Biblioteca Colegio San Marcos Apóstol

Versión 1.0

Catalina Fabres - Diego Hernández - Marialejandra Holguín

26 de diciembre de 2023

Contenidos:

CAPÍTULO 1

appejemplares package

1.1 Submodules

1.2 appejemplares.forms module

class appejemplares.forms.**LibroForm**(*data=None*, *files=None*, *auto_id='id_ %s'*, *prefix=None*, *initial=None*, *error_class=<class 'django.forms.utils.ErrorList'>*, *label_suffix=None*, *empty_permitted=False*, *instance=None*, *use_required_attribute=None*, *renderer=None*)

Bases: `ModelForm`

Formulario para la creación y edición de libros.

Este formulario se basa en el modelo Libro y se utiliza para recopilar información sobre los libros, permitiendo su creación y edición.

Atributos:

model: Clase del modelo asociado al formulario (Libro). fields: Lista de campos del modelo a incluir en el formulario.

Campos Adicionales:

categoria: Campo ChoiceField que representa la categoría del libro.

Métodos:

__init__: Inicializa el formulario y ajusta el widget del campo “categoria”.

CATEGORIAS_CHOICES = [('novelas', 'Novelas'), ('suspenso', 'Suspenso'), ('historia', 'Historia'), ('test', 'Test')]

class Meta

Bases: `object`

fields = '__all__'

model

alias de *Libro*

base_fields = {'autor': <django.forms.fields.CharField object>, 'categoria': <django.forms.fields.ChoiceField object>, 'descripcion': <django.forms.fields.CharField object>, 'ejemplares_disponibles': <django.forms.fields.IntegerField object>, 'ejemplares_prestados': <django.forms.fields.IntegerField object>, 'ejemplares_reservados': <django.forms.fields.IntegerField object>, 'ejemplares_totales': <django.forms.fields.IntegerField object>, 'isbn': <django.forms.fields.CharField object>, 'titulo': <django.forms.fields.CharField object>, 'ubicacion': <django.forms.fields.CharField object>}

declared_fields = {'categoria': <django.forms.fields.ChoiceField object>}

property media

Return all media required to render the widgets on this form.

1.3 appejemplares.models module

class appejemplares.models.**Libro**(**args*, ***kwargs*)

Bases: Model

Modelo que representa un libro en la biblioteca.

Attributes: - isbn (str): Número de identificación único del libro. - titulo (str): Título del libro. - autor (str): Autor del libro. - categoria (str): Categoría del libro (e.g., Novelas, Suspenso). - ubicacion (str): Ubicación física del libro en la biblioteca. - ejemplares_totales (int): Número total de ejemplares del libro. - ejemplares_disponibles (int): Número de ejemplares disponibles para préstamo. - ejemplares_prestados (int): Número de ejemplares actualmente prestados. - ejemplares_reservados (int): Número de ejemplares reservados. - descripcion (str): Descripción del libro (opcional).

Methods: - __str__: Representación en cadena del libro (devuelve el ISBN como identificador). - is_upperclass: Verifica si la categoría del libro está en mayúsculas. - delete: Sobrescribe el método de eliminación para realizar acciones personalizadas.

CATEGORIAS_CHOICES = [('novelas', 'Novelas'), ('suspenso', 'Suspenso'), ('historia', 'Historia'), ('test', 'Test')]

exception DoesNotExist

Bases: ObjectDoesNotExist

exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

autor

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

categoria

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

delete(*using=None*, *keep_parents=False*)

Sobrescribe el método de eliminación para realizar acciones personalizadas.

En este caso, se podría agregar la eliminación de la imagen asociada al libro si se utiliza algún campo de imagen.

Args: - using (str): Nombre de la base de datos. - keep_parents (bool): Indica si se deben mantener las relaciones de clave externa.

descripcion

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

ejemplares_disponibles

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

ejemplares_prestados

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

ejemplares_reservados

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

ejemplares_totales

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

get_categoria_display(*, *field=<django.db.models.fields.CharField: categoria>*)

is_upperclass()

Verifica si la categoría del libro está en mayúsculas.

isbn

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

objects = <django.db.models.manager.Manager object>

prestamo_set

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

titulo

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

ubicacion

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

appejemplares.models.**actualizar_ejemplares_libro**(*sender*, *instance*, ***kwargs*)

Actualiza el número de ejemplares disponibles y prestados cuando se realiza un préstamo.

