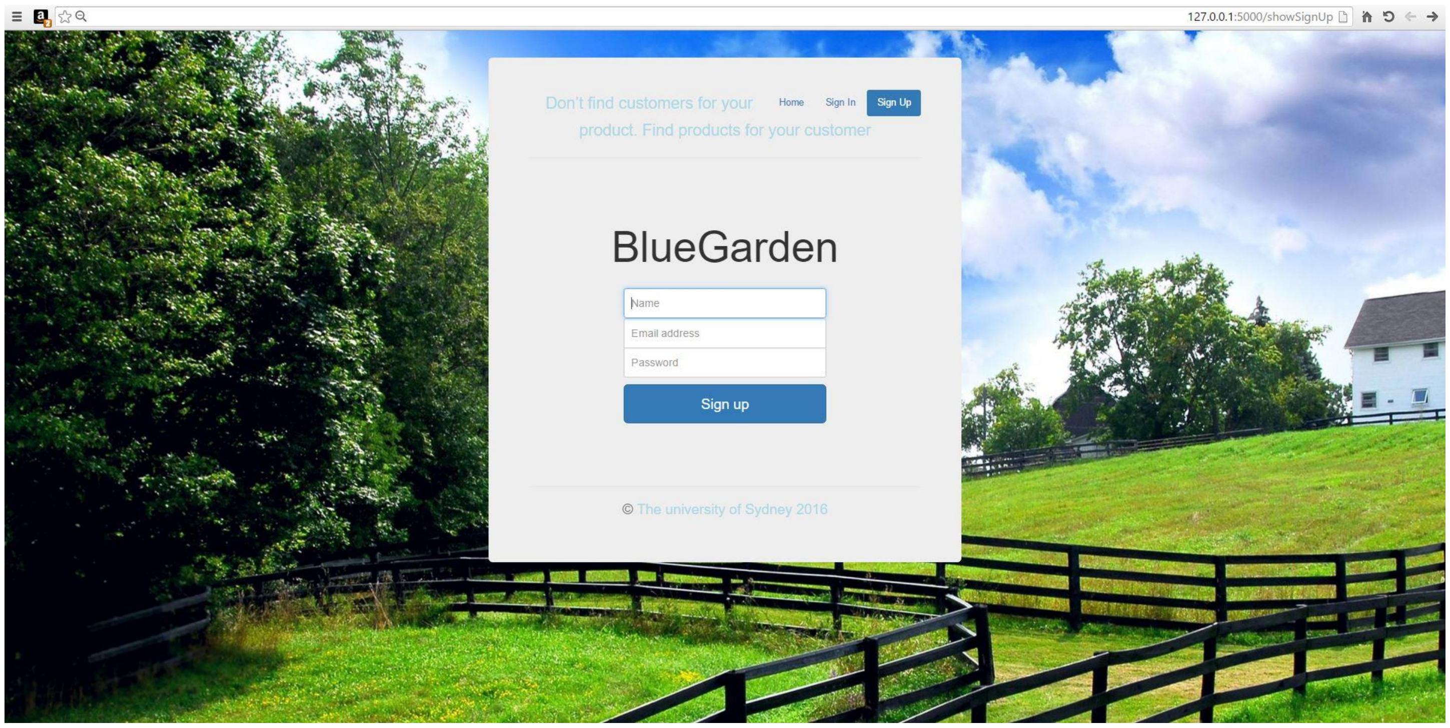
Mohammed (mmoh4187@uni.sydney.edu.au)

© The Univesity of Sydney 2016

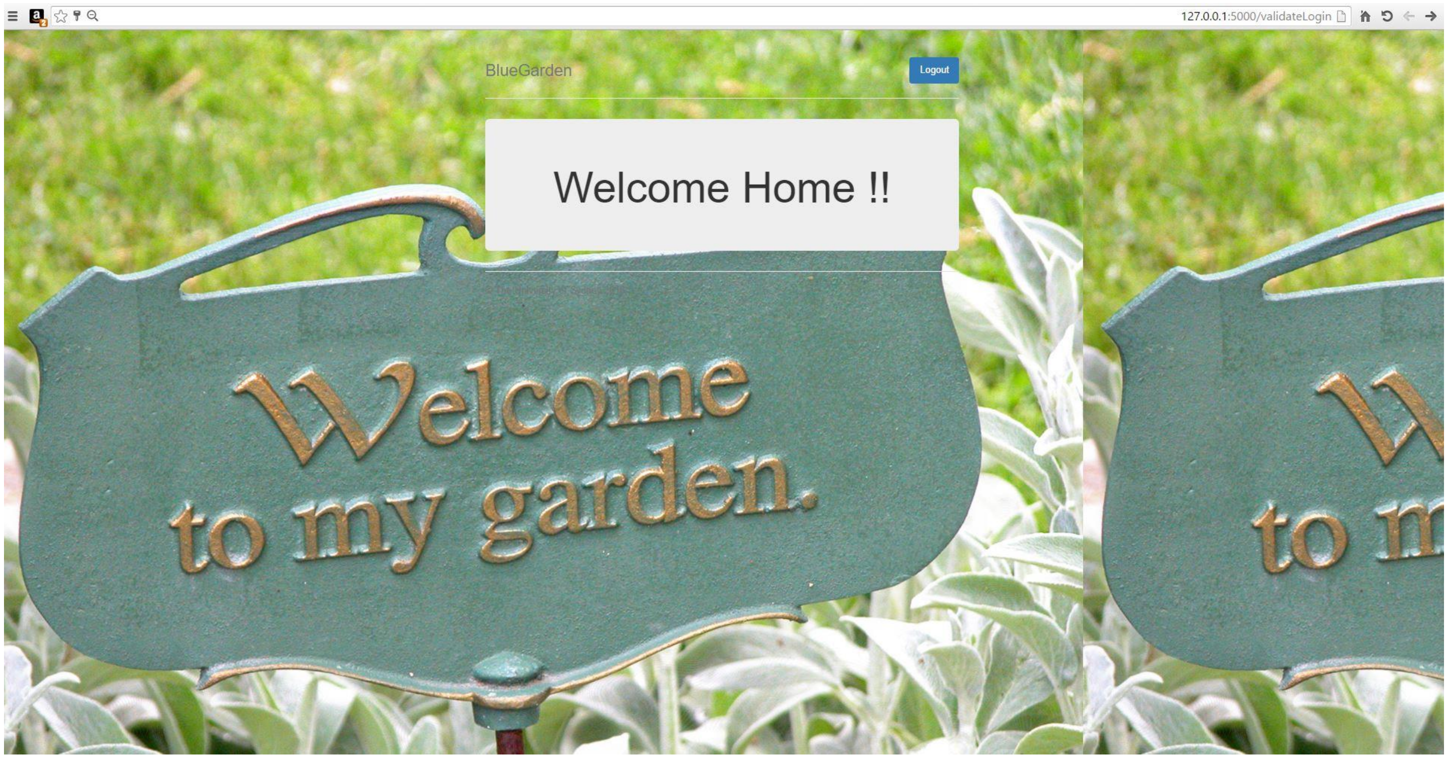
Sign in:



