



Universidad
Nacional
de Loja

[Desarrollo de Software en Ambientes Cloud]

Fundamentos para el desarrollo de Software en ambientes Cloud problema práctico

Software de administración de empleados VIVENTO

Tutor:

ROBERTH GUSTAVO FIGUEROA DIAZ

Elaborado:

Janneth Guamán

Lenin Quizhpe

11 de noviembre de 2022

Problemática:

La gestión de empleados en una empresa mediana o grande es de vital importancia para una solida gestión del departamento de recursos humanos, quienes son los encargados del seguimiento y bienestar de la fuerza laboral. Una empresa que dependa del recurso humano debe manejar un sistema que le permita identificar con facilidad las características propias del empleado, basados en su área de trabajo, experiencia laboral, enfermedades, discapacidad, riesgos, etc. Al conocer estas características y ser llevadas bajo un sistema de administración de empleados, permitirá mantener un ambiente laboral excelente, debido a que la empresa tendrá conocimiento del acontecer del personal y podrá actuar con precisión ante cualquier dificultad que pueda presentarse, así mismo se presentará una mejor organización en las tareas asignadas al empleado, de acuerdo a su experiencia y conocimiento en el área, mejorando la productividad de la empresa y tomando el recurso humano de manera más eficiente.

Tecnologías Utilizadas:

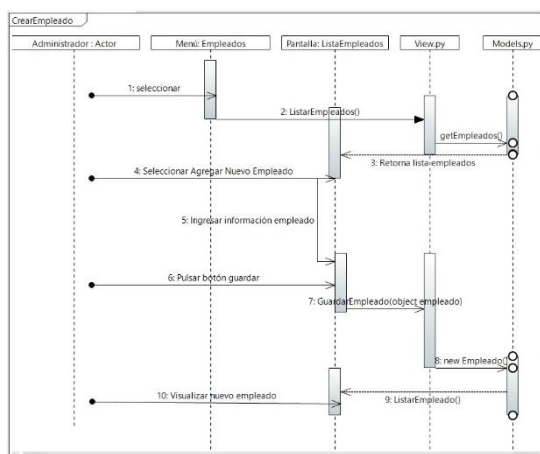
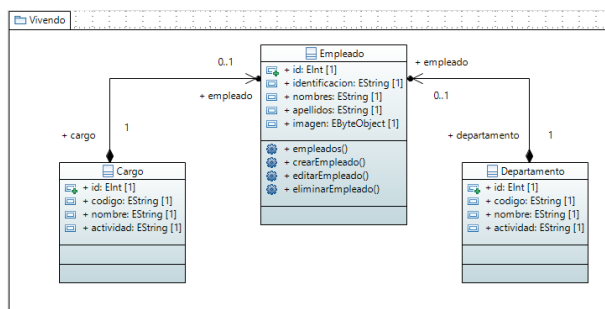
Lenguaje de Programación Python.

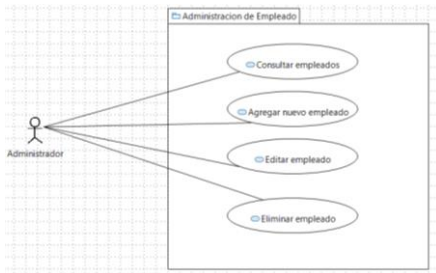
Framework Django

HTML 5

Base datos mysql y configuración en XAMPP

Diagramas:



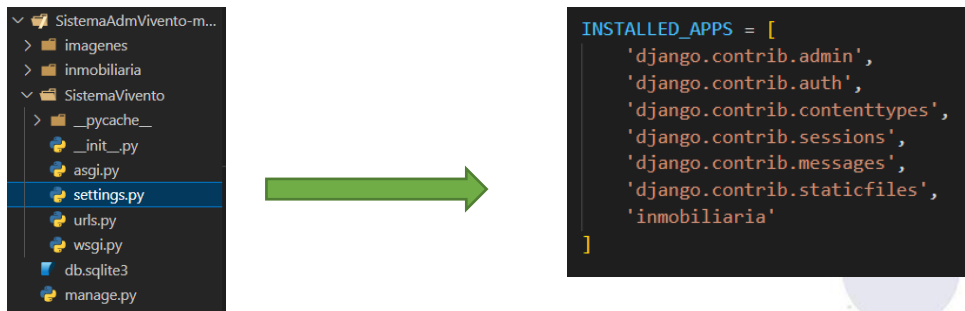


Desarrollo:

El sistema de gestión de empleados, consta de 3 módulos, “Empleado” (Realiza el registro, edición y eliminación de empleados que se vinculen a la empresa), a este empleado se le asigna un “Cargo”, y su pertenencia a algún “Departamento” de la empresa según sus necesidades. El backend a sido construido de la siguiente manera.

