Preinforme: Desarrollo de Malla Curricular Interactiva

Programación de Computadores

Universidad Nacional de Colombia

Presentado por:

Robert Duan Montoya Cardona
Lennys Natalia Villamizar Rodriguez
Neyder Jhonathan Ruiz Peña
David Santiago Villalba Rodriguez

25 de agosto de 2024

Índice

1 Introducción

La programación de computadores ha revolucionado diversas áreas del conocimiento, proporcionando herramientas para automatizar procesos y analizar grandes volúmenes de datos de manera eficiente. En el ámbito académico, la planificación de los estudios es un aspecto crucial para el éxito de los estudiantes. Tradicionalmente, las mallas curriculares se presentan de forma estática, lo que dificulta la visualización de las dependencias entre cursos y la trayectoria académica.

Este proyecto busca innovar en este campo mediante la creación de una malla curricular interactiva que resalte dinámicamente los cursos dependientes de uno seleccionado, utilizando tecnologías modernas como Django, Django REST Framework y SvelteKit para ofrecer una experiencia visual intuitiva y eficiente. La implementación de esta herramienta no solo mejorará la comprensión de la estructura curricular por parte de los estudiantes, sino que también facilitará la toma de decisiones informadas sobre su trayectoria académica.

2 Pregunta Problema

¿Cómo se puede desarrollar una malla curricular interactiva que permita a los estudiantes de la Universidad Nacional de Colombia visualizar de manera dinámica las dependencias y prerrequisitos de los cursos, mejorando la planificación académica y la toma de decisiones?

3 Objetivo General

Desarrollar una plataforma web interactiva para la visualización dinámica de la malla curricular de los programas de pregrado de la Universidad Nacional de Colombia, que permita a los estudiantes identificar claramente los prerrequisitos y cursos dependientes, utilizando Django para el backend, Django REST Framework para la gestión de la API, y SvelteKit para el frontend interactivo.

4 Objetivos Específicos

- Implementar una base de datos estructurada en Django que almacene toda la información relevante de la malla curricular, incluyendo cursos, créditos, prerrequisitos y cursos dependientes.

- Desarrollar una API RESTful con Django REST Framework para manejar la comunicación entre el backend y el frontend, proporcionando datos en formato JSON.
- Diseñar una interfaz de usuario interactiva con SvelteKit que permita a los estudiantes seleccionar un curso y visualizar automáticamente las dependencias (cursos prerrequisitos y cursos dependientes) de manera dinámica y clara.
- Incluir herramientas de visualización avanzada como grafos, donde los cursos se representen como nodos conectados por sus relaciones de prerrequisitos.
- Implementar funciones de búsqueda y filtrado para facilitar la navegación por la malla curricular.
- Asegurar que la plataforma sea completamente responsiva y accesible para dispositivos móviles y usuarios con discapacidades.
- Integrar un sistema de autenticación para permitir a los estudiantes guardar sus progresos y personalizar su visualización de la malla curricular.

5 Justificación del Problema

La malla curricular de la Universidad Nacional de Colombia es compleja y, en muchos casos, los estudiantes tienen dificultades para comprender las relaciones entre los cursos, especialmente en cuanto a los prerrequisitos y las trayectorias académicas. Esto puede llevar a errores en la planificación, lo que resulta en retrasos en la graduación o la elección de cursos inadecuados.

Al desarrollar una malla curricular interactiva, se facilitará la comprensión de estas relaciones, optimizando la experiencia académica y la toma de decisiones por parte de los estudiantes. Además, el uso de tecnologías como Django y SvelteKit permitirá que la plataforma sea moderna, eficiente y capaz de manejar grandes volúmenes de datos de manera ágil y segura.

Los beneficios esperados de este proyecto incluyen:

- Mejora en la planificación académica de los estudiantes.
- Reducción de errores en la selección de cursos.
- Optimización del tiempo de graduación.

- Mayor comprensión de la estructura curricular por parte de estudiantes y docentes.
- Facilidad en la actualización y mantenimiento de la información curricular.
- Posibilidad de extender la herramienta a otros programas y facultades de la universidad.

6 Obtención y Descripción de los Datos

Los datos para este proyecto se obtendrán de la malla curricular oficial de la Universidad Nacional de Colombia, específicamente de programas como Química, cuyos detalles se han extraído del Acuerdo 510 de 2023[1]. Estos datos incluyen la lista de cursos, sus créditos, y las relaciones de prerrequisitos.

