

Informe Final del Proyecto de

Inteligencia Artificial

NOMBRE DEL PROYECTO: *Cooking Courses*: Chatbot educativo con análisis de sentimientos a cursos de cocina.

Integrantes: Richard Padilla, Ariel Sánchez

Carrera / Curso: Tecnología Superior en Desarrollo de software

Docente: Ing. Yadira Franco

Fecha: 28/07/2025

Índice

1. Introducción
2. Objetivos
3. Motivación
4. Estado del Arte
5. Alcance del Proyecto
6. Arquitectura del Sistema
7. Desarrollo del Modelo
8. Integración del Sistema
9. Funcionalidad y Validación
10. Trabajo Colaborativo en GitHub
11. Conclusiones
12. Recomendaciones y Trabajos Futuros
13. Anexos

1. Introducción

El presente proyecto tiene como propósito desarrollar una página web con inteligencia artificial enfoque en el ámbito educativo, lo cual fusiona dos funcionalidades principales: un Chatbot informativo que responde preguntas frecuentes sobre cursos de cocina, y un sistema de análisis de sentimientos que clasifica los comentarios de los usuarios como positivos o negativos. Ambas funcionalidades fueron agregadas en una interfaz amigable construida con flask, usando técnicas de procesamiento de lenguaje natural (NPL).

Esta aplicación ayuda al ámbito educativo como herramienta interactiva que no solo entrega información, sino también la capacidad de poder analizar las respuestas de los usuarios para mejorar el servicio con el que se ha enfocado. El sistema Cooking Courses representa un primer paso hacia la integración de IA en contextos educativos especializados, como la enseñanza culinaria.

2. Objetivos

2.1 Objetivo General

Desarrollar una aplicación en Python que integre un modelo de análisis de sentimientos capaz de clasificar comentarios como positivos o negativos, y un Chatbot informativo sobre cursos de cocina, con una interfaz gráfica amigable y estructura modular. La solución deberá ser funcional, fácil de usar y replicable.

2.2 Objetivos Específicos

- Entrenar un modelo que clasifique el sentimiento de comentarios de usuarios.
- Diseñar un Chatbot informativo que responda dudas sobre los cursos.
- Integrar ambos componentes en una interfaz gráfica.
- Crear una API para consumo desde la interfaz.
- Visualizar métricas y resultados del modelo.
- Publicar el proyecto en GitHub con documentación clara y completa.

3. Motivación

La necesidad de mejorar la experiencia del usuario en plataformas educativas inspiró este proyecto. En el área culinaria, contar con información clara sobre los cursos y medir la opinión del usuario permite mejorar la calidad del servicio. Muchos estudiantes suelen sentirse perdidos al buscar información en línea sobre cocina, y una solución que les oriente y, al mismo tiempo, analice su opinión, podría ser clave para fidelizar a los usuarios.

Al integrar un Chatbot con funciones informativas y un modelo de análisis de sentimientos, se busca proporcionar una herramienta que facilite el aprendizaje, motive la participación y permita identificar áreas de mejora en los cursos ofrecidos. Este enfoque hace uso de tecnologías actuales para crear una experiencia más cercana y personalizada.

4. Estado del Arte

Los sistemas de análisis de sentimientos son ampliamente utilizados en redes sociales y sitios de reseñas. Herramientas como TextBlob, VADER y modelos más recientes basados en transformers (como BERT) permiten clasificar texto con alta eficacia y de forma contextualizada. Estas herramientas han demostrado ser útiles para comprender la opinión de los usuarios en una amplia variedad de dominios, desde el comercio electrónico hasta la política.

En el campo educativo, los Chatbots han ganado terreno como herramientas de asistencia para responder preguntas frecuentes, orientar al estudiante y proporcionar contenidos adaptativos. Plataformas como Coursera, edX y Duolingo han implementado sistemas similares. Nuestro proyecto toma lo mejor de ambos mundos y lo adapta al campo de la enseñanza culinaria.

5. Alcance del Proyecto

El sistema Cooking Courses está diseñado como una herramienta de asistencia educativa especializada en cursos de cocina. Su alcance comprende dos funcionalidades clave: la capacidad de analizar sentimientos en comentarios escritos por los usuarios, y la de brindar respuestas informativas mediante un Chatbot educativo. Esta combinación permite al sistema funcionar tanto como medio de retroalimentación, como canal de consulta directa para estudiantes o interesados en los cursos.

Por un lado, el análisis de sentimientos se limita a detectar si una opinión escrita es positiva o negativa. Esto permite obtener una percepción general del grado de satisfacción de los usuarios con respecto al curso o a la plataforma. Esta funcionalidad es útil para los administradores, quienes pueden tomar decisiones basadas en el tipo de comentarios que reciben.

Por otro lado, el Chatbot responde a una serie de preguntas frecuentes sobre los cursos de cocina. Entre los temas que aborda están el costo de los cursos, la modalidad (presencial o virtual), la duración estimada y si se entrega o no un certificado digital al finalizar. El Chatbot tiene una base de datos con respuestas programadas, por lo que no realiza una comprensión semántica compleja.

La solución incluye una interfaz gráfica diseñada en HTML que facilita la interacción del usuario con ambos módulos. El sistema también se encuentra respaldado por una API desarrollada, que se encarga de procesar los datos del usuario y entregar una respuesta desde el modelo de IA o desde el Chatbot.

Este proyecto está planteado como un prototipo funcional, capaz de ser escalado en el futuro con mejoras como mayor variedad de emociones, y un sistema conversacional más complejo.

No se considera en esta versión:

- Generación de respuestas emocionales o conversacionales complejas.
- Interacción multilingüe o con contexto extendido.
- Generación de respuestas emocionales o conversacionales complejas.

6. Arquitectura del Sistema

6.1 Estructura general

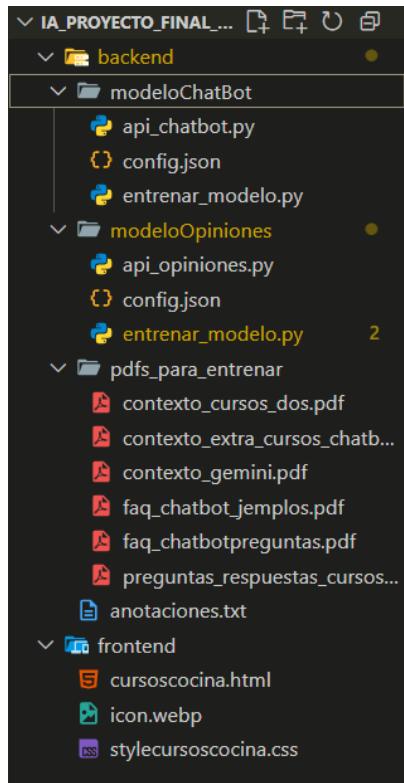


Figura 6.1.1 Estructura del Sistema

El proyecto se encuentra organizado en una estructura dividida principalmente en dos áreas: el backend y el frontend. Esta organización ayuda en la separación de responsabilidades, lo que permite mayor claridad y organización en el desarrollo del sistema.

La **Figura 6.1.** representa cómo están distribuidas las carpetas y archivos en el entorno de desarrollo. Esta estructura evidencia la organización lógica del proyecto en componentes independientes para el análisis de sentimientos, el Chatbot y la interfaz visual.

6.2 Tecnologías utilizadas

Se utilizaron tecnologías modernas para la creación de la aplicación de Cooking Courses que ayudara a poder tener una visualización e interacción lógica con el cliente. Estas librerías son las siguientes:

- Python: lenguaje principal del desarrollo.
- HTML: para la construcción rápida de interfaces web.
- Desarrollo de APIs REST utilizando flask.
- Scikit-learn: herramienta para el entrenamiento del modelo de clasificación.
- Pandas / NLTK: procesamiento y limpieza de datos de texto.
- Git / GitHub: control de versiones y colaboración en equipo.

