

Software Engineering CSC648/848 Spring 2019
~ Project Pegasus ~
Team Number 104

Brenna Tirumalashetty - brennag@mail.sfsu.edu

Omar Alaniz

Zachary Prince

Quan Lu

Adan Guanlao

Fatma Khan

Wafi Mohamed

Milestone 4

May 7th, 2019

| Revision Number | Date | Notes |
|-----------------|---------------|---------------------|
| 1 | May 7th, 2019 | Original submission |
| 2 | | |
| 3 | | |

Product Summary

Pegasus is a unique housing website centered on fostering a student community. Students who are registered on the Pegasus website will be able to not only find a great place to live, but also interact with other Pegasus users in their neighborhoods. Students can use the site to find apartments and roommates, learn about neighborhoods, hear about local events, and help each other with moving and other services. We believe that the Pegasus website will allow students who live off campus to form and maintain connections with other students.

Committed Functions

Post Listings. Landlords can post listings available for rent that include information about a residence, including price, size, type of residence, pet policy, current occupants, amenities, and photos. The site administrator will approve or reject each listing before it can be posted.

Search and Filter. Student users will be able to search listings by city, and filter the results based on size and type of residence, price range, amenities, utilities, current number of tenants, and accessibility.

Registration. Students with an SFSU email address will be able to register, create an account and profile, browse listings, contact landlords and other registered students, and access and post to the student neighbor network. Landlords will be able to register, create an account and profile, post listings, manage listings, post open house dates/times, and respond to student inquiries.

Student Neighbor Network. Registered students will be able to access a forum for interacting with other registered students. The forum will be organized by neighborhood, and will allow students to post general neighborhood information, neighborhood events and services, items for sale, etc. Registered students may create posts, read posts, and reply to posts. Posts will be moderated for content by the administrator.

Administrative Tasks. The website admin will utilize a dashboard to keep track of approved listings and listings that need approval. The admin will also see recent posts to the network that have been flagged for inappropriate content, with the option to remove the posting. The dashboard will contain an activity log that displays all recent admin activity.

Messaging. Registered students will contact landlords or other registered students through the website, without revealing their email addresses.

URL: <http://ec2-18-224-150-8.us-east-2.compute.amazonaws.com/web/demo>

QA Test Plan

Unit Test

While there is quite a long list of functions and features that should be tested via unit tests, for the purposes of this document, we have selected two to present—searching/filtering for listings, and user registration. For searching and filtering posts, we want to verify two major points. One is accuracy; the filters we apply must result in correct output. The other is performance. Searching for all apartments in San Francisco should not take any longer than a few seconds to execute.

In terms of hardware setup, not much is required. A Linux machine with our code repository loaded into it is all that is required. This can be a physical or virtual machine. In terms of software, we require that the system support Python3.6. It need not have Python 3.6 installed—only that it *supports* it being installed, i.e. any system-level libraries like GLIBC need to be pre-configured. In most out-of-the-box Linux OS installations, this is pre-configured.

For our testing framework, we opted to use py.test, as our software stack is Python-based. An example test case for search is given below:

```
34 # Verify listing page generates responses relatively quickly-
35 @pytest.mark.django_db-
36 def test_listing_response(rf):-
37     start = time()-
38     request = rf.post('/demo/listings', {'city': 'San Francisco'})-
39     response = views.listing(request)-
40     assert response.status_code == 200-
41     end = time()-
42     runtime = end - start-
43     logging.info("Listing response time: %s" % runtime)-
44     assert (runtime < 5)-
```

