📚 DOCUMENTACIÓN API - GESTIÓN FCT

Autenticación Global: Todas las rutas (excepto login y descarga de archivos) requieren `auth:sanctum`

🎓 CICLOS

---

1. Listar ciclos

- **Ruta:** `GET /api/ciclos`
- **Controlador:** `CiclosController@index`
- **Descripción:** Devuelve el listado completo de ciclos formativos con su información básica y familia profesional asociada.

---

2. Crear ciclo

- **Ruta:** `POST /api/ciclos`
- **Controlador:** `CiclosController@store`
- **Descripción:** Implementación de un sistema de creación de ciclos formativos que asegura la integridad de los datos mediante validación. El sistema procesa los siguientes campos:

- **Campos del Request:**

  ○ `nombre`: (Obligatorio) Tipo string, máximo 255 caracteres. Debe ser único.

  ○ `familia_profesional_id`: (Obligatorio) Tipo integer. Debe existir en la tabla familias_profesionales.

- **Respuestas:**

- **201**: Si el ciclo se ha creado con éxito.
- **422**: Si la validación falla.
- **500**: Si hay un error al insertar en base de datos.

---

3. Ver detalle de un ciclo

- **Ruta:** GET /api/admin/ciclos/{ciclo}
- **Controlador:** CiclosController@show
- **Descripción:** Devuelve la información completa de un ciclo específico, incluyendo su familia profesional asociada.

- **Parámetros de ruta:**
  - ciclo: (Obligatorio) ID del ciclo.
- **Respuestas:**
  - **200**: Si el ciclo existe.
  - **404**: Si el ciclo no se encuentra.

---

4. Obtener cursos de un ciclo

- **Ruta:** GET /api/ciclo/{ciclo_id}/cursos
- **Controlador:** CiclosController@getCursosByCiclos
- **Descripción:** Devuelve todos los cursos asociados a un ciclo formativo específico.

- **Parámetros de ruta:**
  - ciclo_id: (Obligatorio) ID del ciclo.
- **Respuestas:**
  - **200**: Si el ciclo existe y tiene cursos.
  - **404**: Si el ciclo no se encuentra.

---

5. Obtener tutores de un ciclo

- **Ruta:** GET /api/ciclo/{ciclo_id}/tutores
- **Controlador:** TutorEgibideController@getTutoresByCiclo
- **Descripción:** Devuelve los tutores de Egibide asignados a un ciclo formativo concreto.

- **Parámetros de ruta:**
    - ciclo_id: (Obligatorio) ID del ciclo.
- **Respuestas:**
    - **200**: Si el ciclo existe y tiene tutores asignados.
    - **404**: Si el ciclo no se encuentra.

---

6. Descargar plantilla CSV

- **Ruta:** GET /api/ciclos/plantilla
- **Controlador:** CiclosController@descargarPlantillaCSV
- **Descripción:** Genera y descarga una plantilla CSV vacía para facilitar la importación masiva de datos de ciclos formativos.

- **Respuestas:**
    - **200**: Descarga del archivo plantilla_ciclo.csv con Content-Type text/csv.

---

7. Importar datos desde CSV

- **Ruta:** POST /api/ciclos/importar
- **Controlador:** CiclosController@importarCSV
- **Descripción:** Implementación de un sistema de importación masiva de datos desde archivo CSV. El sistema valida el formato del archivo y procesa los registros de forma controlada.

- **Campos del Request (multipart/form-data):**

- `ciclo_id`: (Obligatorio) Tipo integer. Debe existir en la tabla ciclos.
- `csv_file`: (Obligatorio) Archivo CSV o TXT. Máximo 10240KB (10MB). Debe tener formato CSV válido.

- **Respuestas:**
  - **200**: Si la importación se completó exitosamente.
  - **422**: Si la validación del archivo o datos falla.
  - **500**: Si hay un error durante el proceso de importación.

🏷️ FAMILIAS PROFESIONALES

---

1. Listar familias profesionales

- **Ruta:** `GET /api/familiasProfesionales`
- **Controlador:** `FamiliaProfesionalController@index`
- **Descripción:** Devuelve el listado completo de familias profesionales registradas en el sistema con su código identificador.

