stakeholders

Nov 16, 2019

Contents:

1 Introduction **1**

2 Installing stakeholders **3**

3 Starting stakeholders **5**

4 Using stakeholders **7**

5 Notes on stakeholders **11**

6 Testing stakeholders **13**

7 stakeholders **15**

8 Indices **21**

Python Module Index **23**

Index **25**

CHAPTER 1

Introduction

Stakeholder management is critical to project management. Inspired by the Project Management Body of Knowledge, this app helps project teams to perform the principal tasks associated with managing stakeholders.

CHAPTER 2

Installing stakeholders

stakeholders is available on GitHub at https://github.com/critical-path/stakeholders.

If you do not have pip version 18.1 or higher, then run the following command from your shell.

```
[user@host ~]$ sudo pip install --upgrade pip
```

To install stakeholders with test-related dependencies, run the following commands from your shell.

```
[user@host ~]$ git clone git@github.com:critical-path/stakeholders.git
[user@host ~]$ cd stakeholders
[user@host stakeholders]$ sudo pip install --editable .[test]
```

To install it without test-related dependencies, run the following commands from your shell.

```
[user@host ~]$ git clone git@github.com:critical-path/stakeholders.git
[user@host ~]$ cd stakeholders
[user@host stakeholders]$ sudo pip install .
```

(If necessary, replace `pip` with `pip3`.)

CHAPTER 3

Starting stakeholders

To start stakeholders, run the following commands from your shell.

```
[user@host stakeholders]$ chmod +x ./start-api-and-app.sh
[user@host stakeholders]$ ./start-api-and-app.sh
```

CHAPTER 4

Using stakeholders

Using stakeholders is easy!

4.1 View home page

Point your browser to any of the following URLs.

- `http://localhost:8080/`
- `http://localhost:8080/home`
- `http://localhost:8080/index`

4.2 Add stakeholders

In the navbar, select `stakeholders` and then `add`.

Use the form to identify and evaluate a stakeholder. Enter the stakeholder's name, role, sentiment toward your project, level of power, and level of interest in your project. Then click on the `add` button.

Repeat this process as many times as necessary.

4.3 Show stakeholders

In the navbar, select `stakeholders` and then `show/update/delete`.

Your stakeholders appear here, sorted by their unique identifiers. Each stakeholder has a management approach, which is a function of that stakeholder's levels of power and interest.

Approach	Power	Interest	Flag
Monitor closely	high	high	red
Keep satisfied	high	low	orange
Keep informed	low	high	green
Monitor	low	low	blue
Unknown	invalid	invalid	gray

4.4 Add deliverables

In the navbar, select `deliverables` and then `add`.

Use the form to identify a deliverable, where a deliverable is a reporting requirement. Enter the deliverable's name, kind (type), medium, level of formality, and frequency. Then click on the `add` button.

Repeat this process as many times as necessary.

4.5 Show deliverables

In the navbar, select `deliverables` and then `show/update/delete`.

Your deliverables appear here, sorted by their unique identifiers.

4.6 Add associations

In the navbar, select `associations` and then `add`.

Use the form to identify an association, where an association is the assignment of a deliverable to a stakeholder. Enter a stakeholder and a deliverable appropriate for that stakeholder. Then click on the `add` button.

Repeat this process as many times as necessary. (It is possible to assign multiple deliverables to the same stakeholder as well as to assign the same deliverable to multiple stakeholders.)

4.7 Show associations

In the navbar, select `associations` and then `show/update/delete`.

Your associations appear here, sorted by their unique identifiers.

4.8 View management plan

In the navbar, select `management-plan`.

Your associations appear here, sorted first by management approach, then by stakeholders' unique identifiers, and then by deliverables' unique identifiers.

This is your stakeholder management plan - the whole purpose of this app!

4.9 Make updates

In the navbar, select `stakeholders`, `deliverables`, or `associations` and then `show/update/delete`. Find a stakeholder, deliverable, or association and then click on its `update` or `delete` button.