Sign up:

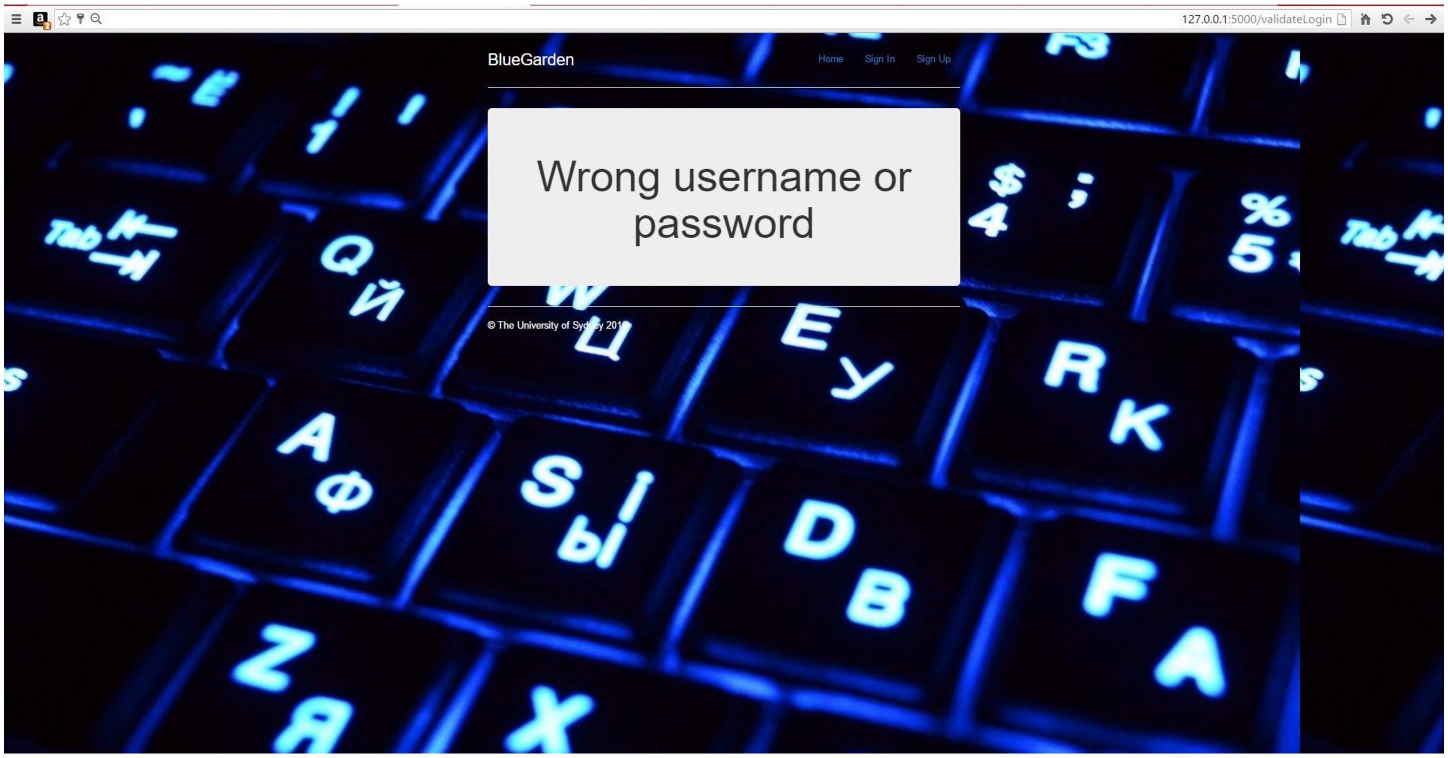

127.0.0.1:5000/showSignUp
Don't find customers for your product. Find products for your customer
Home
Sign In
Sign Up
BlueGarden
Name
Email address
Password
Sign up
© The university of Sydney 2016

User Home



Welcome Home !!

