

Esto es una copia de una conversación entre ChatGPT y Anónimo.

[Informar sobre la conversación](#)

Voy a hacerte un dictado para el desarrollo de una plataforma SaaS en Laravel, diseñada como microservicio API First para gestionar entidades mediante Endpoints CRUD. La plataforma se llamará Academicus, y será una plataforma e-learning.

Perfecto. Dictame nomás.

Para que el dictado quede bien “API First / microservicio” y no se nos escape nada, te propongo esta estructura: vos me vas diciendo y yo lo voy ordenando en un **documento maestro** (tipo spec) con:

- **Contexto del microservicio** (qué problema resuelve dentro de Academicus)
- **Entidades** (campos, validaciones, relaciones)
- **Endpoints CRUD** (routes, request/response JSON, errores)
- **Auth/roles/permisos** (si aplica)
- **Eventos/Jobs/Webhooks** (si querés arquitectura orientada a eventos)
- **Requisitos no funcionales** (rate limit, multi-tenant, auditoría, soft deletes, etc.)

Arranquemos por lo mínimo para definir el primer microservicio:

1. **¿Cuál es el primer microservicio que querés dictar?** (ej: Usuarios/Identidad, Cursos, Lecciones, Inscripciones, Pagos, Evaluaciones, etc.)
2. **¿Es multi-tenant?** (una Academicus por cliente/escuela, o todo en una sola “organización”)
3. **¿Autenticación:** Sanctum / Passport / JWT / tercero (Auth0, Cognito)?

Si querés, podés empezar directamente dictando la **primera entidad** (nombre + campos) y yo te lo voy bajando a: migración + modelo + policy + endpoints + ejemplos de JSON.

Cursos disponibles

1. Ayudantes de IIPP
2. Curso de técnicas de estudio
3. Curso de memorización
4. psicólogos de IIPP

- 5. Juristas de IIPP
- 6. Enfermería de IIPP
- 7. Trabajo social de IIPP
- 8. Cuerpo especial de IIPP

Pensó durante 14s

Anotado  Cursos disponibles (Academicus / entidad Course )

**Entidad:** Course

**Propósito:** catálogo de cursos e-learning disponibles para inscribirse/visualizar.

### Campos mínimos recomendados (API First)

- id (UUID o BIGINT)
- title (string, requerido)
- slug (string, único, requerido)
- description (text, opcional)
- status (enum: draft|published|archived , default published )
- published\_at (datetime, opcional)
- created\_at , updated\_at
- deleted\_at (soft delete, opcional)

### Seed inicial (los 8 cursos que dictaste)

#	title	slug
1	Ayudantes de IIPP	ayudantes-de-iipp
2	Curso de técnicas de estudio	curso-de-tecnicas-de-estudio
3	Curso de memorización	curso-de-memorizacion
4	Psicólogos de IIPP	psicologos-de-iipp
5	Juristas de IIPP	juristas-de-iipp
6	Enfermería de IIPP	enfermeria-de-iipp
7	Trabajo social de IIPP	trabajo-social-de-iipp
8	Cuerpo especial de IIPP	cuerpo-especial-de-iipp

## Endpoints CRUD (v1)

- `GET /api/v1/courses` (paginado + filtros: `status`, `search`, `sort`)
- `GET /api/v1/courses/{id}`
- `POST /api/v1/courses`
- `PATCH /api/v1/courses/{id}`
- `DELETE /api/v1/courses/{id}` (soft delete recomendado)

### Ejemplo de respuesta (GET one)

json

```
{  
  "data": {  
    "id": "uuid-o-id",  
    "title": "Curso de técnicas de estudio",  
    "slug": "curso-de-tecnicas-de-estudio",  
    "description": null,  
    "status": "published",  
    "published_at": "2026-01-05T00:00:00Z",  
    "created_at": "2026-01-05T12:00:00Z",  
    "updated_at": "2026-01-05T12:00:00Z"  
  }  
}
```

 Copiar código

Cuando sigas dictando, el próximo paso lógico es: **qué campos extra necesitas por curso** (precio, duración, modalidad, nivel, portada) y si cada curso tiene **módulos/lecciones** o es solo catálogo por ahora.

 Imagen cargada

Deberá poderse ver los cursos existentes, crear nuevos cursos, editar los cursos existentes, eliminar los cursos existentes, duplicar los cursos existentes. Los cursos estarán organizados por bloques, y dentro de los bloques se encontrarán los temas. Función a destacar a la hora de editar un curso, deberán poderse organizar los bloques y los temas mediante drag and drop. Nota: - PDF principal - Actualizaciones múltiples - Descargas adicionales (resúmenes, esquemas, extras). Que sean descargables, aunque se puedan también leer en la plataforma. Los videos serán alojados en YouTube y deben evitarse las

descargas. Las videoclases son referenciadas a cada tema al entrar en el tema pueda aparecer igual que el tema y pdf o resúmenes o infografías. ¿Qué funcionalidades mínimas debe incluir? - Gestión de cursos - Gestión de bloques: dentro de cada bloque se incluyen los temas, los bloques tenemos que poder crearlos y cambiarlos . Gestión de temas - Gestión de archivos (PDF/video) - Gestión de test y preguntas. Deberá poderse ingresar al curso automáticamente después del pago (Redsys) y manualmente desde el panel. Todos los cursos mensuales tienen que ir con suscripciones. Deben existir cursos cerrados donde el alumno puede ver todo el material y otro que se vea solo mes a mes cuando es por suscripción.

Pensó durante 1m 8s

Perfecto, con lo que dictaste, el **MVP (funcionalidades mínimas)** de Academicus debería cubrir esto (sin "extras" todavía):

## 1) Gestión de cursos (Backoffice)

- Ver listado de cursos + buscador/filtros (estado, tipo, etc.).
- Crear curso
- Editar curso
- Eliminar curso (ideal soft delete)
- Duplicar curso (duplicación "profunda": curso + bloques + temas + recursos + tests)
- Campos mínimos por curso:
  - `title`, `slug`, `description`
  - tipo de acceso:
    - `closed_full` (curso cerrado: ve todo el material)
    - `subscription_drip` (suscripción: contenido "mes a mes")
  - `status` (draft/published/archived)

## 2) Gestión de bloques y temas (estructura del contenido)

### Bloques dentro del curso

- CRUD de bloques (crear/editar/eliminar)
- Ordenar bloques por drag & drop (guardando `sort_order`)
- Ejemplo de bloques (los de tu captura):
  - Bloque I: Organización del Estado y Derecho Administrativo