7. Desarrollo del Modelo

7.1 Dataset usado

Para el desarrollo de este proyecto se utilizó para la creación del modelo de análisis de sentimientos dataset tradicionales. En el modelo de Chatbot en su lugar, se optó por construir nuestro propio dataset a partir de documentos en formato PDF, elaborados manualmente con contenido relevante.

Estos documentos fueron diseñados y descargados para dos fines distintos. Por un lado, se utiliza textos con opiniones en inglés como positivas o negativas para entrenar el modelo de análisis de sentimientos. Por otro, se redactaron documentos con información detallada sobre los cursos de cocina (duración, costos, modalidad, certificaciones, etc.), los cuales fueron utilizados como base de conocimiento para el Chatbot.

Este enfoque permitió personalizar completamente los datos para que se ajustaran al contexto educativo-culinario del proyecto, garantizando mayor coherencia en las respuestas del Chatbot y precisión en la clasificación de sentimientos. Además, nos permitió evitar problemas de licencia o sesgo presentes en datasets genéricos disponibles en internet.

7.2 Preprocesamiento

Debido a que los datos utilizados para entrenar ambos modelos fueron generados a partir de documentos PDF y descargados desde una plataforma exterior, fue necesario aplicar un preprocesamiento cuidadoso para estructurar adecuadamente la información y garantizar resultados óptimos. Este proceso se realizó tanto para las opiniones que alimentarían el modelo de análisis de sentimientos, como para los textos que formarían parte de la base del Chatbot.

Para el módulo de análisis de sentimientos, se siguieron las siguientes etapas:

- Conversión del contenido a minúsculas.
- Eliminación de signos de puntuación y caracteres especiales.
- Tokenización, que consiste en dividir las frases en palabras individuales.
- Lematización, para reducir las palabras a su forma raíz.

En el caso del Chatbot, se realizó una limpieza semántica adicional de los textos extraídos desde los PDF para construir una base estructurada de preguntas frecuentes y respuestas, agrupadas por temas como tipo de curso, modalidad, duración y certificación. Este trabajo permitió mejorar la precisión de las respuestas del Chatbot al momento de recibir consultas específicas.

7.3 Creación y entrenamiento del modelo

7.3.1 Análisis de sentimientos

Para el análisis de sentimientos, se construyó un pipeline con *TfidfVectorizer* para vectorizar el texto y *LogisticRegression* como clasificador. Los datos fueron divididos en conjuntos de entrenamiento y prueba en una proporción de 80/20. A través de múltiples pruebas, se comprobó que este enfoque proporcionaba resultados adecuados para nuestro problema binario.

El modelo fue entrenado utilizando los comentarios creados manualmente, asegurando que existiera un balance entre las opiniones positivas y negativas. Una vez entrenado, el modelo fue encapsulado en una API para ser consumido desde la interfaz.

A continuación se presenta la API REST y el modelo entrenado tal y como se ven en la **Figura 7.3.1.1** y en la **Figura 7.3.1.2**.

```
EXPLORER ... <--> entrenar_modelo.py 9+ x  
IA_PROYECTO_FINAL... D P O S backend  
backend > modeloOpiniones > entrenar_modelo.py > ...  
117 vectorizer = TfidfVectorizer(  
118     ngram_range=tuple(config["ngram_range"]),
119     min_df=config["min_df"]
120 )
121 X_train_vec = vectorizer.fit_transform(tqdm(X_train, desc="Vectorizando train"))
122 X_test_vec = vectorizer.transform(tqdm(X_test, desc="Vectorizando test"))
123
124 # Entrenamos modelo de regresión logística
125 model = LogisticRegression(max_iter=config["max_iter"])
126 for _ in tqdm(range(1), desc="Entrenando modelo"):
127     | model.fit(X_train_vec, y_train)
128
129 # Guardamos modelo
130 with open("modelo_sentimiento.pkl", "wb") as f:
131     | pickle.dump(vectorizer, model, f)
132 print("Modelo guardado como modelo_sentimiento.pkl")
133
134 # ===== MÉTRICAS DE EVALUACIÓN =====
135 y_pred = model.predict(X_test_vec)
136 print(classification_report(y_test, y_pred))
137
138 # Crear y guardar matriz de confusión
139 cm = confusion_matrix(y_test, y_pred)
140 sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xtickLabels=["Negativa", "Positiva"], ytickLabels=["Negativa", "Positiva"])
141 plt.xlabel("Predicción")
142 plt.ylabel("Real")
143 plt.title("Matriz de confusión")
144 plt.savefig("matriz_confusion.png")
145 plt.close()
146 print("Matriz de confusión guardada como matriz_confusion.png")
147
148 # Mostrar tiempo total de entrenamiento
149 total = time.time() - inicio
150 print(f"Tiempo total de entrenamiento y guardado: {total:.2f} segundos")
```

Figura 7.3.1.1 Entrenar modelo

```

EXPLORER
...
IA_PROYECTO_FINAL_ASANCHEZ_RPADILLA
  - backend
    - modeloChatBot
    - modeloOpiniones
      - api_opiniones.py
      - config.json
      - entrenar_modelo.py
      - matriz_confusion.png
      - modelo_sentimiento.pkl
    - pdfs_para_entrenar
    - anotaciones.txt
  - frontend
    - cursoscicina.html
    - icon.webp
    - stylecursoscicina.css

api_opiniones.py 2 ×
backend > modeloOpiniones > api_opiniones.py > ping

9
10   # ===== CARGA DEL MODELO ENTRENADO =====
11   # Se carga el vectorizador TF-IDF y el modelo de regresión logística desde el archivo .pkl
12   with open("modelo_sentimiento.pkl", "rb") as f:
13       vectorizer, model = pickle.load(f)
14
15   # ===== RUTA PRINCIPAL PARA ANALIZAR OPINIONES =====
16   @app.route("/analizar", methods=["POST"])
17   def analizar():
18       data = request.get_json() # Se obtiene el JSON enviado en la petición
19       texto = data.get("texto", "") # Se extrae el campo "texto" del JSON
20
21       # Validación: Si no se recibe texto, se devuelve error 400
22       if not texto:
23           return jsonify({"error": "No se recibió texto"}), 400
24
25       # Transformar el texto recibido con el vectorizador entrenado
26       texto_vec = vectorizer.transform([texto])
27
28       # Usar el modelo para predecir si es positivo (1) o negativo (0)
29       pred = model.predict(texto_vec)[0]
30       sentimiento = "Positiva" if pred == 1 else "Negativa"
31
32       # Devolver el resultado en formato JSON
33       return jsonify({"sentimiento": sentimiento})
34
35   # ===== RUTA DE PRUEBA PARA VER SI LA API ESTÁ ACTIVA =====
36   @app.route("/ping")
37   def ping():
38       return "API activa"
39
40   # ===== EJECUTAR LA APLICACIÓN LOCALMENTE =====
41   # Ejecuta la API en el puerto 5000.
42   if __name__ == "__main__":
43       app.run(host="0.0.0.0", port=5000)

```

Figura 7.3.1.2 API de análisis de sentimientos

7.3.2 Chatbot

Para el Chatbot, se utilizó una estructura basada en recuperación de respuestas desde un archivo indexado de texto. Se implementó una lógica de coincidencia y filtrado por palabras clave, complementada con un sistema de consulta sobre el contenido previamente limpiado de los PDFs. Aunque no se entrenó un modelo con aprendizaje automático para esta parte, se consideró como una forma eficaz y controlada de entregar respuestas informativas de forma rápida y precisa.

