INFORME DEL ENTEGRABLE 4

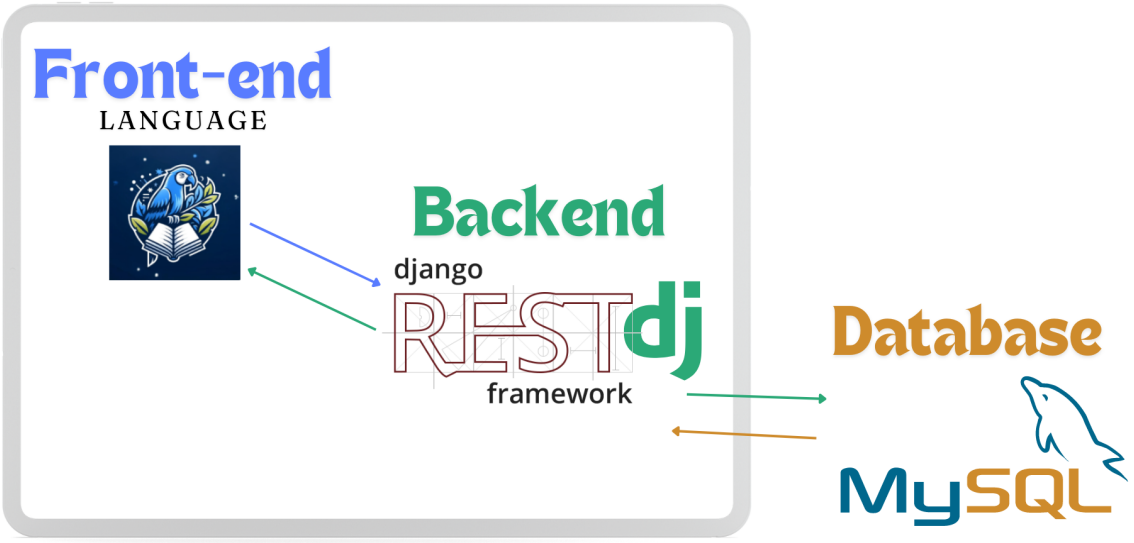
1. Introducción General

El proyecto en curso tiene como objetivo principal el desarrollo y la implementación de una plataforma educativa web y móvil destinada al aprendizaje de los idiomas español y portugués. Esta plataforma se distingue por incorporar tecnología avanzada de reconocimiento de voz y modelos de lenguaje grande (LLM) para mejorar la pronunciación del usuario, además de un sistema de gamificación que evalúa y fomenta el progreso mediante niveles y puntos.

> **Propósito:** de este proyecto es crear una herramienta accesible y eficiente para los estudiantes, profesionales y aficionados a la cultura que deseen mejorar sus habilidades lingüísticas.

2. Arquitectura Empleada

En nuestro proyecto, hemos adoptado una arquitectura desacoplada que consta de tres componentes principales: el frontend, el backend y la base de datos. Esta estructura nos permite una mayor flexibilidad, escalabilidad y facilidad de mantenimiento. A continuación, se describen en detalle cada uno de estos componentes y cómo interactúan entre sí.



3. Integración del Frontend

Enlace: https://ihc-language.vercel.app/

Descripción de Integración en el Front-end

Se utilizo NextJS para realizar integración con Backend, se realizó consultas POST y GET utilizando axios, algunas de las rutas que se llegaron a implementar fueron:

- [POST] /login: se envía un usuario y contraseña al backend, donde se espera la respuesta de un token de autenticación que es guardado por 7 días en las cookies del navegador con la variable authToken,
- [GET] /topics: se envía una petición GET que espera la lista completa de tópicos para mostrarla en distintas secciones de la pagina
- [GET] /topic/{slug}: se envía una petición GET que espera la un topico en especifico con la informacion necesaria para ser mostrada en pantalla
- [GET] /level/{topic_id}: se le enviauna peticion GET con el id del topico y retorna todos los niveles con el ID asociado.