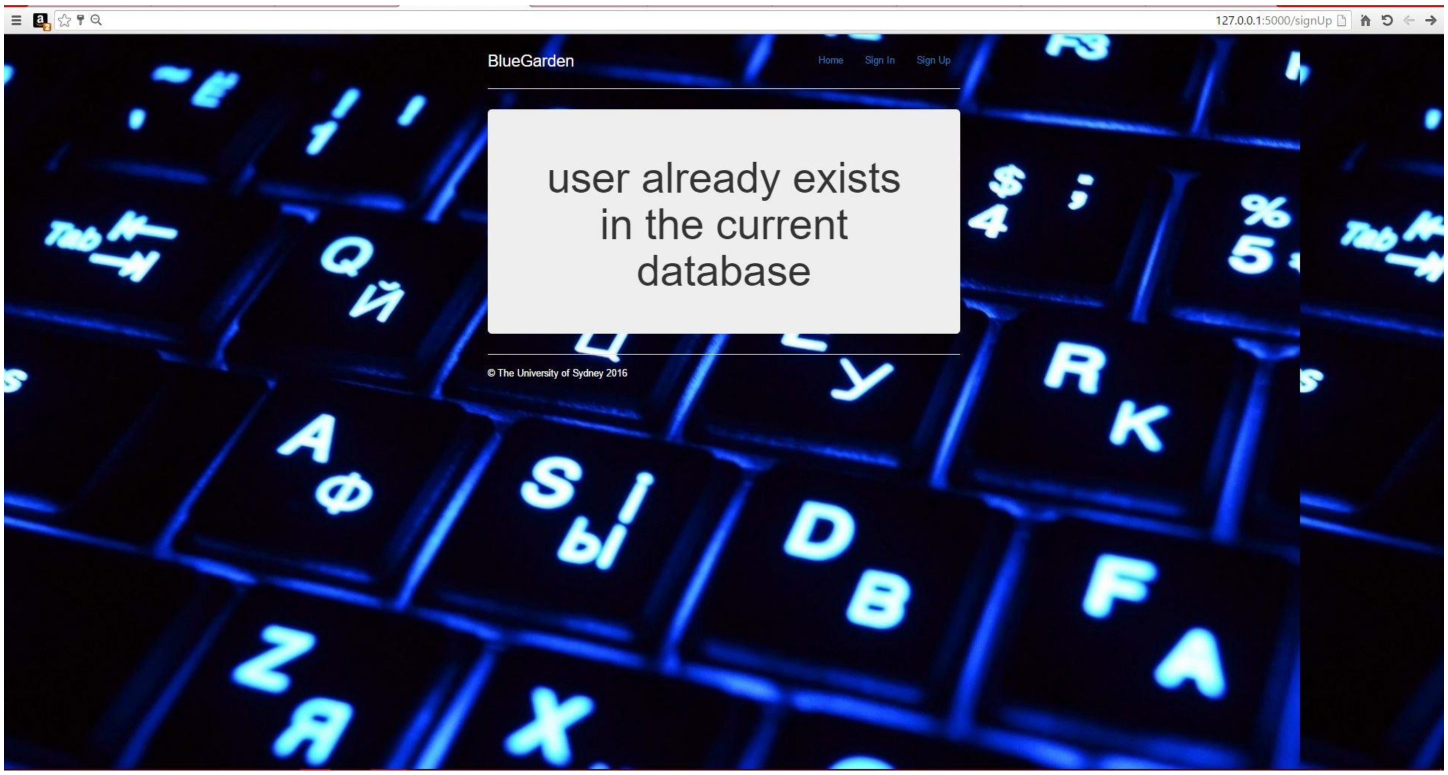
Error (wrong pass or username, unauthorized access):



Wrong username or password

© The University of Sydney 2015

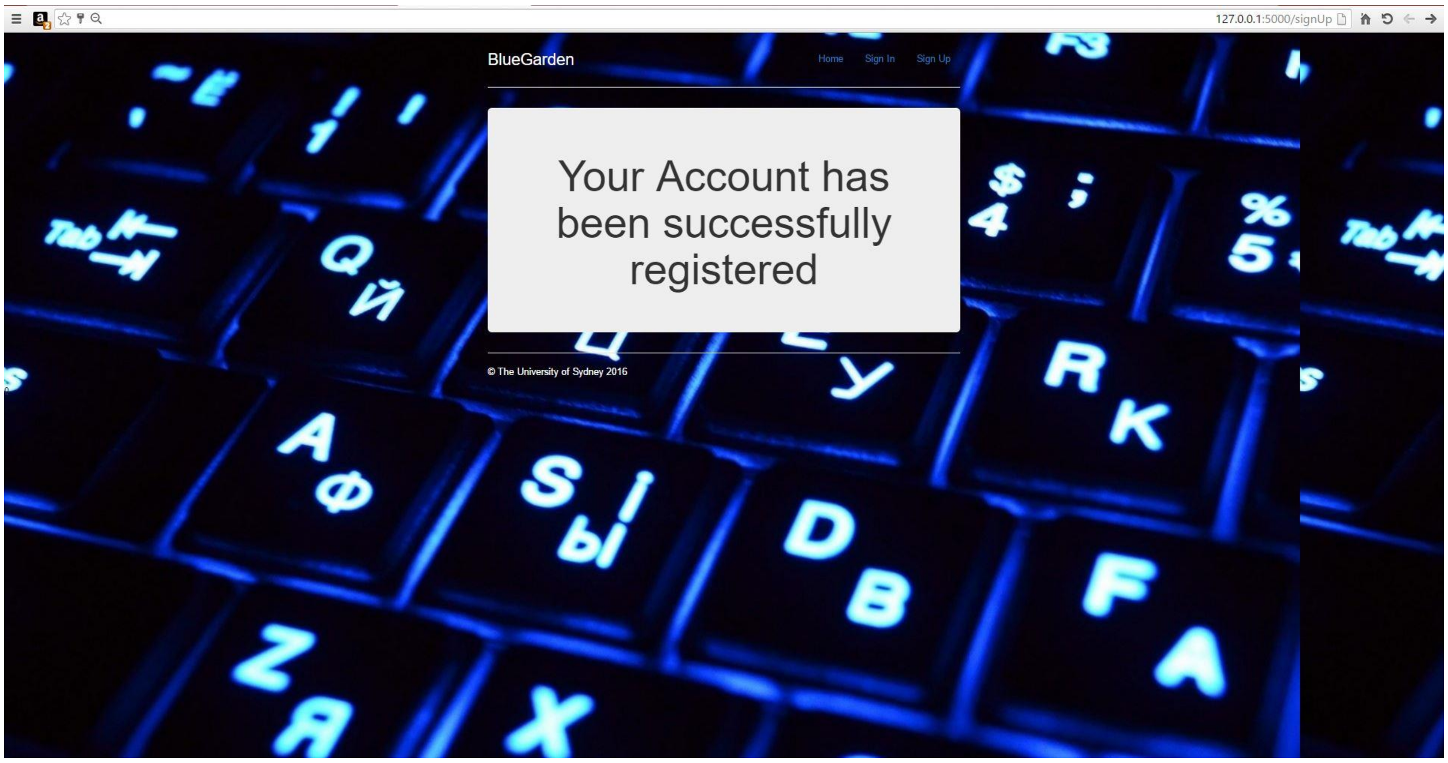
User already exists:



user already exists in the current database

© The University of Sydney 2016

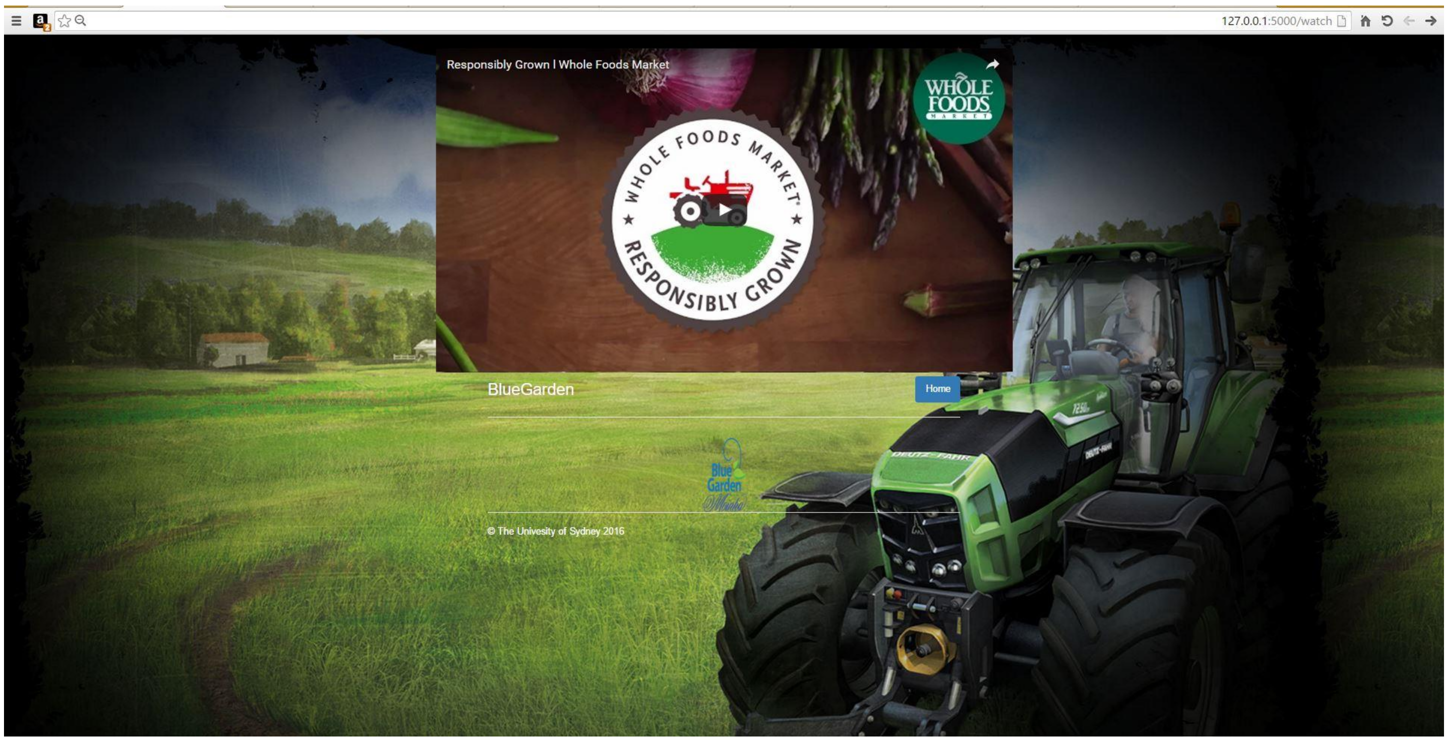
The system registers the user successfully:



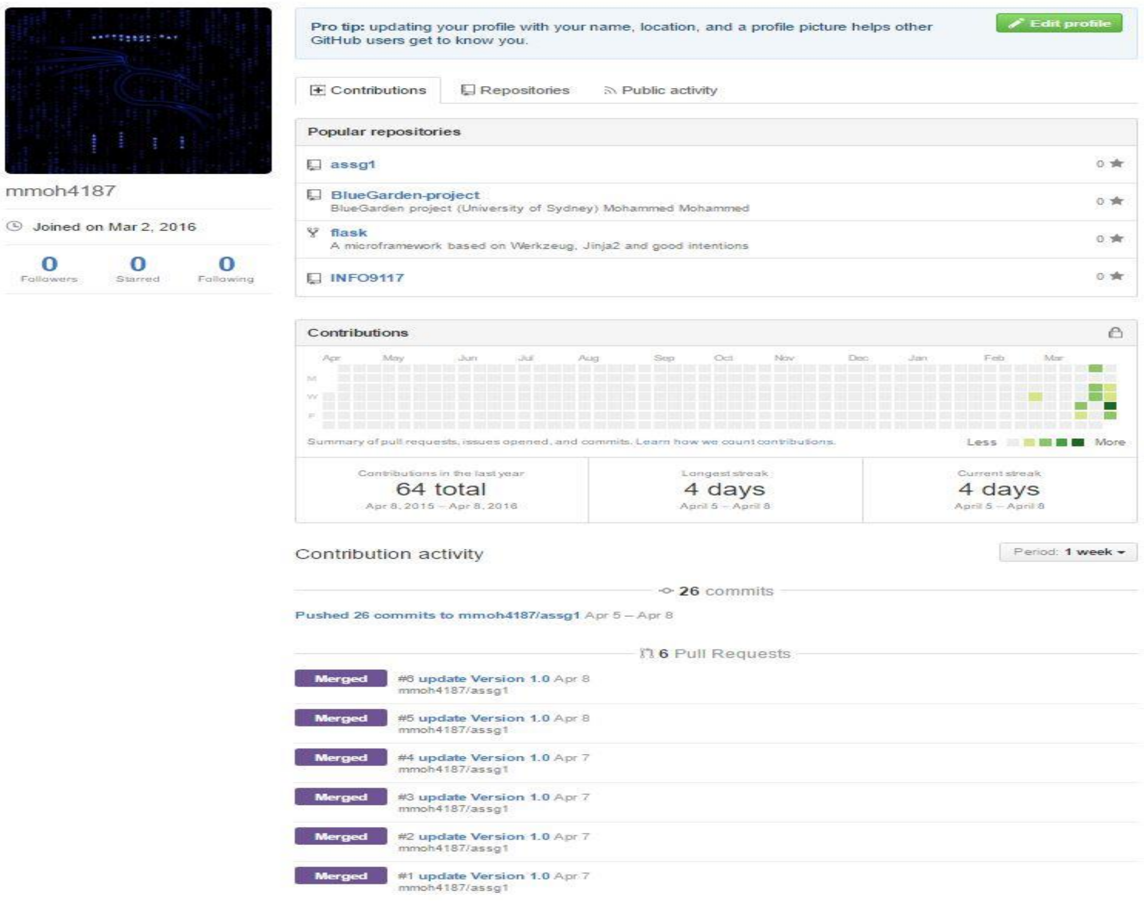
Your Account has been successfully registered

© The University of Sydney 2016

Video page:



GIT Statistics:



mmoh4187

Joined on Mar 2, 2016

0 Followers 0 Starred 0 Following

Pro tip: updating your profile with your name, location, and a profile picture helps other GitHub users get to know you.