- Bloque II: Derecho Penal
- Bloque III: Derecho Penitenciario
- Bloque IV: Conducta humana y habilidades sociales
- Bloque V: Tutorías

### Temas dentro de cada bloque

- CRUD de temas
- Ordenar temas por drag & drop dentro del bloque
- "Subtemas numerados" (Tema 1, Tema 2, etc.) → se resuelve con `sort_order` + numeración automática en frontend

## 3) Materiales y archivos (PDF + extras + video YouTube)

### Requisitos mínimos

- PDF principal del curso
  - Versionado / actualizaciones múltiples (historial de versiones, o al menos "última versión + registro")
  - Lectura en plataforma (visor PDF embebido)
  - Descarga permitida
- Descargas adicionales por curso o por tema (resúmenes, esquemas, infografías, extras)
  - Se pueden leer online y también descargar
- Videos: alojados en YouTube
  - Guardar referencia (`youtube_video_id` / URL embed)
  - Evitar descargas: no almacenar mp4, no entregar enlaces directos a archivos de video
- Asociación por tema: al entrar a un tema se ve:
  - contenido del tema + PDF(s) + extras + videoclases (embebidas)

## 4) Progreso del alumno (barra visual + acciones)

Dentro de cada bloque/curso:

- Barra de progreso visual: "3/15 clases completadas"
- Botón por tema:
  - "Marcar como visto" (estado `completed`)
  - "Revisar más tarde" (estado `review_later`)
- Mínimo de tracking:
  - estado por `user_id + topic_id` (y timestamps)
  - progreso agregado por bloque y por curso (calculado)

## 5) Tests y preguntas

MVP de evaluación:

- CRUD de **tests**
- CRUD de **preguntas**
- Opciones de respuesta (mínimo múltiple choice)
- Intentos del alumno:
  - iniciar intento
  - responder
  - finalizar y obtener resultado
- Asociación: test puede colgar de **curso** o de **tema** (lo más práctico: por tema)

## 6) Pagos (Redsys) + acceso automático y manual

- **Compra/pago Redsys**
  - crear orden/pago
  - endpoint de **notificación/retorno** para confirmar pago
  - al confirmarse pago → **matricular automáticamente** (enrollment)
- **Matrícula manual desde panel**
  - dar acceso a un usuario a un curso (sin pago) o extender acceso

## 7) Suscripciones para cursos mensuales

- "Todos los cursos mensuales" → deben ir con **suscripción**
- Dos comportamientos mínimos:
  1. **Curso cerrado (full)**: el alumno ve todo desde el inicio
  2. **Suscripción mes a mes (drip)**: el alumno ve solo lo liberado según su ciclo
- Implementación mínima de "drip":
  - a **bloques/temas** les agregás `release_month` (1,2,3...)
  - el alumno ve hasta `mes_actual_de_suscripcion`
  - si se cancela/vence → se corta el acceso (o se congela según regla)

---

## Entidades mínimas (para Laravel API)

- `courses`
- `course_blocks` (bloques) con `sort_order`
- `block_topics` (temas) con `sort_order` y opcional `release_month`
- `resources` (archivos/links) con tipo: `main_pdf`, `pdf`, `summary`, `extra`, `infographic`, `youtube_video`
- `topic_progress` (user-topic status: visto / revisar más tarde)

- quizzes , questions , options , attempts , attempt\_answers
  - enrollments (accesos)
  - payments (Redsys)
  - subscriptions (estado, start/end, plan)
- 

Si seguís dictando, lo próximo que necesito para cerrar bien el MVP es **cómo querés vender** cada curso:

- ¿Los "cerrados" son pago único?
- En los de suscripción: ¿se cobra mensual hasta cancelar? ¿hay permanencia mínima?

Mientras tanto, si querés, ya puedo bajarte esto a un **mapa de endpoints CRUD + endpoints especiales** (duplicar curso, reorder drag&drop, marcar visto, redsys notify, etc.) y te lo dejo listo como spec API v1.

Roles: Administrador (acceso y control total) , profesor (solo acceso a contestar preguntas de los usuarios no a ver lo que hacen los usuarios), gestor de contenidos (acceso y control de cursos, bloques y temas), y alumno (solo acceso a cursos, bloques y temas). Se debe permitir el acceso usando Google y Apple.

Pensó durante 23s

Perfecto. Para el MVP, esto se resuelve con **RBAC (roles/permisos) + autenticación por tokens y SSO Google/Apple**.

## Roles y permisos mínimos (RBAC)

### Roles

#### 1. Administrador

- Acceso total: usuarios, cursos, contenidos, matrículas, pagos, suscripciones, tests, auditoría.

#### 2. Gestor de contenidos

- Solo contenido: cursos, bloques, temas, orden drag&drop, recursos (PDF/extras), tests/preguntas.

- **No** puede ver actividad del alumno (progreso/detalle), ni gestionar pagos/suscripciones, ni usuarios (salvo ver datos mínimos si lo necesitás).

### 3. Profesor

- Acceso **solo a responder preguntas** de alumnos (Q&A / soporte académico).
- **No** puede ver progreso, intentos de test, actividad, ni gestión de contenidos/pagos.

### 4. Alumno

- Acceso a sus cursos (según matrícula/suscripción) y a consumir contenido.
  - Puede marcar visto / revisar más tarde, rendir tests.
  - No puede administrar contenido.
- 

## Matriz rápida de acceso (MVP)

Módulo / Acción	Admin	Gestor contenidos	Profesor	Alumno
Cursos CRUD + duplicar	✓	✓	✗	✗
Bloques/Temas CRUD	✓	✓	✗	✗
Reordenar (drag&drop)	✓	✓	✗	✗
Recursos (PDF/extras/YouTube)	✓	✓	✗	✗
Ver/consumir contenido	✓	✓ (preview)	✓ (solo para contexto si querés)	✓ (solo matriculado)
Progreso (ver analytics)	✓	✗	✗	✓ (solo propio)
Marcar visto / revisar más tarde	✓	✗	✗	✓
Tests CRUD	✓	✓	✗	✗
Rendir tests	✓	✗	✗	✓

Módulo / Acción	Admin	Gestor contenidos	Profesor	Alumno
Q&A (preguntas/respuesta s)	✓	✗ (opcional ver)	✓ (responder)	✓ (preguntar)
Pagos Redsys / Suscripciones	✓	✗	✗	✓ (solo pagar/estado propio)
Matrícula manual	✓	✗ (si querés: opcional)	✗	✗
Gestión de usuarios/roles	✓	✗	✗	✗

## Modelo de datos mínimo para RBAC

- `users`
- `roles` (`admin`, `content_manager`, `teacher`, `student`)
- `model_has_roles` (si usás roles múltiples)

Recomendación práctica en Laravel: **spatie/laravel-permission** (roles + permisos finos, policies y middleware).