6.1 Estructura de Datos

La estructura de datos para cada curso incluirá:

- Código del curso
- Nombre del curso
- Número de créditos
- Semestre recomendado
- Lista de prerrequisitos (códigos de cursos)
- Lista de cursos dependientes (códigos de cursos)
- Área o componente del plan de estudios
- Descripción breve del curso

6.2 Fuentes de Datos

- Documentos oficiales de la Universidad Nacional de Colombia
- Sistema de Información Académica (SIA)
- Consultas directas a las coordinaciones de los programas académicos

[1] Universidad Nacional de Colombia, .^Acuerdo 510 de 2023", 2023.

6.3 Procesamiento de Datos

Los datos se extraerán de las fuentes mencionadas y se procesarán para su inclusión en la base de datos de Django. Este proceso incluirá:

- Limpieza y normalización de los datos
- Validación de la integridad de las relaciones entre cursos
- Transformación de los datos al formato requerido por el modelo de Django

7 Metodología

7.1 Backend: Django

7.1.1. Estructura del Modelo de Datos

Se creará un modelo de datos en Django que refleje la estructura de la malla curricular. Cada curso será un objeto en la base de datos, con campos como `nombre`, `créditos`, `prerrequisitos` y `cursos_dependientes`. Django ORM se encargará de la gestión de la base de datos.

```
from django.db import models

class Curso(models.Model):
    codigo = models.CharField(max_length=10, unique=True)
    nombre = models.CharField(max_length=200)
    creditos = models.IntegerField()
    semestre_recomendado = models.IntegerField()
    prerrequisitos = models.ManyToManyField('self',
   symmetrical=False, related_name='depende_de')
    cursos_dependientes = models.ManyToManyField('self',
   symmetrical=False, related_name='depende_de_este')
    area = models.CharField(max_length=100)
    descripcion = models.TextField()

    def __str__(self):
        return f"{self.codigo} - {self.nombre}"
```

Listing 1: Modelo de Datos en Django

7.1.2. API RESTful con Django REST Framework

Se diseñará una API RESTful utilizando Django REST Framework que permita acceder a los datos de los cursos. Se crearán endpoints que devuelvan la información de los cursos, junto con los prerrequisitos y los cursos dependientes.

```
from rest_framework import viewsets, serializers
from .models import Curso

class CursoSerializer(serializers.ModelSerializer):
    prerrequisitos = serializers.PrimaryKeyRelatedField(many=
    True, read_only=True)
    cursos_dependientes = serializers.PrimaryKeyRelatedField(
    many=True, read_only=True)

    class Meta:
        model = Curso
        fields = ['id', 'codigo', 'nombre', 'creditos', '
    semestre_recomendado',
                  'prerrequisitos', 'cursos_dependientes', '
    area', 'descripcion']

class CursoViewSet(viewsets.ModelViewSet):
    queryset = Curso.objects.all()
    serializer_class = CursoSerializer

    def get_queryset(self):
        queryset = Curso.objects.all()
        codigo = self.request.query_params.get('codigo', None
    )
        if codigo is not None:
            queryset = queryset.filter(codigo=codigo)
        return queryset
```

Listing 2: API RESTful con Django REST Framework

7.2 Frontend: SvelteKit

7.2.1. Interfaz de Usuario Interactiva

El frontend en SvelteKit permitirá a los usuarios seleccionar un curso de la malla curricular y ver de manera gráfica las dependencias. Se utilizarán gráficos interactivos para representar las relaciones entre cursos, y la interfaz se actualizará dinámicamente según la selección del usuario.

```
<script>
```

```
  import { onMount } from 'svelte';
  import { fetchCursoDetails } from '../services/api';

  export let cursoId;
  let curso = null;

  onMount(async () => {
    curso = await fetchCursoDetails(cursoId);
  });
</script>

<div class="curso-details">
  {#if curso}
    <h2>{curso.nombre}</h2>
    <p>C digo: {curso.codigo}</p>
    <p>Cr ditos: {curso.creditos}</p>
    <p>Semestre recomendado: {curso.semestre_recomendado}</p>

    <h3>Prerrequisitos:</h3>
    <ul>
      {#each curso.prerrequisitos as prerrequisito}
        <li>{prerrequisito.nombre}</li>
      {/each}
    </ul>

    <h3>Cursos Dependientes:</h3>
    <ul>
      {#each curso.cursos_dependientes as dependiente}
        <li>{dependiente.nombre}</li>
      {/each}
    </ul>
  {:else}
    <p>Cargando informaci n del curso...</p>
  {/if}
</div>
```