Edit profile

Contributions Repositories Public activity

Popular repositories

assg1	0 ★
BlueGarden-project BlueGarden project (University of Sydney) Mohammed Mohammed	0 ★
flask A microframework based on Werkzeug, Jinja2 and good intentions	0 ★
INFO9117	0 ★

Contributions

Summary of pull requests, issues opened, and commits. Learn how we count contributions. Less More

Contributions in the last year	Longest streak	Current streak
64 total	4 days	4 days
Apr 8, 2015 – Apr 8, 2016	April 5 – April 8	April 5 – April 8

Contribution activity

Period: 1 week

26 commits

Pushed 26 commits to mmoh4187/assg1 Apr 5 – Apr 8

6 Pull Requests

Merged #6 update Version 1.0 Apr 8
mmoh4187/assg1

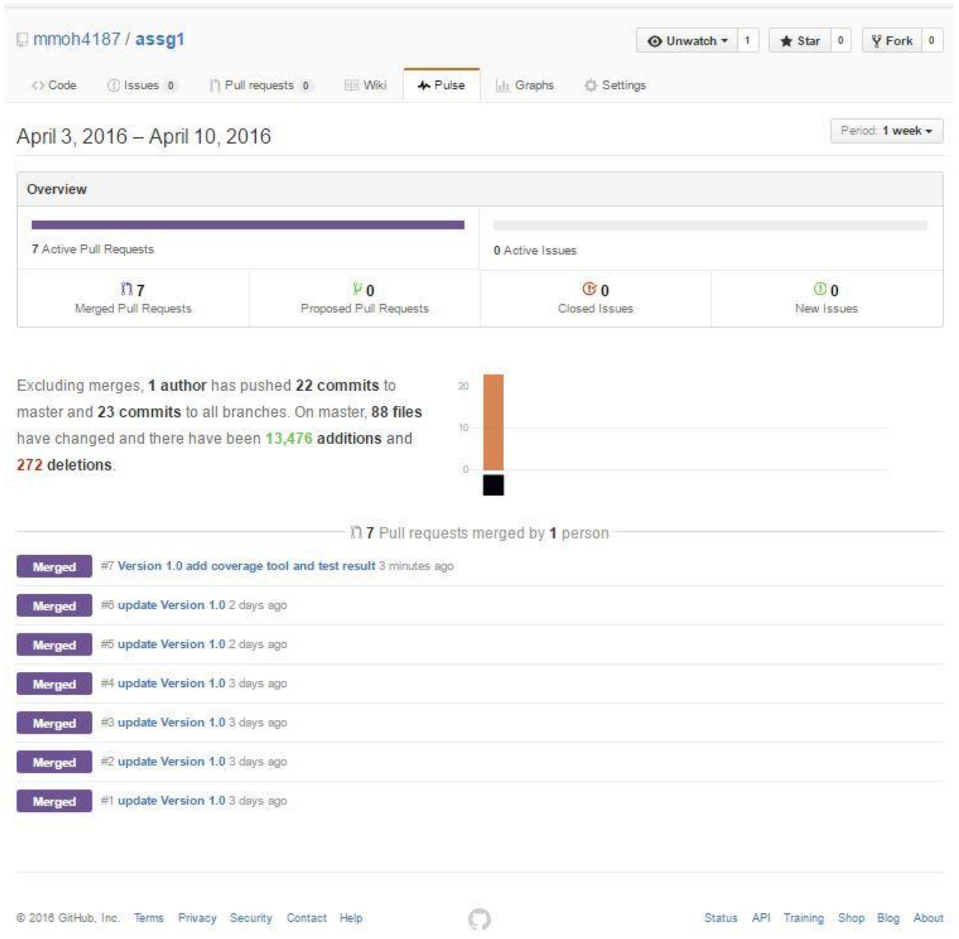
Merged #5 update Version 1.0 Apr 8
mmoh4187/assg1

Merged #4 update Version 1.0 Apr 7
mmoh4187/assg1

Merged #3 update Version 1.0 Apr 7
mmoh4187/assg1

Merged #2 update Version 1.0 Apr 7
mmoh4187/assg1

Merged #1 update Version 1.0 Apr 7
mmoh4187/assg1



April 3, 2016 – April 10, 2016

Overview

7 Active Pull Requests

0 Active Issues

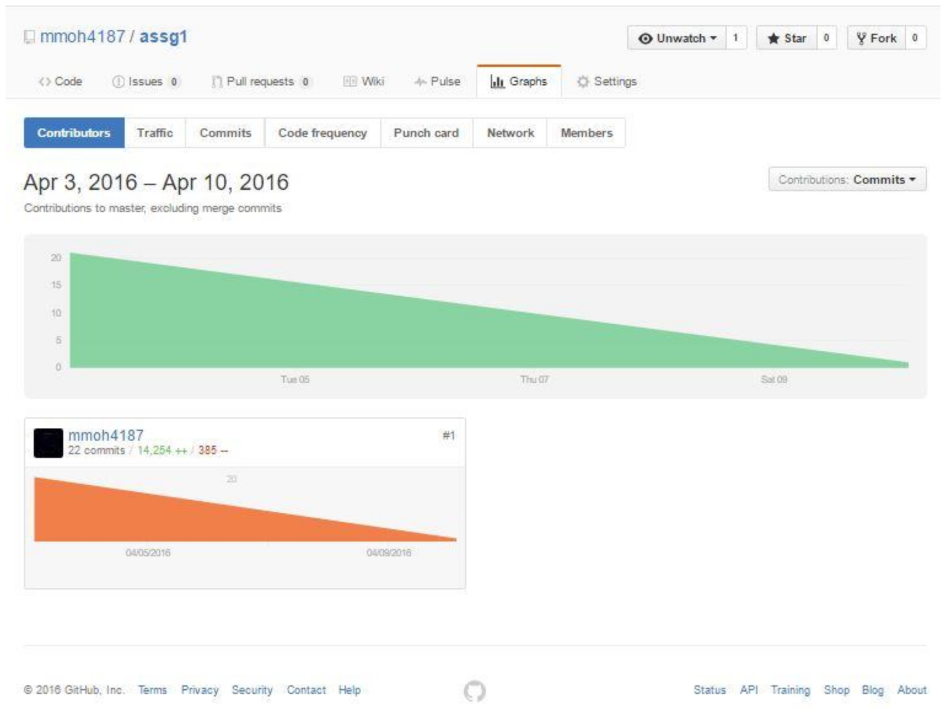
7	0	0	0
Merged Pull Requests	Proposed Pull Requests	Closed Issues	New Issues

