Table of Contents

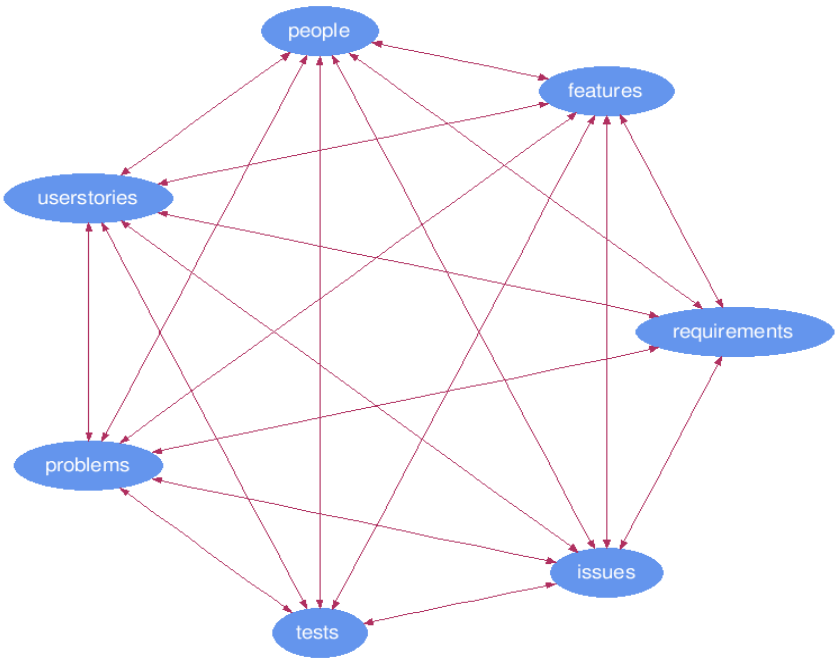
Table of Contents 1
ISSUE-TRACKER 2
1. WHY 2
2. WHAT 2
3. INSTALLATION AND CONFIGURATION 2
3.1. Prerequisites 2
3.2. Fetch the source 3
3.3. run the boot-strap script 3
3.4. Apply the db and issue create scripts 3
3.5. Install the required Perl modules 3
3.6. Start hacking 4
4. ADDITIONAL DOCS 4
5. AND FINALLY THE PROJECT STATUS 4

ISSUE-TRACKER

1. WHY

Why ?! Yet. Another. App ?!
Because work should be inspiring and not overwhelming people.
Because even good intentions without proper commitment, allocation and resourcing and most importantly a mean for tracking advancement of an endeavor in open way reflecting with the reality, might end-up making people less happy, when in fact a really simple solution could be applied for any bigger challenge requiring progress tracking and people communication and coordination... And tons of other reasons we all having been in project disasters know about ... Still here ?! Let's move on !

Figure: 1 issue tracker objects polygon



2. WHAT

An application to manage multiple projects' issues, store them in postgres db, present them into dynamic web pages, write them to txt files, xls files, and publish them into Google Sheets with a lot of cool functionalities.
The full and extensive features and functionalities document cold be read from:
https://github.com/YordanGeorgiev/issue-tracker/blob/master/doc/md/issue-tracker-features-and-functionalities.md
And moreover this application is the reflection of the best practices and principles for tens of years in IT resulting into a product of the Multi-environment instance architecture and the Input-Output Controller Model architecture (more about this in the DevOps guide:
https://github.com/YordanGeorgiev/issue-tracker/blob/master/doc/md/issue-tracker-features-and-functionalities.md

3. INSTALLATION AND CONFIGURATION

This section is the starting point for deploying a fully working instance of the issue-tracker. The installation has been automated as much as possible and the docs have been updated each time when the setup has been changed, so following up the instructions and implementing step by step top to bottom should bring you to a fully functional instance and if there are clear inconsistencies or anything blocking you your feedback will be highly appreciated !

3.1. Prerequisites

The must have binaries are:
bash, perl, zip,postgres 9.6
The nice to have are:
tmux, vim ,ctags
The examples are for Ubuntu - use your OS package manager ...
If you do not have postgres than you would have to follow the longer installation instructions :
https://github.com/YordanGeorgiev/issue-tracker/blob/master/doc/md/issue-tracker-devops-guide.md#1-installations-and-configurations

```
# use your OS package manager ... if you are not on Ubuntu

sudo apt-get autoclean
sudo apt-get install --only-upgrade bash

sudo apt-get install -y perl

# optionally
sudo apt-get install -y excuberant-ctags
sudo apt-get install -y 7z

sudo apt-get upgrade
```

3.2. Fetch the source

Fetch the source from git hub as follows:

```
# got to a dir you have write permissions , for example:
mkdir -p ~/opt/csitea/; cd ~/opt/csitea/

# fetch the source
git clone https://github.com/YordanGeorgiev/issue-tracker.git
# checkit
ls -la

# OR
wget https://github.com/YordanGeorgiev/issue-tracker/archive/0.4.4.zip
ls -al
unzip -o 0.4.4.zip -d .
```

3.3. run the boot-strap script

The bootstrap script will interpolate change the git deployment dir to a "product_instance_dir" (your instance of the issue-tracker, having the same version as this one, but running on a different host with different owner - your)

```
# defiine the latest and greates product_version
export product_version=$(cd issue-tracker;git tag|sort -nr| head -n 1;cd ..)

# check it
echo $product_version

# run the bootstrap script :
bash issue-tracker/src/bash/issue-tracker/bootstrap-issue-tracker.sh

# now go to your product instance dir ( yes this is official term ), note it is a DEV environment
cd /opt/csitea/issue-tracker/issue-tracker.$product_version.dev.$USER
```

3.4. Apply the db and issue create scripts

If you do not have the PostgreSQL (v9.5 >) with current Linux user configured role installed check the instructions in the installations and configurations section of the DevOps guide:
https://github.com/YordanGeorgiev/issue-tracker/blob/master/doc/md/issue-tracker-devops-guide.md#1-installations-and-configurations
If you do have it , apply the db and issue create scripts as follows:

```
# apply the postgre sql scripts
bash src/bash/issue-tracker/issue-tracker.sh -a run-pgsql-scripts
```

3.5. Install the required Perl modules

Just run the prerequisites checker script which will provide you with instruction you could just copy paste.

```
sudo perl src/perl/issue_tracker/script/issue_tracker_preq_checker.pl

# after installing all the modules check the perl syntax of the whole project:
bash src/bash/issue-tracker/issue-tracker.sh -a check-perl-syntax
```

3.6. Start hacking

Start usage:

```
doParseIniEnvVars cnf/issue-tracker-issues.dev.doc-pub-host.cnf

# run all the uncommented out test actions listed in the
# src/bash/issue-tracker/tests/run-issue-tracker-tests.lst file

bash src/bash/issue-tracker/test-issue-tracker.sh
```

4. ADDITIONAL DOCS

Additional docs could be found in the doc/md dir.
A good starting point could be the "features doc:" :
https://github.com/YordanGeorgiev/issue-tracker/blob/master/doc/md/issue-tracker-features-and-functionalities.md d

5. AND FINALLY THE PROJECT STATUS

The issue tracker project status could be tracked by the issue-tracker data stored in the following gsheet:
https://docs.google.com/spreadsheets/d/e/2PACX-1vQ3ijqJkY03mDXiaT3mcvr96NkgRnsONSAHyBGwnukuRezhHTaAZsUxOcoQ6fHfZmcHXP2KpD6kfCPR/pubhtml