A continuación se presenta la API REST y el modelo entrenado tal y como se ven en la **Figura 7.3.2.1** y en la **Figura 7.3.2.2**.

```

EXPLORER ... entrena_modelo.py 7
IA_PROYECTO_FINAL_ASANCHEZ_RPADILLA backend > modeloChatBot > entrena_modelo.py > ...
backend
└── backend
    ├── api_chatbot.py
    ├── config.json
    └── entrena_modelo.py 7
        ├── matriz_confusion_chatbot.pkl
        ├── modelo_chatbot.pkl
        └── respuestas.pkl
    ├── modeloOpiniones
    ├── pdfs_para_entrenar
    └── anotaciones.txt
└── frontend
    ├── cursoscocina.html
    ├── icon.webp
    └── stylecursoscocina.css
> OUTLINE

83     pickle.dump(respuestas_dict, f)
84
85     # Vectorización de texto con TF-IDF (mejora la representación del texto)
86     vectorizer = TfidfVectorizer(
87         ngram_range=tuple(config["ngram_range"]),
88         min_df=config["min_df"]
89     )
90     X = vectorizer.fit_transform(preguntas)
91
92     # Entrenamiento del modelo de clasificación con regresión logística
93     model = LogisticRegression(max_iter=config["max_iter"])
94     model.fit(X, intenciones)
95
96     # Guardar el modelo entrenado
97     with open("modelo_chatbot.pkl", "wb") as f:
98         pickle.dump((vectorizer, model), f)
99
100    # Evaluación del modelo con métricas estándar
101    X_train, X_test, y_train, y_test = train_test_split(preguntas, intenciones, test_size=0.2, random_state=42)
102    X_train_vec = vectorizer.transform(X_train)
103    X_test_vec = vectorizer.transform(X_test)
104    y_pred = model.predict(X_test_vec)
105
106    # Mostrar clasificación y matriz de confusión en consola y guardar gráfica
107    print(classification_report(y_test, y_pred))
108    labels = sorted(list(set(y_test)))
109    cm = confusion_matrix(y_test, y_pred, labels=labels)
110    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=labels, yticklabels=labels)
111    plt.title("Matriz de confusión Chatbot")
112    plt.xlabel("Predicción")
113    plt.ylabel("Real")
114    plt.tight_layout()
115    plt.savefig("matriz_confusion_chatbot.png")
116    plt.close()
117    print("Entrenamiento completo. Modelo y métricas guardadas.")

```

Figura 7.3.2.1 Entrenar modelo

```

EXPLORER ... api_chatbot.py 2
IA_PROYECTO_FINAL_ASANCHEZ_RPADILLA backend > modeloChatBot > api_chatbot.py > ...
backend
└── backend
    ├── config.json
    ├── entrena_modelo.py
    ├── matriz_confusion_chatbot.pkl
    ├── modelo_chatbot.pkl
    └── respuestas.pkl
    ├── modeloOpiniones
    ├── pdfs_para_entrenar
    └── anotaciones.txt
└── frontend
    ├── cursoscocina.html
    ├── icon.webp
    └── stylecursoscocina.css
> OUTLINE

34     def preguntar():
35         p for p, i in fqs
36         if i == intencion_pred and (curso_usuario is None or curso_usuario in p.lower())
37     ]
38
39     # Si no hay preguntas relacionadas con esa intención y curso
40     if not preguntas_de_intencion:
41         return jsonify({"respuesta": "Puedes especificar el curso sobre el que quieres información?"})
42
43     # Buscamos la pregunta más parecida a la del usuario dentro de las filtradas
44     mejor_pregunta = None
45     mejor_similitud = 0
46     for p in preguntas_de_intencion:
47         similitud = difflib.SequenceMatcher(None, texto, p.lower()).ratio()
48         if similitud > mejor_similitud:
49             mejor_similitud = similitud
50             mejor_pregunta = p
51
52     # Si no hay suficiente similitud, devolvemos una respuesta genérica
53     if mejor_similitud < 0.5:
54         respuesta = "Puedes especificar tu pregunta sobre los cursos? Puedo ayudarte con precios, duración, contenido y requisitos."
55     else:
56         # Si hay buena similitud, buscamos la respuesta correspondiente
57         respuesta = respuestas_dict.get((intencion_pred, mejor_pregunta), "Lo siento, no tengo una respuesta para eso.")
58
59     # Devolvemos la respuesta en formato JSON
60     return jsonify({"respuesta": respuesta})
61
62
63     # Endpoint de prueba para verificar si la API está corriendo correctamente
64     @app.route("/ping")
65     def ping():
66         return "API del chatbot activa"
67
68
69     # Ejecutamos el servidor Flask si este archivo es el principal
70     if __name__ == "__main__":
71         app.run(host="0.0.0.0", port=5001)
72
73
74
75
76
77
78
79
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
159
160
161
162
163
164
165
166
167
168
169
169
170
171
172
173
174
175
176
177
178
179
179
180
181
182
183
184
185
186
187
188
189
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
209
210
211
212
213
214
215
216
217
218
219
219
220
221
222
223
224
225
226
227
227
228
229
229
230
231
232
233
234
235
235
236
237
237
238
239
239
240
241
241
242
242
243
243
244
244
245
245
246
246
247
247
248
248
249
249
250
250
251
251
252
252
253
253
254
254
255
255
256
256
257
257
258
258
259
259
260
260
261
261
262
262
263
263
264
264
265
265
266
266
267
267
268
268
269
269
270
270
271
271
272
272
273
273
274
274
275
275
276
276
277
277
278
278
279
279
280
280
281
281
282
282
283
283
284
284
285
285
286
286
287
287
288
288
289
289
290
290
291
291
292
292
293
293
294
294
295
295
296
296
297
297
298
298
299
299
300
300
301
301
302
302
303
303
304
304
305
305
306
306
307
307
308
308
309
309
310
310
311
311
312
312
313
313
314
314
315
315
316
316
317
317
318
318
319
319
320
320
321
321
322
322
323
323
324
324
325
325
326
326
327
327
328
328
329
329
330
330
331
331
332
332
333
333
334
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
```

7.4 Métricas de evaluación

7.4.1 Análisis de sentimientos

Tras el entrenamiento del modelo de análisis de sentimientos con aproximadamente 100,000 ejemplos (equilibrados entre clases positivas y negativas), se obtuvieron los siguientes resultados de desempeño que se muestran en la siguiente **Figura 7.4.1.1**.

	precision	recall	f1-score	support
0	0.93	0.93	0.93	48439
1	0.93	0.93	0.93	51561
accuracy			0.93	100000
macro avg	0.93	0.93	0.93	100000
weighted avg	0.93	0.93	0.93	100000
weighted avg	0.93	0.93	0.93	100000

Figura 7.4.1.1 Tabla de reporte de clasificación

Estas métricas reflejan un modelo robusto y bien balanceado, capaz de distinguir adecuadamente entre opiniones positivas y negativas. La precisión indica que, de todas las predicciones positivas realizadas, el 93% fueron correctas. El recall señala que el modelo identificó correctamente el 93% de todas las opiniones positivas existentes. El F1-score resume el equilibrio entre ambas métricas.