NORMAL test_cases.py

This particular test case is for testing responsiveness for the listing page: it measures the runtime between the server receiving the request and generating the results. This process can be complex—it requires contacting the database for the listings, and then passing those addresses in to the Google Maps API to generate the latitude and longitude for those values to be able to populate the map properly. An example of this test in action is appended below:

```

(venv) 20:43:38 森 /opt/project_pegasus/django_sites/pegasus
$ pytest
===== test session starts =====
platform linux -- Python 3.6.7, pytest-4.4.1, py-1.8.0, pluggy-0.9.0 -- /opt/project_pegasus/django_sites/pegasus/venv/bin/python3.6
cachedir: .pytest_cache
Django settings: pegasus.settings (from ini file)
rootdir: /opt/project_pegasus/django_sites/pegasus, inifile: pytest.ini
plugins: django-3.4.8
collected 3 items

demo/test_cases.py::test_tautology PASSED [ 33%]
demo/test_cases.py::test_user_creation PASSED [ 66%]
demo/test_cases.py::test_listing_response
----- live log call -----
views.py 114 WARNING The 'photo' attribute has no file associated with it.
views.py 114 WARNING The 'photo' attribute has no file associated with it.
views.py 114 WARNING The 'photo' attribute has no file associated with it.
test_cases.py 43 INFO Listing response time: 1.8275625705718994
PASSED [100%]

===== 3 passed in 3.60 seconds =====
(venv) 21:51:17 森 /opt/project_pegasus/django_sites/pegasus
$

```

Integration Test

Our integration testing covers user stories S1, L1, A1 (one example of a student, one example of a landlord, and the site administrator). The testing will be completed on Firefox and Chrome browsers. Testing will be done using the following URL:

<http://ec2-18-224-150-8.us-east-2.compute.amazonaws.com/web/demo>

HW and SW setup

1. Server Host: AWS
2. Operating System: Linux Ubuntu
3. Database: PostgreSQL
4. Web Server: Apache
5. Server-side Language: Python
6. Framework: Django
7. IDE: PyCharm, VSCode
8. Supported Browsers: Chrome, Firefox
9. Additional technologies:
 - a. Google Maps API

Results of our latest round of integration testing demonstrate what work still needs to be done. Examples of the integration testing are shown below. Additional tests can be found in the appendix at the end of this document.

| Integration Test 1 - User Story S1 - Student - Firefox | |
|--|--|
| Test Case ID: S_01 | Test Case Description: Create Student Account and log in |
| Prerequisites: | None |

| Date tested | Scenario | Expected Results | Pass / Fail |
|-------------|------------------------------------|--|-------------|
| 05/06/2019 | Access Pegasus Home | Homepage with search field shown | Pass |
| 05/06/2019 | Click on "Sign up" | Redirect to sign up page | Pass |
| 05/06/2019 | Enter Information and click submit | See account verification page, email sent to account | Pass |
| 05/06/2019 | Check email and click on link | Confirmation of registration | Pass |
| 05/06/2019 | Return to homepage and sign in | Signs into account, go to search listings page | Pass |

| Integration Test 1 - User Story S1 - Student - Chrome | | | |
|---|------------------------------------|--|-------------|
| Test Case ID: S_01 | | Test Case Description: Create Student Account and log in | |
| Prerequisites: | | None | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 5/07/2019 | Access Pegasus Home | Homepage with search field shown | Pass |
| 5/07/2019 | Click on "Sign up" | Redirect to sign up page | Pass |
| 5/07/2019 | Enter Information and click submit | See account verification page, email sent to account | Pass |
| 5/07/2019 | Check email and click on link | Confirmation of registration | Pass |
| 5/07/2019 | Return to homepage and sign in | Signs into account, go to search listings page | Pass |

| Integration Test 2 - User Story S1 - Student - Firefox | |
|--|---|
| Test Case ID: S_02 | Test Case Description: Search listings as a student |