Excluding merges, **1 author** has pushed **22 commits** to master and **23 commits** to all branches. On master, **88 files** have changed and there have been **13,476 additions** and **272 deletions**.

7 Pull requests merged by **1** person

- Merged #7 **Version 1.0 add coverage tool and test result** 3 minutes ago
- Merged #6 **update Version 1.0** 2 days ago
- Merged #5 **update Version 1.0** 2 days ago
- Merged #4 **update Version 1.0** 3 days ago
- Merged #3 **update Version 1.0** 3 days ago
- Merged #2 **update Version 1.0** 3 days ago
- Merged #1 **update Version 1.0** 3 days ago

© 2016 GitHub, Inc. Terms Privacy Security Contact Help Status API Training Shop Blog About


mmoh4187 / assg1
Unwatch 1 Star 0 Fork 0
Code Issues 0 Pull requests 0 Wiki Pulse Graphs Settings
Contributors Traffic Commits Code frequency Punch card Network Members
Contributions: Commits
20
15
10
5
0
Tue 05
Thu 07
Sat 09

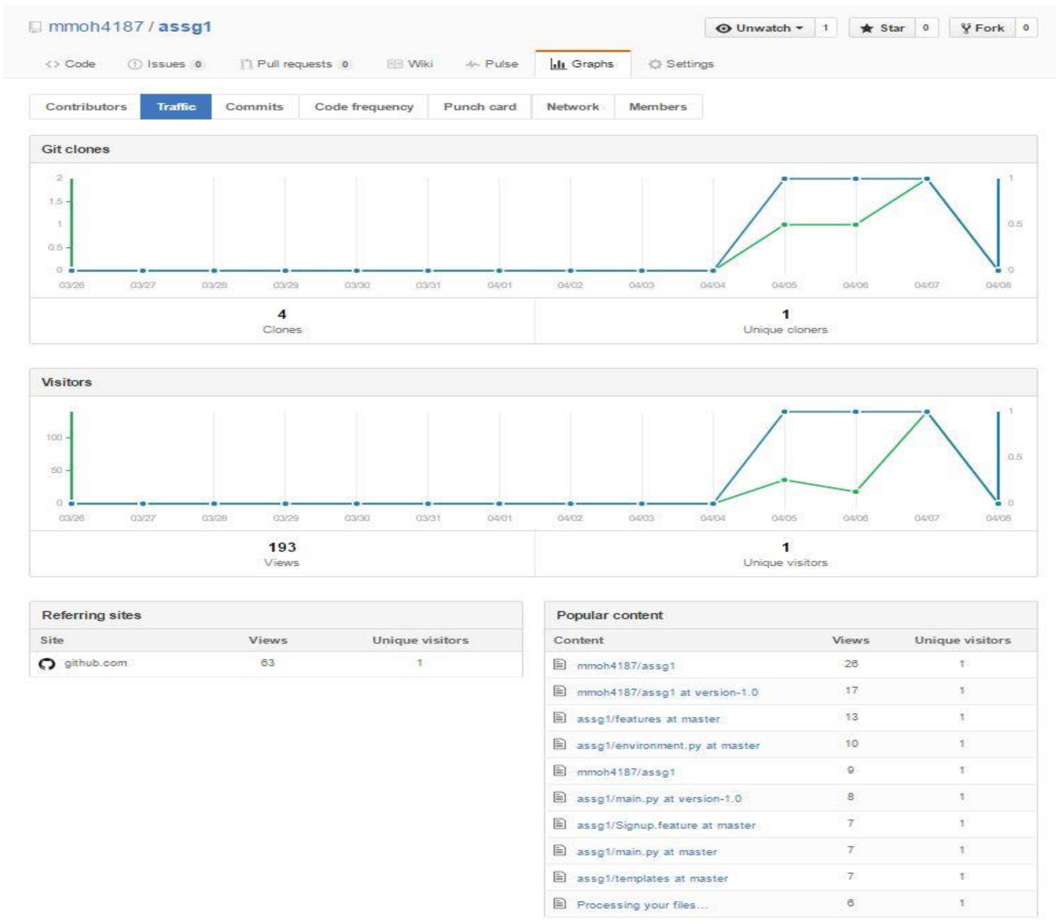
Apr 3, 2016 – Apr 10, 2016

Contributions to master, excluding merge commits

mmoh4187 #1
22 commits / 14,254 ++ / 385 --

© 2016 GitHub, Inc. Terms Privacy Security Contact Help Status API Training Shop Blog About

