DB Frameworks

CSCI 220: Database Management and Systems Design

Today you will learn…

- The benefits of using a framework (Django) for web development

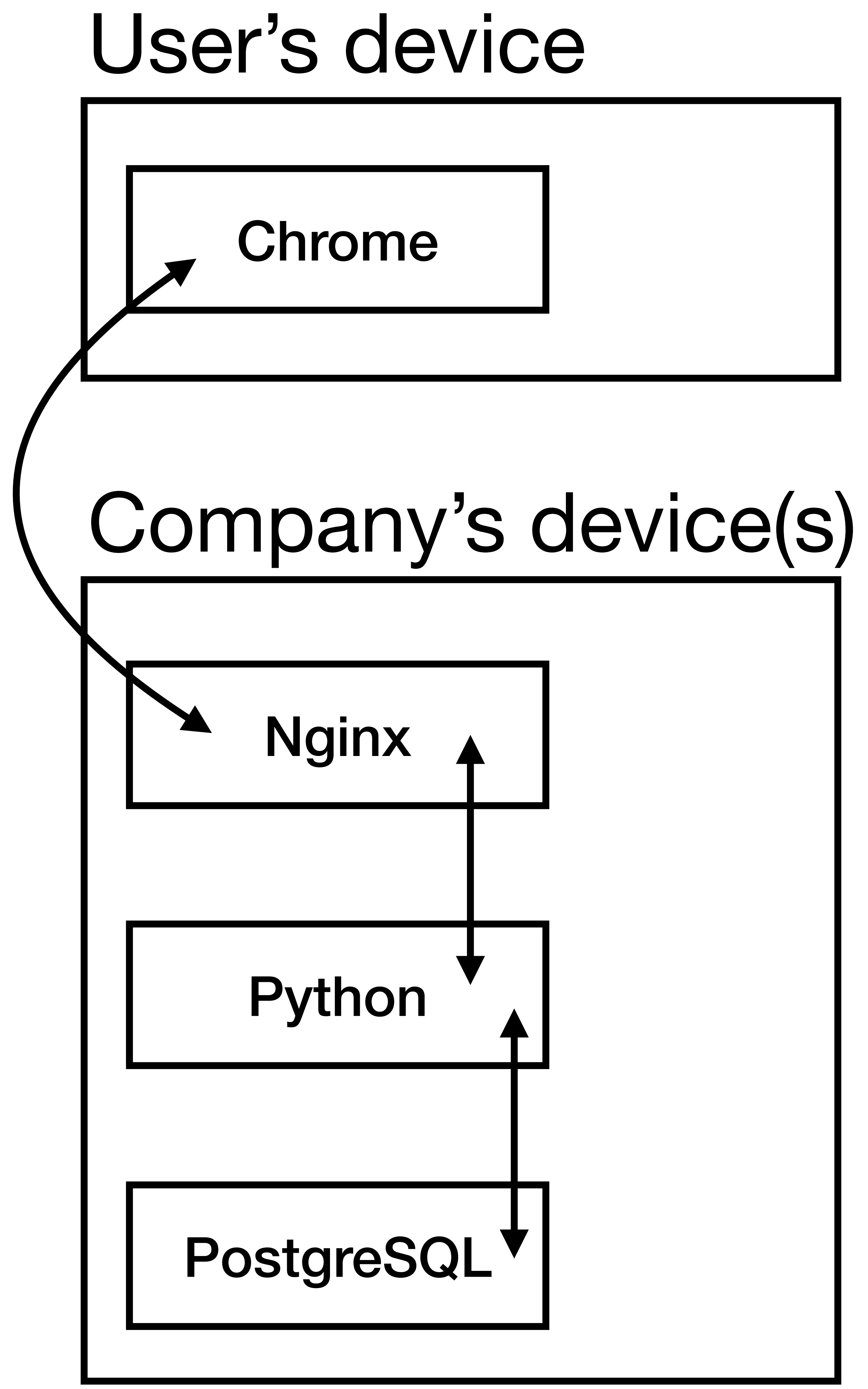
Outline

- Django and WSGI
- Object-relational mapping
- Django admin interface
- Migrations

Django and WSGI

Review: Web App Architecture

- Web browser (e.g., Chrome, Safari) requests pages and renders the application's graphics
- Web server (e.g., nginx, Apache) passes data between the browser and the application code
- Application code (e.g., Django) builds the HTML for dynamic pages, based on data from the database
- The database (e.g., PostgreSQL, MySQL) manages physical storage of the data



Writing a WSGI Web App

- Write a .py file with an `application()` method
- Your code handles all aspects of receiving and responding to web requests
- Pros:
  - Quick to get started
  - You write all the code, so you understand exactly what is happening
- Cons:
  - **Requires writing lots of code** (which will likely be unstructured and unmaintainable)
  - **Challenging to implement securely**