Si querés mantenerlo ultra simple al principio: `users.role` como enum. Pero RBAC real  
te da flexibilidad para crecer (por ejemplo, profesor asignado a ciertos cursos).

## Q&A para “profesor responde”

MVP recomendado:

- `course_questions` (o `topic_questions`)
  - `id`, `course_id / topic_id`, `user_id` (alumno), `title`, `body`, `status`  
(open/answered/closed)
- `question_replies`
  - `id`, `question_id`, `user_id` (profesor/admin), `body`

Permisos clave:

- Alumno: crear pregunta + ver sus preguntas + ver respuestas.

- Profesor: listar preguntas (por curso asignado o global MVP) + responder.
  - Admin: todo.
- 

## Login con Google y Apple (API First)

### Qué hay que soportar

- Google Sign-In (OAuth2 / OpenID Connect)
- Sign in with Apple (OpenID Connect, id\_token JWT)

### Enfoque recomendado para microservicio API

Sanctum + login social por “token exchange”:

- Front (web/mobile) obtiene `id_token` del proveedor.
- Backend recibe `id_token`, lo valida y emite **tu token de API** (Sanctum).

### Endpoints mínimos de auth (v1)

- `POST /api/v1/auth/social/google`
- `POST /api/v1/auth/social/apple`

Body:

json

Copiar código

```
{ "id_token": "..." }
```

Response:

json

Copiar código

```
{ "token": "api_token", "user": { "id": "...", "role": "student", "email": "..." } }
```

Además:

- `POST /api/v1/auth/logout`
- `GET /api/v1/me`

### Tablas mínimas para social

- `social_accounts`
  - `user_id`, `provider` (`google|apple`), `provider_user_id`, `email`, `meta`

Notas importantes:

- Apple puede entregar nombre **solo la primera vez** y puede usar “private relay email”.

- Validación de `id_token` : verificar firma, issuer, audience, expiry.
- 

## Middleware / Policies (lo mínimo)

- Middleware:
    - `auth:sanctum`
    - `role:admin` / `role:content_manager` / etc.
  - Policies:
    - `CoursePolicy` , `BlockPolicy` , `TopicPolicy` , `ResourcePolicy`
    - `ProgressPolicy` (alumno solo su progreso)
    - `QuestionPolicy` (profesor responde, alumno crea)
- 

Si seguís dictando, el próximo punto que conviene definir es: **¿el profesor está asignado a cursos específicos** (y solo ve preguntas de esos cursos) o **ve todas las preguntas** (MVP rápido). Con cualquiera de las dos, el RBAC de arriba te queda sólido.

El menú principal (sidebar) deberá ser así: 1. Dashboard 2. Mi Plan de Estudio (2.1 Cronograma 2.2 Gestión del estudio 2.3 Retos semanales 2.4 Autoestudio 2.5 Progreso) 3. Temario 4. VideoClases 5. Tests 6. Progreso y resultados 7.Tutorías 8. Técnicas de estudio 9. Suscripciones 10. Perfil y ajustes

Queda definido Ese **sidebar** encaja perfecto como menú principal de **Alumno** (y en Admin/Backoffice conviene mostrar un menú distinto o “modo admin”).

Abajo te lo dejo ya bajado a **(A) estructura UI**, **(B) funcionalidades mínimas por sección**, y **(C) mapa de endpoints API v1** para que tu microservicio “Academicus” lo soporte.

---

### A) Sidebar (Alumno) — estructura final

1. Dashboard
2. Mi Plan de Estudio
  - 2.1 Cronograma
  - 2.2 Gestión del estudio

- 2.3 Retos semanales
  - 2.4 Autoestudio
  - 2.5 Progreso
3. Temario
  4. VideoClases
  5. Tests
  6. Progreso y resultados
  7. Tutorías
  8. Técnicas de estudio
  9. Suscripciones
  10. Perfil y ajustes

Nota práctica: "Progreso" aparece dos veces (2.5 y 6). Está bien si 2.5 es **progreso del plan** y 6 es **resultados/estadísticas** (tests + métricas globales). Yo lo separaría así para que no pisen.

---

## B) Funcionalidad mínima por sección (MVP)

### 1) Dashboard

- Resumen: cursos activos, próximo tema, progreso global, próximos hitos del cronograma, últimos tests.
- CTA rápidos: "Continuar", "Reanudar autoestudio", "Ver reto semanal".

### 2) Mi Plan de Estudio

#### 2.1 Cronograma

- Vista calendario/lista con el plan (por curso o global).
- Permitir "arrastrar/reprogramar" (opcional MVP) o editar fecha.

#### 2.2 Gestión del estudio

- Lista de tareas/objetivos (ej: "Bloque II – Tema 3").
- Estados: pendiente / en curso / hecho.

#### 2.3 Retos semanales

- Reto semanal (definido por el sistema o por el alumno).
- Check-in semanal + racha (streak) mínimo.

#### 2.4 Autoestudio

- Sesiones de estudio (timer opcional), notas, "marcar sesión completada".
- Enlazar sesión a curso/bloque/tema.

## 2.5 Progreso

- Progreso del plan: % completado vs planificado, atrasos, temas pendientes.

## 3) Temario

- Navegación curso → bloque → tema
- Dentro del tema:
  - PDF principal + extras descargables/visor
  - Botones: "Marcar como visto" / "Revisar más tarde"
  - (Si aplica) contenido bloqueado por mes (suscripción)

## 4) VideoClases

- Listado de videoclases (filtrar por curso/bloque/tema)
- Reproductor embebido de YouTube (sin descarga)
- Marcar visto / continuar

## 5) Tests

- Listado de tests disponibles (por curso/tema)
- Rendir test (intento), ver puntaje y feedback básico
- Bloqueo por suscripción (si el contenido del mes aún no se liberó)

## 6) Progreso y resultados

- Métricas globales: temas completados, tiempo de estudio (si lo medís), evolución de notas.
- Resultados por test (historial de intentos)

## 7) Tutorías

- Sistema mínimo tipo "soporte/Q&A":
  - Alumno crea pregunta (ideal: por curso/tema)
  - Profesor responde
  - Estados: open / answered / closed