| Prerequisites: | | Student has account and is logged in | |
|----------------|------------------------------|---|---|
| Date tested | Scenario | Expected Results | Pass / Fail |
| 05/06/2019 | Access Pegasus Home | Homepage with search field shown | Pass |
| 05/06/2019 | Enter city in search field | Redirect to listings page with appropriate map and pins | Pass |
| 05/06/2019 | Filter by number of bedrooms | Only correct results displayed | Fail - filtering works, but all listings have the same number of bedrooms except for 1 (not properly connected to database) |
| 05/06/2019 | Filter by price range | Only correct results displayed | Fail - Searched price 3300, resulted in FieldError |
| 05/06/2019 | Filter by home type | Only correct results displayed | Pass |
| 05/06/2019 | Select a listing | Redirect to listing page | Pass |
| 05/06/2019 | View map | Map centers on residence address | Pass |
| 05/06/2019 | Find directions on map | Correctly calculates directions | Fail - Not implemented |
| 05/06/2019 | View listing details | All listing details present | Pass |
| 05/06/2019 | View photos slider | Correct photos display | Fail - not yet connected to database |

| Integration Test 2 - User Story S1 - Student - Chrome | | | |
|---|----------|---|--|
| Test Case ID: S_02 | | Test Case Description: Search listings as a student | |
| Prerequisites: | | Student has account and is logged in | |
| Date tested | Scenario | Expected Results | |

| | | | |
|-----------|------------------------------|---|--------------------------------------|
| 5/07/2019 | Access Pegasus Home | Homepage with search field shown | Pass |
| 5/07/2019 | Enter city in search field | Redirect to listings page with appropriate map and pins | Pass |
| 5/07/2019 | Filter by number of bedrooms | Only correct results displayed | Fail-Errors |
| 5/07/2019 | Filter by price range | Only correct results displayed | Pass |
| | Filter by distance | | Not implemented |
| 5/07/2019 | Filter by home type | Only correct results displayed | Not implemented |
| 5/07/2019 | Select a listing | Redirect to listing page | Not implemented |
| 5/07/2019 | View map | Map centers on residence address | Pass |
| 5/07/2019 | Find directions on map | Correctly calculates directions | Fail - Not implemented |
| 5/07/2019 | View listing details | All listing details present | Not implemented |
| 5/07/2019 | View photos slider | Correct photos display | Fail - not yet connected to database |

| Integration Test 5 - User Story S1 - Student - Firefox | | | |
|--|--------------------------------|--|-------------|
| Test Case ID: S_05 | | Test Case Description: Student interaction with student neighbor network | |
| Prerequisites: | | Assumes successful sign-in and account creation | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 05/06/2019 | Student clicks on forum button | Redirect to forum main page, organized by neighborhood | Pass |
| 05/06/2019 | Student selects neighborhood | Redirect to forum for that neighborhood | Pass |

| | | | |
|------------|---|---|-----------------------------|
| 05/06/2019 | View postings | Relevant subcategories and postings should be displayed for that neighborhood | Pass |
| 05/06/2019 | Post something | Student creates a posting, including text of posting, pictures, tags (for example events, security, services, for sale) | Pass - didn't test pictures |
| 05/06/2019 | Comment on postings | Written comments should show up in the posting thread | Pass |
| 05/06/2019 | Click on usernames to view other student profiles | Names of the posters are clickable to that user's profile | Pass |
| 05/06/2019 | Direct messages to other registered students | Click on send message and popup (or chat window?) opens up | Fail - not implemented |

| Integration Test 5 - User Story S1 - Student - Chrome | | | |
|---|--------------------------------|---|----------------|
| Test Case ID: S_05 | | Test Case Description: Student interaction with student neighbor network | |
| Prerequisites: | | Assumes successful sign-in and account creation | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 5/07/2019 | Student clicks on forum button | Redirect to forum main page, organized by neighborhood | Fail-Errors |
| 5/07/2019 | Student selects neighborhood | Redirect to forum for that neighborhood | Not implements |
| 5/07/2019 | View postings | Relevant subcategories and postings should be displayed for that neighborhood | Pass |