WSGI Hello World

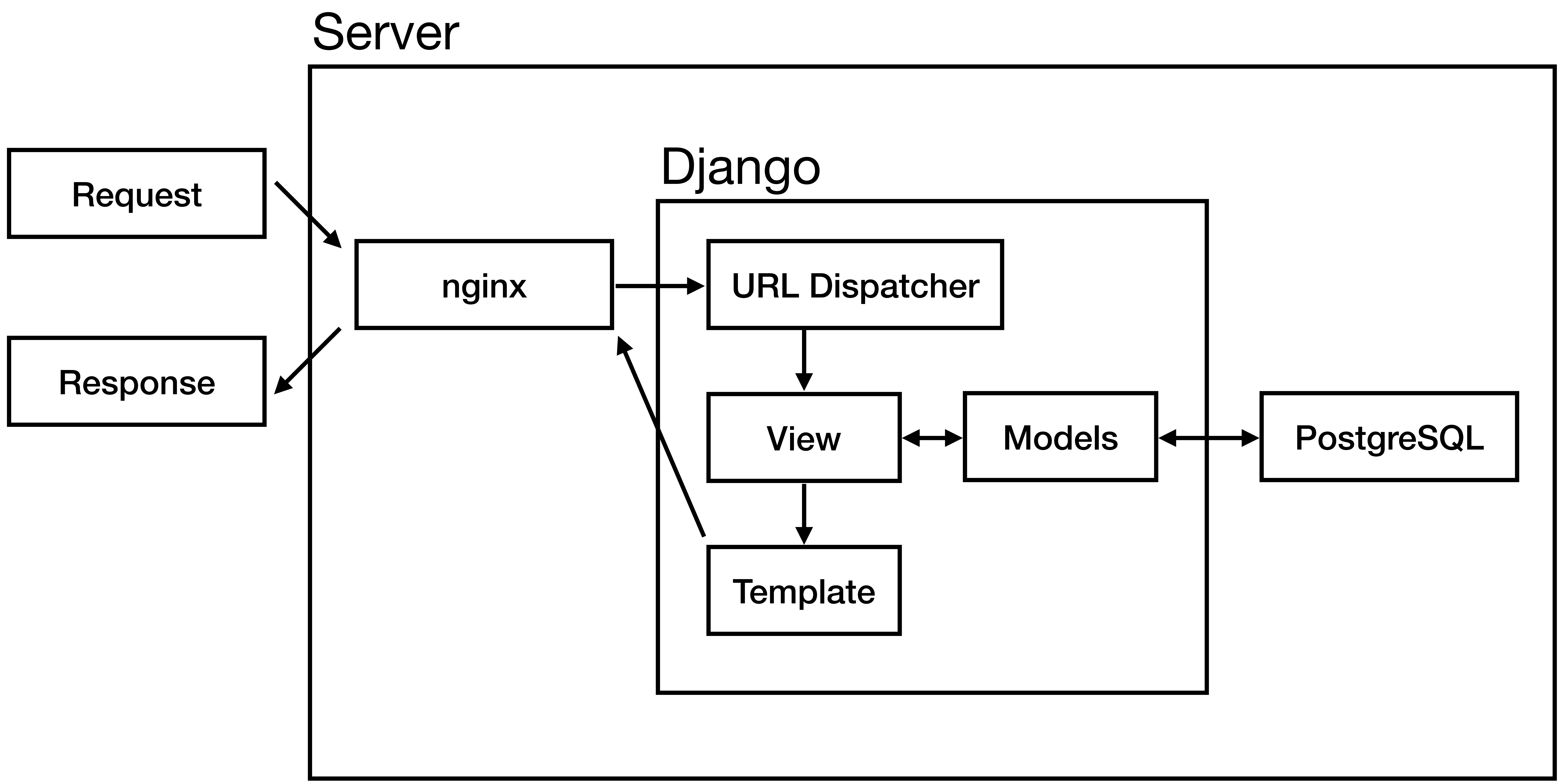
hello_world.py

```python
def application(env, start_response):
    start_response("200 OK", [("Content-Type", "text/html")])
    return [b"Hello World"]
```

Writing a Django Web App

- Use Django to generate a starter project
- Implement models and views using Python, and templates using HTML
- Pros:
  - Requires writing a small amount of well-structured code
  - Django provides features for security and maintainability
- Cons:
  - Requires learning the complex Django framework

Django Components



Django Hello World Skeleton

- `pip install Django`
- `django-admin startproject djangoproject`
- `cd djangoproject`
- `python manage.py startapp hello`

```
djangoproject/
├── djangoproject
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
├── hello
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── migrations
│   │   └── __init__.py
│   ├── models.py
│   ├── tests.py
│   └── views.py
└── manage.py
```

Django Hello World Code

hello/views.py

```
from django.http import HttpResponse

def index(request):
    return HttpResponse("Hello World")
```

hello/urls.py

```
from django.urls import path
from . import views

urlpatterns = [
    path("", views.index, name="index"),
]
```

djangoproject/urls.py

```
from django.contrib import admin
from django.urls import include, path

urlpatterns = [
    path("hello/", include("hello.urls")),
    path("admin/", admin.site.urls),
]
```

https://docs.djangoproject.com/en/5.0/intro/tutorial01/

Django's Secret...

- Django runs as a WSGI application!