-



Git clones

4 Clones | **1** Unique cloners

Visitors

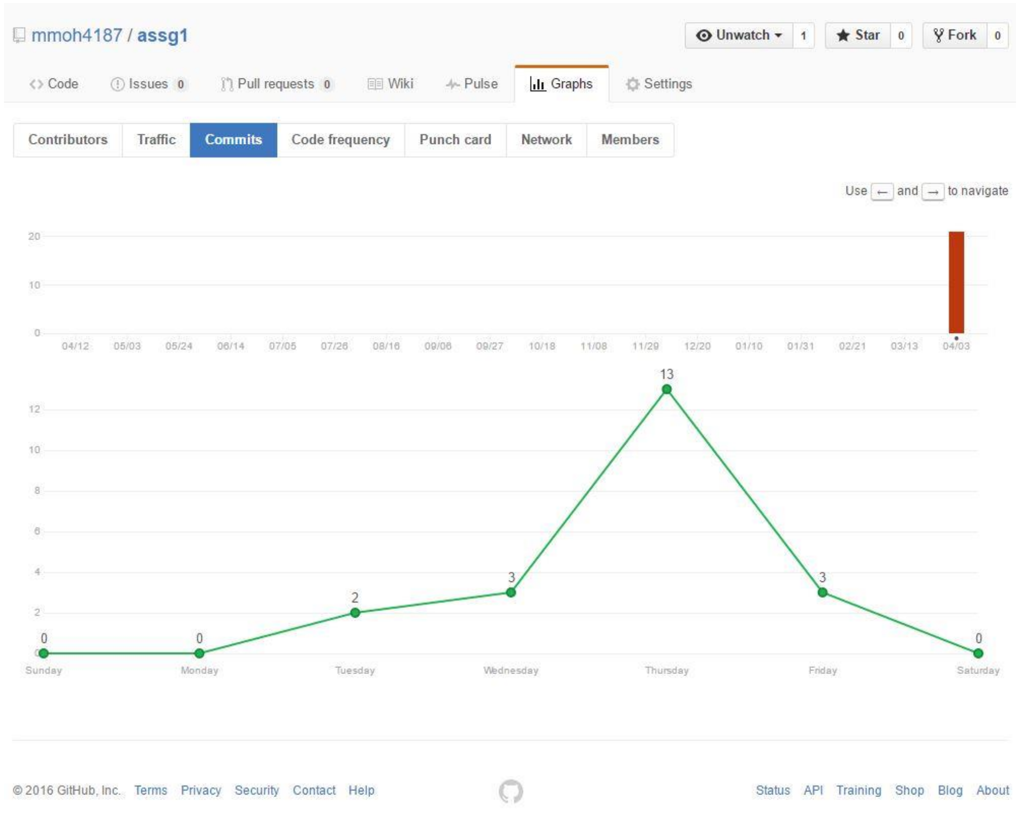
193 Views | **1** Unique visitors

Referring sites

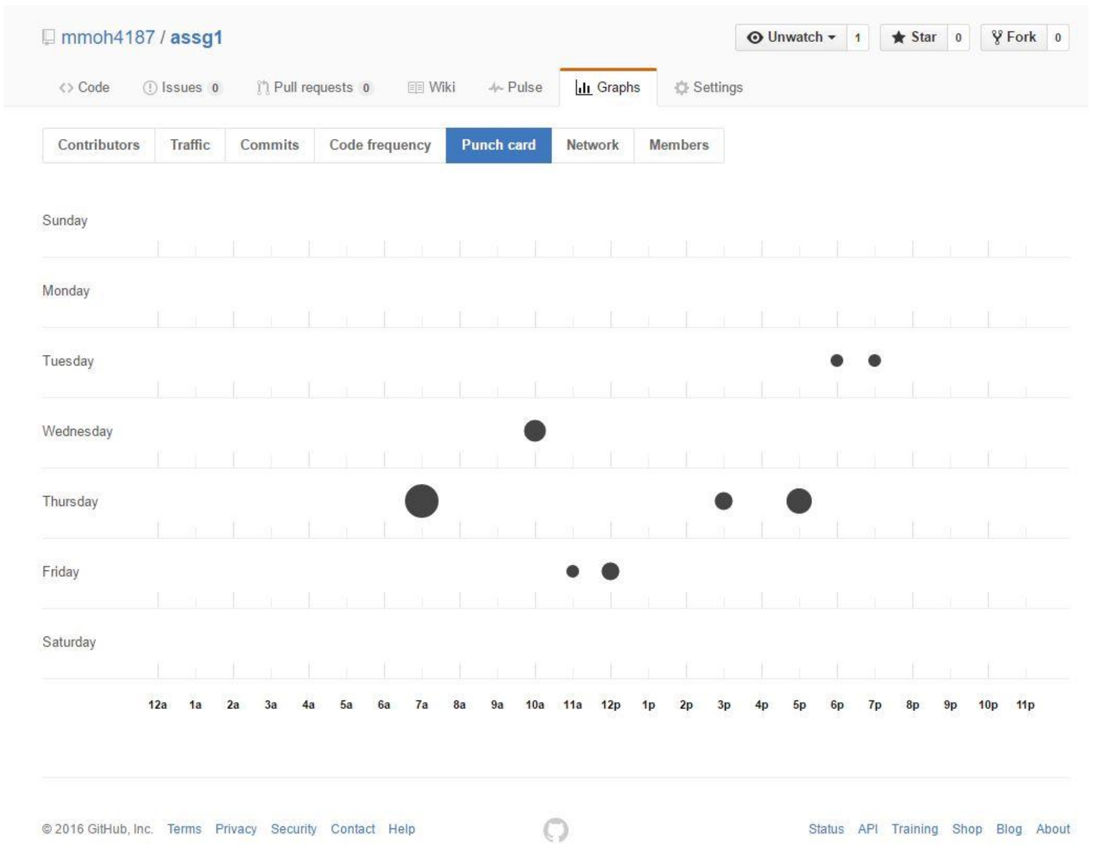
Site	Views	Unique visitors
github.com	63	1