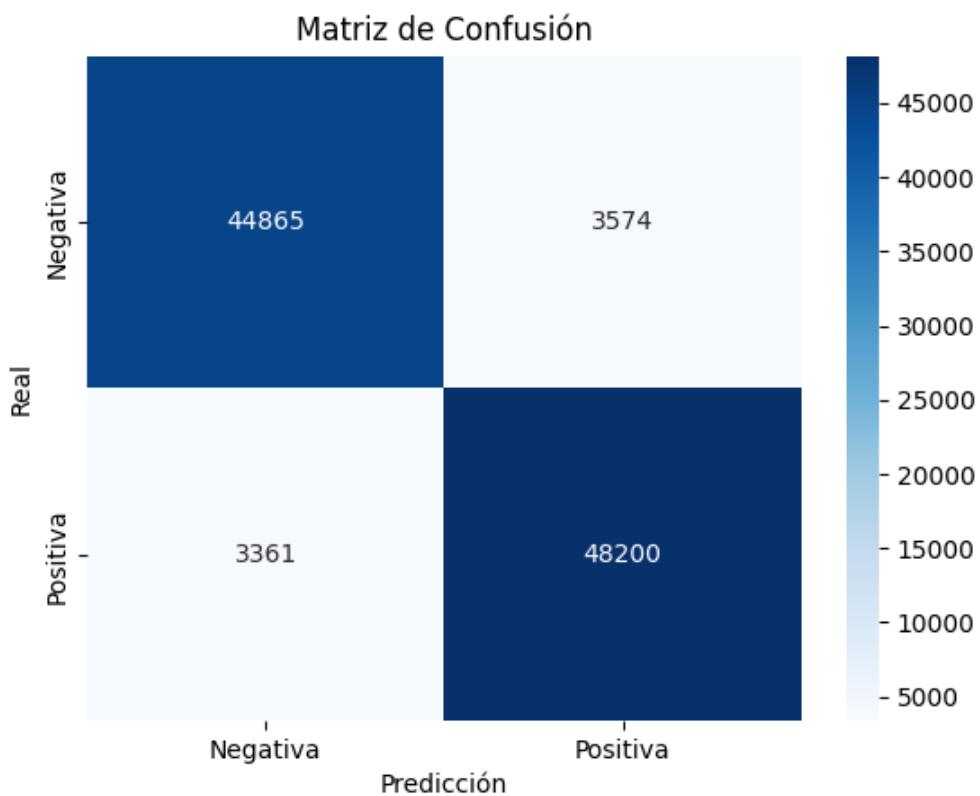


Figura 7.4.1.2 Matriz de confusión

La siguiente **Figura 7.4.1.2** muestra lo siguiente:

- Se clasificaron correctamente 44,865 opiniones negativas y 48,200 opiniones positivas.
- Se produjeron 3,574 falsos positivos y 3,361 falsos negativos, cifras relativamente bajas considerando el volumen total.
- El modelo fue entrenado con *TfidfVectorizer* y *LogisticRegression*, logrando generalizar bien sin sobreajuste.

Este nivel de rendimiento confirma que el modelo es adecuado para su propósito dentro del sistema Cooking Courses, donde se requiere una clasificación binaria clara y eficiente para interpretar la retroalimentación del usuario.

7.4.2 Chatbot

El modelo del Chatbot fue evaluado con un conjunto de datos de prueba etiquetado manualmente en cuatro categorías: *certificado*, *clases*, *contenido* y *precio*. Este modelo tenía como propósito identificar correctamente la intención detrás de una pregunta del usuario para luego brindar una respuesta preprogramada desde su base de datos textual.

Durante las pruebas, se obtuvieron las siguientes métricas de rendimiento como se visualiza en la **Figura 7.4.2.1**:

	precision	recall	f1-score	support
certificado	1.00	1.00	1.00	12
clases	1.00	1.00	1.00	9
contenido	1.00	1.00	1.00	7
precio	1.00	1.00	1.00	37
accuracy			1.00	65
macro avg	1.00	1.00	1.00	65
macro avg	1.00	1.00	1.00	65
macro avg	1.00	1.00	1.00	65
weighted avg	1.00	1.00	1.00	65

Figura 7.4.2.1 Tabla de reporte de clasificación

Estos resultados indican que el modelo fue capaz de clasificar perfectamente las intenciones de las preguntas en cada una de las categorías definidas. En otras palabras, no se cometieron errores de predicción durante la evaluación, lo que evidencia una comprensión clara de las temáticas por parte del sistema.

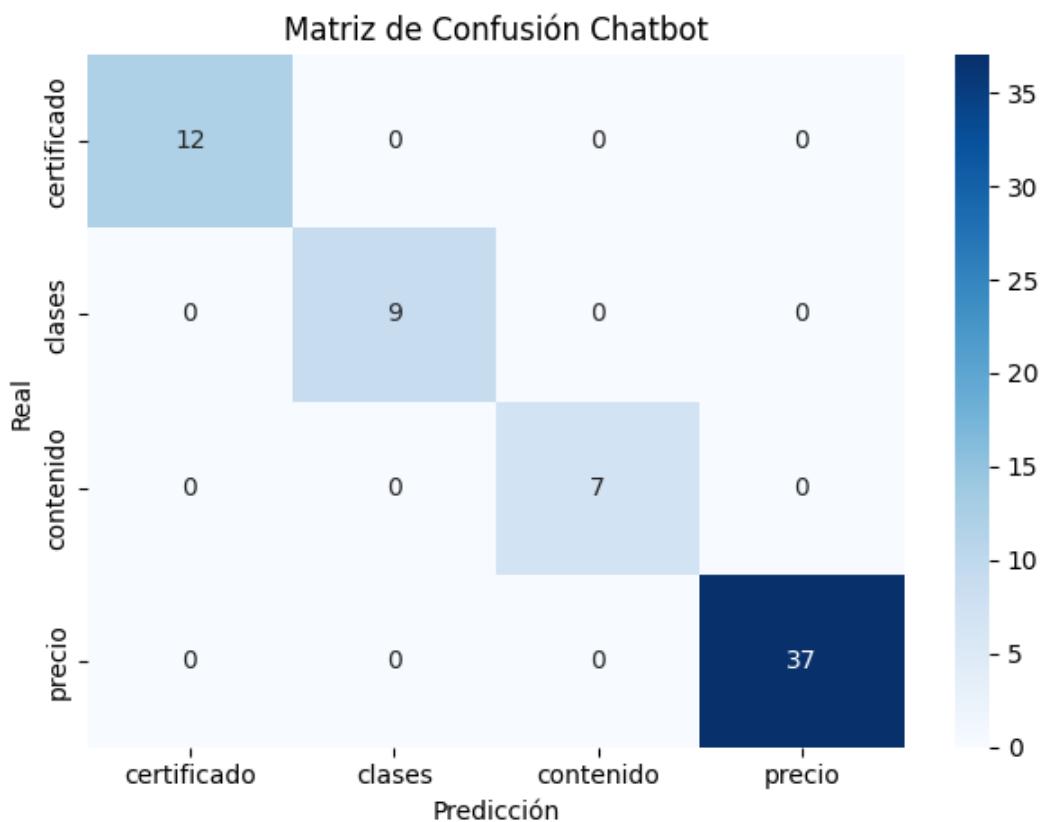


Figura 7.4.2.2 Matriz de confusión

La siguiente **Figura 7.4.2.2** muestra lo siguiente:

- 12 preguntas sobre certificado correctamente clasificadas.
- 9 preguntas sobre clases clasificadas con exactitud.
- 7 preguntas de tipo contenido sin error alguno.
- 37 preguntas sobre precio correctamente predichas.

Todas las preguntas del conjunto de prueba fueron correctamente clasificadas. Esto refleja que el entrenamiento con datos personalizados y controlados desde los PDFs diseñados por el equipo fue altamente efectivo, permitiendo al chatbot entregar respuestas precisas al usuario final. Además, al no depender de modelos complejos, se asegura una rápida respuesta y bajo consumo de recursos.

8. Integración del Sistema

8.1 Interfaz gráfica (GUI)

Se usó HTML y CSS para construir una interfaz que permite al usuario interactuar de forma amigable y visual. Desde esta interfaz se puede consultar información de cursos y enviar comentarios para análisis.