djangoproject/wsgi.py

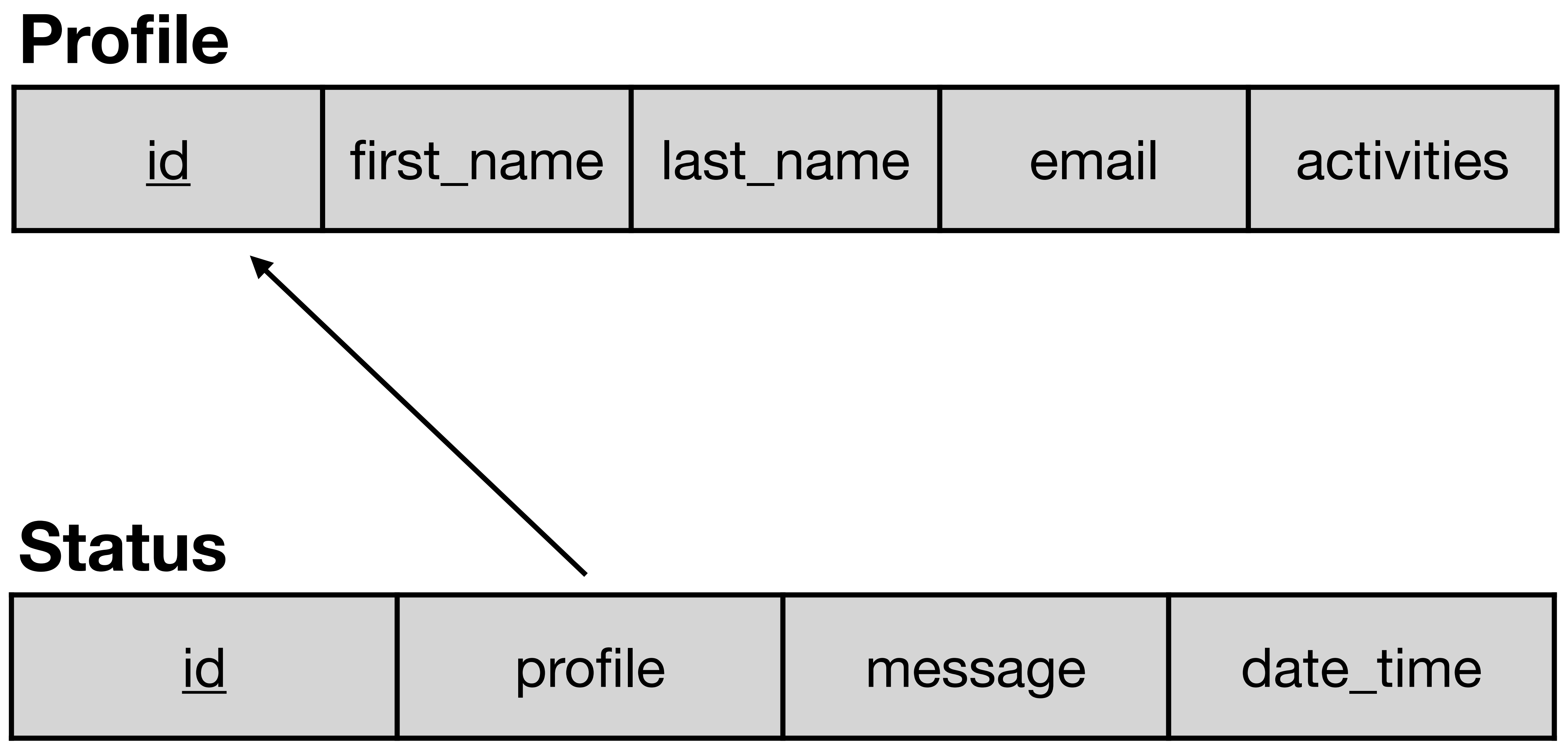
```
import os
from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'djangoproject.settings')

application = get_wsgi_application()
```

Object-Relational Mapping (ORM)

Object-Relational Mapping (ORM)

- Active Record design pattern
  - Classes represent tables, abstracting data access and derived attributes
- Implemented by many frameworks, including:
  - Django
  - Ruby on Rails
- Other ORM patterns: Data Mapper, Repository

MiniFacebook Schema



MiniFacebook Models

```
import uuid

from django.db import models


class Profile(models.Model):
    id = models.UUIDField(primary_key=True, default=uuid.uuid4, editable=False)
    first_name = models.CharField(max_length=100)
    last_name = models.CharField(max_length=100)
    email = models.EmailField()
    activities = models.TextField()


class Status(models.Model):
    id = models.UUIDField(primary_key=True, default=uuid.uuid4, editable=False)
    message = models.TextField()
    date_time = models.DateTimeField()
    profile = models.ForeignKey(Profile, null=False, on_delete=models.CASCADE)
```

