Reference Links:

http://stackoverflow.com/questions/1832709/django-how-to-make-translation-work
https://docs.djangoproject.com/en/1.3/topics/i18n/deployment/

1. Install rosetta , vinaigrette(App for translating database data)

   Note : If you are in virtual environment do not proceed any installation command with sudo because sudo installs packages globally

   ```
   $sudo pip install django-rosetta
   $sudo pip install vinaigrette
   ```

2. Add " rosetta" to INSTALLED_APPS in settings.py file

3. Add following middlewear to MIDDLEWARE_CLASSES in settings.py

   ```
   'django.middleware.locale.LocaleMiddleware',
   ```

   Note : This middleware should be after
   'django.contrib.sessions.middleware.SessionMiddleware',

4. Add languages to be used to settings.py

   ```
   from django.utils.translation import ugettext_lazy as _
   LANGUAGES = (
           ('es', _('Spanish')),
           ('en', 'English'),
           ('gl', _('Galician')),
           ('kn', _('Kannada')),
           ('fr', _('french')),
           )
   ```

5. Make a folder named "locale" in root directory or appdirectory.
   this folder name should be always lower case
   Add path of this local folder to settings.py file with variable name as

   ```
   LOCALE_PATHS = (
                        '/home/mahiti/Django_Projects/lang_test/locale/',
                   )
   ```

6. Add following lines in urls.py

   ```
   url(r'^rosetta/', include('rosetta.urls')),
   url(r'^i18n/', include('django.conf.urls.i18n')),
   ```

7.In all (.html) files or base.html file add following tag at the starting of that file below extends tag (if exists)

```
{% load i18n %}
{% extends “base.html” %}
```

8. For Translating template(.html) strings use {% trans “<string to be translated>” %}
   Ex: {% trans "Welcome to Translations" %}

9. For translating strings from .py files use gettext or ugettext_lazy for that specific strings

```
from django.utils.translation import ugettext_lazy as _
error_message = _(“ Amount greater than basic amount ”)
```

10. To Translate database data add following line in models.py

```
import vinaigrette
from django.utils.translation import ugettext_lazy as _


vinaigrette.register(<model-name>,[“field_name1”, ”filed_name2” , .....]

Ex: class Profile(models.Model) :
            name = models.CharField(max_length=100 ,verbose_name = _(“name”))
           surname = models.CharField(max_length=100 ,verbose_name = _(“surname”))
```

11. To transalate string in views use ugettext_lazy function as

```
from django.utils.translation import ugettext_lazy as _
def sampleview(requset):
        error_msg = _(“Invaid Login”)
```

Note : Simlary forms.py or any .py file

13. Open Terminal cd to project root directory
    To compile messages

```
$django-admin.py makemessages -l kn (or)
$pyhton manage.py makemessages -l kn
```

Note : a) kn - language code which specified in LANGUAGES variable in setting.py file

b) This command parses (crawl) all .html and .py files and extract all the strings

which we have marked for translations using ugettext(.py files),
trans(.html files) and gettext(used for Java Script translations in .html files)
and stores all strings in django.po file as

```
msgid "Welcome to Translations"
msgstr "ಅನುವಾದಗಳನ್ನು ಸ್ವಾಗತ"
```

Where msgid is the string marked for translations
and msgstr is the translated message to that specific language
(Initially msgstr is empty)

c) In order to translate this strings without opening .po files use rosetta app
This app provides a UI from admin to provide strings to be translated.
Link to use rosetta is localhost:8000/rosetta/

d) In order to use rosetta , admin should be logged in as .po files contains
data marked for translations from database

14. To compile messages for all languages use command

```
$django-admin.py compilemessages
```

This command generates .mo files from .po files which contains Byte code.
This files are used by django while translations

15. To provide select box in templates for different languages add the following code

```
<div class="searchform">
        <form action="/i18n/setlang/" method="post">{% csrf_token %}
        <input name="next" type="hidden" value="{{ redirect_to }}" />
        <select name="language">
                {% get_language_info_list for LANGUAGES as languages %}
                {% for language in languages %}
                <option value="{{ language.code }}">{{ language.name }} ({{ language.code
}})</option>
                {% endfor %}
        </select>
        <input type="submit" value="{% trans 'GO'%}" />
        </form>
</div>
```

This select box displays all the languages specified in LANGUAGES in settings.py file

16. How these translations work ?

1. The directories listed in LOCALE_PATHS have the highest precedence, with the ones appearing first having higher precedence than the ones appearing later.
2. Then, it looks for and uses if it exists a locale directory in each of the installed apps listed in INSTALLED_APPS. The ones appearing first have higher precedence than the ones appearing later.
3. Then, it looks for a locale directory in the project directory, or more accurately, in the directory containing your settings file.
4. Finally, the Django-provided base translation in django/conf/locale is used as a fallback.
5. When user selects language and clicks on GO ,control goes to url "/i18n/setlang/" and again to local middleware and returns temporary redirect (302 response)
6. Local middleware makes use of session data so local middleware should be after session middleware in settings.py file