

---

# **BB-Classic Documentation**

***Release 3.0***

**Roman Kozlovskyi**

July 23, 2014



<b>1</b>	<b>Overview</b>	<b>1</b>
<b>2</b>	<b>Contents</b>	<b>3</b>
2.1	Login page . . . . .	4
2.2	Logout page . . . . .	4
2.3	My info page . . . . .	4
2.4	Projects pages . . . . .	4
2.5	Companies pages . . . . .	4
<b>3</b>	<b>Modules source code documentation</b>	<b>5</b>
3.1	bb's module . . . . .	5
3.2	keys's module . . . . .	8
3.3	crypto's module . . . . .	10
<b>4</b>	<b>Acceptance tests</b>	<b>11</b>
4.1	Acceptance tests . . . . .	11
4.2	Generate screenshots . . . . .	15
4.3	Acceptance tests keywords . . . . .	17
4.4	Acceptance tests settings . . . . .	26
4.5	Acceptance tests variables . . . . .	26
<b>5</b>	<b>Indices and tables</b>	<b>27</b>
	<b>Python Module Index</b>	<b>29</b>



---

## Overview

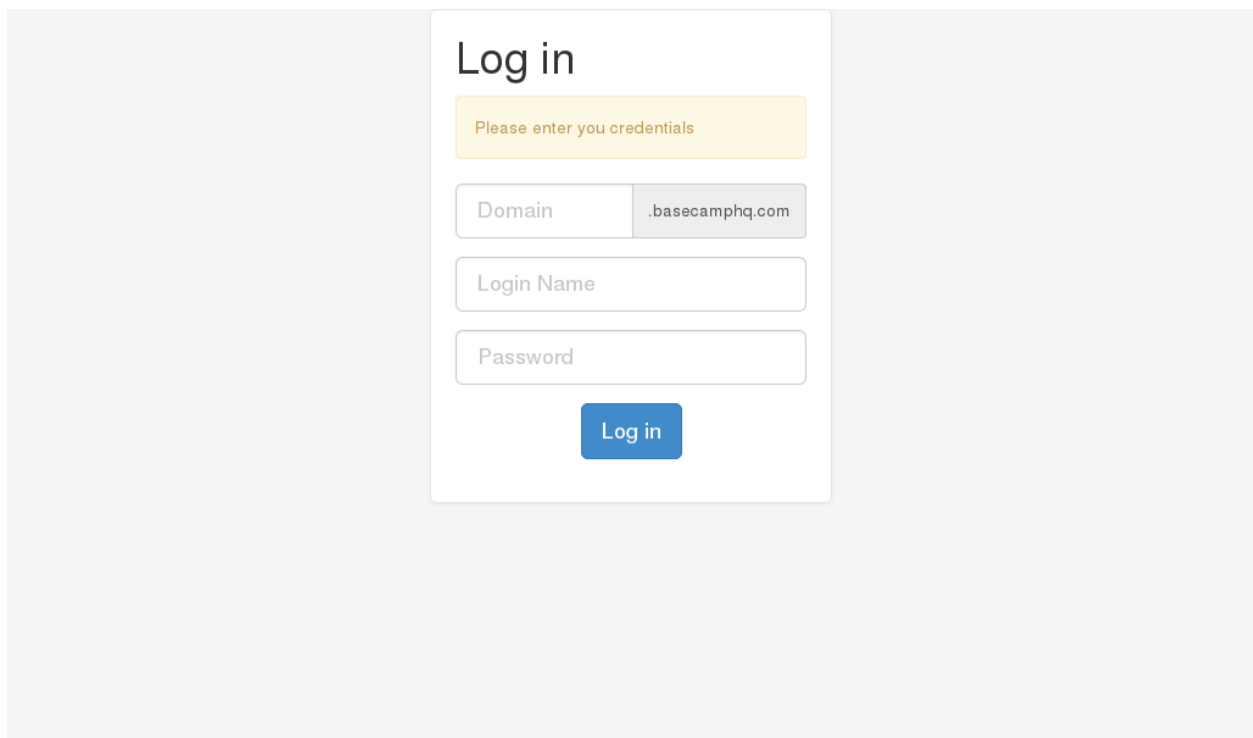
---

Basecamp on Backbone, Twitter Bootstrap, GAE



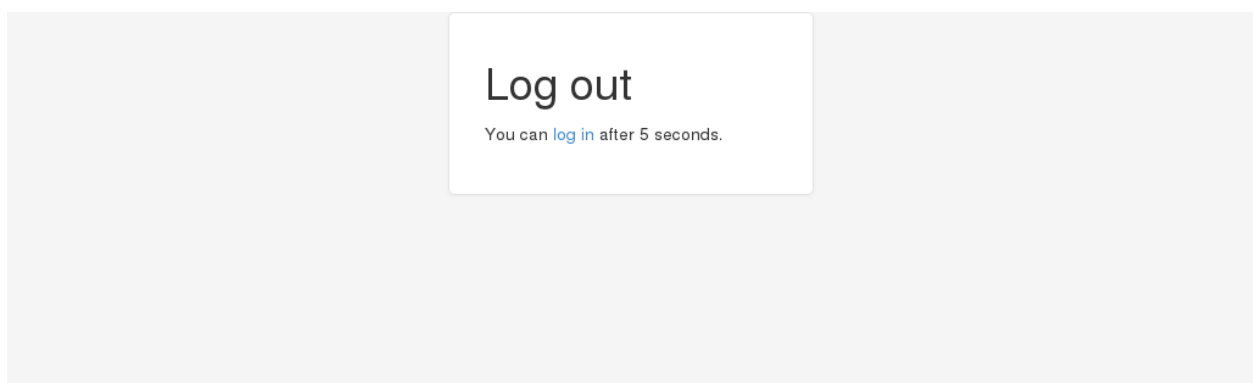


## 2.1 Login page



The screenshot shows a login interface centered on a light gray background. The interface is contained within a white box with rounded corners. At the top of this box, the text "Log in" is displayed in a large, dark font. Below this, a yellow rectangular box contains the text "Please enter you credentials". Underneath the yellow box, there are three input fields: a "Domain" field with the value ".basecampHQ.com" entered, a "Login Name" field, and a "Password" field. At the bottom of the white box, there is a blue button with the text "Log in" in white.

## 2.2 Logout page



The screenshot shows a logout interface centered on a light gray background. The interface is contained within a white box with rounded corners. At the top of this box, the text "Log out" is displayed in a large, dark font. Below this, the text "You can [log in](#) after 5 seconds." is displayed, where "log in" is a blue link.



---

## Modules source code documentation

---

### 3.1 bb's module

#### 3.1.1 Main BB classic app module

- `/` - Main Page Handler
- `/login` - Login Page Handler
- `/logout` - Logout Page Handler
- `/api/.*` - Cross Domain Handler

