Informe de Laboratorio 06

Tema: Laboratorio 06Django (Admin)

Nota

Estudiante	Escuela	Asignatura
Diego Alejandro Carbajal Gonzales, Mariel Alisson Jara Mamani, Jhonatan David Arias Quispe, Ricardo Mauricio Chambilla Perca jariasq@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: II Código: 1702122

Laboratorio	Tema	Duración
06	Laboratorio 06Django (Admin)	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 05 de Junio 2024	Al 08 de Junio 2024

1. Actividades

- Elabore un primer informe grupal, con el modelo de datos de una aplicación que desarrollará durante este semestre.
- Utilicen todas las recomendaciones encontradas en la aplicación library.

1.1. Pregunta

Por cada integrante del equipo, resalte un aprendizaje que adquirió al momento de estudiar esta primera parte de Django (Admin). No se reprima de ser detallista. Coloque su nombre entre parentesis para saber que es su aporte.

2. Equipos, materiales y temas utilizados

- Sistema Operativo ArchCraft GNU Linux 64 bits Kernell
- Termux
- NeoVim
- Git 2.42.0

- Cuenta en GitHub con el correo institucional.
- Latex
- Python 3.10.2
- Django 4.0.2
- Pip
- Virtualenv
- Figma
- Graphviz

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/JhonatanDczel/CoderDojo-UNSA.git`

4. Planificación

Para tener claras las actividades a realizar, se ha realizado una planificación de las mismas, usando la herramienta Figma:

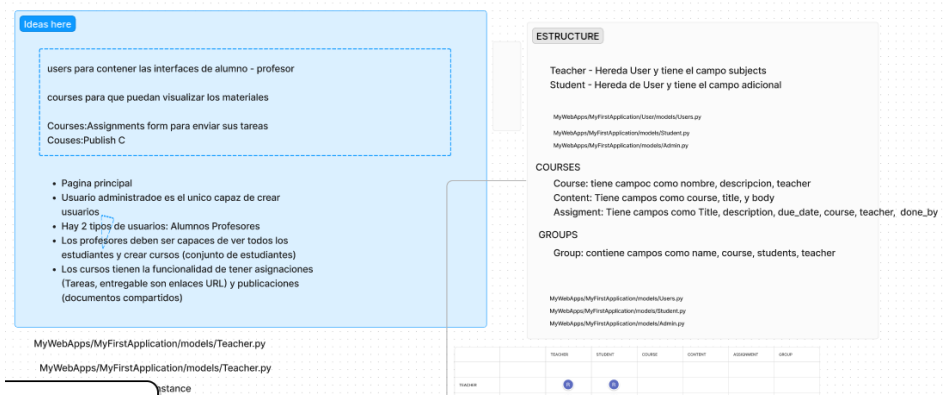

Figura 1: Planificación de actividades

5. Modelo de datos

Para el desarrollo de la aplicación se ha planteado el siguiente modelo de datos:

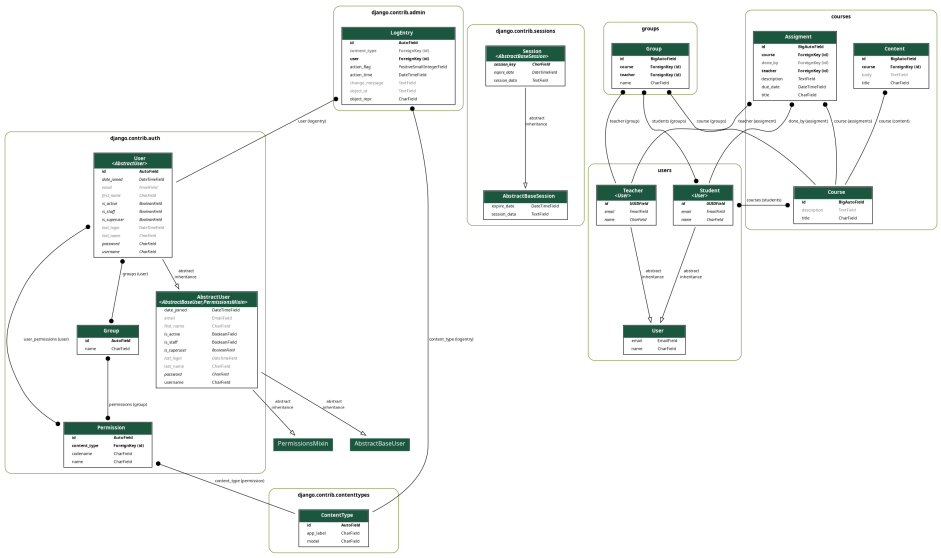

Figura 2: Modelo de datos

A continuacion veremos una explicacion mas detallada sobre los modelos en Django.

