Test cases

1. Unit tests: Tests on individual components

a. Instantiate each object and verify that the object displays its correct information. If an object automatically adds information from a database, for the unit tests you may be required to provide the information manually. This test should be completed as an integration test.

Object-specific unit tests (if there’s anything really special in an object)

b. In User objects:

i. Test that the Donate function properly calls the payment processor.

ii. Test that “browsing” (simplistic search) properly calls the search plugin as indicated.

iii. Test that searching properly calls the search plugin with the information provided.

c. In all profiles, verify that the edit profile function works. Before the GUI is created, this should be simply verifying the getters and setters; after the GUI is created, this is an integration test.

d. For animal types, just verify that our language’s version of instanceof returns the correct type of animal. (For the most part, the animals will not have their own variables, but instead will simply be identified by the type of object they are.)

2. Integration Tests: Tests between components

a. Repeat 1a as an integration test with the database dynamically supplying info.

b. Verify that the payment processor and search plugins return the correct information. (E.g. when called by a user)

c. Verify that the Database generates unique IDs for users, pets, and pet providers.

d. Re-verify 1c after the GUI is created, this time using the GUI functions to access the editing features.

e. Verify that the pages display as intended.

f. Verify that the payment system returns a receipt and information to be stored in the donation info for each pet provider.

g. Verify the user account creation functions.

3. Regression tests: tests to be completed after each substantial change to the system.

a. High priority: Re-verify 2b, 2e, 2f.

b. Moderate priority: Re-verify 2a, 2c, 2d, and 2g.

c. Low priority: Verify that the appearance of the site doesn’t change.

Unit tests should be repeated on any class that is changed before integration tests involving that class are repeated.

4. System tests: Tests as-if end user

a. Browse and search for pets.

b. Donate to a provider.

c. Create a basic user profile and log in.

d. Adopt a pet.

e. View my pets.

f. Alter the pet’s profile, including the medical records (if application)

g. Alter my profile.

h. Donate.

i. Log out, then log in again, then delete my profile.

j. Create a GlobalAdmin.

k. Alter a user’s profile. Alter a PetProvider’s information.

l. Delete a user’s profile. Delete a pet provider.

m. Add a pet provider and assign admin status for that PetProvider to a user account.

n. Log in as admin for pet provider.

o. Perform administrative actions on the Pet Provider page, and re-test all the regular user actions (d—i).

p. Create a Vet account. Re-do the regular user actions.

q. Insert a pet’s medical records, then edit them.

r. View your patients, add a patient, delete a patient.