## 8) Técnicas de estudio

- Contenido "tipo biblioteca" (posts/recursos internos) + descargables
- Puede ser un "curso especial" o un módulo aparte (MVP: módulo aparte con recursos)

## 9) Suscripciones

- Estado de suscripción, plan, próximo cobro, historial de pagos
- Cancelar/renovar (según como integres Redsys)
- Ver qué contenido está liberado "mes a mes"

## 10) Perfil y ajustes

- Datos básicos, seguridad, cerrar sesión
  - Vincular / desvincular Google / Apple (opcional)
  - Preferencias (zona horaria, notificaciones si las agregás)
- 

## C) API v1 mínima para soportar ese menú (microservicio)

### Cursos / Temario

- `GET /api/v1/courses` (mis cursos, según matrícula)
- `GET /api/v1/courses/{course}`
- `GET /api/v1/courses/{course}/blocks`
- `GET /api/v1/blocks/{block}/topics`
- `GET /api/v1/topics/{topic}` (incluye recursos + video refs)

### Recursos (PDF / extras / YouTube)

- `GET /api/v1/topics/{topic}/resources`
- `GET /api/v1/resources/{resource}` (metadata)
- `GET /api/v1/resources/{resource}/download` (solo PDFs/extras; **nunca** videos)

### Progreso (visto / revisar más tarde)

- `POST /api/v1/topics/{topic}/progress`  
Body: { "status": "completed" | "review\_later" | "in\_progress" }
- `GET /api/v1/courses/{course}/progress`
- `GET /api/v1/me/progress/summary` (para Dashboard y "Progreso y resultados")

### Plan de estudio

- `GET /api/v1/me/study-plan`
- `POST /api/v1/me/study-plan/items` (agregar objetivo/tarea)
- `PATCH /api/v1/me/study-plan/items/{id}` (estado/fecha)
- `GET /api/v1/me/weekly-challenges`
- `POST /api/v1/me/weekly-challenges/{id}/checkin`
- `POST /api/v1/me/study-sessions` (autoestudio)
- `GET /api/v1/me/study-sessions`

### Tests

- `GET /api/v1/tests` (filtrable por course/topic)
- `GET /api/v1/tests/{test}`

- POST /api/v1/tests/{test}/attempts
- POST /api/v1/attempts/{attempt}/answers
- POST /api/v1/attempts/{attempt}/submit
- GET /api/v1/me/test-results

## Tutorías (Q&A)

- GET /api/v1/questions (alumno: las suyas; profesor: las asignadas o todas según regla)
- POST /api/v1/questions
- GET /api/v1/questions/{question}
- POST /api/v1/questions/{question}/replies (profesor/admin)

## Suscripciones / Pagos

- GET /api/v1/me/subscriptions
- GET /api/v1/me/payments
- POST /api/v1/payments/redsys/init (crear orden)
- POST /api/v1/payments/redsys/notify (confirmación servidor-servidor)
- GET /api/v1/me/access (qué cursos/meses tiene liberados)

## Perfil / Auth (Google y Apple)

- POST /api/v1/auth/social/google
  - POST /api/v1/auth/social/apple
  - GET /api/v1/me
  - POST /api/v1/auth/logout
- 

## Visibilidad por rol (sidebar dinámico)

- **Alumno:** todo el menú tal cual lo dictaste.
  - **Profesor:** principalmente **Tutorías** (y opcional acceso de lectura a temario/videoclases para contexto, sin ver progreso del alumno).
  - **Gestor de contenidos:** no debería ver "Mi Plan", "Progreso y resultados", "Suscripciones"; sí un **panel de gestión** (Cursos/Bloques/Temas/Recursos/Tests) + reorder drag&drop.
  - **Administrador:** todo + Users/Roles + Matrículas + Pagos + Suscripciones + Auditoría.
-

Si querés, el siguiente paso natural es que me dictes qué se ve exactamente en "Temario" y en "VideoClases" (pantallas y filtros), y te lo transformo en:

- esquema de tablas ( courses/blocks/topics/resources )
- y endpoints de reorder drag&drop para bloques/temas (los "move"/"reindex" que guardan sort\_order ).

Listo, toda la información que te di hasta el momento es un dictado propio. ¿De acuerdo? Quiero que guardes toda esta información. Y a continuación te voy a dar la transcripción de un video del cliente. Que quiero que aproveches y me digas que esto es muy importante, insisto. Si hay cosas que se puedan agregar al dictado que yo hice, que lo destaque. Destaque con el título AGREGADO PARA EL DICTADO. Y debajo de ese encabezado, todo lo que se pueda agregar a lo que yo ya te dije. Y después quiero otro encabezado que sea REQUISITOS ADICIONALES Y debajo de ese encabezado, todo lo que corresponda a información nueva. Importantísimo que lo hagas de esta manera, insisto. Confirmame si entendiste y estás listo para continuar. Y te voy a enviar la transcripción del video del cliente. Y también te voy a enviar dos documentos PDF. Que también quiero que aproveches y coloques en AGREGADO PARA EL DICTADO. Gracias.

Sí, de acuerdo 

- Entendí que todo lo que me diste hasta ahora es tu dictado base para Academicus, y ya lo tengo como referencia.
- Estoy listo para recibir la transcripción del video del cliente y los 2 PDFs.

Cuando me los envíes, voy a devolverte EXACTAMENTE con este formato:

## AGREGADO PARA EL DICTADO

(únicamente cosas que se puedan sumar o encajar directamente en tu dictado actual)

## REQUISITOS ADICIONALES

(información nueva que no estaba en tu dictado, o requisitos extra que surgen del material del cliente)

Envíame ahora la **transcripción** y subí los 2 PDF (adjuntalos acá) y lo hago.