Este método se conecta a la señal pre_save de Prestamo para actualizar automáticamente el número de ejemplares disponibles y prestados en el libro asociado cuando se realiza un préstamo o se finaliza un préstamo.

Args: - sender: Clase que envía la señal (Prestamo en este caso). - instance: Instancia de Prestamo que activa la señal. - **kwargs: Argumentos adicionales.

1.4 appejemplares.urls module

1.5 appejemplares.views module

appejemplares.views.**catalogo**(*request*)

Lista libros y muestra una barra de búsqueda en el catálogo. Para usuarios que reservan libros.

Args:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP renderizada para la página del catálogo.

appejemplares.views.**crear**(*request*)

Crea un nuevo libro y redirige a la página de listado de libros.

Args:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP redirigida a la página de listado de libros.

appejemplares.views.**editar**(*request*, *isbn*)

Edita un libro existente y redirige a la página de listado de libros.

Args:
request (HttpRequest): La solicitud HTTP recibida. isbn (str): El ISBN del libro a editar.

Returns:
HttpResponse: La respuesta HTTP redirigida a la página de listado de libros.

appejemplares.views.**eliminar**(*request*, *isbn*)

Elimina un libro existente y redirige a la página de listado de libros.

Args:
request (HttpRequest): La solicitud HTTP recibida. isbn (str): El ISBN del libro a eliminar.

Returns:
HttpResponse: La respuesta HTTP redirigida a la página de listado de libros.

appejemplares.views.**exit**(*request*)

Cierra la sesión del usuario y redirige a la página de inicio.

Args:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP redirigida a la página de inicio.

appejemplares.views.**inicio**(*request*)

Renderiza la página de inicio.

Args:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP renderizada para la página de inicio.

appejemplares.views.**libros**(*request*)

Lista libros y muestra una barra de búsqueda para bibliotecarios. También realiza la paginación de los resultados.

Args:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP renderizada para la página de listado de libros.

appejemplares.views.**nosotros**(*request*)

Renderiza la página «Nosotros».

Args:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP renderizada para la página «Nosotros».

appejemplares.views.**subir_csv**(*request*)

Sube un archivo CSV con información de libros y actualiza la base de datos.

Args:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP redirigida a la página de listado de libros.

1.6 Module contents

CAPÍTULO 2

apppprestamos package

2.1 Submodules

2.2 apppprestamos.forms module

class apppprestamos.forms.**PrestamoForm**(*data=None*, *files=None*, *auto_id='id_ %s'*, *prefix=None*, *initial=None*, *error_class=<class 'django.forms.utils.ErrorList'>*, *label_suffix=None*, *empty_permitted=False*, *instance=None*, *use_required_attribute=None*, *renderer=None*)

Bases: `ModelForm`

Formulario para el modelo Prestamo.

Attributes:
ESTADOS_CHOICE (list): Opciones para el campo “estado_prestamo”.

Fields:

estado_prestamo (ChoiceField): Campo para seleccionar el estado del préstamo.

- choices: Opciones definidas en ESTADOS_CHOICE.
- widget: Selector de opciones con clase “form-control”.

fecha_devolucion_real (DateField): Campo para ingresar la fecha real de devolución.

- widget: Campo de entrada de fecha con tipo “date”.

Methods:
__init__: Método de inicialización para personalizar la creación del formulario.

```
ESTADOS_CHOICE = [('vigente', 'Préstamo Vigente'), ('finalizado', 'Préstamo
Finalizado'), ('atrasado', 'Devolución Atrasada')]
```

class Meta

Bases: object

fields = '__all__'

model

alias de *Prestamo*

base_fields = {'estado_prestamo': <django.forms.fields.ChoiceField object>, 'fecha_devolucion_calculada': <django.forms.fields.DateField object>, 'fecha_devolucion_real': <django.forms.fields.DateField object>, 'fecha_prestamo': <django.forms.fields.DateField object>, 'isbn': <django.forms.models.ModelChoiceField object>, 'rut_usuario': <django.forms.models.ModelChoiceField object>}

declared_fields = {'estado_prestamo': <django.forms.fields.ChoiceField object>}

property media

Return all media required to render the widgets on this form.

widgets = {'estado_prestamo': <django.forms.widgets.Select object>, 'fecha_devolucion_real': <django.forms.widgets.DateInput object>}

2.3 apppprestamos.models module

class apppprestamos.models.**Prestamo**(**args*, ***kwargs*)

Bases: Model

Modelo que representa un préstamo de un libro a un usuario.