A continuación se presenta la interfaz gráfica del proyecto como se visualiza en la **Figura 8.1.1.**

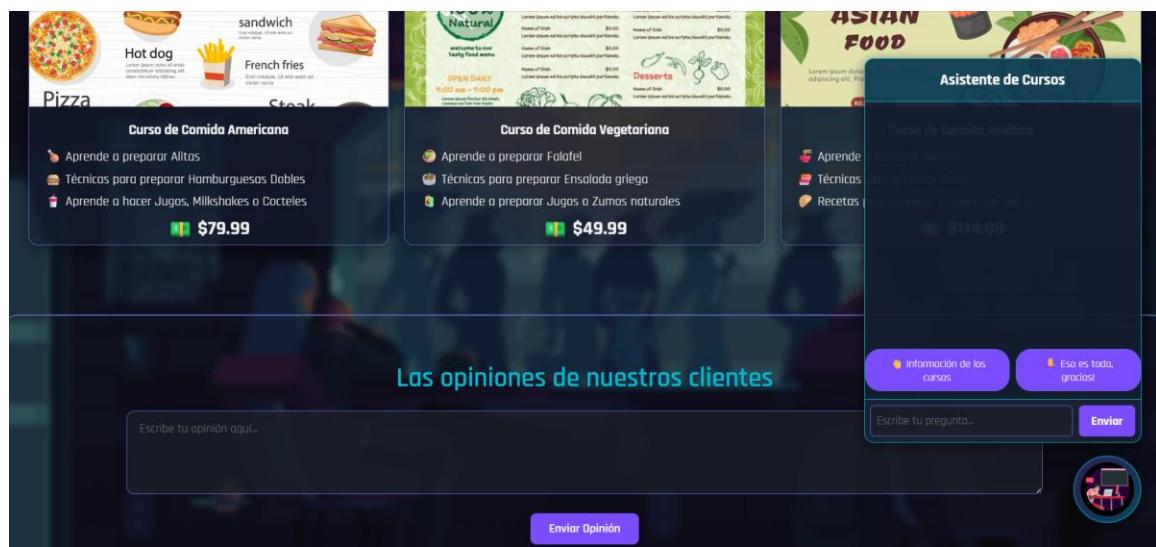


Figura 8.1.1 Pagina web y Chatbot

8.2 API creada

Mediante Flask se crearon dos APIs RESTful que funcionan como intermediarias entre la interfaz gráfica y los modelos desarrollados. Cada una de ellas cumple una función específica dentro del sistema.

8.2.1 Análisis de sentimientos

La primera API corresponde al módulo de **análisis de sentimientos**. Esta expone un endpoint que recibe un comentario del usuario y retorna su clasificación como "positivo" o "negativo". Internamente, el comentario es procesado por el modelo entrenado y devuelto como una respuesta una vez analizada dentro de los dataset para entrenar. Esta API permite medir la opinión del usuario de forma automática y en tiempo real.

Para un mejor entendimiento de lo citado anteriormente, se puede visualizar en la siguiente **Figura 7.3.1.2**.

8.2.2 Chatbot

La segunda API está dedicada al funcionamiento del **Chatbot educativo**. Esta recibe preguntas del usuario relacionadas con los cursos de cocina, como: "¿Cuánto cuesta el curso de cocina mexicana?", "¿El curso incluye certificado?", o "¿Qué duración tiene el curso de comida italiana?". A través de una lógica basada en búsqueda por coincidencias y estructuras predefinidas, la API del Chatbot retorna una respuesta concreta en función de la base de datos textual cargada desde archivos PDF y configuraciones previas.

Para un mejor entendimiento de lo citado anteriormente, se puede visualizar en la siguiente **Figura 7.3.2.2**.

8.3 Consumo del API desde la GUI

8.3.1 Análisis de sentimiento:

La interfaz envía solicitudes POST a la API cada vez que un cliente ingresa un comentario. El resultado se muestra inmediatamente en pantalla, facilitando la comprensión del funcionamiento del modelo.

A continuacion se presenta el consumo de la API desde GIU como se visualiza en la **Figura 8.3.1.1.**

Opinión	Clasificación
Buenos cursos, la profesora Mishelle Padilla que me dio los closes estaba riquisima	Positivo
Que cursos de mierda, una perdida de tiempo y dinero, no los recomiendo para nadie	Positivo
Malos cursos, demasiados temas y poca profundidad	Positivo
Bad courses	Negativo
Do not buy this courses	Negativo
No compren estos cursos	Negativo
Mal curso el de comida rapida americana, esperaba convertirme en gordon ramsey pero solo me convertí en un gordo, no valen la pena	Negativo
Me gusta el curso de comida italiana, me recuerda a mi abuelito el que golpeaba a mi abuela si le ponía piña a la pizza	Positivo

Figura 8.3.1.1 Consumo de la API desde GUI de reseñas.

8.3.2 Chatbot:

La interfaz del Chatbot es solo hacer consultas GET en la API cada vez que un cliente ingresa una pregunta sobre los cursos de cocina. El resultado se muestra inmediatamente en pantalla respondiendo al cliente su cuestionamiento, facilitando la comprensión del funcionamiento del modelo.

A continuacion se presenta el consumo de la API desde GIU como se visualiza en la **Figura 8.3.2.1.**

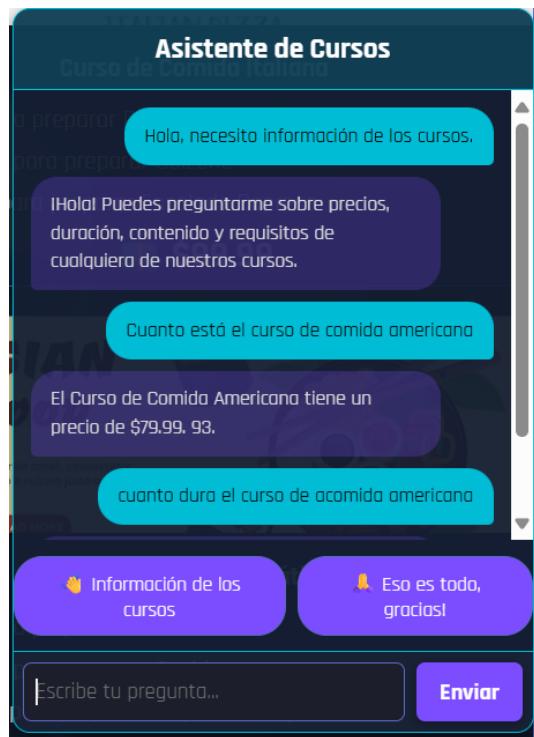


Figura 8.3.2.1 Consumo de la API desde GUI del Chatbot.

8.4 Ejercicio práctico

8.4.1 Análisis de sentimiento

Ejemplo:

reseña: *Los cursos son muy buenos, me gusta que tenga variedad de cursos, sigan así de excelente.*

respuesta: *Positiva*

reseña: *No compren estos cursos*

respuesta: *Negativa*

Este flujo simula un caso real y muestra la funcionalidad integrada del sistema. A continuacion se presenta de manera visual:

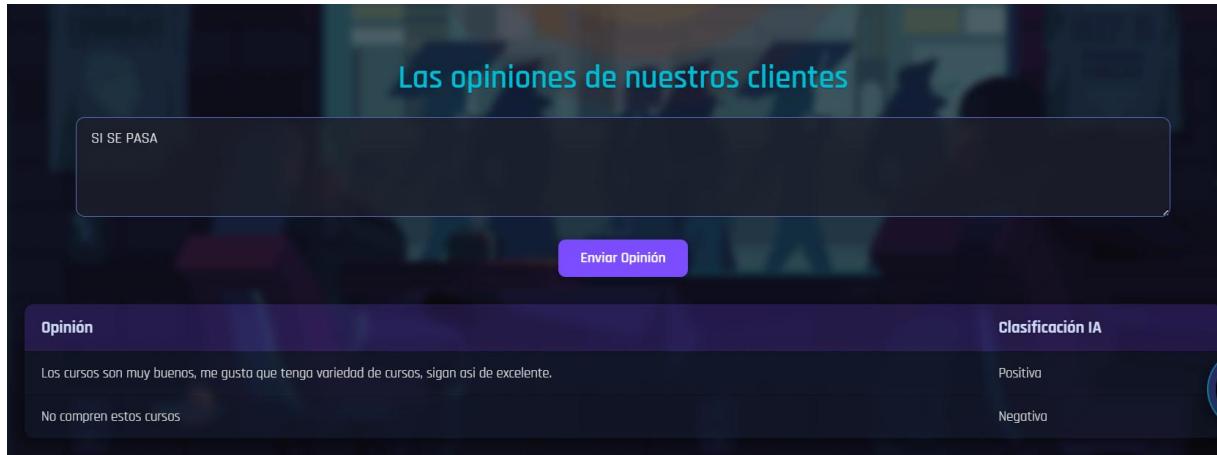


Figura 8.4.1.1 Funcionalidad del sistema en un caso práctico real.