**class** `bb.BaseRequestHandler` (*request=None, response=None*)  
Base Request Handler

**auth\_check** ()  
Check session

**dev**  
Check developement environment

**class** `bb.CacheInfo` (*parent=None, key\_name=None, \_app=None, \_from\_entity=False, \*\*kws*)  
Model for the cached response.

Attributes:

- *url* - Fetch URL
- *status\_code* - Response status
- *headers* - Response headers
- *content* - Response content
- *date* - Date when response was added to the cache

**class** `bb.CrossDomain` (*request=None, response=None*)  
Cross Domain Handler

- `PUT /api/.*` - CrossDomain PUT
- `POST /api/.*` - CrossDomain POST
- `DELETE /api/.*` - CrossDomain DELETE
- `GET /api/.*` - CrossDomain GET

**apiurl**  
get api url

**delete()**  
wrapper for the function

**fetch\_request** (*method*, *data=None*)  
Fetch request

**Parameters**

- **url** (*string*) – [required] request url
- **headers** (*dict*) – [required] request headers

**Returns** request response

**Raises** Exception

**Return type** [Response](#)

**fullurl**  
get full url

**get()**  
wrapper for the function

**jsondata**  
get json data from request body

**post()**  
wrapper for the function

**put()**  
wrapper for the function

**xmldata**  
get xml data from request body

**exception** `bb.GetSubjectException`  
Exception on get subject\_id

**class** `bb.LoginPage` (*request=None*, *response=None*)  
Login Page Handler

- GET /login - [LoginPage GET](#)
- POST /login - [LoginPage POST](#)

**get()**  
GET request

**post()**  
POST request

**class** `bb.LogoutPage` (*request=None*, *response=None*)  
Logout Page Handler

- GET /logout - [LogoutPage GET](#)

**get()**  
GET request

**class** `bb.MainPage` (*request=None*, *response=None*)  
Main Page Handler

•GET / - [MainPage GET](#)

**get** ()

wrapper for the function

`bb.absolute_url (subdomain, relative_url='', params='', query='', fragment='')`

Prepare absolute url for request

#### Parameters

- **subdomain** (*string*) – [required] basecamphq subdomain
- **relative\_url** (*string*) – relative url
- **params** (*string*) – url parameters
- **query** (*string*) – url query
- **fragment** (*string*) – url fragment