| | | | |
|-----------|---|---|------------------------|
| 5/07/2019 | Post something | Student creates a posting, including text of posting, pictures, tags (for example events, security, services, for sale) | Fail-Errors |
| 5/07/2019 | Comment on postings | Written comments should show up in the posting thread | Not implements |
| 5/07/2019 | Click on usernames to view other student profiles | Names of the posters are clickable to that user's profile | Pass |
| 5/07/2019 | Direct messages to other registered students | Click on send message and popup (or chat window?) opens up | Fail - not implemented |

| Integration Test 6 - User Story L1 - Landlord - Firefox | | | |
|---|------------------------------------|---|--|
| Test Case ID: L_01 | | Test Case Description: Create Landlord Account and log in | |
| Prerequisites: | | | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 05/07/2019 | Access Pegasus Home | Homepage with search field shown | Pass |
| 05/07/2019 | Click on "Sign up" | Redirect to sign up page | Pass |
| 05/07/2019 | Enter Information and click submit | See account verification page, email sent to account | Pass |
| 05/07/2019 | Check email and click on link | Confirmation of registration | Pass - Note that if a username or email already exists, it doesn't inform you until you follow the URL |
| 05/07/2019 | Return to homepage and sign in | Signs into account, go to search listings page | Pass |

| Integration Test 6 - User Story L1 - Landlord - Chrome | | | |
|--|------------------------------------|---|-------------------------|
| Test Case ID: L_01 | | Test Case Description: Create Landlord Account and log in | |
| Prerequisites: | | | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 5/07/2019 | Access Pegasus Home | Homepage with search field shown | Pass |
| 5/07/2019 | Click on "Sign up" | Redirect to sign up page | Pass |
| 5/07/2019 | Enter Information and click submit | See account verification page, email sent to account | Fail-cannot submit form |
| 5/07/2019 | Check email and click on link | Confirmation of registration | Fail-above error |
| 05/07/2019 | Return to homepage and sign in | Signs into account, go to search listings page | Fail-above error |

Admin integration testing is still in progress. We experienced a last minute permissions issue with our admin registration that prevented us from completing this test in time for documentation submission.

| Integration Test 9 - User Story A1 - Admin - Firefox | | | |
|--|---|---|-------------|
| Test Case ID: A_01 | | Test Case Description: Admin login and dashboard | |
| Prerequisites: | | Assumes admin account already set up | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| | Access home page and login as admin | Redirects to admin dashboard | |
| | View listings that are pending approval | All listings pending approval should be displayed | |
| | Accept / reject listing | Message should be sent to landlord. If approved, | |

| | | | |
|--|---|--|--|
| | | listing should be ready and searchable | |
| | Click on flagged neighbor network content tab | View all postings that have been flagged by users | |
| | Click on button to remove content | Post should be removed and deleted from the network | |
| | Click on admin log tab | All recent admin activity is displayed (approved/rejected listings, removed content) | |

Beta Testing

Test Objectives

As of the submission of this document, we are not prepared for a beta launch. Our application failed several of the latest integration tests for our core functionality. The plan is to spend until Friday of this week to fix the issues with the site before releasing it for beta testing. At that time, the site will be distributed to friends and family who could be potential users, either as landlords or as students. The website will include all of the basic functionality that we have committed to in this document. Feedback will be collected via user Google survey (<https://forms.gle/tZkhoGuj21FBPtrd9>). Product behavior will be collected by an automatic logging feature.

Code Review

A. Coding style. For Python, we enforce PEP8 coding style. All team members are using Visual Studio Code or Pycharm with a PEP8 plugin. Warning messages are displayed in the IDE any time code does not conform to our agreed-upon style. These styles include:


- Using 4 spaces for indentation, not tabs
- PEP8 style for function names, variable names, and class names
- PEP8 style for whitespace in between functions


For JavaScript and HTML, we enforce the JavaScript style guide and the HTML5 style guide found on w3 schools (https://www.w3schools.com/js/js_conventions.asp , https://www.w3schools.com/html/html5_syntax.asp). Coding style is enforced using the Prettier formatter in Visual Studio Code IDE, which automatically indents our code and makes it easier to read.