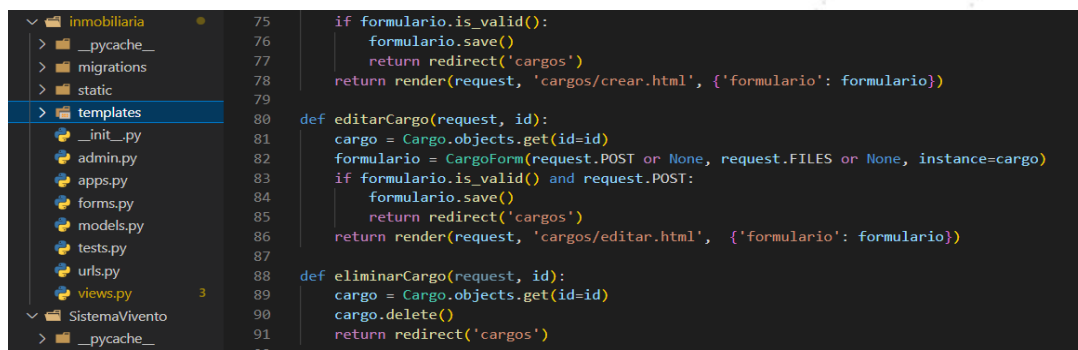
Creación de una aplicación de nombre inmobiliaria a partide del settings.py. como se observa en la Figura 1.

Figura 1: Creación de aplicación “Inmobiliaria”



En la aplicación de inmobiliaria se crea una carpeta “templates”, en la cual django busca las páginas (cargos, departamentos, empleados) que van a interactuar con el cliente mediante código html, y su acceso se lo realiza mediante el archivo view.py, las vistas se crean mediante funciones que generan la solicitud a cada una de las páginas de interfaz de usuario “Cargos”, “Empleados” y “Departamentos”, y sus respectivas acciones “crear”, “editar”, como se observa en la Figura 1.

Figura 1: Creación de clases para las vistas de interfaz de usuario.



Creación de plantilla base.html, la cual es una estructura de contenido que se repetirá en todas las vistas, y la creación de botones de menú de acceso a las diferentes páginas, como se observa en la Figura 2.

Figura 2: Creación de contenido y menú de acceso.

```

> cargos 23
> departamentos 24
> empleados 25
> paginas 26
> static 27
> base.html 28
> _init_.py 29
> 30
class="navbar navbar-expand navbar-light bg-light">
<div class="nav navbar-nav">
  <a class="nav-item nav-link active" href="{% url 'inicio' %}" aria-current="page">Inicio </a>
  <a class="nav-item nav-link active" href="{% url 'cargos' %}" aria-current="page">Cargos </a>
  <a class="nav-item nav-link active" href="{% url 'departamentos' %}" aria-current="page">Departamentos </a>
  <a class="nav-item nav-link active" href="{% url 'empleados' %}" aria-current="page">Empleados </a>
</div>

```

Creación de las vistas de gestión de datos del CRUD, el cual permitirán crear un formulario para el ingreso de los datos del empleado, editar o eliminar, según se requiera como se observa en la Figura 3a, en la Figura 3b la creación de el archivo form.html, el mismo que captura la información ingresada por el formulario.

Figura 3: vistas de gestión y formularios

```

> cargos 10
> departamentos 11
> empleados 12
> crear.html 13
> editar.html 14
> form.html 15
> index.html 16
> paginas 17
> 18
</div>
<div class="card-body">
  <h4 class="card-title">Datos del empleado</h4>
  {{ csrf_token }}
  {{ formulario.as_p }}
  <button type="submit" class="btn btn-primary">Guardar</button>
  <button type="" class="btn btn-primary">Cancelar</button>
</div>

```

a) Se crean las páginas de gestión de datos

```

> cargos 17
> departamentos 18
> empleados 19
> crear.html 20
> editar.html 21
> form.html 22
> index.html 23
> paginas 24
> static 25
> base.html 26
> 27
</div>
{{ csrf_token }}
{{ formulario.as_p }}
<button type="submit" class="btn btn-primary">Guardar</button>
<button type="" class="btn btn-primary">Cancelar</button>
</div>

```

b) Se crea el formulario que guardará la información ingresada.

El archivo index.html permitirá especificar la información contenida en cada página, para nuestro caso el listado de empleados registrados se empleará una tabla detallada como se observa en el Figura 4.