8.4.2 Chatbot

Ejemplo:

Chatbot: ¡Hola! Puedes preguntarme sobre precios, duración, contenido y requisitos de cualquiera de nuestros cursos.

Cliente: Cuanto esta el curso de comida vegetariana?

Chatbot: El Curso de Comida Vegetariana tiene un precio de \$49.99.

Cliente: y incluye certificado?

Chatbot: Sí, al finalizar obtienes certificado digital.

Este flujo simula un caso real y muestra la funcionalidad integrada del sistema. A continuacion se presenta de manera visual:

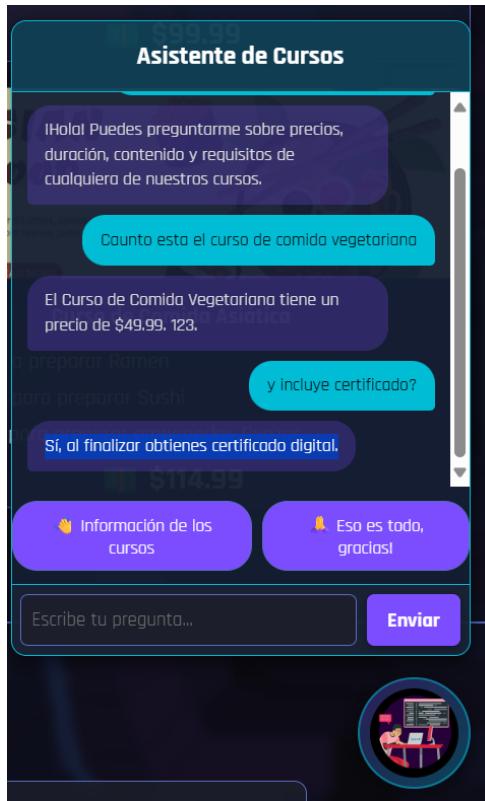


Figura 8.4.2.2 Funcionalidad del sistema en un caso práctico real.

9. Funcionalidad y Validación

Durante las fases de prueba del sistema Cooking Courses, se validaron tanto las funcionalidades principales como el comportamiento de cada componente, abarcando desde el análisis de sentimientos hasta las respuestas del Chatbot y su integración con la interfaz de cliente.

Resultados Positivos

- El sistema es capaz de clasificar comentarios en positivos o negativos con rapidez y un nivel aceptable de precisión.
- El Chatbot responde a preguntas frecuentes relacionadas con los cursos de cocina, tales como precios, tipos de cocina, certificados, entre otros.

- La comunicación entre el frontend, la API de análisis de sentimientos y el modelo del Chatbot se mantiene estable y coherente durante el uso de la plataforma.
- Desde la interfaz se puede interactuar con ambas funcionalidades (Chatbot y análisis de sentimientos), permitiendo probarlas de manera intuitiva y fluida.
- Se ejecutaron pruebas unitarias para validar cada componente por separado, y posteriormente se realizaron pruebas de sistema que confirmaron el correcto funcionamiento en conjunto.

Observaciones y Mejoras Implementadas

- Aunque el clasificador de sentimientos cumple su función en la mayoría de los casos, se observaron algunas predicciones incorrectas (falsos positivos o negativos), especialmente cuando los comentarios contienen sarcasmo o lenguaje ambiguo.
- En ciertos casos, el Chatbot no responde con la coherencia esperada, o bien entrega información irrelevante al contexto de la pregunta. Esto se debe a limitaciones en la fuente de conocimiento y en la lógica de extracción de respuestas.
- Inicialmente, el Chatbot se entrenaba a partir de múltiples archivos PDF extensos, lo cual afectaba el rendimiento y generaba tiempos de respuesta elevados. Para resolver esto:
 - Se redujo el contenido de los PDFs.
 - Se reorganizó su estructura en fragmentos más manejables y temáticamente clasificados.
 - Se implementó una arquitectura modular que permite modificar fácilmente los contenidos de aprendizaje del Chatbot.

10. Trabajo Colaborativo en GitHub

- README documentado con instrucciones claras.
- Evidencia de commits de cada integrante del grupo.
- Organización de carpetas separadas por backend y frontend donde se clasifican para los datos, modelos, interfaz y API.

Repositorio:

https://github.com/ArielSanchez12/IA_PROJECTO_FINAL_ASanchez_RPadilla.git

Despliegue:

<https://proyectofinal-ia-asanchez-rpadilla.netlify.app>

11. Conclusiones

El proyecto logró cumplir con el objetivo principal de integrar técnicas de inteligencia artificial en un entorno educativo, demostrando el potencial del procesamiento de lenguaje natural para mejorar la experiencia de aprendizaje y el acceso a la información. La combinación de análisis de sentimientos e Chatbot representa un aporte significativo en la forma en que los clientes pueden relacionarse con sistemas educativos automatizados, haciéndolos más dinámicos, accesibles y centrados en el cliente.

El módulo de análisis de sentimientos ha demostrado ser una herramienta útil para captar automáticamente la percepción de los usuarios respecto a los cursos ofrecidos. Esta funcionalidad permite obtener retroalimentación de forma rápida, objetiva y escalable, lo que resulta valioso para instituciones que deseen mejorar sus contenidos o metodologías a partir de las opiniones de los estudiantes. Aunque se presentaron algunas predicciones erróneas, estas abren la posibilidad de futuras mejoras mediante la optimización del modelo o la incorporación de técnicas más avanzadas de aprendizaje.

Por su parte, el Chatbot informativo ha facilitado el acceso a información clave sobre los cursos de cocina, resolviendo dudas frecuentes de forma automatizada. Su implementación demostró que es posible construir asistentes educativos con respuestas relevantes y coherentes, aunque también evidenció la necesidad de una base de conocimientos bien estructurada para mejorar la calidad y precisión de las respuestas.

En resumen, este trabajo evidencia cómo la inteligencia artificial puede aportar significativamente al campo educativo, tanto en la mejora de la interacción con los clientes como en la toma de decisiones basada en datos.

12. Recomendaciones y Trabajos Futuros

A partir de los resultados obtenidos y de las observaciones realizadas durante el desarrollo y pruebas del sistema, se proponen varias líneas de mejora y expansión que podrían potenciar significativamente la funcionalidad, accesibilidad y aplicabilidad en contextos reales:

1. Incluir más clases de sentimientos (neutral y mixto)

Actualmente, el modelo de análisis de sentimientos se limita a clasificar las opiniones en positivas o negativas. Sin embargo, muchas opiniones expresadas por los usuarios pueden tener un tono neutral o contener sentimientos mixtos. Incluir estas clases adicionales permitiría un análisis más matizado y realista de las percepciones, mejorando la precisión del sistema y su utilidad para la toma de decisiones.

2. Mejorar la comprensión contextual del Chatbot

Si bien el Chatbot es capaz de responder preguntas frecuentes, su capacidad de comprensión aún es limitada en cuanto al contexto de las conversaciones. Incorporar técnicas avanzadas de modelado conversacional (como seguimiento de contexto, historial de diálogo o modelos de lenguaje más complejos) permitiría ofrecer respuestas más coherentes, naturales y adaptadas a cada usuario.