✉ Archivo cargado ✉ Archivo cargado

Buenos días. Eh, os presento lo que estamos buscando para la plataforma de academia, ¿vale? Primer lugar, en la parte de arriba podéis ver que pone cuerpo especial, acceso libre. Esto sería el nombre del curso con el logotipo y a la izquierda debería estar el logotipo de la empresa, la fecha de del día de hoy y después tenemos lo que sería una campana de notificaciones, un triángulo de alertas y para enviar mensajes. Después esto sería la parte de el nombre que tenga y un y un avatar. Vale, cuando vamos a al avatar, aquí pone atrévete o muere y se despliega mis cursos, mi perfil o cerrar sesión. Vale, dentro de mi perfil vemos que tenemos el avatar que lo puedes usarlo como foto de perfil en tus redes sociales, que lo puedes poner en tus redes sociales, la contraseña, confirmar contraseña, cambiar contraseña o descargar. Aquí le puedes elegir que sea hombre o mujer o puedes ir eligiendo qué uniforme quieras que lleve puesto. Se van abriendo los uniformes dependiendo la puntuación que tengan en la oposición, según sean de los tres tipos de opositores que pueden ser, que son oro, plata y bronce. En esta segunda diapositiva me voy a centrar en lo que es en el lateral de la página web. Eh, queremos que aparezca, no exactamente estos, pero que sean parecidos, ¿vale? Una zona de mi escritorio, una zona de estudio, temario, videoclases, test, simulacros, eh zona de batalla de opositores, estadísticas, materiales, acción tutorial y suscripciones, ¿vale? Esto sería lo que estaría en la parte izquierda de la página de la página web. En mi escritorio, cuando entramos en mi escritorio, vamos a ver en la parte superior eh cómo me encuentro yo respecto a los demás, ¿vale? Cómo voy respecto a los demás, ¿vale? Sale un logotipo y aquí va a ir viendo de cada una de las materias qué nivel o una puntuación que tengo respecto a los demás, cuántos test tengo hechos, cuál es el ranking de las batallas que va a estar relacionado con las batallas entre los opositores. Las batallas de los opositores son que hacen un mismo test de dos opositores y a ver quién gana. Y en la en el último recuadro, ¿cuántas páginas llevo estudiadas? ¿Cuántas

páginas llevo de la primera fase? Va a haber una fase de lectura o de estudio, de subrayado, una fase de repaso y una fase de repaso rápido. Entonces, eh según la fase, pues hay un circulito donde te va diciendo el nivel que tengas. Y después eh otro más que es cuántas páginas llevo estudiadas dependiendo de cada uno de ellos. Esto con un vistazo lo que nos sirve es que que podemos ver directamente qué es lo que lo que ya más llevamos estudiado, lo que llevamos menos dependiendo de del tiempo que llevemos y demás. Y en la parte de abajo habrá como dos cronogramas. ¿Qué clases puedo ver en directo esta semana? Aparecen las clases en directo de esta semana, que estos son los vídeos en directo. Cuando aparece este simbolito es que es son vídeos. ¿Y qué tengo que estudiar esta semana? Esto sería como el cronograma de esta semana que está relacionado con lo que luego ya veremos con los cronogramas y las videoclases. Es decir, aquí lo que me dices es, son qué clases puedo ver esta semana, que va cambiando por semanas y qué es lo que tengo que estudiar. Y todo esto está en la página de inicio, en la página de de mi escritorio. En la zona de estudio, pues tenemos lo que es un cronograma, una gestión de estudio, un planificador de estudio y un progreso de estudio. ¿Vale? El cronograma, pues un cronograma básico en el que eh se puedan poner lo que sean son las videoclases, que cuando demos a la videoclase, pues vayamos a esa videoclase si está grabada y aparece aquí eh del mes de septiembre, por ejemplo, aparece eh aquí todo el mes de septiembre, pues si ya ha sido dada, pues que se vea que se pueda ir directamente a la videoclase. Si es los tests, pues lo mismo, eh, se abre un test y te manda al test si es descargable o al material si es otro tipo de material, ¿vale? Entonces, aquí abajo se ve que, por ejemplo, el 8 solo hay videoclase. En este hay videoclase y que hay material porque están eh aquí abajo encendidos lo que es cada una de las de las aplicaciones que hay. el cronograma. Esto es la zona de estudio, que es el autoestudio guiado mensual, en el que vamos a ir poniendo nosotros qué es lo que tienen que ir estudiando cada día o cada semana, ¿vale? Simplemente se viene relleno, pero nosotros eh lo damos cada mes y ellos no pueden modificarlo. Mientras que en la planificación de estudio aquí lo que hacen es eh, perdón, Vale, esto es el el

autoestudio en el planif en el planificador de estudio, cuando viene el tema tal, el tema que sea, pues en cada uno de los temas se le va a abrir arriba estudio para el día 10 se le va a abrir arriba los tres temas y en qué fase se encuentran, si en la fase uno, en la fase dos o en la fase tres y el número de páginas que hay. Cuando se da clic aquí, pues ellos pueden cambiar el tiempo que han estado estudiando y y poner y hacer test de esa plataforma. de ese de ese tema. Esto sería el progreso de estudio que ellos van llegando, ¿vale? Que van llevando. ¿Veis? Las tres fases. Entonces, dependiendo de la fase que se encuentre, pues se va a iluminar una de las tres fases que se encuentra. Aquí están en la Constitución Española. Horas estudiadas, 2 horas. Tiempo, número de páginas, tanto preguntas, se ha hecho, tanto ratio, tanto último estudio, la fecha del último estudio. Entonces, aquí viene lo que es cada uno el porcentaje que lleva estudiado. Aquí las horas selectivas estudiadas de cada uno de los temas. de organización del Estado. Y aquí es esta parte de aquí es lo que se abre al dar un clic en en el tema. Entonces, en el tema tú puedes decir, "Oye, que he hecho el estudio, tiempo medio para esta fase", te dice el tiempo medio que se necesita para estudiarlo. Tiempo de estudio que he tenido, una hora, tiempo eh de las pausas completadas, tanto tiempo. Fecha eh fecha para la fase completada, el 21 de julio, excluirla o no. Vale, entonces aquí con esto el alumno va metiendo eh las horas que ha que ha ido estudiando, ¿vale? Si hoy ha estudiado media hora, pues se le va a aplicar aquí la media hora que mañana mete otra otra media hora, pues aplica y le va a aplicar el último día del estudio y cuánto tiempo ha estado estudiando. En cuanto al temario, simplemente que aparezcan los distintos bloques y que se dé a descargar, se dé un clic y aparezca el temario eh por bloques y que se y se pueda descargar los distintos PDFs, las videoclases. Pues lo mismo, eh, dentro de videoclases va a ver lo que es el horario, las que son en directo y las que son en diferido. Las que son en directo, pues eh cuando le das a cada una de la clase, te va a aparecer de qué es la materia que se va a dar y quién es el profesor y qué días se van a qué días se van a dar esas clases y y cuáles son las videoclases que hay, que al dar aquí en el clic este iríamos directamente a la videoclase. Una cosa importante que se me ha olvidado