CHAPTER 5

Notes on stakeholders

stakeholders does not enforce constraints on the uniqueness of stakeholders, deliverables, or associations. This is to avoid unnecessary complexity in the code.

CHAPTER 6

Testing stakeholders

To execute tests that do not require a web browser, run the following command from your shell.

```
[user@host stakeholders]$ pytest -m "not browser" --cov --cov-report=term-missing
```

To execute tests that require a web browser, run the following commands from your shell. (Be sure to install Firefox and geckodriver first, however.)

```
[user@host stakeholders]$ chmod +x ./run-browser-tests.sh
[user@host stakeholders]$ ./run-browser-tests.sh
```

CHAPTER 7

stakeholders

7.1 stakeholders package

7.1.1 stakeholders.api module

This module contains the API and API factory.

`stakeholders.api.`**`create_api`**(*file='./stakeholders.sqlite3'*)

This is the API factory. It returns an instance of the API.

Parameters **`file`** (*`str`*) – The name of the database file (*stakeholders.sqlite3* by default).

Returns **api** – An instance of the API as a Flask application object.

Return type Flask application object

7.1.2 stakeholders.app module

This module contains the app and app factory.

`stakeholders.app.`**`create_app`**(*requester=<module 'requests' from '/home/docs/checkouts/readthedocs.org/user_builds/stakeholders/envs/stable/lib/python3.7/si packages/requests/__init__.py'>*, *host='localhost'*, *port='8079'*)

This is the app factory. It returns an instance of the app.

The app is the intermediary between the API and the end-user's web browser.

Parameters

- **`requester`** (*`Python library`*) – The library used to send requests to the API (*requests* for production and *flask.testing.FlaskClient* for testing).

  We need this contrivance only so we can use Flask's *testing* module for unit testing.

- **`host`** (*`str`*) – The name of the host on which the API runs (*localhost* by default).

  The app sends requests to the API's endpoints, and this forms part of the base URI.

- **port** (*str*) – The port on which the API listens (*8079* by default).

  The app sends requests to the API's endpoints, and this forms part of the base URI.

Returns **app** – An instance of the app as a Flask application object.

Return type Flask application object

7.1.3 stakeholders.db module

This module contains the Database class.

class stakeholders.db.**Database**(*file='stakeholders.sqlite3'*)
Bases: object

The *Database* class.

open()
Opens a connection to the database.

close()
Closes the connection to the database.

create_stakeholders_table()
Creates the *stakeholders* table.

create_deliverables_table()
Creates the *deliverables* table.

create_associations_table()
Creates the *associations* table.

drop_stakeholders_table()
Drops the *stakeholders* table.

drop_deliverables_table()
Drops the *deliverables* table.

drop_associations_table()
Drops the *associations* table.

insert_into_stakeholders_table(*name=None*, *role=None*, *sentiment=None*, *power=None*, *interest=None*, *approach=None*)
Inserts a record into the *stakeholders* table, where the record represents a stakeholder.

Parameters

- **name** (*str*) – The stakeholder's name.
- **role** (*str*) – The stakeholder's role.
- **sentiment** (*str*) – The stakeholder's sentiment.
- **power** (*str*) – The stakeholder's level of power.
- **interest** (*str*) – The stakeholder's level of interest.
- **approach** (*str*) – The approach to managing this stakeholder.

insert_into_deliverables_table(*name=None*, *kind=None*, *medium=None*, *formality=None*, *frequency=None*)
Inserts a record into the *deliverables* table, where the record represents a deliverable.

Parameters

- **`name`** (*`str`*) – The deliverable's name.
- **`kind`** (*`str`*) – The deliverable's kind.
- **`medium`** (*`str`*) – The deliverable's medium.
- **`formality`** (*`str`*) – The deliverable's level of formality.
- **`frequency`** (*`str`*) – The deliverable's frequency.