**Returns** absolute url for request

**Return type** string

`bb.authenticated (func)`

decorator for check authentication

`bb.convert (node)`

convert xml to dict

**Parameters** **node** (*string*) – [required] xml node to convert

**Returns** converted node

**Raises** Exception

**Return type** tuple(string, valuetype)

`bb.convertchilds (childs)`

convert childs

**Parameters** **childs** (*list*) – [required] list of nodes

**Returns** converted childs

**Raises** Exception

**Return type** list

`bb.convertsubnodes (childs)`

convert subnodes

**Parameters** **childs** (*list*) – [required] list of nodes

**Returns** converted nodes

**Raises** Exception

**Return type** dict[string]

`bb.dict2xml (data, tags)`

convert dict to xml

#### Parameters

- **data** (*dict*) – [required] dict with data to convert
- **tags** (*tuple*) – [required] tuple of base tags

**Returns** pretty xml

**Return type** string

`bb.get_headers (username, password)`

Prepare request headers

**Parameters**

- **username** (*string*) – [required] username
- **password** (*string*) – [required] password

**Returns** headers dict

**Return type** dict

`bb.get_subject_id (username, password, subdomain)`

Get 'subject\_id' for report query - it is id of logged in user.

**Parameters**

- **username** (*string*) – [required] username
- **password** (*string*) – [required] password
- **subdomain** (*string*) – [required] subdomain

**Returns** id of logged in user

**Raises** [GetSubjectException](#)

**Return type** string

`bb.get_xml_for_request (url)`

Get xml base tags for request

**Parameters** **url** (*string*) – [required] url

**Returns** tuple of base tags or None

**Return type** tuple or None

`bb.save_request (url, result)`

Save request

**Parameters**

- **url** (*string*) – [required] request url
- **result** (*Response*) – [required] request response

## 3.2 keys's module

### 3.2.1 keyring module

`keys._db_get ()`

get keyring from db

**Returns** keyring

**Return type** [Keyring](#)

**class** `keys.Keyring` (*parent=None, key\_name=None, \_app=None, \_from\_entity=False, \*\*kws*)

Keyring db model

Attributes:

• *data* - encoded key data

`keys.current()`

get current key

**Returns** get next key

**Return type** string

`keys.data()`

get keys

**Returns** key

**Return type** string

`keys.delete()`

delete keyring

`keys.generate_key()`

generate key

**Returns** generated key pair

**Return type** tuple of string

`keys.generate_marker()`

regenerate\_raw\_key implementation satisfies needs

**Returns** generated key

**Return type** string

`keys.generate_raw_key()`

generate raw key

**Returns** generated key

**Return type** string

`keys.refresh()`

refresh keys

**Returns** fresh keys

**Return type** list of key pairs

`keys.rotate()`

rotate keys

`keys.setkeys(keys)`

set keys

**Parameters** `keys` (*list*) – [required] list of key pairs

## 3.3 crypto's module

### 3.3.1 crypto module

- [/genkeys](#) - Generate keys handler
- [/addkey](#) - Add key handler

**class** `crypto.AddkeyPage`

Add key handler

- GET [/addkey](#) - AddkeyPage GET
- POST [/addkey](#) - AddkeyPage POST

**class** `crypto.BaseHandler`

Base crypto module handler

- GET [/. \\*](#) - BaseHandler GET
- POST [/. \\*](#) - BaseHandler POST

**get** ()

GET request

**post** ()

POST request

**exception** `crypto.DecodeError`

Decode Error

**class** `crypto.GenkeysPage`

Generate keys handler

- GET [/genkeys](#) - GenkeysPage GET
- POST [/genkeys](#) - GenkeysPage POST

`crypto.decode_data (source, delimiter='\\n')`  
decode data

`crypto.decrypt (source, key)`

`crypto.encode_data (values, delimiter='\\n')`  
encode data

`crypto.encrypt (source, key)`

`crypto.main ()`

---

## Acceptance tests

---

### 4.1 Acceptance tests

#### 4.1.1 Login page should have inputs and validate submit

Login page should have input[name=subdomain], input[name=username], input[name=password] and should validate inputs on submit.

```
Given an bb-classic app login page
Then page should contain Please enter your credentials
And page should contain element name=subdomain
And page should contain element name=username
And page should contain element name=password
When enter subdomain field
And click button Log in
Then page should contain Please enter all fields!
When enter username field
And click button Log in
Then page should contain Please enter all fields!
When enter password field
And click button Log in
Then Wait Until Page Contains First Last
```

#### 4.1.2 Navigation menu should work correctly

Navigation menu should change location and heading correctly.

```
Given an bb-classic app
When click on "Projects" link in navbar
Then location should be ${APP_URL}/#projects
And heading should be "Projects"
And title should be Projects - BB
When click on "Companies" link in navbar
Then location should be ${APP_URL}/#companies
And heading should be "Companies"
And title should be Companies - BB
And wait for data loading
When click link Name of #0
Then location should be ${APP_URL}/#companies/0
And heading should be "Name of #0"
And title should be Name of #0 - Companies - BB
```

```
When click on "To-Dos" link in navbar
Then location should be ${APP_URL}/#todos
And heading should be "My to-dos"
And title should be My to-dos - BB
When click on "Time" link in navbar
Then location should be ${APP_URL}/#time_report
And heading should be "Time report"
And title should be Time report - BB
When click on "People" link in navbar
Then location should be ${APP_URL}/#people
And heading should be "People"
And title should be People - BB
When click link First#0 Last
Then location should be ${APP_URL}/#people/0
And heading should be "First#0 Last"
And title should be First#0 Last - People - BB
```