6. Modelos Python

Los siguientes modelos fueron creados en Django, en archivos separados, en la carpeta models de cada aplicación:

6.1. Student

El modelo Student hereda de User, y tiene una relación de muchos a uno con Course, es decir, un estudiante puede estar inscrito en varios cursos, pero un curso solo puede tener un estudiante.

```
class Student(User):
    courses = models.ForeignKey(Course, related_name='students', on_delete=models.CASCADE)
```

6.2. Teacher

El modelo Teacher hereda de User, y tiene una relación de muchos a uno con Groups, que maneja la lógica de los grupos de estudiantes y sus cursos.

```
class Teacher(User):
    pass
```

6.3. Course

El modelo Course tiene un título y una descripción, y una relación de muchos a uno con Teacher, es decir, un curso solo puede tener un profesor, pero un profesor puede tener varios cursos.

Los campos title y description son obligatorios, y el campo teacher es opcional.

```
class Course(models.Model):
  title = models.CharField(max_length=255, blank=False, null=False)
```

```
  description = models.TextField(blank=True, null=False)

  #teacher = models.ForeignKey(Teacher, on_delete=models.CASCADE)
  def __str__(self):
    return self.title
```

6.4. Content

El modelo Content tiene un título y un cuerpo, y una relación de muchos a uno con Course, es decir, un contenido solo puede pertenecer a un curso, pero un curso puede tener varios contenidos.

Representa el contenido de un curso, como una lección, un video, un archivo, etc.

```
class Content(models.Model):
  course = models.ForeignKey(Course, on_delete=models.CASCADE)
  title = models.CharField(blank=False, max_length=255, null=False)
  body = models.TextField(blank=True, null=False)

  class Meta:
    ordering = ['course', 'title', 'body']

  def __str__(self):
    return self.title
```

6.5. Assigment

El modelo Assignment tiene un título, una descripción, una fecha de entrega, y una relación de muchos a uno con Course y Teacher, y una relación de uno a uno con Student, es decir, una tarea solo puede ser asignada a un estudiante, pero un estudiante puede tener varias tareas.

Los estudiantes enviarán sus tareas a través de la aplicación, y los profesores podrán calificarlas, mediante un link.

```
class Assignment(models.Model):
    title = models.CharField(max_length=255, blank=False, null=False)
    description = models.TextField(blank=False, null=False)
    due_date = models.DateTimeField()
    course = models.ForeignKey(Course, related_name='assigments', on_delete=models.CASCADE)
    teacher = models.ForeignKey('users.Teacher', on_delete=models.CASCADE)
    done_by = models.ForeignKey('users.Student', null=True, blank=True, on_delete=models.SET_NULL)

    def __str__(self):
        return self.title
```

6.6. Groups

El modelo Group tiene un nombre, una relación de muchos a uno con Course y Teacher, y una relación de muchos a muchos con Student, es decir, un grupo solo puede tener un profesor y un curso, pero un profesor puede tener varios grupos, y un grupo puede tener varios estudiantes.

Los grupos representan la división de los estudiantes en clases, y cada grupo tiene un profesor y un curso asignado.

```
class Group(models.Model):

    name = models.CharField(max_length=255, blank=False, null=False)
    course = models.ForeignKey(Course, related_name='groups', on_delete=models.CASCADE)
    students = models.ManyToManyField('users.Student', related_name='groups')
    teacher = models.ForeignKey('users.Teacher', on_delete=models.CASCADE)
```

```
    def __str__(self):
        return self.name
```

7. Admin CRUD

Para poder realizar el CRUD de los modelos en Django, se debe registrar los modelos en el archivo admin.py de cada aplicación. A continuación se muestra un ejemplo de cómo se registra el modelo Course en el archivo admin.py de la aplicación courses:

```
from django.contrib import admin
from courses.models import Course
from courses.models import Content
from courses.models import Assignment
from users.models import Student
from users.models import Teacher
from groups.models import Group


# Register your models here.
admin.site.register(Course)
admin.site.register(Content)
admin.site.register(Assignment)
admin.site.register(Student)
admin.site.register(Teacher)
admin.site.register(Group)
```

Esta es una forma de registrar los modelos en el panel de administración de Django, para poder realizar el CRUD de los modelos.

Aqui se muestra el acceso por medio de la URL /admin, donde se puede ver los modelos registrados en el panel de administración de Django.

