Application Objective

WiSStack Dashboard App is a web application for so-called ticketing. It is used in IT, construction and programming industries, among others. The application streamlines the reporting of problems and their immediate handling. In the near future we plan to introduce chat applications. Administration and assignment of profiles to given companies. WiSStack Dashboard app is easy to im

plement and use.

Installation

UNZIP the sources or clone the repository. After getting the code, open a terminal and navigate to the working directory, with product source code. Or use bash:

$ git clone <https://github.com/igorwisniewski/Djangoapp>

$cd Djangoapp

Then you must install modules via ‘venv’:

$ virtualenv env

$ source env/bin/activate

$ pip3 install -r requiraments.txt

Set up Database:

$ python manage.py makemigrations

$ python manage.py migrate

Start the app:

$python manage.py runserver

At this point, the app runs at `http://127.0.0.1:8000/`.

Successes and failures

Beginning with successes met general assumptions such as logging and registration, creating, assigning, displaying, accepting and updating tickets. Creating a comfortable and favorable UX/UI. Failures: failure to create conversations that automatically update charts and search by page. Which I plan to add soon. This project I hope I will be able to cure and make it fully usable for private and small business use.

Models

from django.db import models

from django.contrib.auth.models import AbstractUser

class User(AbstractUser):

is\_enginer = models.BooleanField(default=False)

is\_customer = models.BooleanField(default=False)

def \_\_str\_\_(self):

return self.username

A user model. I Tried to keep it pretty simple

import uuid

from django.db import models

from users.models import User

# Create your models here.

class Ticket(models.Model):

status\_choice = (

('Active', 'Active'),

('Completed', 'Completed'),

('Pending', 'Pending')

)

ticket\_number = models.UUIDField(default= uuid.uuid4)

title = models.CharField(max\_length=100)

description = models.TextField()

created\_by = models.ForeignKey(User, on\_delete = models.CASCADE, related\_name='created\_by')

date\_created = models.DateTimeField(auto\_now\_add=True)

assinged\_to = models.ForeignKey(User, on\_delete = models.DO\_NOTHING, null = True, blank =True)

is\_resolved = models.BooleanField(default = False)

accepted\_data = models.DateTimeField(null=True, blank=True)

closed\_date = models.DateTimeField(null=True, blank=True)

ticket\_status = models.CharField(max\_length=15, choices =status\_choice)

def \_\_str\_\_(self):

return self.title

A ticket model is not very complicated\