WebUITestDocumentation

version 1.0

Ryan Wilkins

February 29, 2016

Contents

Welcome to WebUITest's documentation! 1

- Tests 1
  - WebUI package 1
    - Submodules 1
    - WebUI.WebUI module 1
    - Module contents 3
  - conftest module 3
  - setup_db module 3
  - test_create_account module 3
  - test_edit_account module 4
  - test_login module 4
  - test_password module 5
  - test_security module 6
  - test_upload module 6
  - test_username module 6

Indices and tables 7

Index 9

Python Module Index 11

Welcome to WebUITest's documentation!

Contents:

Tests

WebUI package

Submodules

WebUI.WebUI module

These are the web user interface utilities used to drive the web tests.

class `WebUI.WebUI.`**`WebUI`** (`baseurl='http://localhost:25396/'`)

This class keeps track of the web session. It uses Firefox as the web driver. Its baseurl is the start page of the software. By default it is the localhost on port 25396.

`approve_account` (`username`)

Logs in as the system admin and approves an account with the given username

Parameters: **username** -- Username of the account to approve

Returns:

`check_login` (`t=5`)

This will verify you are logged in. It checks for the manage account link that only appears if you are logged in.

Parameters: **t** -- A timeout in seconds to wait for the management link to appear

Returns:

`check_logoff` (`t=10`)

This will check that there is no logged in user for the session

Parameters: **t** -- time in seconds to wait for the login button to appear. The default is 10.

Returns:

`check_request_account` ()

Validates that the account request was successful

Returns:

`confirm_account_info` (`email=''`, `address=''`, `phonenum=''`)

This procedure will confirm account information of whatever information is provided.

Parameters:

- **email** -- the expected email address
- **address** -- the expected home address
- **phonenum** -- the expected phone number

Returns:

`create_patient` (`user`, `pwd`, `email`, `first_name`, `last_name`, `address`, `phone_number`)

This will create a patient account. It expects the physician to be logged in when called.

Parameters:
- **user** --
- **pwd** --
- **email** --
- **first_name** --
- **last_name** --
- **address** --
- **phone_number** --

Returns:

delete_account (user**)**
This logs in as the system admin and deletes the specified user

Parameters: **user** -- username of the account to delete

Returns:

driver = *<selenium.webdriver.firefox.webdriver.WebDriver (session="0da33f1b-159d-411b-b705-e2f7475bdc99")>*

get_page ()
Helper procedure to return the current web pages source useful to validate responses on the page

Returns:

go_home ()
Navigates the web session back to the base url

Returns:

login (user**,** pwd**)**
This will make sure you are logged in as the user specified. It will log you out if you are already logged in.

Parameters:
- **user** -- Username to log in as
- **pwd** -- Password for the given username

Returns:

logoff ()
This will log the logged in user off.

Returns:

request_account (account_type**,** user**,** pwd**,** email**,** first_name**,** last_name**,** address**,** phone_number**)**
This will fill out a request for an account

Parameters:
- **account_type** -- The type of account to request. can either be Physician or Exp Admin
- **user** -- The username of the new account
- **pwd** -- The password of the new account
- **email** -- The email address of the new account
- **first_name** -- The first name of the user
- **last_name** -- The last name of the user
- **address** -- The home address of the user
- **phone_number** -- The phone number of the user

Returns:

reset_user_password (user**,** new_pwd**)**
This procedure will make the system admin reset an accounts password

Parameters:
- **user** -- username of the account to touch
- **new_pwd** -- the new password to give the account

Returns:

set_account_info (pwd='', current_pwd='', email='', address='', phonenum=''**)**
This procedure will set the account info to any info provided if provided It assumes the user is already logged in

Parameters:
- **pwd** -- the new password desired
- **current_pwd** -- the current password. needed to change passwords
- **email** -- the new email address
- **address** -- the new home address
- **phonenum** -- the new phone number

Returns:

upload_files (files**,** activity_dict**)**
This will uplaod files for a patient with the given activities

Parameters:
- **files** -- Relative paths to files to be uploaded
- **activity_dict** -- A dictionary with index of activity and elements of stime and ftime

Returns:

Module contents

conftest module

conftest.**login_sysadmin** ()

conftest.**login_texpadmin** ()

conftest.**login_tpatient** ()

conftest.**login_tphysician** ()

conftest.**logoff** ()

conftest.**pre_existing_users** ()

setup_db module

setup_db.**main** ()

test_create_account module

These test cases are designed to test the ability to create and delete accounts

test_create_account.**test_create_experiment_admin** (logoff)
This test will request an Exp Admin account of awong, approve the account and login as this user Validation is done at the request account and login steps.