- **Respuestas:**
  - **200**: Si hay familias profesionales registradas.
  - **401**: Si el usuario no está autenticado.

🏢 EMPRESAS

---

1. Listar empresas

- **Ruta:** GET /api/empresa
- **Controlador:** EmpresasController@index
- **Descripción:** Devuelve el listado completo de empresas colaboradoras registradas en el sistema con su información de contacto.

- **Respuestas:**
    - **200**: Si hay empresas registradas.
    - **401**: Si el usuario no está autenticado.

---

2. Crear empresa

- **Ruta:** POST /api/empresa
- **Controlador:** EmpresasController@store
- **Descripción:** Implementación de un sistema de registro de empresas colaboradoras que asegura la integridad de los datos mediante validación. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - nombre: (Obligatorio) Tipo string, máximo 150 caracteres.
    - cif: (Obligatorio) Tipo string, máximo 20 caracteres. Debe ser único.
    - email: (Obligatorio) Formato email válido, máximo 120 caracteres. Debe ser único.
    - direccion: (Opcional) Tipo string, máximo 250 caracteres.
    - telefono: (Opcional) Tipo string, máximo 20 caracteres.

- **Respuestas:**
    - **201**: Si la empresa se ha creado con éxito.
    - **422**: Si la validación falla.
    - **500**: Si hay un error interno del servidor.

🎓 ALUMNOS

---

1. Listar alumnos

- **Ruta:** GET /api/alumnos
- **Controlador:** AlumnosController@index
- **Descripción:** Devuelve el listado completo de alumnos registrados en el sistema con su información personal y académica.

- **Respuestas:**
    - **200**: Si hay alumnos registrados.
    - **401**: Si el usuario no está autenticado.

---

2. Crear alumno

- **Ruta:** POST /api/alumnos
- **Controlador:** AlumnosController@store
- **Descripción:** Implementación de un sistema de registro de alumnos que crea simultáneamente el usuario asociado y su estancia inicial. El sistema asegura la integridad de los datos mediante validación y procesa los siguientes campos:

- **Campos del Request:**
    - nombre: (Obligatorio) Tipo string, máximo 100 caracteres.
    - apellidos: (Obligatorio) Tipo string, máximo 150 caracteres.
    - email: (Obligatorio) Formato email válido, máximo 255 caracteres. Debe ser único en la tabla users.
    - password: (Obligatorio) Tipo string, mínimo 8 caracteres. Debe guardarse encriptado.
    - curso: (Obligatorio) Tipo integer. Debe existir en la tabla cursos.

- `tutor`: (Obligatorio) Tipo integer. Debe existir en la tabla tutores.
- `telefono`: (Opcional) Tipo string, máximo 20 caracteres.
- `ciudad`: (Opcional) Tipo string, máximo 120 caracteres.

- **Respuestas:**
  - **201**: Si el alumno se ha creado con éxito.
  - **422**: Si la validación falla.

---

3. Obtener perfil del alumno autenticado

- **Ruta:** `GET /api/me/alumno`
- **Controlador:** `AlumnosController@me`
- **Descripción:** Devuelve la información completa del perfil del alumno asociado al usuario autenticado, incluyendo datos personales y de contacto.

- **Respuestas:**
  - **200**: Si el alumno existe.
  - **401**: Si el usuario no está autenticado.
  - **404**: Si no existe un alumno asociado al usuario.

---

4. Inicio del alumno (Dashboard)

- **Ruta:** `GET /api/me/inicio`
- **Controlador:** `AlumnosController@inicio`
- **Descripción:** Devuelve la información de dashboard para el alumno autenticado, incluyendo datos de su estancia actual, empresa asignada, entregas pendientes y seguimientos recientes.

- **Respuestas:**
  - **200**: Si el alumno existe (con o sin estancia asignada).

- **401**: Si el usuario no está autenticado.

📦 ENTREGAS

---

1. Listar mis entregas

- **Ruta:** `GET /api/entregas/mias`
- **Controlador:** `EntregaController@mias`
- **Descripción:** Devuelve todas las entregas del cuaderno de prácticas del alumno autenticado, ordenadas por fecha.