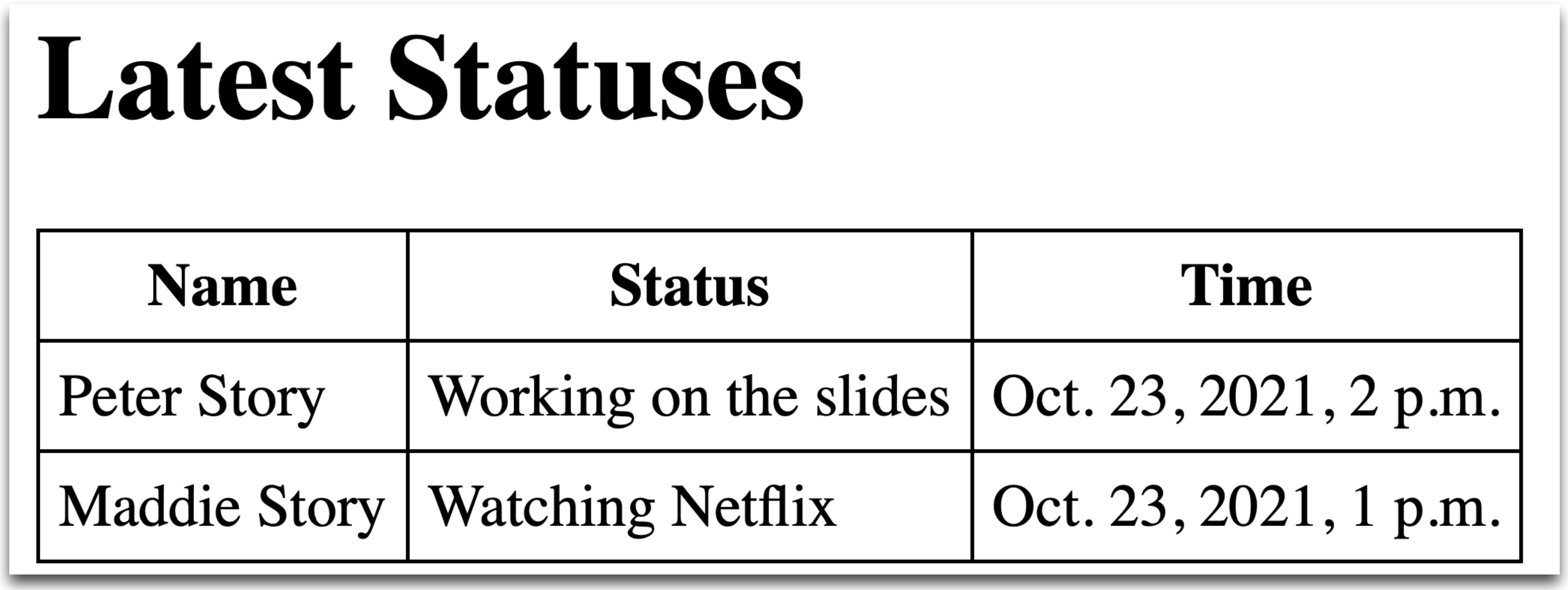
MiniFacebook View

```
from django.shortcuts import render

from .models import Profile

def index(request):
    context = {"profiles": Profile.objects.all()}
    return render(request, "index.html", context)
```

MiniFacebook Template

```
<h1>Latest Statuses</h1>

<table>
<tr><th>Name</th><th>Status</th><th>Time</th></tr>
{% for profile in profiles %}
  <tr>
    <td>{{profile.first_name}} {{profile.last_name}}</td>
    <td>{{profile.latest_status.message}}</td>
    <td>{{profile.latest_status.date_time}}</td>
  </tr>
{% endfor %}
</table>
```

MiniFacebook Rendered



Latest Statuses

Name	Status	Time
Peter Story	Working on the slides	Oct. 23, 2021, 2 p.m.
Maddie Story	Watching Netflix	Oct. 23, 2021, 1 p.m.