Parameters: **logoff** -- Makes sure the web session is in a logged off state at the start of the test

Returns:

test_create_account.**test_create_patient** (login_tphysician)
This test will create a patient account of pfry from the tphysician login and login as pfry Validation is done on the create patient step and login step

Parameters: **login_tphysician** -- Makes sure the web session is logged in as tphysician at the start of the test

Returns:

test_create_account.**test_create_physician** (logoff)
This test will request an Physician account of hfarnsworth, approve the account and login as this user Validation is done at the request account and login steps.

Parameters: **logoff** -- Makes sure the web session is in a logged off state at the start of the test

Returns:

test_create_account.**test_delete_experiment_admin** (logoff)
This test will delete an experiment admin account using the System Admin account and verify the user cannot login Validation is done on the login step

Parameters: **logoff** -- Makes sure the web session is in a logged off state at the start of the test

Returns:

test_create_account.**test_delete_patient** (logoff)
This test will delete a patient account using the System Admin account and verify the user cannot login Validation is done on the login step

Parameters: **logoff** -- Makes sure the web session is in a logged off state at the start of the test

Returns:

test_create_account.**test_delete_physician** (logoff)
This test will delete a physician account using the System Admin account and verify the user cannot login Validation is done on the login step

Parameters: **logoff** -- Makes sure the web session is in a logged off state at the start of the test

Returns:

test_edit_account module

These tests are here to verify that the accounts can have their information edited.

test_edit_account.**test_physician_can_edit_his_account** (login_tphysician)
This test will edit the testPhysician's email address and verify the email address got changed Validation is on both the edit step and the confirmation step

Parameters: **login_tphysician** -- Makes sure the session is logged in as the testPhysician at the start of the test

Returns:

test_edit_account.**test_sys_admin_can_edit_exp_admin_account** (login_sysadmin)

test_edit_account.**test_sys_admin_can_edit_patient_account** (login_sysadmin)

test_edit_account.**test_sys_admin_can_edit_physician_account** (login_sysadmin)

test_login module

These tests are to verify that the accounts can be logged into

test_login.**test_login_bad_pass** (logoff)
This test will verify that a user cannot login with an incorrect password Validation is done by checking for "Invalid login attempt" in the web page

Parameters: **logoff** -- Makes sure the session is in a logged off state at the start of the test

Returns:

test_login.**test_login_bad_user** (logoff)
This test will verify that a user cannot login with an incorrect username Validation is done by checking for "Invalid login attempt" in the web page

Parameters: **logoff** -- Makes sure the session is in a logged off state at the start of the test
Returns:

test_login.**test_login_experiment_admin** (logoff)
This test will login as the testExpAdmin. Validation is on the login step.

Parameters: **logoff** -- Makes sure the session is in a logged off state at the start of the test
Returns:

test_login.**test_login_patient** (logoff)
This test will login as the testPatient. Validation is on the login step.

Parameters: **logoff** -- Makes sure the session is in a logged off state at the start of the test
Returns:

test_login.**test_login_physician** (logoff)
This test will login as the testPhysician. Validation is on the login step.

Parameters: **logoff** -- Makes sure the session is in a logged off state at the start of the test
Returns:

test_login.**test_login_system_admin** (logoff)
This test will login as the fitadmin. Validation is on the login step.

Parameters: **logoff** -- Makes sure the session is in a logged off state at the start of the test
Returns:

test_password module

These tests are to make sure password functionality meets the requirements

test_password.**test_good_password** (logoff)
This test tries to create an account with a password that meets all the requirements Validation is done by checking the web page's response after submitting the request

Parameters: **logoff** -- Makes sure the session is in a logged off state before the test starts
Returns:

test_password.**test_password_case** (logoff)
This test tries to create an account with a password that is has no uppercase Validation is done by checking the web page's response after submitting the request

Parameters: **logoff** -- Makes sure the session is in a logged off state before the test starts
Returns:

test_password.**test_password_digit** (logoff)
This test tries to create an account with a password that has no numbers Validation is done by checking the web page's response after submitting the request

Parameters: **logoff** -- Makes sure the session is in a logged off state before the test starts
Returns:

test_password.**test_password_length** (logoff)
This test tries to create an account with a password that is too short Validation is done by checking the web page's response after submitting the request

Parameters: **logoff** -- Makes sure the session is in a logged off state before the test starts
Returns:

test_password.**test_password_special_char** (logoff)

This test tries to create an account with a password that is too short Validation is done by checking the web page's response after submitting the request

Parameters: **logoff** -- Makes sure the session is in a logged off state before the test starts

Returns:

test_password.**test_reset_password** (login_sysadmin)
This test has the system admin reset a users password. Then the user tries to login with the new password Validation is done on the reset step and the login step

Parameters: **login_sysadmin** --

Returns:

test_security module

test_security.**test_exp_admin_cannot_create_patient** ()
This procedure tests that an experiment admin cannot create another patient. :return:

test_security.**test_patient_cannot_create_patient** ()
This procedure tests that a patient cannot create another patient.