`insert_into_associations_table`(*stakeholder_id=None*, *deliverable_id=None*)
Inserts a record into the *associations* table, where a record represents a relationship between a stakeholder and a deliverable.

Parameters

- **`stakeholder_id`** (*`int`*) – A stakeholder's unique *id* as found in the *stakeholders* table.
- **`deliverable_id`** (*`int`*) – A deliverable's unique *id* as found in the *deliverables* table.

`select_all_from_stakeholders_table`()
Selects all records and fields from the *stakeholders* table.

Returns A list of records.

Return type list

`select_all_from_deliverables_table`()
Selects all records and fields from the *deliverables* table.

Returns A list of records.

Return type list

`select_all_from_associations_table`()
Selects all records and fields from the *associations* table.

Returns A list of records.

Return type list

`select_from_stakeholders_table`(*id=None*)
Selects one record and all fields from the *stakeholders* table.

Parameters **`id`** (*`str`*) – A stakeholder's unique *id* as found in the *stakeholders* table.

Returns A list with one record.

Return type list

`select_from_deliverables_table`(*id=None*)
Selects one record and all fields from the *deliverables* table.

Parameters **`id`** (*`int`*) – A deliverable's unique *id* as found in the *deliverables* table.

Returns A list with one record.

Return type list

`select_from_associations_table`(*id=None*)
Selects one record and all fields from the *associations* table.

Parameters **`id`** (*`int`*) – An association's unique *id* as found in the *associations* table.

Returns

Return type A list with one record.

`update_stakeholders_table`(*id=None*, *name=None*, *role=None*, *sentiment=None*, *power=None*, *interest=None*, *approach=None*)

Updates a record in the *stakeholders* table.

Parameters

- **`id`** (`str`) – A stakeholder's unique *id* as found in the *stakeholders* table.
- **`name`** (`str`) – The stakeholder's name.
- **`role`** (`str`) – The stakeholder's role.
- **`sentiment`** (`str`) – The stakeholder's sentiment.
- **`power`** (`str`) – The stakeholder's level of power.
- **`interest`** (`str`) – The stakeholder's level of interest.
- **`approach`** (`str`) – The approach to managing this stakeholder.

`update_deliverables_table`(*id=None*, *name=None*, *kind=None*, *medium=None*, *formality=None*, *frequency=None*)

Updates a record in the *deliverables* table.

Parameters

- **`id`** (`int`) – A deliverable's unique *id* as found in the *deliverables* table.
- **`name`** (`str`) – The deliverable's name.
- **`kind`** (`str`) – The deliverable's kind.
- **`medium`** (`str`) – The deliverable's medium.
- **`formality`** (`str`) – The deliverable's level of formality.
- **`frequency`** (`str`) – The deliverable's frequency.

`update_associations_table`(*id=None*, *stakeholder_id=None*, *deliverable_id=None*)

Updates a record in the *associations* table.

Parameters

- **`id`** (`int`) – An association's unique *id* as found in the *associations* table.
- **`stakeholder_id`** (`int`) – A stakeholder's unique *id* as found in the *stakeholders* table.
- **`deliverable_id`** (`int`) – A deliverable's unique *id* as found in the *deliverables* table.

`delete_from_stakeholders_table`(*id=None*)

Deletes a record from the *stakeholders* table.

Parameters **`id`** (`str`) – A stakeholder's unique *id* as found in the *stakeholders* table.

`delete_from_deliverables_table`(*id=None*)

Deletes a record from the *deliverables* table.

Parameters **`id`** (`str`) – A deliverable's unique *id* as found in the *deliverables* table.

`delete_from_associations_table`(*id=None*)

Deletes a record from the *associations* table.

Parameters **`id`** (`str`) – An association's unique *id* as found in the *associations* table.

7.1.4 stakeholders.utils module

This module contains utils.

`stakeholders.utils.`**`compute_approach`**`(`*power*, *interest*`)`

Computes the approach to managing a stakeholder according to the power/interest model.