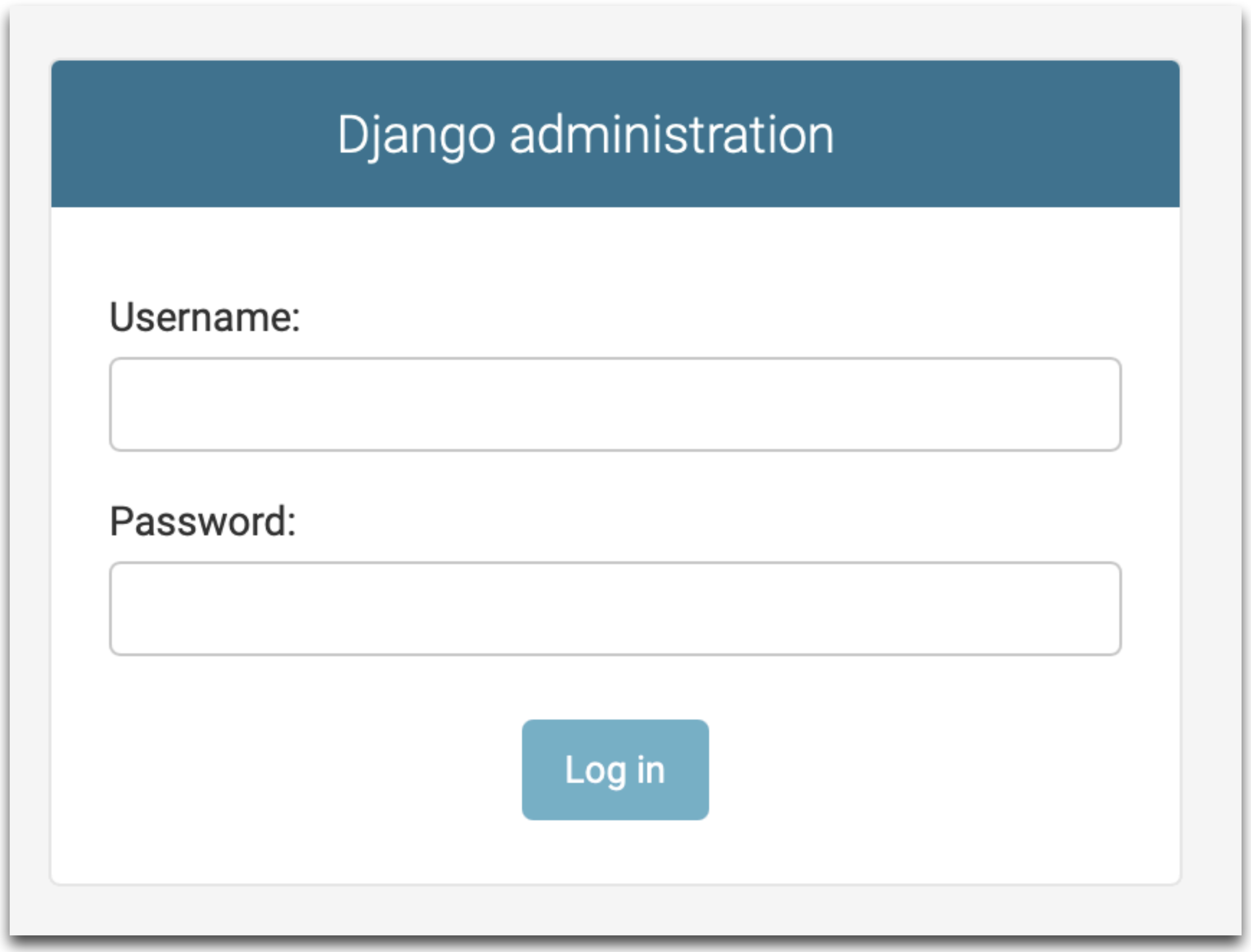
MiniFacebook Code

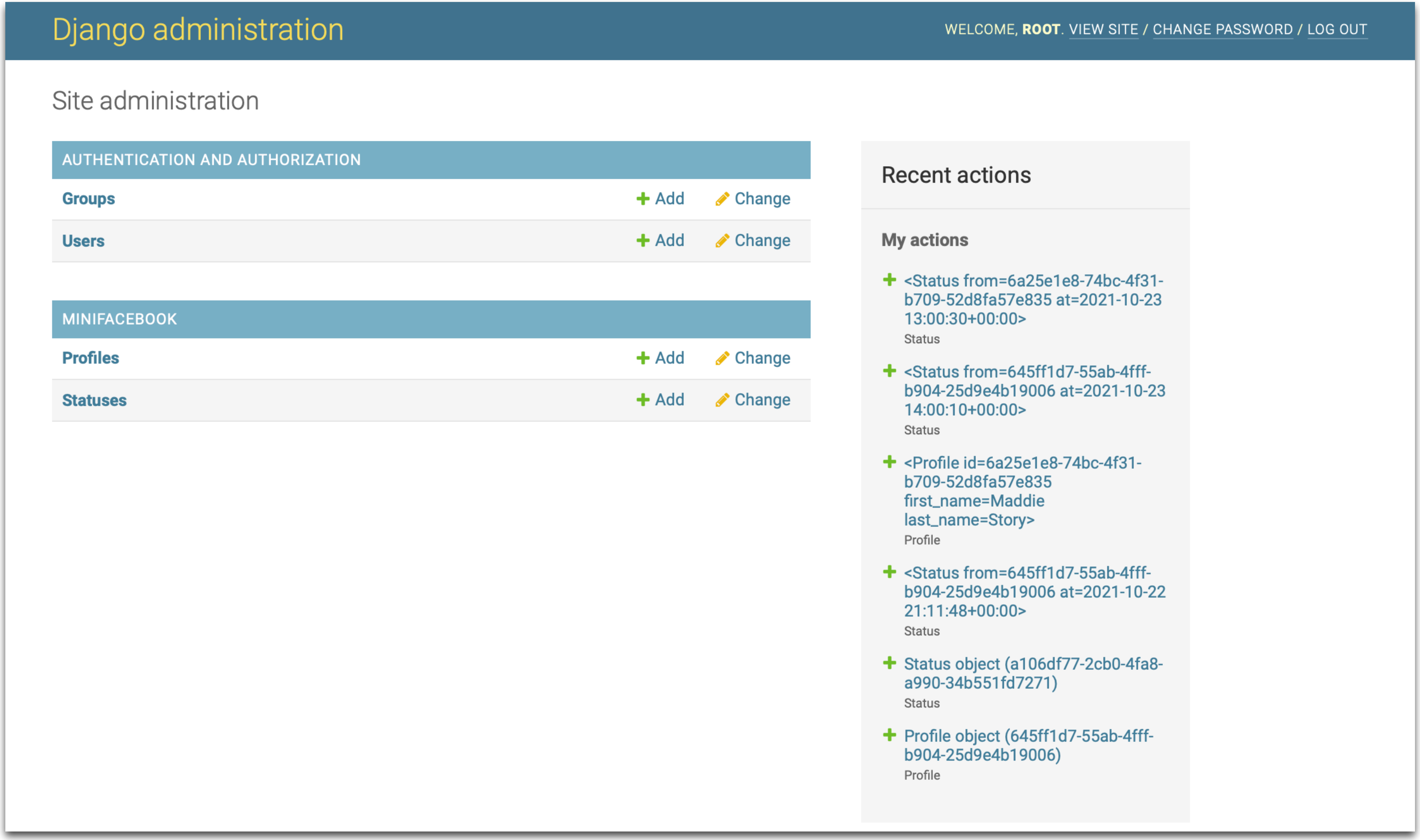
- Simple WSGI examples: https://github.com/ClarkuCSCI/csci220-uwsgi
- Django examples: https://github.com/ClarkuCSCI/csci220-django

Django Admin Interface

Django Admin Interface

- After you’ve implemented your models, you get a fully-featured application in <5 lines of code
- Convenient and secure
- Ideal for standard CRUD (Create Read Update Delete) apps