Listing 3: Componente Svelte para la visualización de cursos

7.2.2. Visualización Gráfica con D3.js

Para la visualización avanzada, se utilizará D3.js para construir un grafo interactivo que muestre los cursos y sus relaciones de prerrequisitos. Al seleccionar un curso, el grafo resaltará los cursos dependientes y los prerrequisitos.

```
<script>
  import { onMount } from 'svelte';
  import * as d3 from 'd3';
  import { fetchMallaCurricular } from '../services/api';
```

```
let svg;
let data = { nodes: [], links: [] };

onMount(async () => {
  data = await fetchMallaCurricular();
  renderGraph();
});

function renderGraph() {
  const width = 800;
  const height = 600;

  const simulation = d3.forceSimulation(data.nodes)
    .force("link", d3.forceLink(data.links).id(d => d.id))
    .force("charge", d3.forceManyBody())
    .force("center", d3.forceCenter(width / 2, height / 2))
 ;

  const svg = d3.select(svg)
    .attr("viewBox", [0, 0, width, height]);

  const link = svg.append("g")
    .selectAll("line")
    data.links)
    .join("line")
    .attr("stroke", "#999")
    .attr("stroke-opacity", 0.6);

  const node = svg.append("g")
    .selectAll("circle")
    .data(data.nodes)
    .join("circle")
    .attr("r", 5)
    .attr("fill", d => d.group === 1 ? "#ff0000" : "#1f77b4
");

  node.append("title")
    .text(d => d.id);

  simulation.on("tick", () => {
    link
      .attr("x1", d => d.source.x)
      .attr("y1", d => d.source.y)
      .attr("x2", d => d.target.x)
      .attr("y2", d => d.target.y);

    node
      .attr("cx", d => d.x)
```

```
        .attr("cy", d => d.y);
    });
  }
</script>

<svg bind:this={svg}></svg>
```