3. Ampliar la base de conocimientos del Chatbot

Una mejora sustancial sería la expansión de la base de datos de respuestas y la incorporación de una mayor cantidad de preguntas frecuentes. Esto permitiría al sistema cubrir un espectro más amplio de inquietudes relacionadas con los cursos de cocina, reduciendo las ocasiones en que el Chatbot no puede ofrecer una respuesta relevante.

13. Anexos

13.1 Capturas de la interfaz grafica

BIENVENIDO A COOKING COURSES

CURSOS DE COCINA

En nuestra Escuela de Cocina ofrecemos una amplia variedad de cursos de cocina. Tanto si eres un aficionado a la cocina como un experto cocinero nuestros talleres de cocina son ideales para ti. Mejorarás tus conocimientos gastronómicos, recetas y técnicas culinarias y pasarás un rato muy divertido.

Por eso, en nuestros talleres de cocina, aprenderás no solo a preparar deliciosos platos sino también a ampliar tus creaciones culinarias. Nuestros chefs expertos te ayudarán a preparar cada receta, te animarán a participar y a practicar en todo momento.

¿A qué esperas para disfrutar de una inolvidable y divertida experiencia gastronómica en COOKING COURSES?

Todos nuestros talleres de cocina



Todos nuestros talleres de cocina



Curso de Comida Rápida	Curso de Comida Mexicana	Curso de Comida Italiana
Aprende a preparar Pizza Técnicas para preparar Hamburguesas Papas fritas crocantes	Aprende a preparar Tacos Técnicas para preparar Burritos Receta para preparar Guacamole	Aprende a preparar Pesto Técnicas para preparar Calzone Receta para preparar Ensalada Caprese
\$29.99	\$59.99	\$99.99

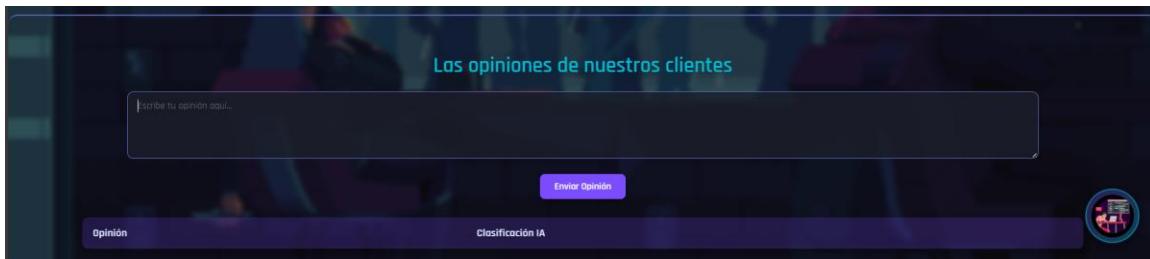
Curso de Comida Americana	Curso de Comida Vegetariana	Curso de Comida Asiática
Aprende a preparar Altas Técnicas para preparar Hamburguesas Dobles Aprende a hacer Jugs, Milkshakes o Cocteles	Aprende a preparar Falafel Técnicas para preparar Ensalada griego Aprende a preparar Jugs o Zumos naturales	Aprende a preparar Romanos Técnicas para preparar Sushi Recetas para preparar empanados Gyozas
\$79.99	\$49.99	\$114.99

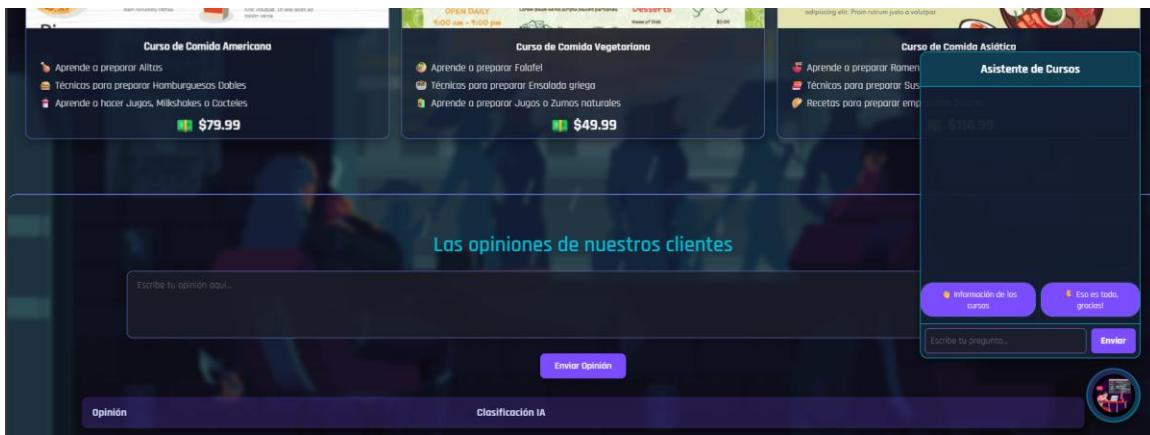
Las opiniones de nuestros clientes

Escribe tu opinión aquí...

Enviar Opinión

Opinión Clasificación IA





13.2 Capturas del código

EXPLORER

```

IA_PROYECTO_FINAL_ASANCHEZ_RPADILLA
└── backend
    ├── modeloChatBot
    └── modeloOpiniones
        └── pdfs_para_entrenar
            └── anotaciones.txt
└── frontend
    ├── cursoscocina.html
    ├── icon.webp
    └── stylecursoscocina.css

```

frontend > cursoscocina.html > html > body

```

2   <html lang="es">
3     <body>
4       <div id="chatbot-window">
5         </div>
6         <div id="chatbot-input-area">
7           <input id="chatbot-input" type="text" placeholder="Escribe tu pregunta...">
8           <button id="chatbot-send">Enviar</button>
9         </div>
10    </div>
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161

```

<!-- Script para opiniones y chatbot -->

```

<script>
document.getElementById('submitReview').addEventListener('click', async () => {
  const reviewText = document.getElementById('reviewInput').value.trim();
  if (reviewText === '') return;

  const response = await fetch('http://localhost:5000/analizar', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ texto: reviewText })
  });

  let sentimiento = "Error";
  if (response.ok) {
    const data = await response.json();
    sentimiento = data.sentimiento;
  } else {
    sentimiento = "Error al conectar con IA";
  }

  const table = document.getElementById('reviewTableBody');
  const newRow = table.insertRow();
  const cellReview = newRow.insertCell(0);
  const cellClassification = newRow.insertCell(1);

```

RichardPadilla07 (1 day ago) · Ln 136, Col 3 · Spaces: 2 · UTF-8 · CRLF · 4 HTML · 8 · Port: 5500

EXPLORER

IA_PROYECTO_FINAL...

- backend
 - modeloChatBot
 - modeloOpiniones
 - pdfs_para_entrenar
 - anotaciones.txt
- frontend
 - cursoscocina.html
 - icon.webp
 - stylecursoscocina.css

frontend > stylecursoscocina.css > ...

```

1  /* === FUENTE GLOBAL === */
2  input, button, textarea, select {
3    font-family: 'Rajdhani', sans-serif;
4  }
5
6  body {
7    font-family: 'Rajdhani', sans-serif;
8    margin: 0;
9    padding: 0;
10   background: url('https://giffles.alphacoders.com/213/213164.gif') no-repeat center center fixed;
11   background-size: cover;
12   min-height: 100vh;
13   text-align: center;
14   color: #eae0e0;
15   position: relative;
16   font-size: 25px;
17 }
18
19 /* Overlay difuminado */
20 body::before {
21   content: "";
22   position: fixed;
23   top: 0;
24   left: 0;
25   width: 100%;
26   height: 100%;
27   background: rgba(0, 0, 0, 0.4);
28   backdrop-filter: blur(4px);
29   z-index: 0;
30 }
31
32 body > * {
33   position: relative;
34   z-index: 1;
35 }
36
37 /* === EFECTO GLOBAL DE ESCALA === */

```

> OUTLINE

> TIMELINE

EXPLORER

IA_PROYECTO_FINAL...