4. Integración del Backend

Enlace: https://brasilearn-api-gateway.fly.dev/

Para la integración del backend, hemos seleccionado Django, una de las tecnologías más avanzadas y completas. Específicamente, utilizaremos **Django REST framework**, que nos permitirá desarrollar de manera eficiente una API REST en Python, proporcionando una base sólida y flexible para nuestro proyecto.

Requerimientos : Para construir nuestro API backend basado en Django Rest Framework, es necesario realizar las siguientes instalaciones y configuraciones, las cuales darán como resultado una infraestructura robusta y escalable con Django:

```
pip install Django
pip install Django mysqlclient
pip install djangorestframework
```

Base de datos: Decidimos prescindir de la base de datos predeterminada SQLite3 de Django debido a varios problemas que surgían al trabajar con GitHub. En particular, la base de datos SQLite3 se modificaba constantemente durante los push y commits, lo que dificultaba la colaboración y gestión del proyecto.

Para solucionar este problema, acordamos utilizar MySQL, una base de datos más robusta y adecuada para entornos de desarrollo colaborativos. MySQL está alojado en el siguiente host:

`Host:` `162.246.16.66`

`Puerto:` `3306`

`Nombre de la Base de Datos:` `jhanerco_brasilearn`

`Usuario:` `jhanerco_ihc`

`Contraseña:` `gRm*********XM`

Configuraciones en **settings:**

```
INSTALLED_APPS = [
    ...
    'rest_framework',
    'myapp',
]

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'jhanerco_brasilearn',
        'USER': 'jhanerco_ihc',
        'PASSWORD': 'gRmuHlJvdPvhFnXM',
        'HOST': '162.246.16.66',
        'PORT': '3306',
    }
}
```

Descripción de las Rutas de Backend

En esta etapa del proyecto, se han creado y hospedado diversas rutas de backend. A continuación, se presenta una descripción de las rutas más importantes, contenidas en el archivo **urls.py:**

Autenticación y Carga de Audio:

```
# Ruta para la autenticación de la API.
path('api-auth/', include('rest_framework.urls', namespace='rest_framework'))

# Ruta para la carga de archivos de audio.
path('upload-audio/', AudioUploadView.as_view(), name='upload_audio')
```

Operaciones con Usuarios y Contenido

```
router = routers.DefaultRouter()
# Ruta para manejar usuarios
router.register(r'users', UserViewSet)

# Ruta para manejar las habilidades de los usuarios
router.register(r'user-skills', UserSkillsViewSet)

# Ruta para manejar los temas
router.register(r'topics', TopicViewSet)

# Ruta para manejar los niveles
router.register(r'levels', LevelViewSet)

# Ruta para manejar las preguntas
router.register(r'questions', QuestionViewSet)

# Ruta para manejar los desafíos semanales
router.register(r'weekly-challenges', WeeklyChallengeViewSet)

# Ruta para manejar las puntuaciones
router.register(r'scores', ScoreViewSet)

# Ruta para manejar la comunidad
router.register(r'comunidade', ComunidadeViewSet)

# Ruta para manejar la puntuación de usuario por nivel
router.register(r'pontuation', PontuationUserLevelViewSet)

# Ruta para manejar los intereses de temas de los usuarios
router.register(r'userTopicInterest', UserTopicInterestViewSet)

# Ruta para manejar las comunidades de los usuarios
router.register(r'userComunity', UserComunityViewSet)
```

Rutas GET

```
# Ruta para obtener preguntas de un nivel específico y de un tipo específico
path('questions/<int:id_level>/<str:q_type>/', get_questions, name='get_quest
```

```
# Ruta para obtener los niveles de un tema específico
path('level/<int:id_topic>/', get_levels, name='get_levels'),

# Ruta para obtener los desafíos semanales de un tema específico dentro de un
path('weeklyChallenge/<int:id_topic>/<str:start_date>/<str:end_date>/', get_w

# Ruta para obtener la puntuación de un usuario en un desafío específico
path('user/<int:id_user>/<int:id_challenge>/', get_user_score, name='get_user

# Ruta para obtener la puntuación de un usuario en un tema y nivel específico
path('puntuacion/<int:user_id>/<int:id_topic>/<int:id_level>/', get_user_pont
```

