

DATABASE

```
class Post(models.Model):
    title = models.CharField(max_length=150)
    content = RichTextField(blank=True, null=True)
    date_posted = models.DateTimeField(default=timezone.now)
    date_updated = models.DateTimeField(auto_now=True)
    author = models.ForeignKey(User, on_delete=models.CASCADE)
    likes = models.ManyToManyField(User, related_name="blogpost",
blank=True)
    saves = models.ManyToManyField(User, related_name="blogsave",
blank=True)

    def total_likes(self):
        return self.likes.count()

    def total_saves(self):
        return self.saves.count()

    def __str__(self):
        return self.title

    def get_absolute_url(self):
        return reverse('post-detail', kwargs={"pk": self.pk}) #pk is a
slug

""" Comment model """

class Comment(models.Model):
    post = models.ForeignKey(Post, related_name="comments",
on_delete=models.CASCADE)
    name = models.ForeignKey(User, on_delete=models.CASCADE)
    body = models.TextField(max_length=200)
    date_added = models.DateTimeField(auto_now_add=True)
    likes = models.ManyToManyField(User, related_name="blogcomment",
blank=True)
    reply = models.ForeignKey('self', null=True, related_name="replies",
on_delete=models.CASCADE)

    def total_clikes(self):
        return self.likes.count()

    def __str__(self):
        return '%s - %s - %s' % (self.post.title, self.name, self.id)

    def get_absolute_url(self):
        return reverse('post-detail', kwargs={"pk": self.pk})
```

```

class Room(models.Model):
    # room_id = models.UUIDField(primary_key=True, default=uuid.uuid4,
    # editable=False)
    room_id = models.AutoField(primary_key=True)
    author = models.ForeignKey(User, related_name='author_room',
    on_delete=models.CASCADE)
    friend = models.ForeignKey(User, related_name='friend_room',
    on_delete=models.CASCADE)
    created = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return f"{self.room_id}-{self.author}-{self.friend}"

class Chat(models.Model):
    room_id = models.ForeignKey(Room, on_delete=models.CASCADE,
    related_name='chats')
    author = models.ForeignKey(User, on_delete=models.CASCADE,
    related_name='author_msg')
    friend = models.ForeignKey(User, on_delete=models.CASCADE,
    related_name='friend_msg')
    text = models.CharField(max_length=300)
    date = models.DateTimeField(auto_now_add=True)
    has_seen = models.BooleanField(default=False)

    def __str__(self):
        return '%s - %s' % (self.id, self.date)

```

```

class FriendList(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE,
    related_name='user')
    friends = models.ManyToManyField(User, blank=True,
    related_name='friends')

    def __str__(self):
        return self.user.username

    def add_friend(self, account):
        if not account in self.friends.all():
            self.friends.add(account)
            self.save()

    def remove_friend(self, account):
        if account in self.friends.all():
            self.friends.remove(account)
            self.save()

    def unfriend(self, removee):
        remover_friends_list = self
        remover_friends_list.remove_friend(removee)

        friends_list = FriendList.objects.get(user=removee)

```

```

        friends_list.remove_friend(self.user)

    def is_mutual_friend(self, friend):
        if friend in self.friends.all():
            return True
        return False

""" Friend Request model """

class FriendRequest(models.Model):
    sender = models.ForeignKey(User, on_delete=models.CASCADE,
related_name='sender')
    receiver = models.ForeignKey(User, on_delete=models.CASCADE,
related_name='receiver')
    is_active = models.BooleanField(blank=True, null=True, default=True)
    timestamp = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return self.sender.username

    def accept(self):
        # update both sender and receiver friend list
        receiver_friend_list = FriendList.objects.get(user=self.receiver)
        if receiver_friend_list:
            receiver_friend_list.add_friend(self.sender)
        sender_friend_list = FriendList.objects.get(user=self.sender)
        if sender_friend_list:
            sender_friend_list.add_friend(self.receiver)
            self.is_active = False
            self.save()

    def decline(self):
        self.is_active = False
        self.save()

    def cancel(self):
        self.is_active = False
        self.save()

```

```

class Profile(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    is_online = models.BooleanField(default=False)
    following = models.ManyToManyField(User, related_name="following",
blank=True)
    friends = models.ManyToManyField(User, related_name='my_friends',
blank=True)
    bio = models.CharField(default="", blank=True, null=True,
max_length=350)
    date_of_birth = models.DateField(blank=True)
    updated = models.DateTimeField(auto_now=True)
    created = models.DateTimeField(auto_now_add=True)
    image = models.ImageField(default='default.jpg',
upload_to='profile_pics', blank=True, null=True)

```

```

def profile_posts(self):
    return self.user.post_set.all()

def get_friends(self):
    return self.friends.all()

def get_friends_no(self):
    return self.friends.all().count()

def __str__(self):
    return f'{self.user.username} Profile'

STATUS_CHOICES = (
    ('send', 'send'),
    ('accepted', 'accepted')
)


class Relationship(models.Model):
    sender = models.ForeignKey(Profile, on_delete=models.CASCADE,
related_name='friend_sender')
    receiver = models.ForeignKey(Profile, on_delete=models.CASCADE,
related_name='friend_receiver')
    status = models.CharField(max_length=8, choices=STATUS_CHOICES)
    updated = models.DateTimeField(auto_now=True)
    created = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return f"{self.sender}-{self.receiver}-{self.status}"

```

ADMIN LOGIN

127.0.0.1:3000/admin/login/meet=/admin/



Welcome to the library

admin

...

Log in

Activate Windows
Go to Settings to activate Windows.

solloquy

admin

Dashboard

Authentication and Authorization

Groups

Users

Accounts

Email addresses

Blog

Comments

Posts

Chat

Chats

Rooms

Friend

Friend lists

Friend requests

Notification

Notifications

Sites

Sites

Users

Profiles

Relationships

Recent actions

akhil Profile

Changed Image.

akhil Profile

Changed Image.

Swooping golden orbs

Changed Title.

I felt nothing but peace and happiness

Changed Title and Content.

I had the vision of seeing a white light

Changed Title and Content.

I felt my soul or something come right out of my body

Changed Title and Content.

Activate Windows
Go to Settings to activate Windows.