- backend
 - modeloChatBot > api_chatbot.py
 - api_chatbot.py
 - config.json
 - entrenar_modelo.py
 - matriz_confusion_chat...
 - modelo_chatbot.pkl
 - respuestas.pkl
- frontend
 - cursoscocina.html
 - icon.webp
 - stylecursoscocina.css

backend > modeloChatBot > api_chatbot.py > ...

```

1 from flask import Flask, request, jsonify
2 import pickle
3 import difflib
4 from flask_cors import CORS
5
6 app = Flask(__name__)
7 CORS(app)
8
9 with open("modelo_chatbot.pkl", "rb") as f:
10   vectorizer, model = pickle.load(f)
11
12 with open("respuestas.pkl", "rb") as f:
13   respuestas_dict = pickle.load(f)
14
15 # Cargar todas las preguntas e intenciones
16 faqs = []
17 for (intencion, pregunta) in [(k[0], k[1]) for k in respuestas_dict.keys()]:
18   faqs.append((pregunta, intencion))
19
20 def detectar_curso(texto, lista_cursos):
21   texto = texto.lower()
22   for curso in lista_cursos:
23     if curso in texto:
24       return curso
25   return None
26
27 @app.route("/preguntar", methods=["POST"])
28 def preguntar():
29   data = request.get_json()
30   texto = data.get("texto", "").lower()
31   if not texto:
32     return jsonify({"error": "No se recibió ninguna pregunta"}), 400
33
34   # Respuestas fijas para saludo/despedida
35   if "hola" in texto or "informacion" in texto:
36     return jsonify({"respuesta": "¡Hola! Puedes preguntarme sobre precios, duración, contenido y requisitos de nuestros"})

```

> OUTLINE

EXPLORER

```

IA_PROYECTO_FINAL... ...
  backend > modeloChatBot > entrenar_modelo.py > ...
    api_chatbot.py
    config.json
    entrenar_modelo.py
      matriz_confusion_chat...
      modelo.chatbot.pkl
      respuestas.pkl
  frontend
    cursoscocina.html
    icon.webp
    stylecursoscocina.css

```

```

backend > modeloChatBot > entrenar_modelo.py > ...
1 import os
2 import json
3 import re
4 import pickle
5 import fitz # PyMuPDF
6 from sklearn.feature_extraction.text import TfidfVectorizer
7 from sklearn.linear_model import LogisticRegression
8 from sklearn.model_selection import train_test_split
9 from sklearn.metrics import classification_report, confusion_matrix
10 import matplotlib.pyplot as plt
11 import seaborn as sns
12
13 def extraer_faq_desde_pdf(ruta):
14     faqs = []
15     if not os.path.exists(ruta):
16         return faqs
17     doc = fitz.open(ruta)
18     texto_total = ""
19     for pagina in doc:
20         texto_total += pagina.get_text()
21     bloques = texto_total.split("Pregunta:")
22     for bloque in bloques:
23         if "Respuesta:" in bloque:
24             partes = bloque.split("Respuesta:")
25             pregunta = partes[0].strip()
26             respuesta = partes[1].strip()
27             respuesta = re.sub(r'\s*\d+\.\s*', '', respuesta)
28             if pregunta and respuesta:
29                 faqs.append((pregunta, respuesta))
30
31     return faqs
32
33 def detectar_intencion(pregunta):
34     p = pregunta.lower()
35     if any(x in p for x in ["precio", "cuánto cuesta", "valor"]):
36         return "precio"
37     if any(x in p for x in ["clase", "cuántas clases", "duración"]):
38         return "clases"

```

OUTLINE

TIMELINE

main* ⚡ 0 △ 0

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF ⓘ Python 🖼 3.13.5 ⚡ Port: 5500 ⚡ Prettier

EXPLORER

```

IA_PROYECTO_FINAL... ...
  backend > modeloOpiniones > api_opiniones.py > analizar
    config.json
    entrenar_modelo.py
    matriz_confusion.png
    modelo_sentimiento.pkl
  frontend
    cursoscocina.html
    icon.webp
    stylecursoscocina.css

```

```

backend > modeloOpiniones > api_opiniones.py > analizar
1 import pickle
2
3 app = Flask(__name__)
4 CORS(app) # Esto habilita CORS para todas las rutas
5
6 with open("modelo_sentimiento.pkl", "rb") as f:
7     vectorizer, model = pickle.load(f)
8
9 @app.route("/analizar", methods=["POST"])
10 def analizar():
11     data = request.get_json()
12     texto = data.get("texto", "")
13     if not texto:
14         return jsonify({"error": "No se recibió texto"}), 400
15
16     texto_vec = vectorizer.transform([texto])
17     pred = model.predict(texto_vec)[0]
18     sentimiento = "Positiva" if pred == 1 else "Negativa"
19     return jsonify({"sentimiento": sentimiento})
20
21 @app.route("/ping")
22 def ping():
23     return "API activa"
24
25 if __name__ == "__main__":
26     app.run(host="0.0.0.0", port=5000)

```

The screenshot shows a code editor interface with two tabs open: `entrenar_modelo.py` and `api_opiniones.py`. The left pane displays the project structure under 'EXPLORER' for a folder named 'IA_PROYECTO_FINAL...'. The right pane shows the content of `entrenar_modelo.py`.

```

19     def leer_reseñas_pdf(ruta_pdf):
20         reseñas = []
21         if os.path.exists(ruta_pdf):
22             doc = fitz.open(ruta_pdf)
23             for page in doc:
24                 texto = page.get_text()
25                 for linea in texto.splitlines():
26                     if linea.strip():
27                         reseñas.append(linea.strip())
28
29     # ===== CARGAR CONFIGURACIÓN =====
30     with open("config.json") as f:
31         config = json.load(f)
32
33     # ===== MEDIR TIEMPO DE ENTRENAMIENTO =====
34     inicio = time.time()
35
36     dataset2 = load_dataset("imdb")
37
38     # ===== CARGAR DATOS DESDE HUGGING FACE (LIMITADO) =====
39     print("Cargando datasets desde Hugging Face (máx 500,000 registros en total)...")
40     MAX_REGISTROS = 500_000 # máximo total de registros
41     textos = []
42     etiquetas = []
43     total = 0
44
45     def agregar_registros(textos, etiquetas, nuevos_textos, nuevas_etiquetas, max_total):
46         restantes = max_total - len(textos)
47         if restantes <= 0:
48             return textos, etiquetas
49         textos += nuevos_textos[:restantes]
50         etiquetas += nuevas_etiquetas[:restantes]
51         return textos, etiquetas
52
53     # Dataset 1: amazon_polarity
54

```

13.3 Enlace del video explicativo

Enlace:

<https://youtu.be/8hOE3q9i1Hk>

2 casos a revisar

Caso 1: [Nombre del Proyecto]

Área de aplicación:

(Ejemplo: Salud / Agricultura / Educación / Logística, etc.)

Descripción del proyecto:

Breve resumen del objetivo, funcionalidad principal y tecnología base.

Tecnologías empleadas:

- Tipo de IA utilizada (machine learning, redes neuronales, NLP, etc.)
- Herramientas o lenguajes (Python, TensorFlow, Keras, etc.)

Resultados obtenidos:

- Beneficios alcanzados
- Precisión o rendimiento del modelo (si aplica)

Impacto del proyecto:

- Público beneficiado
- Problema resuelto

Aspectos destacados / reflexión:

- ¿Qué llamó la atención del proyecto?
- ¿Qué lo hace innovador o diferente?
- ¿Qué se podría adaptar para el proyecto propio?