B. Example. We have been using GitHub pull requests, along with Slack, to discuss changes that need to be made before merging into dev branch. Below are some examples of branches that were requested to merge, comments on the request, and subsequent incorporation of the changes.

Initial Commit with Comments

```
3  from . import views
4
5  urlpatterns = [
6  +  path('index', views.index, name='index'),
7      path('test/', views.test, name='test'),
8      path('admin/', views.admin, name='admin'),
9  +  path('', views.home, name='home'),
10
11      path('signup/', views.signup, name='signup'),
12      path('add_new_property/', views.create_listing, name='add_new_property'),
13      path('description/', views.description, name='description'),
14      path('manager_profile/', views.manager_profile, name='manager_profile'),
15      path('survey/', views.survey, name='survey'),
16      path('user_profile/', views.user_profile, name='user_profile'),
17  +  path('profile/', views.base_profile, name='profile'),
18
19      # Administrative paths
20      path('login/', views.user_login, name='login'),
```

 aguanlao 12 minutes ago + 🗨️ ...
Might want to change path names. The empty path usually leads to the index. Swapping the paths may make things confusing.

 Reply...

```
10      path('signup/', views.signup, name='signup'),
11      path('add_new_property/', views.create_listing, name='add_new_property'),
12      path('description/', views.description, name='description'),
13      path('manager_profile/', views.manager_profile, name='manager_profile'),
14      path('survey/', views.survey, name='survey'),
15      path('user_profile/', views.user_profile, name='user_profile'),
16  +  path('profile/', views.base_profile, name='profile'),
17
18      # Administrative paths
19      path('login/', views.user_login, name='login'),
```

Commit with Changes Applied

```
1  from django.urls import path, re_path

2  from . import views
3
4  urlpatterns = [
5  +   path('', views.index, name='index'),
6  +   path('listing', views.listing, name='listing'),
7
8      path('add_new_property/', views.create_listing, name='add_new_property'),
9      path('description/', views.description, name='description'),
10     path('manager_profile/', views.manager_profile, name='manager_profile'),
11     path('survey/', views.survey, name='survey'),
12
13     # Administrative paths
14     path('login/', views.user_login, name='login'),
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31     path('<int:listing_id>', views.view_listing, name='view_listing'),
32     path('<int:listing_id>/edit/', views.edit_listing, name='edit_listing'),
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
```



Wafi fatma fd pages #30

fatmakhan95 wants to merge 5 commits into [dev](#) from [wafi-fatma-fd-pages](#)



aguanlao reviewed 5 days ago

[View changes](#)

django_sites/pegasus/demo/templates/demo/login.html

| | | | |
|-----|-----|-------------------|---|
| ... | ... | @@ -10,7 +10,7 @@ | <h1 class="text-center mb-5">Welcome Back!</h1> |
| 10 | 10 | | |
| 11 | 11 | | <form method="post" class="text-center text-white text-size"> |
| 12 | 12 | | {% csrf_token %} |
| 13 | - | | {{ context.login_form }} |
| | 13 | + | {{ context.login_form }} |



aguanlao 5 days ago

+ 😊 ...

Suggested change ⓘ

| | | | |
|----|---|--|---------------------------|
| 13 | - | | {{ context.login_form }} |
| 13 | + | | {{ context.login_form }} |

Commit suggestion ▼



Backend base #22

TFarron merged 5 commits into `dev` from `backend-base` 13 days ago



aguanlao commented 13 days ago • edited ▾

+ 😊 ...

Backend base updated to dev. Merging back to dev.



aguanlao added some commits 15 days ago



Removed duplicate EditUserForm

eb6e7d9



Added CreateListingForm; functionality for adding listings

d3aca90



Added basic view listing

a2e7661



Added ability to edit listings

533cdad



Merge remote-tracking branch 'origin/dev' into backend-base

de79356



TFarron reviewed 13 days ago

[View changes](#)

TFarron left a comment

+ 😊 ...