explicar en la planificación del estudio es que aquí lo que buscamos la parte la parte que es nuestra está el cronograma cerrado, pero necesitamos un cronograma diferente que en el que ellos puedan ir cambiando lo que les falta. Es decir, hoy han estudiado esto, lo dan como estudiado, como la lección completada, por ejemplo, y la pasan para dentro de 3 días para repasarla y se lo y lo pueden ir modificando ellos cuándo es lo que quieren que estudien y cuándo es lo que ellos no quieren que estudien. La puedan dar la lección, por ejemplo, cada una de las partes que la puedan dar eh completada o que la puedan dar e que necesita repaso, ¿vale? Que se que se pueda se pueda saber eh que ellos puedan decidir qué tanto por ciento de del tema lo llevan y dependiendo de ello lo puedan modificar dentro de su calendario para ir estudiando ellos como ellos quieran, ¿vale? Nosotros les damos en la programación normal, pero que ellos tengan un calendario en el que puedan modificar lo que son los temas. Tema uno de tal lo pongo aquí, tema dos, lo pongo para tal día, tema 80 para tal día y que ellos puedan seguir con la programación. de lo que van haciendo y que en ese tema aparezca eh qué tipo o qué porcentaje eh llevan ellos de de aprendizaje. Si llevan un 25% de cada uno de los puntos que tiene el tema, pues este el 100% me lo sé, no lo paso aquí. El otro sí que me lo sé, pues o no me lo sé bien lo paso para el día 3 que lo tengo que volver a repasar porque no me lo he aprendido. La otra parte no me la sé, la paso para el día 5 porque no tengo tiempo y que creen ellos un un que puedan modificar ellos su cronograma atendiendo a su situación personal. Vale. Eh, según íbamos el temario, eh, aquí hemos visto los las videoclases, ¿vale? Dentro de las videoclases en diferida aparece un esto más o menos eh todo lo al final lo buscamos que sea como más tipo más cuadrado y no haya tanta letra, ¿vale? Que sea más fácil de ver. Derecho penitenciario, temas, pues cuando abres los temas, que se vean los distintos temas y ya coges las videoclases. Repasos por temas, si son en talleres, si son directos en diferido, ¿vale? Entonces, cuando entras ya dentro, pues derecho. penitenciario temas, ves los subidos recientemente, que son los últimos, pues si quieres verlos h y seguir con ellos con lo último que se ha visto y después puedes ir y buscar por los distintos temas. Ay,

pues voy a ir al tema uno. El tema uno va a tener la los distintos puntos que tema que tiene el tema uno y las distintas clases que vienen recogidas ahí. ¿Vale? Lo que también sería interesante es que dentro del tema, a ver que pasamos en la siguiente pantalla, eh, estas son una clase en diferido que simplemente el vídeo, introducción, material, sesión, enviar una duda o añadir una anotación. Añadir anotación no es importante, pero aquí sí que podría aparecer eh que podamos incluir algún otro archivo como podría ser el tema supuestos que que se vayan a ver, que se puedan ver y que se puedan descargar desde aquí, ¿vale? No solamente desde la de la parte del temario. Aquí las videoclases, pues lo que buscamos eh, perdón, el test, pues lo que buscamos lo mismo, hacer test en vez de poner posición global, organización, derecho penitenciario, que es cada uno de los bloques que tiene esta oposición. que se pueda que se que en vez de que aparezcan aquí que una en el momento que desaltan ya eh por bloques para poder elegir. Esto es la posición global de los tests. Es la eh es un un en cuanto al test, esto es la posición global, es decir, en la que intentamos que vea el alumno qué percentil va teniendo con respecto al resto, el percentil de de las preguntas dependiendo del nivel, la nota media global, la calidad de los tests, eh si son generales, pues qué porcentaje lleva, si es de cada uno y que si es de cada uno de los de las de los de los bloques, ¿vale? Para que sepa en global y para que sepa en cada uno de los bloques. Aquí sería como sería un test personalizado a la carta, ¿vale? Tú puedes elegir lo que poníamos arriba, el bloque que queremos en bloques, cada uno de los bloques o o coger un multibloque, ¿vale? Y después debajo te aparecen los tests que se van creando porque tú les vas creando, no son que hayan que estén fijos, sino que los coges escogiéndolos. ¿Y cómo se escogen? Pues desde aquí, por ejemplo, puedes coger multibloque, que es el más el más global de todos, en el que aparecen todas las materias. Tú eliges las materias que quieras y el tipo de preguntas, todas, las acertadas, las falladas, las no contestadas, las nunca vistas de exámenes oficiales con video explicación, los distintos tipos de dificultad, si seleccionas por temas y el número de preguntas y entonces esto lo que hace es crearte un test y ese test pues si entra entra dentro multibloque. Después

las respuestas primero te verifica de que es lo que has cogido, que has explicado esto y después te saca las preguntas. Las preguntas puedes marcarla la correcta, puedes desmarcar la respuesta y eh puedes marcarla como duda, ¿vale? Puedes marcarla como duda. ¿Para qué? Porque luego eh se hace un análisis de también cuántas de las preguntas que se falla que que de las que se tienen dudas, qué porcentaje aciertan para que la no sepa qué nivel tiene en cuanto a las preguntas que también tiene dudas. Una vez que has hecho el test, pues salen los resultados. Eh, estadísticas del test, las acertadas, las falladas, las no contestadas, el porcentaje de preguntas por dificultad, análisis por dificultad de la pregunta y después eh saldría lo que es la pregunta, si es fallada, la explicación, si es correcta también se podría poner la explicación y aquí es super importante ponerla como favorita. si quieras que se te vuelva a poner, eh, marcar una anotación, si quieras poner alguna anotación, impugnarla o enviar una duda. En cuanto a las batallas, al final son simplemente test, eh, que el alumno lo que hace es eh contacta con con otro, le solicita poder hacer un test del tema o del bloque que sea. Entonces, lo solicita si los dos están de acuerdo, hacen como una especie de batalla, tienen un tiempo para hacer desde que ellos quieran les llegue les llega una anotación, por ejemplo, les llega al escritorio una notificación para que hagan el test. Dan al test que se ha creado, se da el mismo y el que y el que gana pues obtiene más puntos que el otro y dependiendo de eso pues va pasando dependiendo del porcentaje si es oro, plata o bronce. ¿Vale? Y esto serían las estadísticas, ¿vale? Que lo puedes hacer por por las distintas materias, derecho penitenciario, derecho penal, función pública y las estadísticas pues pueden ser globales o pueden ser también por materias y que se pueden comparar con otro tipo de alumnos, los de bronce, plata u oro, ¿vale? Esto del bronce, plata o oro se les va eh metiendo dependiendo de las horas que hacen, de las horas que estudio que meten y del tiempo que llevan también en la academia y se les va dando una valoración, ¿vale? ¿Cuál es la nota media? ¿Cuántos test es realizado? Contesto preguntas o se puede poner un listado con con los mejores por test, los mejores en derecho. penitenciario de este T son este, este, este y se les da una puntuación la

nota que hayan sacado. Y después pues materiales en los que se pueden ¿Qué materiales han subido en los talleres? Pues que se los puedan descargar o en los anexos de clase o en los directos o en los tesis supuestos o en las guías esquematizadas. Estos son los materiales. Y después pues en el lateral también acción tutorial, que puede ser más que acción tutorial también eh que se pueda mandar un mensaje al tutor eh y también que se pueda aparezcan aquí lo que serían videoclases de técnicas de estudio, quiénes son tus tutores y un panel de suscripción.