- **Respuestas:**
  - **200**: Si hay entregas registradas.
  - **401**: Si el usuario no está autenticado.

---

2. Crear entrega

- **Ruta:** `POST /api/entregas`
- **Controlador:** `EntregaController@store`
- **Descripción:** Implementación de un sistema de subida de entregas del cuaderno de prácticas que valida el formato del archivo y asegura que el alumno solo puede subir entregas a su propio cuaderno. El sistema procesa los siguientes campos:

- **Campos del Request (multipart/form-data):**
  - `archivo`: (Obligatorio) Archivo PDF. Máximo 10240KB (10MB). Solo se aceptan archivos con formato PDF.
  - `fecha`: (Obligatorio) Tipo date, formato YYYY-MM-DD.
  - `cuaderno_practicas_id`: (Obligatorio) Tipo integer. Debe existir en la tabla cuadernos_practicas y pertenecer al alumno autenticado.

- **Respuestas:**
    - **201**: Si la entrega se ha creado con éxito.
    - **422**: Si la validación falla.
    - **403**: Si el alumno intenta subir a un cuaderno que no le pertenece.

---

3. Eliminar entrega

- **Ruta:** DELETE /api/entregas/{id}
- **Controlador:** EntregaController@destroy
- **Descripción:** Permite al alumno autenticado eliminar una de sus entregas del cuaderno de prácticas. El archivo asociado también se elimina del almacenamiento.

- **Parámetros de ruta:**
    - id: (Obligatorio) ID de la entrega.

- **Respuestas:**
    - **200**: Si la entrega se eliminó correctamente.
    - **403**: Si el alumno intenta eliminar una entrega que no le pertenece.
    - **404**: Si la entrega no existe.

---

4. Descargar archivo de entrega

- **Ruta:** GET /api/entregas/{entrega}/archivo
- **Controlador:** EntregaController@archivo
- **Auth:** ❌ No requiere autenticación
- **Descripción:** Permite descargar el archivo PDF asociado a una entrega específica. Esta ruta es pública para permitir que los tutores puedan acceder a las entregas.

- **Parámetros de ruta:**

- entrega: (Obligatorio) ID de la entrega.

- **Respuestas:**
  - **200**: Descarga del archivo PDF.
  - **404**: Si el archivo no existe o ha sido eliminado.

📘 SEGUIMIENTOS

---

1. Listar seguimientos de un alumno

- **Ruta:** GET /api/seguimientos/alumno/{alumno_id}
- **Controlador:** SeguimientosController@seguimientosAlumno
- **Descripción:** Devuelve todos los seguimientos registrados para un alumno específico, ordenados cronológicamente. Incluye información sobre visitas, llamadas y otras acciones de seguimiento.

- **Parámetros de ruta:**
  - alumno_id: (Obligatorio) ID del alumno.

- **Respuestas:**
  - **200**: Si hay seguimientos registrados.
  - **401**: Si el usuario no está autenticado.
  - **404**: Si el alumno no existe.

---

2. Crear seguimiento

- **Ruta:** POST /api/nuevo-seguimiento
- **Controlador:** SeguimientosController@nuevoSeguimiento
- **Descripción:** Implementación de un sistema de registro de seguimientos de prácticas que permite documentar las interacciones

con los alumnos y empresas. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - `accion`: (Obligatorio) Tipo string, máximo 150 caracteres. Describe el tipo de acción realizada (Visita empresa, Llamada telefónica, Email, etc.).
    - `fecha`: (Obligatorio) Tipo date, formato YYYY-MM-DD.
    - `estancia_id`: (Obligatorio) Tipo integer. Debe existir en la tabla estancias.
    - `descripcion`: (Opcional) Tipo text. Observaciones o comentarios detallados sobre el seguimiento.
    - `via`: (Opcional) Tipo string, máximo 50 caracteres. Medio utilizado (Presencial, Teléfono, Email, Videollamada, etc.).
- **Respuestas:**
    - **201**: Si el seguimiento se ha creado con éxito.
    - **422**: Si la validación falla.

---

3. Eliminar seguimiento

- **Ruta:** `DELETE /api/seguimientos/{seguimiento}`
- **Controlador:** `SeguimientosController@destroy`
- **Descripción:** Permite eliminar un seguimiento específico del sistema. Solo puede ser eliminado por el usuario que lo creó.