Will approve if following features are confirmed to be working with this update:

- Login/Logout
- View/Edit User accounts



TFarron commented 13 days ago

+ 😊 ...

Talked with Adan, features should be working.



TFarron approved these changes 13 days ago

[View changes](#)



TFarron merged commit `ea8e4f3` into `dev` 13 days ago

[Revert](#)



TFarron deleted the `backend-base` branch 13 days ago

[Restore branch](#)

Self-check: Adherence to Original Non-Functional Specifications

| Specification | Status |
|---|---|
| Application shall be developed, tested, and deployed using tools specified in M0 and reviewed and approved by Nicholas Stepanov. | DONE |
| Application shall be optimized for standard desktop/laptop browsers and will render correctly on the two latest versions of two major browsers. | DONE |
| Data shall be stored in our chosen database on our deployment server. | DONE |
| Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users. | ON TRACK |
| Application shall be easy to use and intuitive. | ON TRACK |
| No email clients shall be allowed. | DONE |
| Pay functionality shall not be implemented nor simulated. | DONE |
| Basic site security best practices shall be applied. Web traffic should travel over HTTPS. Users must log in to be able to post or apply to listings. Login credentials must be stored in an encrypted format (preferably SHA-512). | ISSUE - Amazon self signed certificate still a problem for HTTPS. May switch to HTTP. |
| Before posted live, all content (apartment listings and images) must be approved by site administrator. | DONE |
| Modern software engineering processes and practices shall be used as specified in the class, including collaborative and continuous software development. | DONE |
| The website shall prominently display the following exact text on all pages: "SFSU Software Engineering Project CSC 648-848, Spring 2019. For Demonstration Only." at the top of the WWW page. | ON TRACK |
| The website shall be fast. Searching for and applying to listings should not take more than 3 seconds per event. | ON TRACK |
| The website must have high fault tolerance. | ON TRACK |

| | |
|---|----------|
| The website will be scalable to allow expansion to additional universities. | ON TRACK |
| The software will be readable, expandable, and maintainable. | ON TRACK |

APPENDIX

Below are integration tests currently with poor outcomes or not able to be completed.

| Integration Test 3 - User Story S1 - Student - Firefox | | | |
|--|---|--|------------------------|
| Test Case ID: S_03 | | Test Case Description: Save listing / contact landlord / sign up for open house | |
| Prerequisites: | | Assumes accessing an individual listing's page | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 05/06/2019 | Click on save listing | Adds listing to student profile page saved listings | Fail - not implemented |
| 05/06/2019 | Click on Contact button | Redirect to messaging page (or popup?) | Fail - not implemented |
| 05/06/2019 | Message landlord | Write a message, and the message sends to landlord inbox | Fail - not implemented |
| 05/06/2019 | Sign up for open house (click on desired open house time) | Message is sent to landlord with student profile, the event shows up in student's calendar | Fail - not implemented |

| Integration Test 4 - User Story S1 - Student - Firefox | | | |
|--|-------------------------|--|------------------------|
| Test Case ID: S_04 | | Test Case Description: Student interaction with own profile and settings | |
| Prerequisites: | | Assumes successful sign-in and account creation | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 05/06/2019 | Click on View Profile | Student profile displays | Pass |
| 05/06/2019 | Click on saved listings | All saved listings that are still valid are displayed (and are clickable to each listing's page) | Fail - Not implemented |

| | | | |
|------------|-----------------------|---|---|
| 05/06/2019 | Click on forum button | Redirect to student neighbor network forum | Pass |
| 05/06/2019 | View calendar | Calendar is populated with all open house times the student has signed up for | Fail - not implemented |
| 05/06/2019 | Edit profile | Should be able to edit profile and save | Fail - tried to edit the bio, and it gives the field already exists error |
| 05/06/2019 | View inbox | All private messages sent to the student should show up in the inbox | Pass |