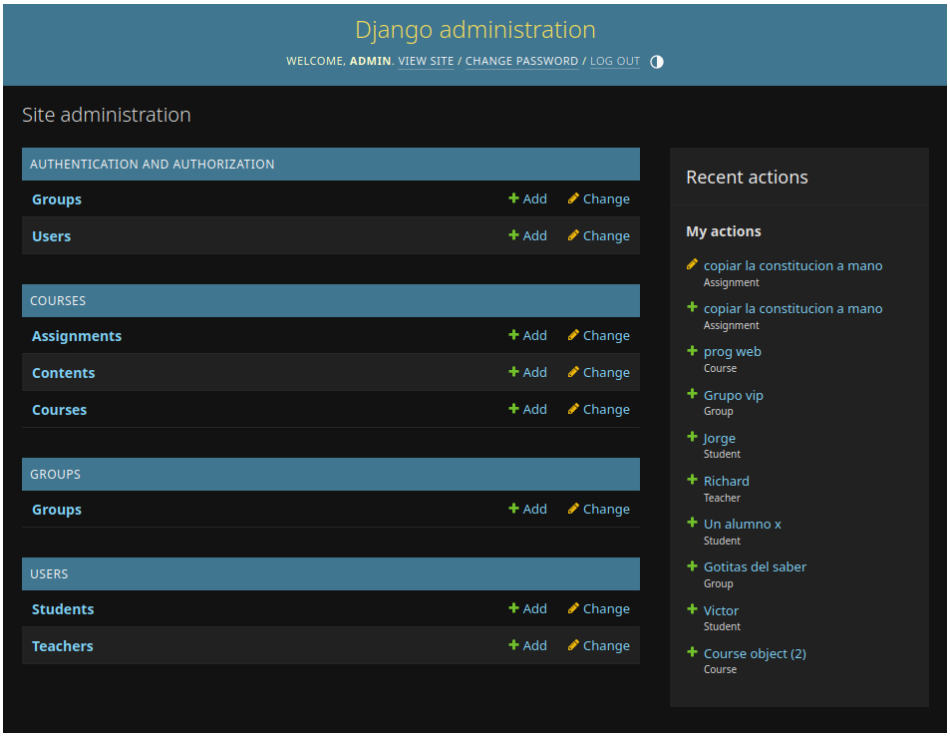

Figura 3: Panel de administración de Django

8. Commits más recientes

- A continuación se presentan los commits más recientes.
- Commit 56934394e531bc0bb79a154a410895e6d7e01256
  Autor: JhonatanDczel ¡jariasq@unsa.edu.pe¿
  Fecha: Viernes 17 de mayo de 2024 -05:00
  Descripción: Ejercicio 03 completado
- Commit d8386a3942377190621a037ae0e62410330fd8ad
  Autor: JhonatanDczel ¡jariasq@unsa.edu.pe¿
  Fecha: Viernes 17 de mayo de 2024 -05:00
  Descripción: Ejercicio 02 completo
- Commit ccec8d074a312d17012a3c76eb4a54b8277b27ae
  Autor: JhonatanDczel ¡jariasq@unsa.edu.pe¿
  Fecha: Viernes 17 de mayo de 2024 -05:00
  Descripción: Reestructurando el directorio
- Commit f276cd63e77e35bfe0ba970503ab5325766c1772
  Autor: JhonatanDczel ¡jariasq@unsa.edu.pe¿
  Fecha: Viernes 17 de mayo de 2024 -05:00
  Descripción: Haciendo el teclado random
- Commit 2de93beb630cc8a0e1327950dfa71d4a51ce7273
  Autor: JhonatanDczel ¡jariasq@unsa.edu.pe¿
  Fecha: Viernes 17 de mayo de 2024 -05:00
  Descripción: Termina ejercicio 03

- Commit 523e2c994f290bb09876b5feaad519411a367168
  Autor: JhonatanDczel ¡jariasq@unsa.edu.pe¿
  Fecha: Viernes 17 de mayo de 2024 -05:00
  Descripción: Nav completo

9. Rúbricas

9.1. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 1: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 2: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Repositorio se pudo clonar y se evidencia la estructura adecuada para revisar los entrega bles. (Se descontará puntos por error o onser vación)	4	X	4	
2. Commits	Hay porciones de código fuente asociado a los commits planificados con explicaciones deta lladas. (El profesor puede preguntar para re frendar calificación).	4	X	4	
3. Ejecución	Se incluyen comandos para ejecuciones y prue bas del código fuente explicadas gradualmente que permitirían replicar el proyecto. (Se des contará puntos por cada omisión)	4	X	1.5	
4. Pregunta	Se responde con completitud a la pregunta for mulada en la tarea. (El profesor puede pregun tar para refrendar calificación)	2	X	2	
5. Ortografía	El documento no muestra errores ortográficos. (Se descontará puntos por error encontrado)	2	X	1.5	
6. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente con explicaciones puntuales pero precisas, agre gando diagramas generados a partir del códi go fuente y refleja un acabado impecable. (El profesor puede preguntar para refrendar cali ficación).	4	X	3	
Total		20		16	