Returns:

test_security.**test_patient_to_patient** ()
This procedure tests one patient trying to view a different patients data

Returns:

test_security.**test_physician_to_patient** ()
This procedure tests a physician cannot view a patient's data that belongs to a patient who belongs to a different physician.

Returns:

test_upload module

test_upload.**test_basis_peak_data_upload** ()

test_upload.**test_mband_data_upload** ()

test_upload.**test_multi_file_multi_activity** ()

test_upload.**test_multi_file_no_activity** (login_tpatient)

test_upload.**test_multi_file_one_activity** ()

test_upload.**test_single_file_multi_activity** ()

test_upload.**test_single_file_no_activity** (login_tpatient)

test_upload.**test_single_file_one_activity** ()

test_upload.**test_zephyr_data_upload** ()

test_username module

These tests are to make sure the username functions correctly

test_username.**test_username_cannot_be_copied** (logoff)
This test creates a new user, tries to create another user with the same name, then creates third unique user Validatin is done on all three steps. This verifies that the username must be unique.

Parameters: **logoff** -- Makes sure the session is in a logged off state before the test starts.

Returns:

Indices and tables

- `genindex`
- `modindex`
- `search`

Index

A

approve_account() (WebUI.WebUI.WebUI method)

C

check_login() (WebUI.WebUI.WebUI method)

check_logoff() (WebUI.WebUI.WebUI method)

check_request_account() (WebUI.WebUI.WebUI method)

confirm_account_info() (WebUI.WebUI.WebUI method)

conftest (module)

create_patient() (WebUI.WebUI.WebUI method)

D

delete_account() (WebUI.WebUI.WebUI method)

driver (WebUI.WebUI.WebUI attribute)

G

get_page() (WebUI.WebUI.WebUI method)

go_home() (WebUI.WebUI.WebUI method)

L

login() (WebUI.WebUI.WebUI method)

login_sysadmin() (in module conftest)

login_texpadmin() (in module conftest)

login_tpatient() (in module conftest)

login_tphysician() (in module conftest)

logoff() (in module conftest)

(WebUI.WebUI.WebUI method)

M

main() (in module setup_db)

P

pre_existing_users() (in module conftest)

R

request_account() (WebUI.WebUI.WebUI method)

reset_user_password() (WebUI.WebUI.WebUI method)

S

set_account_info() (WebUI.WebUI.WebUI method)

setup_db (module)

T

test_basis_peak_data_upload() (in module test_upload)

test_create_account (module)

test_create_experiment_admin() (in module test_create_account)

test_create_patient() (in module test_create_account)

test_create_physician() (in module test_create_account)

test_delete_experiment_admin() (in module test_create_account)

test_delete_patient() (in module test_create_account)

test_delete_physician() (in module test_create_account)

test_edit_account (module)

test_exp_admin_cannot_create_patient() (in module test_security)

test_good_password() (in module test_password)

test_login (module)

test_login_bad_pass() (in module test_login)

test_login_bad_user() (in module test_login)

test_login_experiment_admin() (in module test_login)

test_login_patient() (in module test_login)

test_login_physician() (in module test_login)

test_login_system_admin() (in module test_login)

test_mband_data_upload() (in module test_upload)

test_multi_file_multi_activity() (in module test_upload)

test_multi_file_no_activity() (in module test_upload)

test_multi_file_one_activity() (in module test_upload)

test_password (module)

test_password_case() (in module test_password)

test_password_digit() (in module test_password)

test_password_length() (in module test_password)

test_password_special_char() (in module test_password)

test_patient_cannot_create_patient() (in module test_security)

test_patient_to_patient() (in module test_security)

test_physician_can_edit_his_account() (in module test_edit_account)

test_physician_to_patient() (in module test_security)

test_reset_password() (in module test_password)

test_security (module)

test_single_file_multi_activity() (in module test_upload)

test_single_file_no_activity() (in module test_upload)

test_single_file_one_activity() (in module test_upload)

test_sys_admin_can_edit_exp_admin_account() (in module test_edit_account)

test_sys_admin_can_edit_patient_account() (in module test_edit_account)

test_sys_admin_can_edit_physician_account() (in module test_edit_account)

test_upload (module)

test_username (module)

test_username_cannot_be_copied() (in module test_username)

test_zephyr_data_upload() (in module test_upload)

U

upload_files() (WebUI.WebUI.WebUI method)

W

WebUI (class in WebUI.WebUI)

(module)

WebUI.WebUI (module)

Python Module Index

c

conftest

s

setup_db

t

test_create_account

test_edit_account

test_login

test_password

test_security

test_upload

test_username

w

WebUI

WebUI.WebUI