| | | | |
|---|-------------------------|--|------------------------|
| Integration Test 4 - User Story S1 - Student - Chrome | | | |
| Test Case ID: S_04 | | Test Case Description: Student interaction with own profile and settings | |
| Prerequisites: | | Assumes successful sign-in and account creation | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 5/07/2019 | Click on View Profile | Student profile displays | Pass |
| 5/07/2019 | Click on saved listings | All saved listings that are still valid are displayed (and are clickable to each listing's page) | Fail - Not implemented |
| 5/07/2019 | Click on forum button | Redirect to student neighbor network forum | Fail-Not implemented |
| 5/07/2019 | View calendar | Calendar is populated with all open house times the student has signed up for | Fail - not implemented |
| 5/07/2019 | Edit profile | Should be able to edit profile and save | Pass |
| 5/07/2019 | View inbox | All private messages | Not implemented |

| | | | |
|--|--|---|--|
| | | sent to the student should show up in the inbox | |
|--|--|---|--|

| Integration Test 7 - User Story L1 - Landlord - Firefox | | | |
|---|---|--|-----------------------------------|
| Test Case ID: L_02 | | Test Case Description: Post Listings / delete listings | |
| Prerequisites: | | Assumes successful login | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 05/07/2019 | Access landlord profile | Redirect to landlord profile page | Fail - Redirected to student page |
| 05/07/2019 | Click on add listing | Page for adding a listing is displayed, and landlord can enter information and upload photos | Fail - Because of above |
| 05/07/2019 | Pending approval for listing | After submitting new listing, landlord receives message that the listing is pending review. | Fail - Because of above |
| 05/07/2019 | Getting approval for listing | After the listing is approved by admin, the landlord receives a message and the listing is displayed in the landlord profile | Fail - Because of above |
| 05/07/2019 | Creating open house dates and times - Landlord sets available showing times in "Calendar" tab | Clickable timeslot buttons are displayed on the listing. | Fail - Not implemented |
| 05/07/2019 | On each listing, click on remove listing | Listing status changes to invalid | Fail - Because of above |

| |
|--|
| Integration Test 7 - User Story L1 - Landlord - Chrome |
|--|

| Test Case ID: L_02 | | Test Case Description: Post Listings / delete listings | |
|--------------------|---|--|-----------------|
| Prerequisites: | | Assumes successful login | |
| Date tested | Scenario | Expected Results | Pass / Fail |
| 05/07/2019 | Access landlord profile | Redirect to landlord profile page | Pass |
| 05/07/2019 | Click on add listing | Page for adding a listing is displayed, and landlord can enter information and upload photos | Not implemented |
| 05/07/2019 | Pending approval for listing | After submitting new listing, landlord receives message that the listing is pending review. | Not implemented |
| 05/07/2019 | Getting approval for listing | After the listing is approved by admin, the landlord receives a message and the listing is displayed in the landlord profile | Not implemented |
| 05/07/2019 | Creating open house dates and times - Landlord sets available showing times in "Calendar" tab | Clickable timeslot buttons are displayed on the listing. | Not implemented |
| 05/07/2019 | On each listing, click on remove listing | Listing status changes to invalid | Not implemented |

This integration test will not be done until Landlord sign-up directs to Landlord profile

| Integration Test 8 - User Story L1 - Landlord - Firefox/Chrome | | | |
|--|----------|---|-------------|
| Test Case ID: L_03 | | Test Case Description: Interactions with students | |
| Prerequisites: | | Assumes successful login | |
| Date tested | Scenario | Expected Results | Pass / Fail |

| | | | |
|--|---|---|--|
| | Access landlord inbox in profile | Messages from student users are displayed | |
| | On calendar (or listing?), click on "Attendees" for open house. | List of students who have signed up, with links to their profiles is displayed | |
| | Contact open house attendees through attendees mailing list | Message will be sent to only those students who have signed up for the open house | |
| | Cancel open house - click cancel button | Registered attendees will be notified via the messaging feature that the open house is canceled | |