### 4.1.3 My profile page should contains my info

My profile page should contains info about me.

```
Given an bb-classic app
When go to my profile page
Then my info be visible
```

### 4.1.4 Projects page should contains links to projects

Projects page should contains correct links to projects if company and projects state selected.

```
Given an projects page
And wait for data loading
When click on "Company name #0" in active state
Then link to Name of #0 should be visible
And link to Name of #5 should not be visible
And link to Name of #10 should not be visible
And link to Name of #15 should be visible
And link to Name of #20 should not be visible
And link to Name of #25 should not be visible
When click on "Company name #0" in on_hold state
Then link to Name of #0 should not be visible
And link to Name of #5 should not be visible
And link to Name of #10 should be visible
And link to Name of #15 should not be visible
And link to Name of #20 should not be visible
And link to Name of #25 should be visible
When click on "Company name #0" in archived state
Then link to Name of #0 should not be visible
And link to Name of #5 should be visible
And link to Name of #10 should not be visible
And link to Name of #15 should not be visible
And link to Name of #20 should be visible
And link to Name of #25 should not be visible
```



## 4.1.5 Project navigation should work correctly

Projects navigation should change location and heading correctly.

```

Given an projects page
And wait for data loading
When click link Name of #0
Then location should be ${APP_URL}/#projects/0
And heading should be "Name of #0"
And title should be Name of #0 - Projects - Company name #0 - Companies - BB
When click on "Messages" in projectnav
Then location should be ${APP_URL}/#projects/0/posts
And heading should be "Posts"
And title should be Posts - Name of #0 - Projects - Company name #0 - Companies - BB
And wait for data loading
When click link Title #0
Then location should be ${APP_URL}/#projects/0/posts/0
And heading should be "Title #0"
And title should be Title #0 - Posts - Name of #0 - Projects - Company name #0 - Companies - BB
When click element css=.itemcomments
Then location should be ${APP_URL}/#projects/0/posts/0/comments
And heading should be "Comments"
And title should be Comments - Title #0 - Posts - Name of #0 - Projects - Company name #0 - Companies - BB
When click on "To-Dos" in projectnav
Then location should be ${APP_URL}/#projects/0/todo_lists
And heading should be "To-dos"
And title should be To-dos - Name of #0 - Projects - Company name #0 - Companies - BB
And wait for data loading
When click link css=a[href$='todo_lists/0']
Then location should be ${APP_URL}/#projects/0/todo_lists/0
And heading should be "Name of #0"
And title should be Name of #0 - To-dos - Name of #0 - Projects - Company name #0 - Companies - BB
And wait for data loading
When click link Todo content #0
Then location should be ${APP_URL}/#projects/0/todo_lists/0/0
And heading should be "Todo content #0"
And title should be Todo content #0 - Name of #0 - To-dos - Name of #0 - Projects - Company name #0 - Companies - BB
When click element css=.itemcomments
Then location should be ${APP_URL}/#projects/0/todo_lists/0/0/comments
And heading should be "Comments"
And title should be Comments - Todo content #0 - Name of #0 - To-dos - Name of #0 - Projects - Company name #0 - Companies - BB
When click on "Calendar" in projectnav
Then location should be ${APP_URL}/#projects/0/calendar
And heading should be "Calendar"
And title should be Calendar - Name of #0 - Projects - Company name #0 - Companies - BB
And wait for data loading
When click link Title #0
Then location should be ${APP_URL}/#projects/0/calendar/0
And heading should be "Title #0"
And title should be Title #0 - Calendar - Name of #0 - Projects - Company name #0 - Companies - BB
When click element css=.itemcomments
Then location should be ${APP_URL}/#projects/0/calendar/0/comments
And heading should be "Comments"
And title should be Comments - Title #0 - Calendar - Name of #0 - Projects - Company name #0 - Companies - BB
When click on "Time" in projectnav
Then location should be ${APP_URL}/#projects/0/time_entries
And heading should be "Time"
And title should be Time - Name of #0 - Projects - Company name #0 - Companies - BB
When click on "Files" in projectnav