- **Parámetros de ruta:**
    - `seguimiento`: (Obligatorio) ID del seguimiento.
- **Respuestas:**
    - **200**: Si el seguimiento se eliminó correctamente.
    - **404**: Si el seguimiento no existe.

👨‍🏫 TUTOR EGIBIDE

---

1. Inicio del tutor (Dashboard)

- **Ruta:** GET /api/tutorEgibide/inicio
- **Controlador:** TutorEgibideController@inicioTutor
- **Descripción:** Devuelve la información de dashboard para el tutor de Egibide autenticado, incluyendo estadísticas de alumnos asignados, estancias activas y seguimientos pendientes.

- **Respuestas:**
    - **200**: Si el tutor existe.
    - **401**: Si el usuario no está autenticado.
    - **404**: Si no existe un tutor asociado al usuario.

---

2. Obtener perfil del tutor autenticado

- **Ruta:** GET /api/me/tutor-egibide
- **Controlador:** TutorEgibideController@me
- **Descripción:** Devuelve la información completa del perfil del tutor de Egibide autenticado, incluyendo datos personales y de contacto.

- **Respuestas:**
    - **200**: Si el tutor existe.
    - **401**: Si el usuario no está autenticado.
    - **404**: Si no existe un tutor asociado al usuario.

---

3. Obtener tutores por ciclo

- **Ruta:** GET /api/ciclo/{ciclo_id}/tutores
- **Controlador:** TutorEgibideController@getTutoresByCiclo
- **Descripción:** Devuelve los tutores de Egibide asignados a un ciclo formativo específico para facilitar la asignación de alumnos.

- **Parámetros de ruta:**
    - ciclo_id: (Obligatorio) ID del ciclo.
- **Respuestas:**
    - **200**: Si el ciclo existe y tiene tutores asignados.
    - **404**: Si el ciclo no se encuentra.

---

4. Obtener instructores disponibles para un alumno

- **Ruta:** GET /api/alumnos/{alumno_id}/instructores
- **Controlador:** TutorEgibideController@getInstructoresParaAlumno
- **Descripción:** Devuelve los instructores de empresa disponibles que pertenecen a la empresa donde el alumno realiza sus prácticas. Facilita la asignación del tutor de empresa.

- **Parámetros de ruta:**

    - alumno_id: (Obligatorio) ID del alumno.

- **Respuestas:**
    - **200**: Si el alumno existe y tiene empresa asignada.
    - **404**: Si el alumno no existe o no tiene empresa asignada.

---

5. Asignar instructor a alumno

- **Ruta:** POST /api/alumnos/asignar-instructor
- **Controlador:** TutorEgibideController@asignarInstructorAlumno
- **Descripción:** Implementación de un sistema de asignación de instructores de empresa a alumnos que valida que el instructor

pertenezca a la empresa donde el alumno realiza sus prácticas. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - `alumno_id`: (Obligatorio) Tipo integer. Debe existir en la tabla alumnos y tener una estancia activa.
    - `instructor_id`: (Obligatorio) Tipo integer. Debe existir en la tabla instructores y pertenecer a la empresa de la estancia del alumno.
- **Respuestas:**
    - **200**: Si el instructor se asignó correctamente.
    - **400**: Si el instructor no pertenece a la empresa donde el alumno realiza sus prácticas.
    - **404**: Si el alumno no existe o no tiene estancia activa.
    - **422**: Si la validación falla.

🏭 TUTOR EMPRESA / INSTRUCTORES

---

1. Listar instructores

- **Ruta:** `GET /api/instructores`
- **Controlador:** `TutorEmpresaController@index`
- **Descripción:** Devuelve el listado completo de instructores/tutores de empresa registrados en el sistema con su empresa asociada.
- **Respuestas:**
    - **200**: Si hay instructores registrados.
    - **401**: Si el usuario no está autenticado.

---

2. Listar instructores disponibles

- **Ruta:** GET /api/instructores/disponibles
- **Controlador:** TutorEmpresaController@getInstructoresDisponibles
- **Descripción:** Devuelve los instructores que están disponibles para tutorizar alumnos, mostrando su capacidad restante y alumnos actualmente asignados.