Attributes:

id_prestamo (AutoField): Campo autoincremental para identificar un préstamo de manera única. isbn (ForeignKey): Clave foránea que relaciona el préstamo con un libro. rut_usuario (ForeignKey): Clave foránea que relaciona el préstamo con un usuario. fecha_prestamo (DateField): Fecha en que se realiza el préstamo. fecha_devolucion_calculada (DateField): Fecha estimada de devolución calculada automáticamente. fecha_devolucion_real (DateField, optional): Fecha real de devolución (opcional, puede ser nula). estado_prestamo (CharField): Estado actual del préstamo (vigente, finalizado, atrasado).

Methods:

__str__: Devuelve una representación legible en cadena del objeto Prestamo.

exception DoesNotExist

Bases: ObjectDoesNotExist

ESTADOS_CHOICE = [('vigente', 'Préstamo Vigente'), ('finalizado', 'Préstamo Finalizado'), ('atrasado', 'Devolución Atrasada')]

exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

estado_prestamo

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

fecha_devolucion_calculada

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

fecha_devolucion_real

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

fecha_prestamo

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

get_estado_prestamo_display(*, *field=<django.db.models.fields.CharField: estado_prestamo>*)

get_next_by_fecha_devolucion_calculada(*, *field=<django.db.models.fields.DateField: fecha_devolucion_calculada>*, *is_next=True*, ***kwargs*)

get_next_by_fecha_prestamo(*, *field=<django.db.models.fields.DateField: fecha_prestamo>*, *is_next=True*, ***kwargs*)

get_previous_by_fecha_devolucion_calculada(*, *field=<django.db.models.fields.DateField: fecha_devolucion_calculada>*, *is_next=False*, ***kwargs*)

get_previous_by_fecha_prestamo(*, *field=<django.db.models.fields.DateField: fecha_prestamo>*, *is_next=False*, ***kwargs*)

id_prestamo

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

isbn

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Child.parent` is a `ForwardManyToOneDescriptor` instance.

isbn_id

objects = <django.db.models.manager.Manager object>

rut_usuario

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Child.parent` is a `ForwardManyToOneDescriptor` instance.

rut_usuario_id

apppprestamos.models.**actualizar_fecha_devolucion_real**(*sender*, *instance*, ***kwargs*)

apppprestamos.models.**fecha_devolucion**()

apppprestamos.models.**fecha_hoy**()

2.4 apppprestamos.urls module

2.5 apppprestamos.views module

apppprestamos.views.**editar_prestamo**(*request*, *id_prestamo*)

Vista para editar un préstamo existente.

Parameters:
request (HttpRequest): La solicitud HTTP recibida. id_prestamo (int): El ID del préstamo a editar.

Returns:
HttpResponse: La respuesta HTTP que muestra el formulario para editar el préstamo.

apppprestamos.views.**eliminar_prestamo**(*request*, *id_prestamo*)

Vista para eliminar un préstamo existente.

Parameters:
request (HttpRequest): La solicitud HTTP recibida. id_prestamo (int): El ID del préstamo a eliminar.

Returns:
HttpResponse: La respuesta HTTP que redirige a la lista de préstamos después de la eliminación.

apppprestamos.views.**nuevo_prestamo**(*request*)

Vista para crear un nuevo préstamo.

Parameters:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP que muestra el formulario para un nuevo préstamo.

apppprestamos.views.**prestamos**(*request*)

Vista para mostrar la lista de préstamos y filtrar por búsqueda.

Parameters:
request (HttpRequest): La solicitud HTTP recibida.

Returns:
HttpResponse: La respuesta HTTP que muestra la lista de préstamos.

2.6 Module contents

CAPÍTULO 3

appusuarios package

3.1 Submodules

3.2 appusuarios.forms module

class appusuarios.forms.**UsuarioForm**(*data=None*, *files=None*, *auto_id='id_ %s'*, *prefix=None*, *initial=None*, *error_class=<class 'django.forms.utils.ErrorList'>*, *label_suffix=None*, *empty_permitted=False*, *instance=None*, *use_required_attribute=None*, *renderer=None*)

Bases: `ModelForm`

Formulario para la creación y edición de usuarios.

