**models.py – user**

***class*models.User**

**username** Required. 30 characters or fewer. Usernames may contain alphanumeric, \_, @, +, . and - characters.

**first\_name** Optional. 30 characters or fewer.

**last\_name** Optional. 30 characters or fewer.

**email** Optional. Email address.

**password** Required. A hash of, and metadata about, the password. (Django doesn’t store the raw password.) Raw passwords can be arbitrarily long and can contain any character.

**groups** Many-to-many relationship to [Group](https://docs.djangoproject.com/en/1.6/ref/contrib/auth/#django.contrib.auth.models.Group)

**user\_permissions** Many-to-many relationship to [Permission](https://docs.djangoproject.com/en/1.6/ref/contrib/auth/#django.contrib.auth.models.Permission)

**is\_staff** Boolean. Designates whether this user can access the admin site.

**is\_active** Boolean. Designates whether this user account should be considered active. We recommend that you set this flag to Falseinstead of deleting accounts; that way, if your applications have any foreign keys to users, the foreign keys won’t break.

**is\_superuser** Boolean. Designates that this user has all permissions without explicitly assigning them.

**last\_login** A datetime of the user’s last login. Is set to the current date/time by default.

**date\_joined** A datetime designating when the account was created. Is set to the current date/time by default when the account is created.

**models.py – user\_profiles**

from django.db import models

from django.contrib.auth.models import User

class UserProfile(models.Model):

username = models.OneToOneField(User)

user\_phone = models.CharField(max\_length=12)

avatar = models.ImageField(upload\_to='userprofiles', blank=True)

calification = models.SmallIntegerField(default=3)

**models.py – store\_requests**

from contacts.models import Contact

from django.db import models

from tabulators.models import Tabulator

from status\_types.models import Status\_Type

from addresses.models import Address

from django.contrib.auth.models import User

class Store(models.Model):

store\_name = models.CharField(max\_length=255)

activity = models.CharField(max\_length=255)

dimentions\_width = models.FloatField()

dimentions\_length = models.FloatField()

address = models.ForeignKey(Address)

store\_phone = models.CharField(max\_length=12)

image1 = models.ImageField(upload\_to='stores')

image2 = models.ImageField(upload\_to='stores', blank=True)

image3 = models.ImageField(upload\_to='stores', blank=True)

image4 = models.ImageField(upload\_to='stores', blank=True)

image5 = models.ImageField(upload\_to='stores', blank=True)

website = models.CharField(max\_length=255, blank=True)

facebook = models.CharField(max\_length=255, blank=True)

twitter = models.CharField(max\_length=255, blank=True)

youtube = models.CharField(max\_length=255, blank=True)

comments = models.TextField(blank=True)

contact = models.ForeignKey(Contact)

tabulator = models.ForeignKey(Tabulator)

status = models.ForeignKey(Status\_Type)

date\_created = models.DateField(null=True, blank=True)

date\_approved = models.DateField(null=True, blank=True)

username = models.ForeignKey(User)

wireless = models.BooleanField(default=False)

stands = models.BooleanField(default=False)

repisas = models.BooleanField(default=False)

boards = models.BooleanField(default=False)

lighting = models.BooleanField(default=False)

electricity = models.BooleanField(default=False)

water = models.BooleanField(default=False)

airconditioning = models.BooleanField(default=False)

toilets = models.BooleanField(default=False)

heating = models.BooleanField(default=False)

elevator = models.BooleanField(default=False)

parkinglot = models.BooleanField(default=False)

counter = models.BooleanField(default=False)

phoneline = models.BooleanField(default=False)

storehouse = models.BooleanField(default=False)

dressingroom = models.BooleanField(default=False)

others1 = models.BooleanField(default=False)

others2 = models.BooleanField(default=False)

others3 = models.BooleanField(default=False)

price = models.FloatField(blank=True)

**models.py – store\_requests**

from django.db import models

from stores.models import Store

from rent\_types.models import Rent\_Type

from status\_types.models import Status\_Type

from django.contrib.auth.models import User

from contacts.models import Contact

class Store\_Request(models.Model):

request\_code = models.CharField(max\_length=6)

store = models.ForeignKey(Store)

username = models.ForeignKey(User)

contact = models.ForeignKey(Contact)

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

rent\_type = models.ForeignKey(Rent\_Type)

rent\_price = models.FloatField(null=True, blank=True, default=0)

status\_req = models.ForeignKey(Status\_Type)

start\_date = models.DateField()

ending\_date = models.DateField()

**models.py – contacts**

from django.db import models

class Contact(models.Model):

firstname = models.CharField(max\_length=255)

lastname = models.CharField(max\_length=255)

contact\_phone = models.CharField(max\_length=12)

email = models.CharField(max\_length=255)

**models.py – tabulators**

from django.db import models

from towns.models import Town

class Tabulator(models.Model):

tab\_zone = models.CharField(max\_length=255)

town = models.ForeignKey(Town)

min\_price = models.IntegerField()

max\_price = models.IntegerField()

media\_price = models.IntegerField()

suggested\_price = models.IntegerField(blank=True)

**models.py – news**

from django.db import models

from django.contrib.auth.models import User

class News(models.Model):

news\_text = models.TextField()

username = models.ForeignKey(User)

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

**models.py – topups**

from django.db import models

class Topup(models.Model):

topup\_name = models.CharField(max\_length=255)

image1 = models.ImageField(upload\_to='topups')

image2 = models.ImageField(upload\_to='topups', blank=True)

image3 = models.ImageField(upload\_to='topups', blank=True)

**models.py – popular\_topups**

from django.db import models

from stores.models import Store

from topups.models import Topup

class Popular\_Topup(models.Model):

topup = models.ForeignKey(Topup)

store\_n = models.ManyToManyField('stores.Store')

priority = models.IntegerField()

**models.py – byzone\_topups**

from django.db import models

from stores.models import Store

from towns.models import Town

class Byzone\_Topup(models.Model):

zone\_name = models.CharField(max\_length=255)

store = models.ForeignKey(Store)

town = models.ForeignKey(Town)

priority = models.PositiveIntegerField()

**models.py – rent\_types**

from django.db import models

class Rent\_Type(models.Model):

name\_type = models.CharField(max\_length=255)

**models.py – status\_types**

from django.db import models

class Status\_Type(models.Model):

status\_name = models.CharField(max\_length=11)

**models.py – addresses**

from django.db import models

from postal\_codes.models import Postal\_Code

class Address(models.Model):

address\_line1 = models.CharField(max\_length=255)

neighborhood = models.CharField(max\_length=255)

postal\_code = models.ForeignKey(Postal\_Code)

**models.py – postal\_codes**

from django.db import models

from towns.models import Town

class Postal\_Code(models.Model):

postal\_code\_key = models.CharField(max\_length=5)

town = models.ForeignKey(Town)

**models.py – towns**

from django.db import models

from cities.models import City

class Town(models.Model):

town\_name = models.CharField(max\_length=255)

city = models.ForeignKey(City)

**models.py – cities**

from django.db import models

from states\_a.models import State

class City(models.Model):

city\_name = models.CharField(max\_length=255)

state = models.ForeignKey(State)

**models.py – states\_a**

from django.db import models

from countries\_a.models import Country

class State(models.Model):

state\_name = models.CharField(max\_length=255)

country = models.ForeignKey(Country)

**models.py – countries\_a**

from django.db import models

class Country(models.Model):

country\_name = models.CharField(max\_length=255)