- **Respuestas:**
    - **200**: Si hay instructores disponibles.
    - **401**: Si el usuario no está autenticado.

---

3. Crear instructor

- **Ruta:** POST /api/instructores/crear
- **Controlador:** TutorEmpresaController@crearInstructor
- **Descripción:** Implementación de un sistema de registro de instructores de empresa que crea simultáneamente el usuario asociado con rol tutor_empresa. El sistema asegura la integridad de los datos mediante validación y procesa los siguientes campos:

- **Campos del Request:**
    - nombre: (Obligatorio) Tipo string, máximo 100 caracteres.
    - apellidos: (Obligatorio) Tipo string, máximo 150 caracteres.
    - email: (Obligatorio) Formato email válido, máximo 255 caracteres. Debe ser único en la tabla users.
    - password: (Obligatorio) Tipo string, mínimo 8 caracteres. Debe guardarse encriptado.
    - empresa_id: (Obligatorio) Tipo integer. Debe existir en la tabla empresas.
    - telefono: (Opcional) Tipo string, máximo 20 caracteres.
    - ciudad: (Opcional) Tipo string, máximo 120 caracteres.

- **Respuestas:**

- **201**: Si el instructor se ha creado con éxito.
- **422**: Si la validación falla.

---

4. Asignar instructor a estancia

- **Ruta:** `POST /api/instructores/asignar`
- **Controlador:** `TutorEmpresaController@asignarInstructor`
- **Descripción:** Implementación de un sistema de asignación de instructores a estancias de alumnos que valida que el instructor pertenezca a la misma empresa de la estancia. El sistema procesa los siguientes campos:

- **Campos del Request:**
  - `estancia_id`: (Obligatorio) Tipo integer. Debe existir en la tabla estancias.
  - `instructor_id`: (Obligatorio) Tipo integer. Debe existir en la tabla instructores y pertenecer a la empresa de la estancia.

- **Respuestas:**
  - **200**: Si el instructor se asignó correctamente.
  - **400**: Si el instructor no pertenece a la empresa de la estancia.
  - **422**: Si la validación falla.

---

5. Inicio del instructor (Dashboard)

- **Ruta:** `GET /api/tutorEmpresa/inicio`
- **Controlador:** `TutorEmpresaController@inicioInstructor`
- **Descripción:** Devuelve la información de dashboard para el instructor de empresa autenticado, incluyendo estadísticas de alumnos tutorizados y seguimientos realizados.

- **Respuestas:**

- **200**: Si el instructor existe.
- **401**: Si el usuario no está autenticado.
- **404**: Si no existe un instructor asociado al usuario.

---

6. Obtener alumnos del instructor

- **Ruta:** GET /api/tutorEmpresa/alumnos
- **Controlador:** TutorEmpresaController@getAlumnosByCurrentInstructor
- **Descripción:** Devuelve los alumnos asignados al instructor de empresa autenticado, incluyendo información de sus estancias y puestos de trabajo.

- **Respuestas:**
  - **200**: Si hay alumnos asignados.
  - **401**: Si el usuario no está autenticado.

---

7. Obtener perfil del instructor autenticado

- **Ruta:** GET /api/me/tutor-empresa
- **Controlador:** TutorEmpresaController@me
- **Descripción:** Devuelve la información completa del perfil del instructor de empresa autenticado, incluyendo datos personales y empresa asociada.

- **Respuestas:**
  - **200**: Si el instructor existe.
  - **401**: Si el usuario no está autenticado.
  - **404**: Si no existe un instructor asociado al usuario.

🎯 COMPETENCIAS

---

1. Listar competencias

- **Ruta:** GET /api/competencias
- **Controlador:** CompetenciasController@index
- **Descripción:** Devuelve el listado completo de competencias técnicas y transversales disponibles en el sistema, separadas por tipo.

- **Respuestas:**
    - **200**: Si hay competencias registradas.
    - **401**: Si el usuario no está autenticado.