Figura 4: Presentación de información en la página empleados.

```

> empleados 7
> crear.html 8
> editar.html 9
> form.html 10
> index.html 11
> paginas 12
> static 13
> base.html 14
> _init_.py 15
> admin.py 16
> apps.py 17
> forms.py 18
> models.py 19
> 20
<div class="card">
  <div class="card-header">
    <a name="" id="" class="btn btn-primary" href="{% url 'crearEmpleado' %}" role="button">Agregar</a>
  </div>
  <div class="card-body">
    <h4 class="card-title">Empleado</h4>
    <div class="table-responsive">
      <table class="table table-primary">
        <thead>
          <tr>
            <th scope="col">Identificación</th>
            <th scope="col">Apellidos</th>
            <th scope="col">Nombres</th>
            <th scope="col">Cargo</th>

```

Para el acceso a las pantallas se requiere mediante el archivo `views.py` en el cual se encuentran las funciones que ligán con los archivos `html`, en la Figura 5, se presenta el llamado de las pantallas para mostrar, crear o editar el registro de empleados.

Figura 5: Acceso a las páginas de listar, crear, o editar información del empleado.



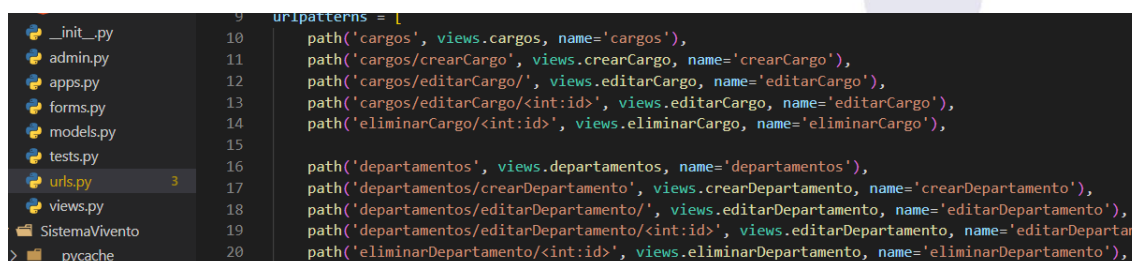
```

18 def empleados(request):
19     empleados = Empleado.objects.all()
20     return render(request, 'empleados/index.html', {'empleados': empleados})
21
22 def crearEmpleado(request):
23     formulario = EmpleadoForm(request.POST or None, request.FILES or None)
24     if formulario.is_valid():
25         formulario.save()
26         return redirect('empleados')
27     return render(request, 'empleados/crear.html', {'formulario': formulario})
28
29 def editarEmpleado(request, id):
30     empleado = Empleado.objects.get(id=id)
31     formulario = EmpleadoForm(request.POST or None, request.FILES or None, instance=empleado)
32     if formulario.is_valid() and request.POST:
33         formulario.save()
34         return redirect('empleados')
35

```

El archivo `url.py` permite establecer el acceso por url y navegación a las páginas de administración de empleados. Figura 6.

Figura 6: acceso a páginas por url.

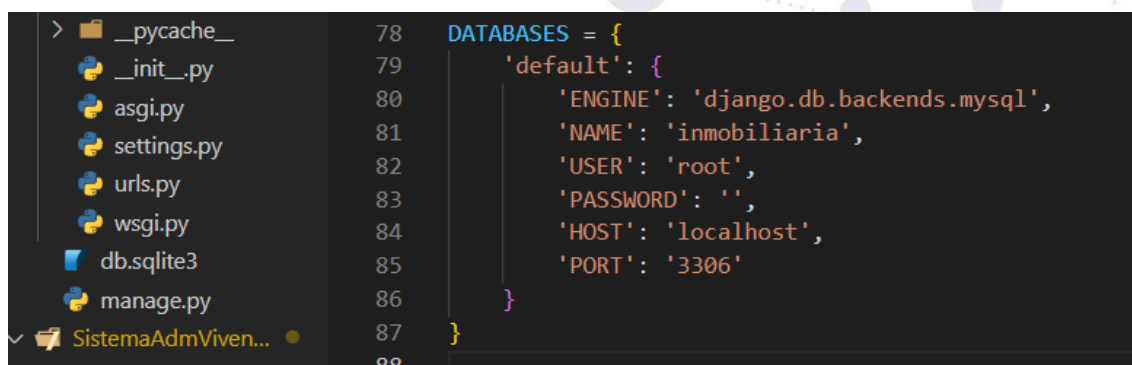


```

9 urlpatterns = [
10     path('cargos', views.cargos, name='cargos'),
11     path('cargos/crearCargo', views.crearCargo, name='crearCargo'),
12     path('cargos/editarCargo/', views.editarCargo, name='editarCargo'),
13     path('cargos/editarCargo/<int:id>', views.editarCargo, name='editarCargo'),
14     path('eliminarCargo/<int:id>', views.eliminarCargo, name='eliminarCargo'),
15
16     path('departamentos', views.departamentos, name='departamentos'),
17     path('departamentos/crearDepartamento', views.crearDepartamento, name='crearDepartamento'),
18     path('departamentos/editarDepartamento/', views.editarDepartamento, name='editarDepartamento'),
19     path('departamentos/editarDepartamento/<int:id>', views.editarDepartamento, name='editarDepartamento'),
20     path('eliminarDepartamento/<int:id>', views.eliminarDepartamento, name='eliminarDepartamento'),
21