Django administration

WELCOME, **ROOT**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

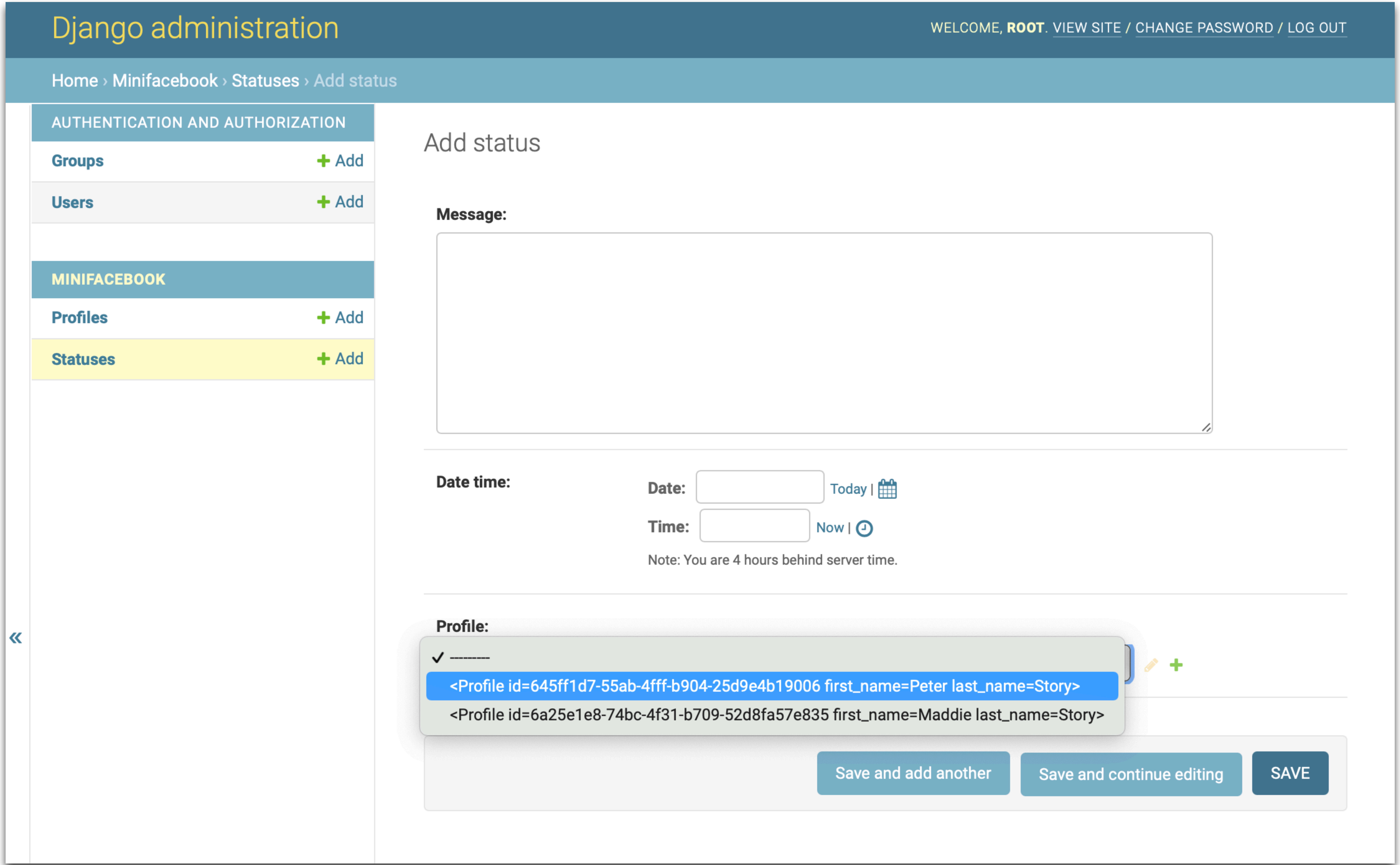
AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

MINIFACEBOOK		
Profiles	+ Add	Change
Statuses	+ Add	Change

Recent actions

My actions

+ <Status from=6a25e1e8-74bc-4f31-b709-52d8fa57e835 at=2021-10-23 13:00:30+00:00>
  Status
+ <Status from=645ff1d7-55ab-4fff-b904-25d9e4b19006 at=2021-10-23 14:00:10+00:00>
  Status
+ <Profile id=6a25e1e8-74bc-4f31-b709-52d8fa57e835 first_name=Maddie last_name=Story>
  Profile
+ <Status from=645ff1d7-55ab-4fff-b904-25d9e4b19006 at=2021-10-22 21:11:48+00:00>
  Status
+ Status object (a106df77-2cb0-4fa8-a990-34b551fd7271)
  Status
+ Profile object (645ff1d7-55ab-4fff-b904-25d9e4b19006)
  Profile


Django administration

WELCOME, **ROOT**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Minifacebook › Statuses › Add status

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

MINIFACEBOOK

Profiles + Add

Statuses + Add

Add status

Message:

Date time: Date: Today | Time: Now |

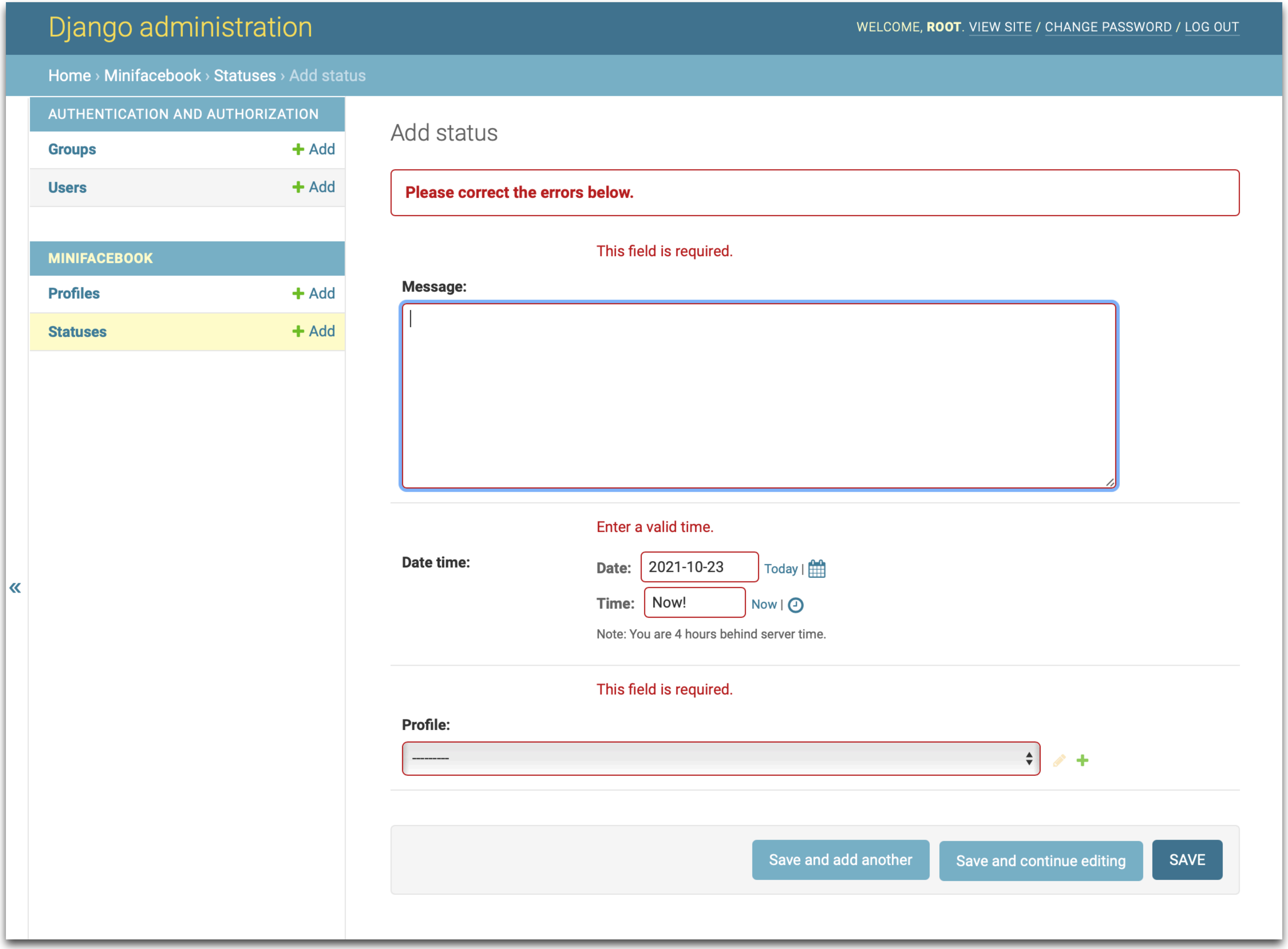
Note: You are 4 hours behind server time.

Profile:

✓ ---------

<Profile id=645ff1d7-55ab-4fff-b904-25d9e4b19006 first_name=Peter last_name=Story>

<Profile id=6a25e1e8-74bc-4f31-b709-52d8fa57e835 first_name=Maddie last_name=Story>



Django administration

WELCOME, **ROOT**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Minifacebook › Statuses › Add status

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

MINIFACEBOOK

Profiles + Add

Statuses + Add

Add status

Please correct the errors below.

This field is required.

Message:

Enter a valid time.

Date time: **Date:** 2021-10-23 Today |

Time: Now! Now |

Note: You are 4 hours behind server time.

This field is required.

Profile: ---------

Save and add another | Save and continue editing | SAVE

Migrations

Migrations

- If your application is used by real people, it will need to be changed
    - Sometimes the database schema will change
- Database migrations encode changes to the database
- Frameworks like Django offer advanced features:
    - Automatic migration generation
    - Rollbacks
    - Squashing migrations (merging them)

Django Migrations

1. Generate migrations
    - `python manage.py makemigrations`
2. Apply migrations
    - `python manage.py migrate`

MiniFacebook Migrations

0001_initial.py

```
from django.db import migrations, models
import django.db.models.deletion
import uuid


class Migration(migrations.Migration):
    initial = True
    dependencies = []
    operations = [
        migrations.CreateModel(
            name='Profile',
            fields=[
                ('id', models.UUIDField(default=uuid.uuid4, editable=False, primary_key=True, serialize=False)),
                ('first_name', models.CharField(max_length=100)),
                ('last_name', models.CharField(max_length=100)),
                ('email', models.EmailField(max_length=254)),
                ('activities', models.TextField()),
            ],
        ),
...
```

MiniFacebook Migrations

0001_initial.py

```
...
        migrations.CreateModel(
            name='Status',
            fields=[
                ('id', models.UUIDField(default=uuid.uuid4, editable=False, primary_key=True, serialize=False)),
                ('message', models.TextField()),
                ('date_time', models.DateTimeField()),
                ('profile', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
to='minifacebook.profile')),
            ],
        ),
    ]
```

MiniFacebook Migrations

0002_alter_status_options.py

```
from django.db import migrations


class Migration(migrations.Migration):

    dependencies = [
        ('minifacebook', '0001_initial'),
    ]

    operations = [
        migrations.AlterModelOptions(
            name='status',
            options={'verbose_name_plural': 'Statuses'},
        ),
    ]
```