---

2. Obtener competencias técnicas del alumno

- **Ruta:** GET /api/competencias/tecnicas/alumno/{alumno_id}
- **Controlador:** CompetenciasController@getCompetenciasTecnicasByAlumno
- **Descripción:** Devuelve las competencias técnicas del ciclo formativo del alumno, incluyendo sus resultados de aprendizaje asociados.

- **Parámetros de ruta:**
    - alumno_id: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe.
    - **404**: Si el alumno no existe.

---

3. Obtener competencias técnicas asignadas al alumno

- **Ruta:** GET /api/competencias/tecnicas/asignadas/{alumno_id}
- **Controlador:** CompetenciasController@getCompetenciasTecnicasAsignadasByAlumno
- **Descripción:** Devuelve las competencias técnicas que ya han sido asignadas a la estancia del alumno para su evaluación.

- **Parámetros de ruta:**
    - alumno_id: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe.
    - **404**: Si el alumno no existe.

---

4. Obtener competencias transversales del alumno

- **Ruta:** GET /api/competencias/transversales/alumno/{alumno_id}
- **Controlador:** CompetenciasController@getCompetenciasTransversalesByAlumno
- **Descripción:** Devuelve las competencias transversales de la familia profesional del ciclo del alumno.

- **Parámetros de ruta:**
    - alumno_id: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe.
    - **404**: Si el alumno no existe.

---

5. Obtener calificaciones de competencias técnicas

- **Ruta:** GET /api/competencias/tecnicas/calificaciones/{alumno_id}
- **Controlador:** CompetenciasController@getCalificacionesCompetenciasTecnicas
- **Descripción:** Devuelve las calificaciones de las competencias técnicas evaluadas para un alumno específico.

- **Parámetros de ruta:**
    - alumno_id: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe y tiene calificaciones.
    - **404**: Si el alumno no existe.

---

6. Obtener calificaciones de competencias transversales

- **Ruta:** GET /api/competencias/transversales/calificaciones/{alumno_id}
- **Controlador:** CompetenciasController@getCalificacionesCompetenciasTransversales
- **Descripción:** Devuelve las calificaciones de las competencias transversales evaluadas para un alumno específico.

- **Parámetros de ruta:**
    - alumno_id: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe y tiene calificaciones.
    - **404**: Si el alumno no existe.

---

7. Crear competencia técnica

- **Ruta:** POST /api/competencias/tecnica
- **Controlador:** CompetenciasController@storeTecnica

- **Descripción:** Implementación de un sistema de creación de competencias técnicas asociadas a ciclos formativos. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - `descripcion`: (Obligatorio) Tipo text. Descripción detallada de la competencia técnica.
    - `ciclo_id`: (Obligatorio) Tipo integer. Debe existir en la tabla ciclos.
- **Respuestas:**
    - **201**: Si la competencia se ha creado con éxito.
    - **422**: Si la validación falla.

---

8. Crear competencia transversal

- **Ruta:** `POST /api/competencias/transversal`
- **Controlador:** `CompetenciasController@storeTransversal`
- **Descripción:** Implementación de un sistema de creación de competencias transversales asociadas a familias profesionales. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - `descripcion`: (Obligatorio) Tipo text. Descripción detallada de la competencia transversal.
    - `familia_profesional_id`: (Obligatorio) Tipo integer. Debe existir en la tabla familias_profesionales.
    - `nivel`: (Opcional) Tipo string, máximo 50 caracteres. Nivel de la competencia (Básico, Intermedio, Avanzado).
- **Respuestas:**
    - **201**: Si la competencia se ha creado con éxito.
    - **422**: Si la validación falla.

---

9. Asignar competencias técnicas a estancia

- **Ruta:** POST /api/competencias/tecnicas/asignar
- **Controlador:** CompetenciasController@storeCompetenciasTecnicasAsignadas
- **Descripción:** Implementación de un sistema de asignación de competencias técnicas a las estancias de los alumnos para su posterior evaluación. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - estancia_id: (Obligatorio) Tipo integer. Debe existir en la tabla estancias.
    - competencias: (Obligatorio) Tipo array de integers, mínimo 1 elemento. Cada elemento debe existir en la tabla competencias_tec.
- **Respuestas:**
    - **201**: Si las competencias se asignaron con éxito.
    - **422**: Si la validación falla.