Pensó durante 2m 4s

## AGREGADO PARA EL DICTADO

**MUY IMPORTANTE** (del video + PDFs): el cliente le da muchísimo peso a **gamificación, estadísticas comparativas, cronogramas (uno “cerrado” de la academia + otro editable del alumno), y batallas/simulacros** como motor de retención.

### UI / Layout (alumno)

- **Header superior:** nombre del curso + logo; a la izquierda **logo empresa + fecha del día**; iconos de **notificaciones (campana)**, **alertas (triángulo)** y **mensajes**; a la derecha **nombre + avatar** con menú desplegable (“Mis cursos / Mi perfil / Cerrar sesión”).
- **Sidebar:** el cliente muestra un layout “tipo” con secciones parecidas a las que ya definiste, pero con naming alternativo: “Mi escritorio” (Dashboard), “Zona de estudio” (Mi plan), “Temario”, “Videoclases”, “Test”, “Simulacros”, “Batallas”, “Estadísticas”, “Materiales”, “Acción tutorial”, “Suscripciones”.

### Perfil (avatar + gamificación visual)

- En “Mi perfil”: avatar configurable (descargable) + cambio de contraseña.
- Personalización: género (hombre/mujer) y **uniformes desbloqueables** según nivel del alumno (**oro / plata / bronce**) (gamificación visual ligada al progreso).

### Dashboard “Mi escritorio” (lo que debe mostrar)

- Tarjetas de “cómo voy respecto a los demás” por materias (comparativas).
- Métricas: tests realizados, ranking de batallas, y **páginas estudiadas**.
- Progreso por **fases** (3 estados): fase de estudio/lectura + repaso + repaso rápido (indicador visual).
- Dos cronogramas en la home:
  - “**Clases en directo esta semana**” (videoclases en directo).
  - “**Qué tengo que estudiar esta semana**” (tareas/temario).

## Mi Plan de Estudio (detallado)

- Cronograma tipo calendario con tareas por día, sincronizado con temario/test y con colores por tipo de tarea (temario/test/supuestos/repaso). [proyecto\\_academicus](#)
- Gestión del estudio: **temporizador tipo Pomodoro**, drag&drop para reorganizar, registro de tiempo por bloque/tema, informe horas reales vs planificadas y métricas (horas, temas vistos, tests, % aciertos). [proyecto\\_academicus](#)
- Reto semanal: retos definidos por usuario (o sistema), alertas, recomendaciones automáticas y gamificación ligera. [proyecto\\_academicus](#)
- **Doble enfoque de cronograma (clave del video):**
  1. cronograma mensual "cerrado" que define la academia;
  2. **cronograma editable del alumno** donde puede mover temas, marcarlos completados / "necesita repaso", reprogramarlos y registrar avance porcentual por partes del tema.

## Temario / recursos

- Temario por bloques y temas; cada tema debe permitir **descargar PDF principal + ver actualizaciones** y descargar extras. [BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
- Los recursos deben poder verse/descargarse también **desde dentro de la videoclase** (no solo desde "Temario") (lo recalca el cliente en el video).

## Videoclases / Tutorías (estructura y UX)

- Videoclases: al abrir, ver **cronograma mensual de clases** y otro acceso "por bloques/temas". [proyecto\\_academicus](#)
- Dentro de cada bloque: subtemas numerados + barra de progreso + "marcar como visto / revisar más tarde". [proyecto\\_academicus](#)
- En clases: enviar dudas, puntuar, mandar feedback, y permitir adjuntar/descargar archivos (tema, actualizaciones, resúmenes, esquemas, tests en PDF).  
[proyecto\\_academicus](#)
- Tutorías: mismas lógicas que videoclases (directo + grabado) y **buscador** de dudas por bloques/temas. [proyecto\\_academicus](#)

## Tests + Supuestos (funcionalidad ampliada)

- Mantener "test a la carta" y separar claramente **Test vs Supuestos**. [proyecto\\_academicus](#)
- Opciones clave del test (del video):
  - filtros por acertadas/falladas/no contestadas/nunca vistas, exámenes oficiales, dificultad, temas, nº de preguntas;
  - marcar pregunta como **duda**;

- post-test: métricas por dificultad + revisión con explicación cuando exista, y marcar como **favorita**, anotar, impugnar, enviar duda.
- Confirmado: repetir **solo falladas y control de tiempo** por test/pregunta.

[!\[\]\(0bfc65f44c8abd2b170640998d4b431b\_img.jpg\) BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

## Batallas + ranking (muy importante para engagement)

- "Batallas entre opositores": emparejamiento, mismo test para ambos, notificaciones, puntos y ranking; mensual. [!\[\]\(a07aa36d78c53e31e70392f2e6464460\_img.jpg\) BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
- Gamificación por niveles oro/plata/bronze y notificaciones para volver a la plataforma (retos/batallas/tareas). [!\[\]\(287c4fb1f5e17c57768ce0ff15937137\_img.jpg\) BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

## Progreso y resultados (métricas "core")

- Panel tipo dashboard con % avance temario, promedio aciertos, horas, racha, aciertos por bloque/tema, evolución temporal y comparativa con media de alumnos. [!\[\]\(a860f4cfbb09c556b57af1b6bdb25c78\_img.jpg\) BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
  - Recomendaciones automáticas ("mapa de rendimiento") y mensajes motivacionales que aparezcan periódicamente (idealmente desactivables). [!\[\]\(77a60bdb82ebaf837021a162aa5e2acb\_img.jpg\) proyecto\\_academicus](#)
- 

## REQUISITOS ADICIONALES

### Tipos de cursos / productos (más variedad que el dictado base)

- Se piden cursos de **12 meses y de 18 meses** y además distintos "formatos":
  - curso grabado completo 12m y 18m,
  - cursos de planificaciones mensuales (cada mes como curso distinto),
  - curso presencial (temas + test + supuestos + planificación + tutoría),
  - curso "solo test",
  - y un acceso de **simulacros de examen** para opositores. [!\[\]\(b5bfbe237dcc6fc7340d789169daa234\_img.jpg\) proyecto\\_academicus](#)
- Además: un apartado de **simulacros gratuitos** accesible a cualquier usuario registrado aunque no pague (solo ven esa sección + clasificación).