```

```
Then location should be ${APP_URL}/#projects/0/files
And heading should be "Files"
And title should be Files - Name of #0 - Projects - Company name #0 - Companies - BB
And wait for data loading
When click link css=a[href$='files/0']
Then location should be ${APP_URL}/#projects/0/files/0
And heading should be "Name of #0"
And title should be Name of #0 - Files - Name of #0 - Projects - Company name #0 - Companies - BB
When click on "Categories" in projectnav
Then location should be ${APP_URL}/#projects/0/categories
And heading should be "Categories"
And title should be Categories - Name of #0 - Projects - Company name #0 - Companies - BB
And wait for data loading
When click link css=a[href$='categories/0']
Then location should be ${APP_URL}/#projects/0/categories/0
And heading should be "Name of #0"
And title should be Name of #0 - Categories - Name of #0 - Projects - Company name #0 - Companies - BB
When click on "People" in projectnav
Then location should be ${APP_URL}/#projects/0/people
And heading should be "People"
And title should be People - Name of #0 - Projects - Company name #0 - Companies - BB
And wait for data loading
When click link First#0 Last
Then location should be ${APP_URL}/#projects/0/people/0
And heading should be "First#0 Last"
And title should be First#0 Last - People - Name of #0 - Projects - Company name #0 - Companies - BB
```

### 4.1.6 User can edit delete and sort time entries on time report page

User should edit, delete and sort time entries.

```
Given an time report page
When user edit time entry
Then page should contain edited entry
When user delete time entry
Then page should not contain deleted entry
When user sort time entries
Then page should contain sorted entries
```

### 4.1.7 User can add edit delete and sort time entries on project time page

User should add, edit, delete and sort time entries.

```
Given an project time page
When user edit time entry
Then page should contain edited entry
When user delete time entry
Then page should not contain deleted entry
When user sort time entries
Then page should contain sorted entries
When user add time entry
Then page should contain added entry
```

### 4.1.8 User can view todos for it and other users

User should view to-dos for it, other users and unassigned.

```
Given an to-dos page
When select from list target First#1 Last
Then Wait Until Page Contains First#1 Last's to-do items across all projects
And heading should be "First#1 Last's to-dos"
When select from list target Nobody
Then Wait Until Page Contains Unassigned to-do items across all projects
And heading should be "Unassigned to-dos"
```

### 4.1.9 User can add edit and delete todo lists and items

User should add, edit and delete todo lists and items.

```
Given an project todos page
When user add todo list
Then page should contain added todo list
When user edit todo list
Then page should contain edited todo list
When user delete todo list
Then page should not contain deleted todo list
When click link css=a[href$='todo_lists/0']
When user add todo item
Then page should contain added todo item
When user edit todo item
Then page should contain edited todo item
When user delete todo item
Then page should not contain deleted todo item
```

### 4.1.10 User can edit and delete calendar entries on project calendar page

User should edit and delete calendar entries.

```
Given an project calendar page
When user edit calendar entry
Then page should contain edited entry
When user delete calendar entry
Then page should not contain deleted entry
```

## 4.2 Generate screenshots

### 4.2.1 Generate screenshots

```
Sleep 1
Capture Page Screenshot ${base_login}
Test user login
Wait for data loading
Capture Page Screenshot ${base_projects}
Click link Companies
Wait for data loading
Capture Page Screenshot ${base_companies}
```

```
Click link Name of #0
Wait for data loading
Capture Page Screenshot ${base_company}
Click link To-Dos
Wait for data loading
Capture Page Screenshot ${base_todos}
Click link Time
Wait for data loading
Capture Page Screenshot ${base_time_report}
Click link People
Wait for data loading
Capture Page Screenshot ${base_people}
Click link First#0 Last
Wait for data loading
Capture Page Screenshot ${base_person}
Click link First Last
Click link My profile
Capture Page Screenshot ${base_mypage}
Click link Projects
Click link Name of #0
Capture Page Screenshot ${base_project_overview}
Click on "Messages" in projectnav
Capture Page Screenshot ${base_project_messages}
Click link Title #0
Wait for data loading
Capture Page Screenshot ${base_project_message}
Click element css=.itemcomments
Wait for data loading
Capture Page Screenshot ${base_project_message_comments}
Click on "To-Dos" in projectnav
Capture Page Screenshot ${base_project_todos}
Click link css=a[href$='todo_lists/0']
Wait for data loading
Capture Page Screenshot ${base_project_todolist}
Click link Todo content #0
Wait for data loading
Capture Page Screenshot ${base_project_todo}
Click element css=.itemcomments
Wait for data loading
Capture Page Screenshot ${base_project_todo_comments}
Click on "Calendar" in projectnav
Capture Page Screenshot ${base_project_calendar}
Click link Title #0
Wait for data loading
Capture Page Screenshot ${base_project_calendar_entry}
Click element css=.itemcomments
Wait for data loading
Capture Page Screenshot ${base_project_calendar_entry_comments}
Click on "Time" in projectnav
Capture Page Screenshot ${base_project_time}
Click on "Files" in projectnav
Capture Page Screenshot ${base_project_files}
Click link css=a[href$='files/0']
Wait for data loading
Capture Page Screenshot ${base_project_file}
Click on "Categories" in projectnav
Capture Page Screenshot ${base_project_categories}
Click link css=a[href$='categories/0']
```

```
Wait for data loading
Capture Page Screenshot  ${base_project_category}
Click on "People" in projectnav
Capture Page Screenshot  ${base_project_people}
Click link  First#0 Last
Wait for data loading
Capture Page Screenshot  ${base_project_person}
Test user logout
Capture Page Screenshot  ${base_logout}
```