---

10. Calificar competencias técnicas

- **Ruta:** POST /api/competencias/tecnicas/calificar
- **Controlador:** CompetenciasController@storeCompetenciasTecnicasCalificadas
- **Descripción:** Implementación de un sistema de calificación de competencias técnicas que valida el rango de notas y asegura que las competencias están asignadas a la estancia. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - estancia_id: (Obligatorio) Tipo integer. Debe existir en la tabla estancias.

- calificaciones: (Obligatorio) Tipo array de objetos, mínimo 1 elemento. Cada objeto debe contener:

  - competencia_id: (Obligatorio) Tipo integer. Debe existir en la tabla competencias_tec.

  - nota: (Obligatorio) Tipo decimal. Entre 0 y 10, máximo 2 decimales.

- **Respuestas:**

  - **201**: Si las calificaciones se guardaron con éxito.

  - **422**: Si la validación falla.

---

11. Calificar competencias transversales

- **Ruta:** POST /api/competencias/transversales/calificar
- **Controlador:** CompetenciasController@storeCompetenciasTransversalesCalificadas
- **Descripción:** Implementación de un sistema de calificación de competencias transversales que valida el rango de notas y asegura la integridad de los datos. El sistema procesa los siguientes campos:

- **Campos del Request:**

  - estancia_id: (Obligatorio) Tipo integer. Debe existir en la tabla estancias.

  - calificaciones: (Obligatorio) Tipo array de objetos, mínimo 1 elemento. Cada objeto debe contener:

  - competencia_id: (Obligatorio) Tipo integer. Debe existir en la tabla competencias_trans.

  - nota: (Obligatorio) Tipo decimal. Entre 0 y 10, máximo 2 decimales.

- **Respuestas:**

  - **201**: Si las calificaciones se guardaron con éxito.

  - **422**: Si la validación falla.

📊 NOTAS

---

1. Obtener notas técnicas del alumno

- **Ruta:** `GET /api/notas/tecnicas/{alumno_id}`
- **Controlador:** `NotasController@obtenerNotasTecnicas`
- **Descripción:** Devuelve las calificaciones de las competencias técnicas evaluadas durante la estancia en empresa del alumno especificado.

- **Parámetros de ruta:**
    - `alumno_id`: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe y tiene calificaciones.
    - **404**: Si el alumno no existe.

---

2. Obtener notas transversales del alumno

- **Ruta:** `GET /api/notas/transversales/{alumno_id}`
- **Controlador:** `NotasController@obtenerNotasTransversales`
- **Descripción:** Devuelve las calificaciones de las competencias transversales evaluadas durante la estancia en empresa del alumno especificado.

- **Parámetros de ruta:**
    - `alumno_id`: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe y tiene calificaciones.
    - **404**: Si el alumno no existe.

---

3. Obtener notas de Egibide del alumno

- **Ruta:** GET /api/notas/egibide/{alumno_id}
- **Controlador:** NotasController@obtenerNotasEgibide
- **Descripción:** Devuelve las calificaciones de las asignaturas cursadas en el centro educativo Egibide por el alumno especificado.

- **Parámetros de ruta:**
    - alumno_id: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe y tiene calificaciones.
    - **404**: Si el alumno no existe.

---

4. Guardar notas de Egibide

- **Ruta:** POST /api/notas/egibide
- **Controlador:** NotasController@guardarNotasEgibide
- **Descripción:** Implementación de un sistema de registro de calificaciones de asignaturas del centro educativo que permite guardar o actualizar múltiples notas simultáneamente. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - alumno_id: (Obligatorio) Tipo integer. Debe existir en la tabla alumnos.
    - notas: (Obligatorio) Tipo array de objetos, mínimo 1 elemento. Cada objeto debe contener:

    - asignatura_id: (Obligatorio) Tipo integer. Debe existir en la tabla asignaturas.

    - nota: (Obligatorio) Tipo decimal. Entre 0 y 10, máximo 2 decimales.

- anio: (Obligatorio) Tipo year (integer). Entre 2000 y 2100.

- **Respuestas:**
    - **201**: Si las notas se guardaron con éxito.
    - **422**: Si la validación falla.