Parameters

- **`power`** (*`str`*) – The stakeholder's level of power, either *high* or *low*.
- **`interest`** (*`str`*) – The stakeholder's level of interest, either *high* or *low*.

Returns The approach to managing this stakeholder: *monitor closely*, *keep satisfied*, *keep informed*, *monitor*, or *unknown*.

Return type str

`stakeholders.utils.`**`transform`**`(`*record*`)`

Transforms (maps) a record.

Parameters **`record`** (*`dict`*) – The record to transform.

Returns The transformed record.

Return type dict

`stakeholders.utils.`**`recursive_fold`**`(`*left*, *right*`)`

Recursively folds (reduces) two records.

Parameters

- **`left`** (*`dict`*) – The "left-hand", or current, record.
- **`right`** (*`dict`*) – The "right-hand", or next, record.

Returns **left** – The folded (reduced) record.

Return type dict

`stakeholders.utils.`**`map_reduce`**`(`*records*`)`

Maps and reduces a list of records.

Calls the *transform* and *recursive_fold* functions.

Parameters **`records`** (*`list of dicts`*) – The records to map and reduce.

Returns The mapped and reduced records.

Return type dict

`stakeholders.utils.`**`sort`**`(`*records*`)`

Sorts records by the approach to managing stakeholders.

Parameters **`records`** (*`dict`*) – The records to sort.

Returns The sorted records.

Return type OrderedDict

7.1.5 Module contents

Manage stakeholders like a pro!

CHAPTER 8

Indices

- genindex
- modindex
- search

Python Module Index

S

stakeholders, 19
stakeholders.api, 15
stakeholders.app, 15
stakeholders.db, 16
stakeholders.utils, 19

Index

C

close() (*stakeholders.db.Database method*), 16
compute_approach() (*in module stakeholders.utils*), 19
create_api() (*in module stakeholders.api*), 15
create_app() (*in module stakeholders.app*), 15
create_associations_table() (*stakeholders.db.Database method*), 16
create_deliverables_table() (*stakeholders.db.Database method*), 16
create_stakeholders_table() (*stakeholders.db.Database method*), 16

D

Database (*class in stakeholders.db*), 16
delete_from_associations_table() (*stakeholders.db.Database method*), 18
delete_from_deliverables_table() (*stakeholders.db.Database method*), 18
delete_from_stakeholders_table() (*stakeholders.db.Database method*), 18
drop_associations_table() (*stakeholders.db.Database method*), 16
drop_deliverables_table() (*stakeholders.db.Database method*), 16
drop_stakeholders_table() (*stakeholders.db.Database method*), 16

I

insert_into_associations_table() (*stakeholders.db.Database method*), 17
insert_into_deliverables_table() (*stakeholders.db.Database method*), 16
insert_into_stakeholders_table() (*stakeholders.db.Database method*), 16

M

map_reduce() (*in module stakeholders.utils*), 19

O

open() (*stakeholders.db.Database method*), 16

R

recursive_fold() (*in module stakeholders.utils*), 19

S

select_all_from_associations_table() (*stakeholders.db.Database method*), 17
select_all_from_deliverables_table() (*stakeholders.db.Database method*), 17
select_all_from_stakeholders_table() (*stakeholders.db.Database method*), 17
select_from_associations_table() (*stakeholders.db.Database method*), 17
select_from_deliverables_table() (*stakeholders.db.Database method*), 17
select_from_stakeholders_table() (*stakeholders.db.Database method*), 17
sort() (*in module stakeholders.utils*), 19
stakeholders (*module*), 19
stakeholders.api (*module*), 15
stakeholders.app (*module*), 15
stakeholders.db (*module*), 16
stakeholders.utils (*module*), 19

T

transform() (*in module stakeholders.utils*), 19

U

update_associations_table() (*stakeholders.db.Database method*), 18
update_deliverables_table() (*stakeholders.db.Database method*), 18
update_stakeholders_table() (*stakeholders.db.Database method*), 17