Attributes: - model (Usuario): Modelo asociado al formulario. - fields (list): Lista de campos a incluir en el formulario.

class Meta

Bases: `object`

Clase Meta para configuración adicional del formulario.

Attributes: - model (Usuario): Modelo asociado al formulario. - fields (list): Lista de campos a incluir en el formulario.

fields = '__all__'

model

alias de *Usuario*

base_fields = {'curso': <django.forms.fields.CharField object>, 'email': <django.forms.fields.EmailField object>, 'nombre_usuario': <django.forms.fields.CharField object>, 'rut_usuario': <django.forms.fields.CharField object>}

declared_fields = {}

property media

Return all media required to render the widgets on this form.

3.3 appusuarios.models module

class appusuarios.models.**Usuario**(**args*, ***kwargs*)

Bases: Model

Modelo que representa a un usuario en el sistema.

Attributes: - rut_usuario (str): Rut del usuario, clave primaria. - nombre_usuario (str): Nombre del usuario. - email (EmailField): Dirección de correo electrónico del usuario. - curso (str): Curso al que pertenece el usuario.

Methods: - __str__(): Devuelve una representación en cadena del usuario.

Meta: - verbose_name (str): Nombre amigable para el modelo en singular.

exception DoesNotExist

Bases: ObjectDoesNotExist

exception MultipleObjectsReturned

Bases: MultipleObjectsReturned

curso

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

email

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

nombre_usuario

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

objects = <django.db.models.manager.Manager object>

prestamo_set

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Parent.children is a ReverseManyToOneDescriptor instance.

Most of the implementation is delegated to a dynamically defined manager class built by create_forward_many_to_many_manager() defined below.

rut_usuario

A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

3.4 appusuarios.urls module

3.5 appusuarios.views module

appusuarios.views.**crear**(*request*)

Vista para crear un nuevo usuario.

Permite el ingreso de información para crear un nuevo usuario.

Args:
request (HttpRequest): La solicitud HTTP.

Returns:
HttpResponse: Respuesta HTTP que renderiza el formulario de creación de usuario.

appusuarios.views.**editar**(*request*, *rut_usuario*)

Vista para editar un usuario existente.

Permite modificar la información de un usuario existente.

Args:
request (HttpRequest): La solicitud HTTP. rut_usuario (str): El RUT del usuario a editar.

Returns:
HttpResponse: Respuesta HTTP que renderiza el formulario de edición de usuario.

appusuarios.views.**eliminar**(*request*, *rut_usuario*)

Elimina un usuario existente.

Args:
request (HttpRequest): La solicitud HTTP. rut_usuario (str): El RUT del usuario a eliminar.

Returns:
HttpResponse: Respuesta HTTP de redirección a la lista de usuarios.

appusuarios.views.**subir_usuarios**(*request*)

Sube usuarios desde un archivo CSV.

Args:
request (HttpRequest): La solicitud HTTP.

Returns:
HttpResponse: Respuesta HTTP de redirección a la lista de usuarios.

appusuarios.views.**usuarios**(*request*)

Vista para listar usuarios y realizar búsquedas.

Permite la paginación de usuarios y búsquedas por nombre, rut y email.

Args:
request (HttpRequest): La solicitud HTTP.

Returns:
HttpResponse: Respuesta HTTP que renderiza la lista de usuarios.

3.6 Module contents

CAPÍTULO 4

sistema package

4.1 Submodules

4.2 sistema.asgi module

ASGI config for sistema project.

It exposes the ASGI callable as a module-level variable named `application`.

For more information on this file, see https://docs.djangoproject.com/en/4.2/howto/deployment/asgi/

4.3 sistema.settings module

Django settings for sistema project.

Generated by “django-admin startproject” using Django 4.2.7.

For more information on this file, see https://docs.djangoproject.com/en/4.2/topics/settings/

For the full list of settings and their values, see https://docs.djangoproject.com/en/4.2/ref/settings/

4.4 sistema.urls module

4.5 sistema.wsgi module

WSGI config for sistema project.

It exposes the WSGI callable as a module-level variable named `application`.

For more information on this file, see https://docs.djangoproject.com/en/4.2/howto/deployment/wsgi/

4.6 Module contents

CAPÍTULO 5

Indices y tablas

- genindex
- modindex
- search

Índice de Módulos Python

a

s