MiniFacebook Migrations

0003_poke.py

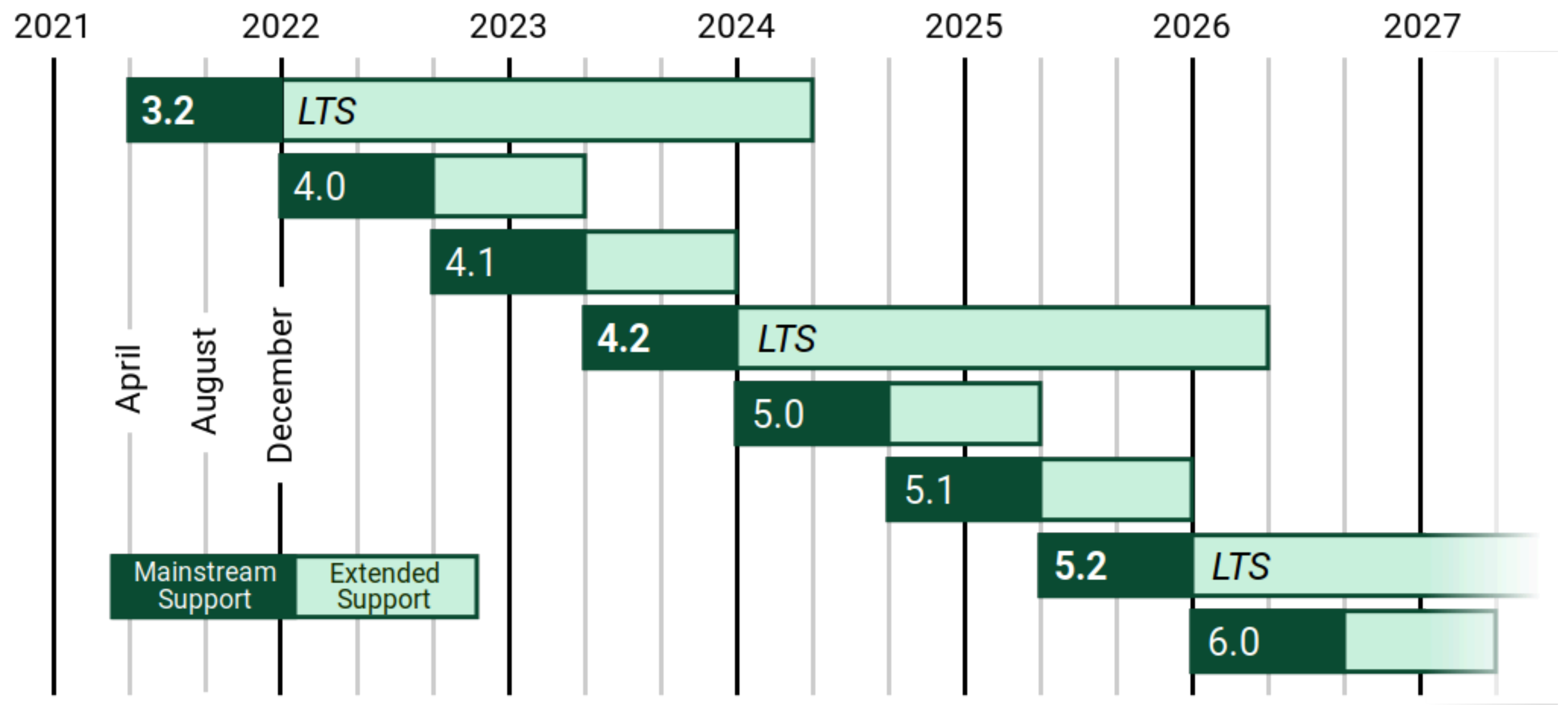
```
from django.db import migrations, models
import django.db.models.deletion

class Migration(migrations.Migration):

    dependencies = [('minifacebook', '0002_alter_status_options'),]

    operations = [
        migrations.CreateModel(
            name='Poke',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, 
verbose_name='ID')),
                ('date_time', models.DateTimeField()),
                ('pokee', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, 
related_name='poke_pokee', to='minifacebook.profile')),
                ('poker', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, 
related_name='poke_poker', to='minifacebook.profile')),
            ],
        ),
    ]
```

Migration SQL

```
> python manage.py sqlmigrate minifacebook 0001
BEGIN;
--
-- Create model Profile
--
CREATE TABLE "minifacebook_profile" ("id" uuid NOT NULL PRIMARY KEY, "first_name" varchar(100) NOT NULL,
"last_name" varchar(100) NOT NULL, "email" varchar(254) NOT NULL, "activities" text NOT NULL);
--
-- Create model Status
--
CREATE TABLE "minifacebook_status" ("id" uuid NOT NULL PRIMARY KEY, "message" text NOT NULL, "date_time"
timestamp with time zone NOT NULL, "profile_id" uuid NOT NULL);
ALTER TABLE "minifacebook_status" ADD CONSTRAINT "minifacebook_status_profile_id_dfb04e9b_fk_minifaceb" FOREIGN
KEY ("profile_id") REFERENCES "minifacebook_profile" ("id") DEFERRABLE INITIALLY DEFERRED;
CREATE INDEX "minifacebook_status_profile_id_dfb04e9b" ON "minifacebook_status" ("profile_id");
COMMIT;
```

Django Version Differences

- Django is frequently updated
- I recommend using <u>the latest long-term support (LTS) version</u>
- Ensure you're reading the correct documentation version



Django 4.2 LTS Documentation

- Django design philosophy
- Django models
- Django model fields
- Django queries
- Django migrations

Homework