## 4.3 Acceptance tests keywords

### 4.3.1 go to my profile page

Go to my profile page thru the user actions menu.

```
Click link  First Last
Click link  My profile
Location should be  ${APP_URL}/#me
```

### 4.3.2 my info be visible

Check if page contains user info.

```
Page should contain element  link=name@domain.com
Page should contain element  link=example
Page should contain  Mobile: +000 (00) 000-0000
Page should contain  Home: +000 (00) 000-0001
Page should contain  Fax: +000 (00) 000-0002
Page should contain  Office: +000 (00) 000-0003
Page should contain  Time zone: Europe/Kiev
```

### 4.3.3 an projects page

Go to projects page.

```
Given Test user login
Then click link  Projects
```

### 4.3.4 click on \${text} in \${state} state

Click on link in state block.

```
click on ${text} in ${state}    Click on ${text} in ${state} state and wait effect
```

### 4.3.5 link to \${name} should be visible

Check if link is visible.

```
link to ${name}      Element Should Be Visible  link=${name}
```

### 4.3.6 link to \${name} should not be visible

Check if link is not visible.

```
link to ${name}      Element Should Not Be Visible  link=${name}
```

### 4.3.7 an bb-classic app

Go to application page.

```
an bb-classic app
    Test user login
```

### 4.3.8 an bb-classic app login page

Go to application login page.

```
an bb-classic app login page
    Reload Page Until  Location should contain  login
    Page should contain  BB
    Title should be  BB
```

### 4.3.9 enter \${name} field

Enter “test” text in input field.

```
enter ${name}      Input Text  name=${name}  test
```

### 4.3.10 click on \${name} link in navbar

Click on link in navbar.

```
click on ${name}      Click link  xpath=//*[contains(@class,"navbar")]//a[contains(text(),${name})]
```

### 4.3.11 heading should be \${heading}

Check if heading contain text.

```
heading should be  ${heading}
    Page should contain element  xpath=//h1[contains(text(),${heading})]
```

### 4.3.12 Wait for data loading

Waiting for loading data from server.

```
Wait Until Keyword Succeeds 10 sec 0.1 sec Element Should Not Be Visible css=.alert
```

#### 4.3.13 Click on \${text} in \${state} state and wait effect

Switch state filter, wait, click on link in state block and wait.

```
Click on ${text} in ${state} Click link ${state}
Wait Until Keyword Succeeds 5 sec 0.1 sec Element Should Be Visible xpath=//*[@id="projects_
Click element xpath=//*[@id="projects_${state}"]//a[contains(text(),${text})]
Wait Until Keyword Succeeds 5 sec 0.1 sec Element Should Be Visible xpath=//*[@id="projects_
```

#### 4.3.14 Click on \${text} in projectnav

Click on link in project navigation.

```
Click on ${text} Click element xpath=//*[contains(@class,"projectnav")]//a[contains(text(),${text})]
Wait for data loading
```

#### 4.3.15 Start app

Start application.

```
${PRO_TEST} = Get Environment Variable PRO_TEST False
Run Keyword If '${PRO_TEST}' != 'True' Start Process bin/dev_appserver app --skip_sdk_update_
```

#### 4.3.16 Start Browser For Test

Start browser.

```
Start browser
Set Window Size 1024 768
```

#### 4.3.17 Close All Browsers and Report Status

Close browser and report test status.

```
Close All Browsers
Report test status
```

#### 4.3.18 Stop app

Stop application.

```
Run pkill -f dev_appserver
${out} = Read Process Output
Append to file app.log ${out}
Stop All Processes
```

### 4.3.19 Reload Page Until

Reload page 10 times until condition will be true.

```
: FOR ${try} IN RANGE 10
\   ${status}= Run Keyword And Ignore Error @{condition}
\   ${raise error}= Set Variable If ${status}[0]=='PASS' 'False' 'True'
\   Run Keyword If ${status}[0]=='PASS' Exit For Loop
\   Sleep 1 Waiting for loading server
\   Go to ${APP_URL}
Run Keyword If ${raise error}=="True" Fail "${condition}"
```

### 4.3.20 Test user login

Login by test user.

```
Reload Page Until Location should contain login
Input Text name=subdomain test
Input Text name=username test
Input Password name=password test
Click button Log in
Sleep 2 Waiting for loading app
Wait Until Page Contains First Last 15 sec
```