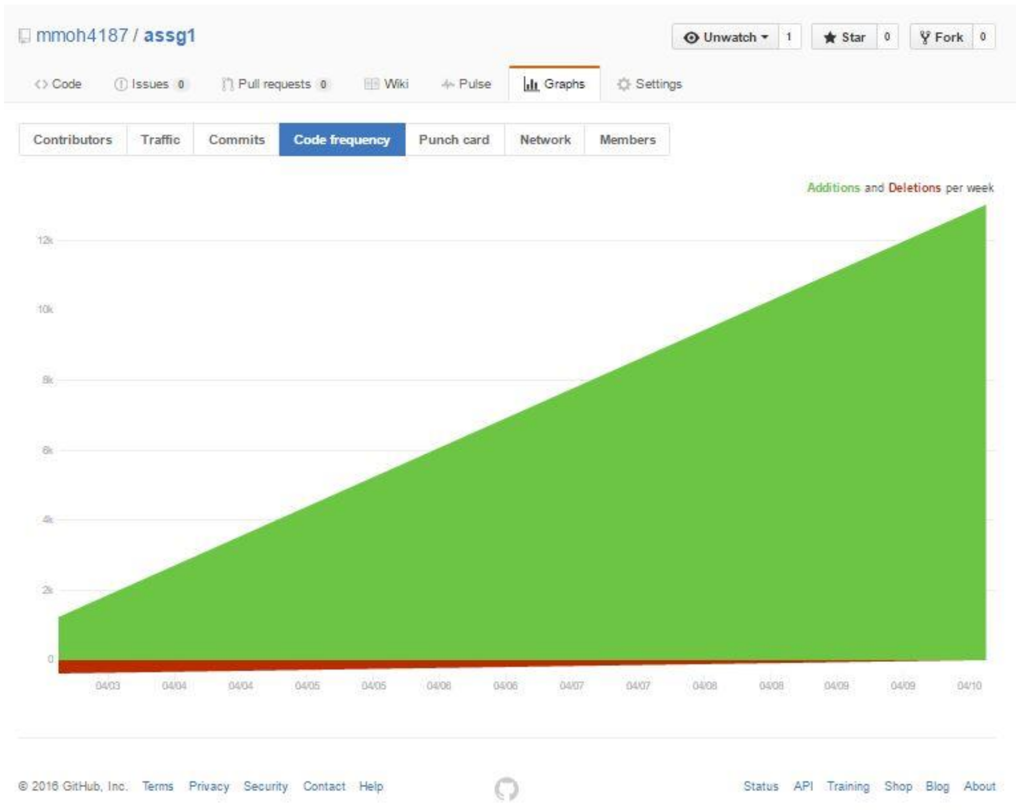
Popular content

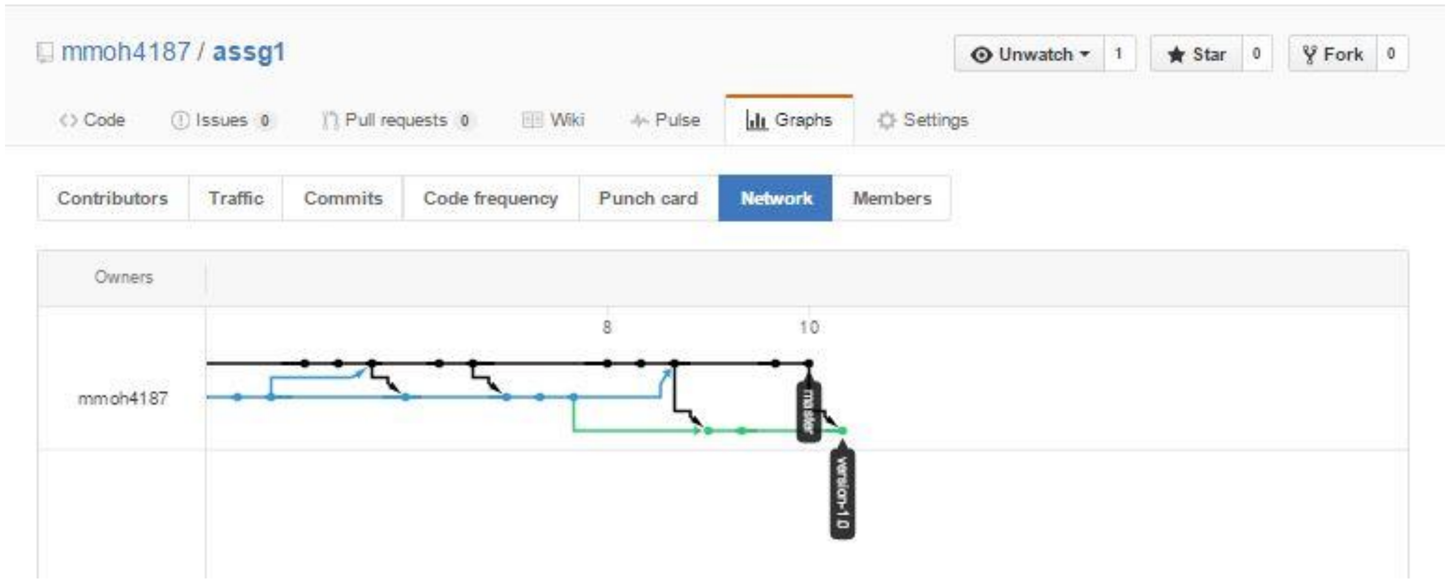
Content	Views	Unique visitors
mmoh4187/assg1	26	1
mmoh4187/assg1 at version-1.0	17	1
assg1/features at master	13	1
assg1/environment.py at master	10	1
mmoh4187/assg1	9	1
assg1/main.py at version-1.0	8	1
assg1/Signup.feature at master	7	1
assg1/main.py at master	7	1
assg1/templates at master	7	1
Processing your files...	6	1



© 2016 GitHub, Inc. Terms Privacy Security Contact Help Status API Training Shop Blog About


mmoh4187 / assg1
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Wiki
Pulse
Graphs
Settings
Contributors
Traffic
Commits
Code frequency
Punch card
Network
Members
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
12a 1a 2a 3a 4a 5a 6a 7a 8a 9a 10a 11a 12p 1p 2p 3p 4p 5p 6p 7p 8p 9p 10p 11p
© 2016 GitHub, Inc. Terms Privacy Security Contact Help
Status API Training Shop Blog About


mmoh4187 / assg1
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Wiki
Pulse
Graphs
Settings
Contributors
Traffic
Commits
Code frequency
Punch card
Network
Members
Additions and Deletions per week
12k
10k
8k
6k
4k
2k
0
04/03 04/04 04/04 04/05 04/05 04/06 04/06 04/07 04/07 04/08 04/08 04/09 04/09 04/10
© 2016 GitHub, Inc. Terms Privacy Security Contact Help
Status API Training Shop Blog About


mmoh4187 / assg1
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Wiki
Pulse
Graphs
Settings
Contributors
Traffic
Commits
Code frequency
Punch card
Network
Members
Owners
8
10
mmoh4187
master
version-1.0