Rutas POST

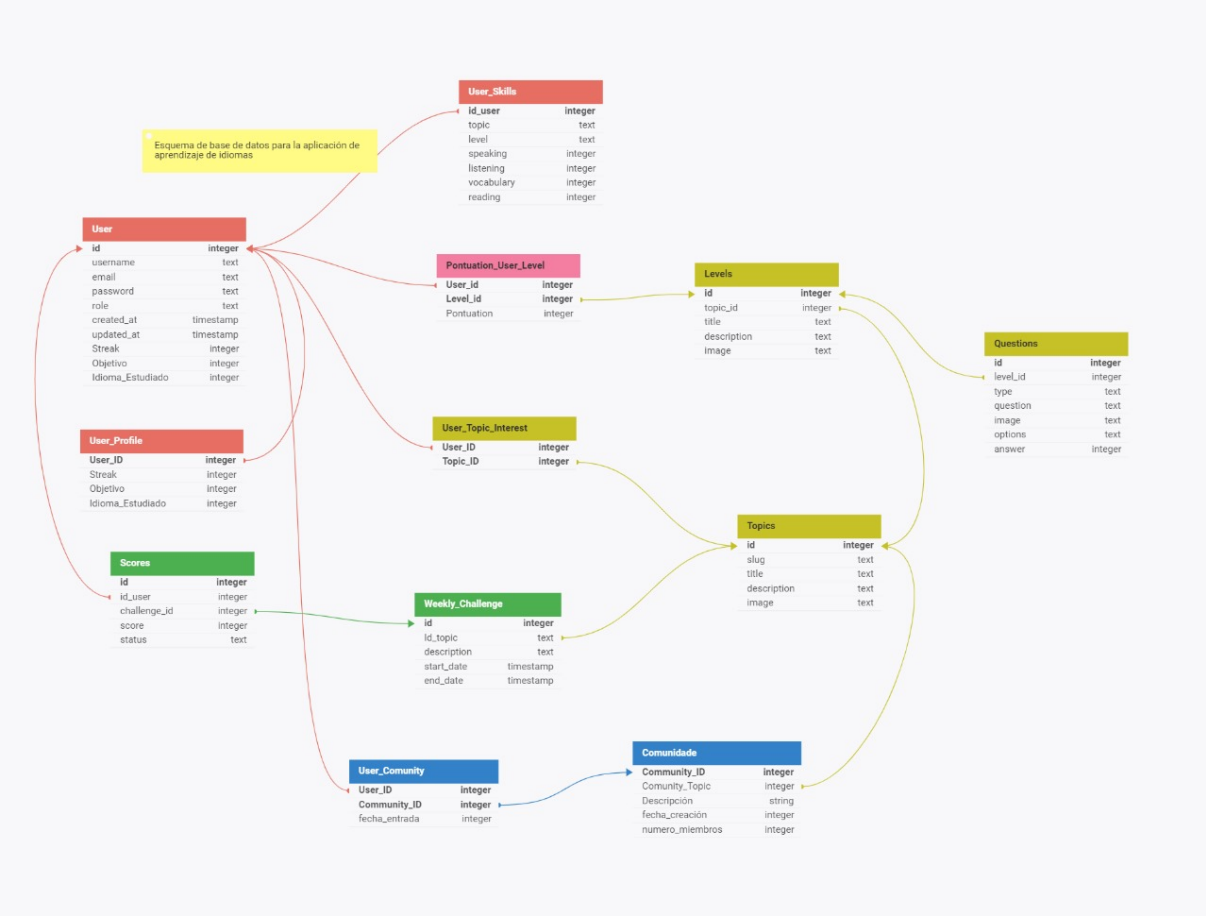
```
# Ruta para el chatbot de pathLLM
path('pathLLM-chatbot/', pathLLM_chatbot, name='pathLLM_chatbot'),

# Ruta para cargar puntuación
path('load_pontuation/', load_pontuation, name='load_pontuation'),

# Ruta para obtener el perfil de usuario
path("GetUserProfile/", get_user_profile, name="get_user_profile"),

# Ruta para devolver un ID (puede estar relacionado con algún recurso específ
path("ReturnID/", return_id, name="return_id"),

# Ruta para devolver un tema específico basado en un slug
path('return_topic/<str:topic_slug>/', return_topic, name='return_topic'),
```