### 4.3.21 Test user logout

Logout.

```
Click link First Last
Click link Logout
```

### 4.3.22 Start browser

Prepare environment and start browser.

```
${BROWSER} = Get Environment Variable ROBOT_BROWSER Firefox
${REMOTE_URL} = Get Environment Variable ROBOT_REMOTE_URL ${REMOTE_URL}
${BUILD_MANUAL} = Evaluate random.randint(0,10**6) random
${BUILD_NUMBER} = Get Environment Variable ROBOT_BUILD_NUMBER manual-${BUILD_MANUAL}
${DESIRED_CAPABILITIES} = Get Environment Variable ROBOT_DESIRED_CAPABILITIES platform:Linux
${BUILD_TAGS} = Evaluate " ".join(${TEST_TAGS})
${BUILD_INFO} = Set variable build:${BUILD_NUMBER},name:${TEST_NAME},tags:${BUILD_TAGS},public
Open browser ${APP_URL} ${BROWSER} remote_url=${REMOTE_URL} desired_capabilities=${DESIRED_C
Run keyword unless '${REMOTE_URL}' == '' Run keyword and ignore error Set session id
```

### 4.3.23 Set session id

Get session id and set variable.

```
Keyword should exist Get session id
${SESSION_ID} = Get session id
Set test variable ${SESSION_ID} ${SESSION_ID}
```



#### 4.3.24 Report test status

Report test status to saucelabs.

```
Run keyword unless '${SESSION_ID}' == '' Report sauce status ${SESSION_ID} ${TEST_STATUS}
```

#### 4.3.25 an time report page

```
Given an bb-classic app
When click on "Time" link in navbar
Then wait for data loading
```

#### 4.3.26 an project time page

```
Given an bb-classic app
When click link Name of #0
And click on "Time" in projectnav
Then wait for data loading
```

#### 4.3.27 user add time entry

user can add time entry

```
Wait Until Page Contains Element css=tr.addtime button.additem
Input Text css=tr.addtime [name="description"] added description
Click Button css=tr.addtime button.additem
```

#### 4.3.28 page should contain added entry

added item should be present

```
Wait Until Page Contains added description
```

#### 4.3.29 user edit time entry

user can edit time entry

```
Wait Until Page Contains Element css=tr[data-id="0"] button.edit
Click Button css=tr[data-id="0"] button.edit
Wait Until Page Contains Element css=tr.edittime[data-id="0"] [name="description"]
Input Text css=tr.edittime[data-id="0"] [name="description"] edited description
Click Button css=tr[data-id="0"] button.save
```

#### 4.3.30 page should contain edited entry

edited item should be present

```
Wait Until Page Contains edited description
```

### 4.3.31 user delete time entry

user can delete time entry

```
Wait Until Page Contains Element  css=tr[data-id="0"]  button.remove
Click Button  css=tr[data-id="0"]  button.remove
```

### 4.3.32 page should not contain deleted entry

deleted item should not be present

```
Page should not contain  edited description
```

### 4.3.33 user sort time entries

sort time entries by date

```
Click Element  xpath=//table/thead/tr/th[contains(text(), "date")]
```

### 4.3.34 page should contain sorted entries

time entries must be sorted

```
Page should contain  2012-12-24
Page should not contain  2001-01-01
```

### 4.3.35 an to-dos page

**an to-dos page**

```
Given an bb-classic app
When click on "To-Dos" link in navbar
Then wait for data loading
And heading should be "My to-dos"
```

### 4.3.36 an project todos page

```
Given an bb-classic app
When click link  Name of #0
And click on "To-Dos" in projectnav
Then wait for data loading
```

### 4.3.37 user add todo list

user can add todo list

```
Click button  Add an item
Input Text  todoName  added todo list
Input Text  todoDescription  added description
Click Element  css=.additem
```

#### 4.3.38 page should contain added todo list

added item should be present

```
Wait Until Page Contains  added todo list
Wait Until Page Contains  added description
```

#### 4.3.39 user edit todo list

user can edit todo list

```
Click Element  css=.todolist.edititem[data-id="30"]
Input Text    todoName30  edited todo list
Input Text    todoDescription30  edited description
Click Element  css=.save
```

#### 4.3.40 page should contain edited todo list

edited item should be present

```
Wait Until Page Contains  edited todo list
Wait Until Page Contains  edited description
```

#### 4.3.41 user delete todo list

user can delete todo list

```
Click Element  css=.todolist.removeitem[data-id="30"]
```

#### 4.3.42 page should not contain deleted todolist

deleted item should not be present

```
Page should not contain  edited todo list
Page should not contain  edited description
```

#### 4.3.43 user add todo item

user can add todo item

```
Click button  Add an item
Input Text    todoContent  added todo item
Click Element  css=.additem
```