Listing 4: Visualización de grafo con D3.js

7.3 Infraestructura y Desarrollo

7.3.1. Despliegue en el VPS

El proyecto será desplegado en un servidor VPS usando Nginx como proxy inverso para manejar el tráfico hacia las aplicaciones Django y SvelteKit. Se utilizará Gunicorn para servir la aplicación Django, y la comunicación entre el backend y el frontend se hará a través de la API.

```
server {
    listen 80;
    server_name malla.ejemplo.com;

    location / {
        proxy_pass http://localhost:3000;  # SvelteKit
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    location /api {
        proxy_pass http://localhost:8000;  # Django
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

Listing 5: Configuración de Nginx

7.3.2. Pruebas y Ajustes

A lo largo del desarrollo, se realizarán pruebas de usuario para asegurar que la interfaz sea intuitiva y que los datos se presenten de manera clara y comprensible. Se implementarán las siguientes estrategias de prueba:

- Pruebas unitarias para el backend Django
- Pruebas de integración para la API
- Pruebas de componentes para el frontend SvelteKit
- Pruebas de usabilidad con un grupo selecto de estudiantes
- Pruebas de rendimiento y carga

8 Arquitectura del Sistema

La arquitectura del sistema se basará en un modelo cliente-servidor, con una clara separación entre el backend y el frontend. Esta arquitectura permitirá una mayor flexibilidad y escalabilidad del sistema.

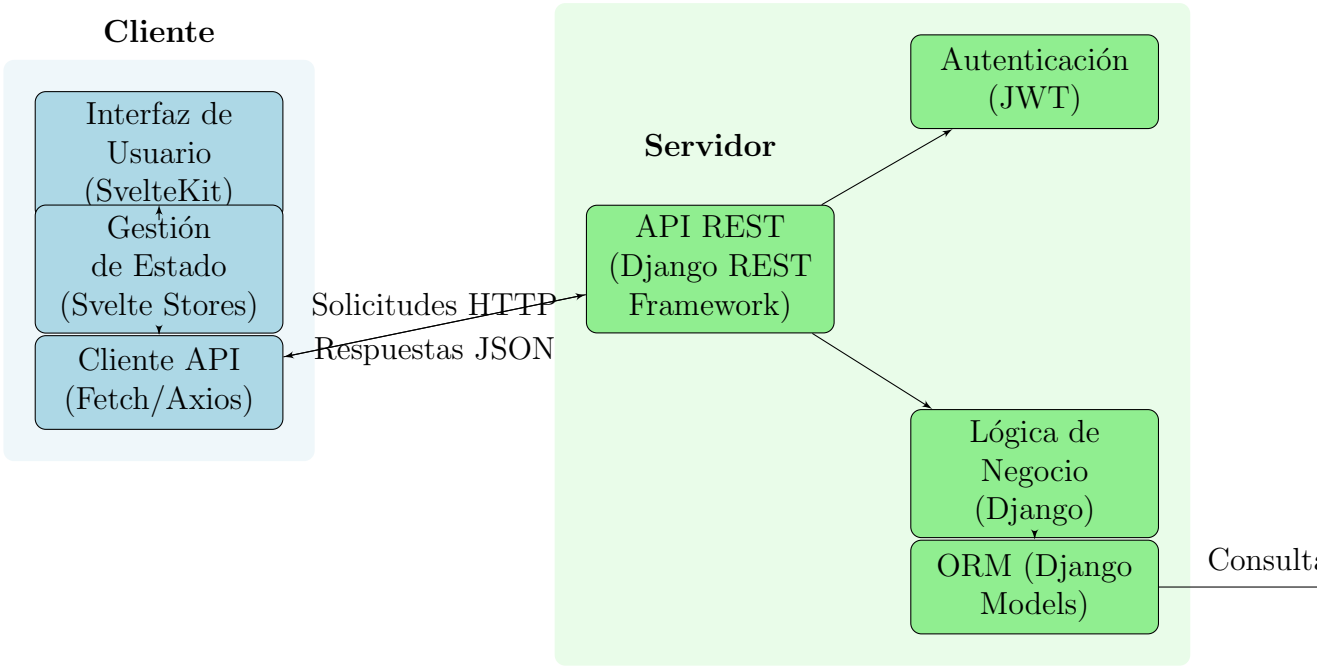

Figura 1: Arquitectura Detallada del Sistema de Malla Curricular Interactiva

8.1 Componentes Principales

- **Frontend (SvelteKit):** Interfaz de usuario interactiva y responsiva.
- **Backend (Django):** Lógica de negocio y gestión de datos.

- **API (Django REST Framework):** Capa de comunicación entre frontend y backend.
- **Base de Datos (PostgreSQL):** Almacenamiento persistente de datos.

9 Consideraciones de Seguridad

La seguridad es un aspecto crucial en el desarrollo de aplicaciones web. Se implementarán las siguientes medidas de seguridad:

- Autenticación y autorización robusta utilizando JWT (JSON Web Tokens).
- Encriptación de datos sensibles en la base de datos.
- Implementación de HTTPS para todas las comunicaciones.
- Protección contra ataques comunes como XSS, CSRF, y SQL Injection.
- Limitación de tasas para prevenir ataques de fuerza bruta.

10 Plan de Trabajo

El desarrollo del proyecto se dividirá en las siguientes fases:

1. **Fase 1: Planificación y Diseño (2 semanas)**
   - Análisis detallado de requisitos
   - Diseño de la arquitectura del sistema
   - Creación de mockups y prototipos de la interfaz de usuario
2. **Fase 2: Desarrollo del Backend (3 semanas)**
   - Implementación de modelos de datos en Django
   - Desarrollo de la API RESTful
   - Configuración de la base de datos
3. **Fase 3: Desarrollo del Frontend (3 semanas)**
   - Implementación de la interfaz de usuario en SvelteKit

- Integración con la API del backend
- Desarrollo de la visualización de grafos con D3.js

4. **Fase 4: Integración y Pruebas (2 semanas)**
   - Integración de todos los componentes del sistema
   - Realización de pruebas exhaustivas
   - Corrección de errores y optimización del rendimiento
5. **Fase 5: Despliegue y Documentación (1 semana)**
   - Despliegue del sistema en el servidor de producción
   - Elaboración de la documentación técnica y de usuario
   - Preparación del informe final del proyecto

11 Conclusiones y Trabajo Futuro

El desarrollo de una malla curricular interactiva para la Universidad Nacional de Colombia representa un avance significativo en la forma en que los estudiantes pueden visualizar y planificar su trayectoria académica. Este proyecto no solo mejorará la experiencia de los estudiantes, sino que también proporcionará valiosas herramientas para la administración académica.

Como trabajo futuro, se podrían considerar las siguientes extensiones del proyecto:

- Integración con el sistema de matrícula de la universidad
- Implementación de un sistema de recomendaciones basado en el historial académico del estudiante
- Expansión a otros programas y facultades de la universidad
- Desarrollo de una aplicación móvil complementaria
- Implementación de análisis predictivo para identificar posibles dificultades en la trayectoria académica de los estudiantes

12 Referencias

Referencias

[1] Django Software Foundation. (2023). Django Documentation. `https://docs.djangoproject.com/`

[2] Django REST Framework. (2023). Django REST Framework Documentation. `https://www.django-rest-framework.org/`

[3] Svelte. (2023). Svelte Documentation. `https://svelte.dev/docs`

[4] Bostock, M. (2023). D3.js - Data-Driven Documents. `https://d3js.org/`

[5] Universidad Nacional de Colombia. (2023). Acuerdo 510 de 2023. Bogotá, Colombia.