---

5. Obtener nota del cuaderno de prácticas

- **Ruta:** GET /api/notas/cuaderno/{alumno_id}
- **Controlador:** NotasController@obtenerNotaCuadernoByAlumno
- **Descripción:** Devuelve la calificación del cuaderno de prácticas del alumno especificado.

- **Parámetros de ruta:**
    - alumno_id: (Obligatorio) ID del alumno.

- **Respuestas:**
    - **200**: Si el alumno existe y tiene calificación del cuaderno.
    - **404**: Si el alumno no existe o no tiene cuaderno.

---

6. Guardar nota del cuaderno de prácticas

- **Ruta:** POST /api/notas/cuaderno
- **Controlador:** NotasController@guardarNotasCuaderno
- **Descripción:** Implementación de un sistema de registro de la calificación del cuaderno de prácticas que valida el rango de notas. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - cuaderno_id: (Obligatorio) Tipo integer. Debe existir en la tabla cuadernos_practicas.
    - nota: (Obligatorio) Tipo decimal. Entre 0 y 10, máximo 2 decimales.

- **Respuestas:**
    - **201**: Si la nota se guardó con éxito.
    - **422**: Si la validación falla.

🔐 AUTENTICACIÓN

---

1. Login

- **Ruta:** POST /api/login
- **Controlador:** AuthController@authenticate
- **Auth:** ❌ No requiere autenticación
- **Descripción:** Implementación de un sistema de autenticación que valida las credenciales del usuario y genera un token Sanctum para acceder a las rutas protegidas. El sistema procesa los siguientes campos:

- **Campos del Request:**
    - email: (Obligatorio) Formato email válido, máximo 255 caracteres.
    - password: (Obligatorio) Tipo string.

- **Respuestas:**
    - **200**: Si la autenticación fue exitosa. Devuelve el token de acceso y los datos del usuario.
    - **401**: Si las credenciales son incorrectas.
    - **422**: Si la validación falla.

---

2. Obtener usuario autenticado

- **Ruta:** GET /user
- **Descripción:** Devuelve la información completa del usuario actualmente autenticado, incluyendo su rol y datos básicos.

- **Respuestas:**
    - **200**: Si el usuario está autenticado.
    - **401**: Si el token no es válido o ha expirado.

⚙️ ADMIN

---

1. Dashboard de administración

- **Ruta:** GET /api/admin/inicio
- **Controlador:** AdminController@inicioAdmin
- **Descripción:** Devuelve estadísticas generales del sistema para el panel de administración, incluyendo totales de usuarios, estancias activas y seguimientos del mes.

- **Respuestas:**
    - **200**: Si el usuario es administrador.
    - **401**: Si el usuario no está autenticado.
    - **403**: Si el usuario no tiene rol de administrador.

---

2. Detalle completo de alumno

- **Ruta:** GET /api/admin/alumnos/{alumno_id}
- **Controlador:** AdminController@detalleAlumno
- **Descripción:** Devuelve información completa de un alumno específico incluyendo estancia, notas de Egibide, notas de competencias y seguimientos realizados.

- **Parámetros de ruta:**
    - `alumno_id`: (Obligatorio) ID del alumno.
- **Respuestas:**
    - **200**: Si el alumno existe.
    - **401**: Si el usuario no está autenticado.
    - **403**: Si el usuario no tiene rol de administrador.
    - **404**: Si el alumno no existe.

📋 CÓDIGOS DE ESTADO HTTP

Código	Descripción
200	OK - Petición exitosa
201	Created - Recurso creado exitosamente
400	Bad Request - Error de negocio o petición malformada
401	Unauthorized - No autenticado (falta token o token inválido)
403	Forbidden - No autorizado (sin permisos para esta acción)

404	Not Found - Recurso no encontrado
422	Unprocessable Entity - Error de validación de datos
500	Internal Server Error - Error interno del servidor

🔑 ROLES DE USUARIO

Rol	Descripción
alumno	Estudiante en prácticas. Acceso a entregas, perfil y estancia
tutor_egibide	Tutor del centro educativo. Gestiona alumnos y seguimientos
tutor_empresa	Instructor en empresa. Evalúa competencias y realiza seguimientos
admin	Administrador del sistema. Acceso completo a todas las funcionalidades