#### 4.3.44 page should contain added todo item

added item should be present

```
Wait Until Page Contains  added todo item
```

#### 4.3.45 user edit todo item

user can edit todo item

```
Click Element  css=.todo.edititem[data-id="30"]
Input Text    todoContent30  edited todo item
Click Element  css=.save
```

#### 4.3.46 page should contain edited todo item

edited item should be present

```
Wait Until Page Contains  edited todo item
```

#### 4.3.47 user delete todo item

user can delete todo item

```
Click Element  css=.todo.removeitem[data-id="30"]
```

#### 4.3.48 page should not contain deleted todoitem

deleted item should not be present

```
Page should not contain  edited todo item
```

#### 4.3.49 an project calendar page

```
Given an bb-classic app
When click link  Name of #0
And click on "Calendar" in projectnav
Then wait for data loading
```

#### 4.3.50 user edit calendar entry

user can edit calendar entry

```
Wait Until Page Contains Element  css=.edititem[data-id="0"]
Click Element  css=.edititem[data-id="0"]
Wait Until Page Contains Element  css=.editcalendar[data-id="0"] [name="title"]
Input Text    css=.editcalendar[data-id="0"] [name="title"]  edited title
Click Button  css=[data-id="0"] button.save
```

#### 4.3.51 page should contain edited entry

edited item should be present

```
Wait Until Page Contains  edited title
```

### 4.3.52 user delete calendar entry

user can delete calendar entry

```
Wait Until Page Contains Element  css=.removeitem[data-id="0"]
Click Element  css=.removeitem[data-id="0"]
```

### 4.3.53 page should not contain deleted entry

deleted item should not be present

```
Page should not contain  edited title
```

### 4.3.54 Keywords for tests

`keywords.compare_screenshot_to_base (baseline, diff=100)`

Calculate the exact difference between two images.

#### Parameters

- **baseline** (*string*) – [required] base screenshot to compare
- **diff** (*int*) – value of maximum difference

Example:

```
Compare screenshot to base  base_screenshot.jpg
```

`keywords.get_session_id()`

Get session id

**Returns** session id

**Return type** string

Example:

```
${SESSION_ID} =  Get Session Id
```

`keywords.report_sauce_status (job_id, test_status)`

Report test status to Sauce service

#### Parameters

- **job\_id** (*string*) – [required] saucelabs job id
- **test\_status** (*string*) – [required] status of test

**Returns** request status code

**Return type** int or string

Example:

```
Report sauce status  ${SESSION_ID}  ${TEST_STATUS}
```

## 4.4 Acceptance tests settings

**Library** `Selenium2Library` `timeout=15` `implicit_wait=0.1`

**Resource** `keywords.txt`

**Suite Setup** `Start app`

**Suite Teardown** `Stop app`

**Test Setup** `Start Browser For Test`

**Test Teardown** `Close All Browsers and Report Status`

## 4.5 Acceptance tests variables

```
${PORT} = 8080
${DOMAIN} = localhost
${APP_URL} = http://${DOMAIN}:${PORT}
${base_path} = docs/
${base_login} = ${base_path}login.png
${base_logout} = ${base_path}logout.png
${base_mypage} = ${base_path}mypage.png
${base_projects} = ${base_path}projects.png
${base_companies} = ${base_path}companies.png
${base_company} = ${base_path}company.png
${base_time_report} = ${base_path}time_report.png
${base_todos} = ${base_path}todos.png
${base_people} = ${base_path}people.png
${base_person} = ${base_path}person.png
${base_project_overview} = ${base_path}project_overview.png
${base_project_messages} = ${base_path}project_messages.png
${base_project_todos} = ${base_path}project_todos.png
${base_project_calendar} = ${base_path}project_calendar.png
${base_project_time} = ${base_path}project_time.png
${base_project_files} = ${base_path}project_files.png
${base_project_categories} = ${base_path}project_categories.png
${base_project_people} = ${base_path}project_people.png
${base_project_message} = ${base_path}project_message.png
${base_project_todolist} = ${base_path}project_todolist.png
${base_project_todo} = ${base_path}project_todo.png
${base_project_calendar_entry} = ${base_path}project_calendar_entry.png
${base_project_file} = ${base_path}project_file.png
${base_project_category} = ${base_path}project_category.png
${base_project_person} = ${base_path}project_person.png
${base_project_message_comments} = ${base_path}project_message_comments.png
${base_project_calendar_entry_comments} = ${base_path}project_calendar_entry_comments.png
${base_project_todo_comments} = ${base_path}project_todo_comments.png

${REMOTE_URL} =
${SESSION_ID} =
```

---

## Indices and tables

---

- *genindex*
- *modindex*
- *search*





## **b**

bb, [5](#)

## **c**

crypto, [10](#)

## **k**

keys, [8](#)

keywords, [25](#)