Ruta load_audio

La "ruta de upload_audio" es un componente crítico del sistema que posibilita la evaluación de la pronunciación de los usuarios. Esta funcionalidad utiliza el servicio Google Cloud Speech-to-Text para transcribir el audio que se envía desde el frontend, con el fin de evaluar la precisión de la pronunciación del usuario. A continuación, se proporciona una descripción detallada del proceso:

1. **Grabación del Audio en el Frontend**: El usuario graba su pronunciación usando un micrófono. El audio se envía al backend en formato JSON, que incluye los bytes de audio codificados en base64 y la tasa de muestreo.
2. **Recepción y Procesamiento del Audio en el Backend**: El backend recibe el JSON y decodifica los bytes de audio. El audio se procesa utilizando la biblioteca pydub para asegurarse de que está en el formato correcto.
3. **Transcripción del Audio con Google Cloud Speech-to-Text**: El audio decodificado se envía a la API de Google Cloud Speech. La API transcribe el audio y devuelve el texto resultante.
4. **Evaluación de la Pronunciación**: La transcripción proporcionada por Google Cloud Speech se compara con el texto base que el usuario estaba destinado a pronunciar. La precisión se evalúa en función de la correspondencia entre el texto base y el texto transcrito.

Tablas usadas



El archivo SQL proporcionado define diversas tablas fundamentales para el funcionamiento de nuestra plataforma educativa:

1. **Tabla users**: Alberga la información de los usuarios registrados en la plataforma. Los campos incluyen ID de usuario, nombre completo, nombre de usuario, contraseña, correo electrónico, entre otros.
2. **Tabla user_skills**: Registra las habilidades de los usuarios. Los campos incluyen ID de habilidad, ID de usuario, descripción de la habilidad, nivel de competencia.
3. **Tabla topics**: Contiene los temas disponibles en la plataforma. Los campos incluyen ID de tema, nombre del tema, descripción, slug (identificador único).
4. **Tabla levels**: Define los niveles de dificultad para cada tema. Los campos son ID de nivel, ID de tema, nombre del nivel, descripción.
5. **Tabla questions**: Almacena las preguntas utilizadas en los ejercicios y desafíos. Los campos son ID de pregunta, ID de nivel, tipo de pregunta, contenido de la pregunta.
6. **Tabla weekly_challenges**: Registra los desafíos semanales propuestos a los usuarios. Los campos son ID de desafío, ID de tema, fecha de inicio, fecha de fin, descripción.

7. **Tabla scores**: Contiene las puntuaciones obtenidas por los usuarios en diversos ejercicios y desafíos. Los campos son ID de puntuación, ID de usuario, ID de desafío, puntuación obtenida, fecha de registro.
8. **Tabla pontuation_user_level**: Registra la puntuación de los usuarios en los distintos niveles. Los campos son ID de registro, ID de usuario, ID de nivel, puntuación.
9. **Tabla user_topic_interest**: Almacena los intereses temáticos de los usuarios. Los campos son ID de registro, ID de usuario, ID de tema.
10. **Tabla comunidade**: Registra la información sobre las comunidades de usuarios. Los campos son ID de comunidad, nombre de la comunidad, descripción.
11. **Tabla user_comunity**: Contiene las relaciones entre los usuarios y las comunidades a las que pertenecen. Los campos son ID de registro, ID de usuario, ID de comunidad.

Estas tablas son esenciales para soportar las funcionalidades de la plataforma, permitiendo una gestión eficiente de usuarios, sus habilidades, los contenidos educativos y la interacción entre ellos.