```

Para la conexión a la base de datos accedemos a la raíz principal del proyecto y en `settings.py` y realizar la configuración que se requiera, en el presente caso se usó `mysql` y `XAMPP`, la Figura 7, se observa los parámetros para la base de datos.



```

78 DATABASES = {
79     'default': {
80         'ENGINE': 'django.db.backends.mysql',
81         'NAME': 'inmobiliaria',
82         'USER': 'root',
83         'PASSWORD': '',
84         'HOST': 'localhost',
85         'PORT': '3306'
86     }
87 }
88

```

En el archivo `_init_.html` creamos la interacción con la base de datos mediante el siguiente código

```
import pymysql
pymysql.install_as_MySQLdb()
```

Creación de modelos para ingreso de información, se realiza mediante el archivo models.py, en él se crean las clases para la estructura de datos a ser ingresados por el usuario, inclusive el ingreso de una imagen para identificar al empleado.

```
index.html
> paginas
> static
base.html
__init__.py
admin.py
apps.py
forms.py
models.py 2
tests.py
urls.py 3
views.py

class Empleado(models.Model):
    id = models.AutoField(primary_key=True)
    identificacion = models.CharField(max_length=10, verbose_name='Identificación')
    nombres = models.CharField(max_length=250, verbose_name='Nombres')
    apellidos = models.CharField(max_length=250, verbose_name='Apellidos')
    cargo = models.CharField(max_length=250, verbose_name='Cargo')
    departamento = models.ForeignKey(Departamento, verbose_name='Departamentos', null=False, blank=False)
    imagen = models.ImageField(upload_to='imagenes/', verbose_name='Imagen', null=True)

    def __str__(self):
        fila = "Nombres: " + self.nombres + " " + "Apellidos:" + self.apellidos
        return fila
```

Registro de información para el administrativo de django

```
from django.contrib import admin
from .models import *
# Register your models here.
admin.site.register(Cargo)
admin.site.register(Departamento)
admin.site.register(Empleado)
```

A continuación se presentan las interfaz de usuario en el que permitirá ingresar a nuevos empleados, editar su información, o eliminarlos, así mismo para las pantallas de cargo y departamentos.

UNIVERSIDAD NACIONAL DE LOJA

MAESTRIA EN INGENIERIA EN SOFTWARE

Sitio Administrativo de Vivienda y
Financiamiento VIVENTO

Administración de Empleados.

INTEGRANTES:

- Janneth Guamán
- Lenin Quizhpe



1859



Universidad
Nacional
de Loja

[Agregar nuevo empleado](#)

Empleado

Identificación	Apellidos	Nombres	Cargo	Imagen	Acciones
0604704395	Oscar Estalin	Vizúete Tactaiza	Ing. Sistemas		Editar Borrar
0605823656	Jose Fernando	Flores Guamán	Ing. Civil		Editar Borrar
0604068593	Lenin Israel	Quizipe Narvaéz	Ing. Electrico		Editar Borrar
1700253697	Janneth Patricia	Guamán Siguenza	Secretaria		Editar Borrar
1896320254	Anita Elizabeth	Ayala Vizúete	Contadora		Editar Borrar
0502369852	Gissela Elizabeth	Mendez Guerrero	Ing. Sistemas		Editar Borrar

[Agregar nuevo cargo](#)

Cargos

Codigo	Nombre	Actividad	Acciones
001	Jefe Financiero	Área Financiera	Editar Borrar
002	Jefe de TIC's	Coordinador de infraestructura de tecnologías	Editar Borrar

Cargos VIVENTO

[Agregar nuevo departamento](#)

Departamentos

Codigo	Nombre	Actividad	Acciones
001	Gerencia	Trámites administrativa	Editar Borrar
002	Atención al Cliente	Atender a los clientes / usuarios Vivento	Editar Borrar
003	Administrativo	Administración de información de vivento	Editar Borrar
004	TIC	Área tecnología de vivento	Editar Borrar