[!\[\]\(92604bff2a286d454d073adc13337191\_img.jpg\) BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

## Registro / acceso

- Un alumno puede estar en **varios cursos a la vez** y se pide **validación de email**.

[!\[\]\(821acc15c470127e6ad8ef362d2c5a7f\_img.jpg\) BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

- En un documento se menciona candado/descarga por tema y desplegar actualizaciones. [BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
- En el brief se indica que **no** hace falta candado porque cada alumno verá lo de su curso (mensual/global). [BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)  
➡ Esto hay que resolver como regla de negocio: **bloqueo por suscripción/mes vs "solo por pertenencia al curso".**

## Directos / infraestructura de clases

- Directos: aún no está elegida la plataforma (mencionan “tipo Zoom” y que se recomienda una). El link debería aparecer en el cronograma y opcionalmente enviarse por notificación. [BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
- Hosting de vídeo: YouTube privado, con posibilidad de Vimeo.  
[BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

## Pagos, facturación y promociones

- Métodos de pago: tarjeta + **transferencia** con alta manual desde panel.  
[BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
- **Cupones/descuentos.** [BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
- Área para que el alumno pueda **descargar factura.** [BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

## Tipos de supuestos (más complejos)

- Supuestos de:
  - tipo test (4 opciones / 1 correcta),
  - tipo test “5 respuestas”,
  - y supuestos con **respuesta escrita.**
  - También pruebas de “rellenar la palabra que falta del artículo”.

[BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

## No funcional / operación

- Pico estimado: **~50 alumnos simultáneos.** [BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
- Backup automático: lo consideran “buena idea” (requisito operativo).  
[BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)
- Necesidad de **migrar contenido fácilmente.** [BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

## Entregables y mantenimiento

- Entregable: **video explicando cómo usar la plataforma + formación.**  
[BRIEF\\_TÉCNICO\\_PARA DESARROLLO\\_D...](#)

- Mantenimiento mensual post-garantía y evolución futura (especialmente gamificación). [!\[\]\(d20e99f063ba78fb860cdc65719a5813\_img.jpg\) BRIEF\\_TÉCNICO\\_PARA\\_DESARROLLO\\_D...](#)

Fuentes

Checklist con absolutamente todo para auditar el trabajo de programación y desarrollo web

A continuación tenés un **checklist ultra completo** (auditoría end-to-end) para revisar **programación + desarrollo web** de Academicus (Laravel SaaS, API-First, microservicio CRUD, e-learning, roles, pagos, suscripciones, contenidos, tests, gamificación, estudio). Está pensado para que puedas auditar **con evidencias**: PRs, logs, capturas, endpoints en Postman, métricas, scripts, etc.

---

## 0) Evidencias mínimas que el equipo debe entregarte

- **Repo(s)** con historial de commits claro (no “update” repetido).
  - **README** de instalación + `.env.example` completo.
  - **Postman/Insomnia collection** + environment(s).
  - **OpenAPI/Swagger** actualizado (ideal).
  - **Credenciales de staging** (admin, content manager, teacher, student).
  - **Dataset seed** (cursos, bloques, temas, recursos, tests) para probar.
  - **Guía de deploy** (staging + prod) + rollback.
  - **Registro de decisiones** (tradeoffs) y lista de pendientes (tech debt).
- 

## 1) Trazabilidad: el desarrollo cumple el dictado y lo del cliente

### 1.1 Matriz de requisitos (imprescindible)

- Existe tabla/matriz “Requisito → Implementación → Endpoint/Screen → Test → Evidencia”.
- Todos los puntos del dictado base están mapeados:
  - Cursos CRUD + duplicar

- Bloques/Temas CRUD + drag&drop (reorder)
  - PDF principal versionable + extras descargables/legibles
  - YouTube embed (evitar descargas)
  - Progreso (visto / revisar más tarde) + barra
  - Tests/preguntas
  - Acceso post-pago Redsys + matrícula manual
  - Suscripciones: cerrado vs drip mes a mes
  - Roles (admin, gestor contenidos, profesor Q&A, alumno)
  - Login Google + Apple
  - Sidebar alumno (con secciones)
- Puntos “AGREGADO PARA EL DICTADO” del cliente están contemplados:
    - Dashboard “Mi escritorio” con métricas comparativas
    - Doble cronograma: academia cerrado + editable del alumno
    - Simulacros, batallas, ranking, oro/plata/bronze, notificaciones
    - Perfil con avatar/uniformes desbloqueables (si se incluyó)
  - Todo lo “no hecho” está en un **backlog con alcance** y “no confundir MVP vs fase 2”.
- 

## 2) Arquitectura: API-First / microservicio bien planteado

- Separación clara: **API vs frontend** (o BFF si existe).
  - Versionado de API: `/api/v1/...`
  - Convenciones REST consistentes (status codes, errores, paginación).
  - Estructura de carpetas Laravel ordenada (Domains/Modules o similar).
  - No hay lógica de negocio en controllers (uso de Services/Actions).
  - Eventos/Jobs/Queues definidos donde corresponda (pagos, notificaciones, async).
  - Feature flags o configuración para activar módulos (batallas, simulacros).
- 

## 3) Calidad de código y mantenibilidad

### 3.1 Estándares y revisión

- PSR-12 / Pint / PHP-CS-Fixer configurado y ejecutado en CI.
- PHPStan/Psalm (nivel razonable) en CI.
- Nombres claros: clases, métodos, endpoints, variables.
- No hay “god classes” ni helpers gigantes.
- Reutilización correcta (traits, policies, resources).

## 3.2 Manejo de errores

- Errores con formato consistente (ej. `code`, `message`, `errors`).
  - Validaciones con mensajes útiles.
  - Manejo de excepciones global (Handler) sin filtrar datos sensibles.
- 

## 4) Seguridad (crítico)

### 4.1 Auth y sesiones/tokens

- Autenticación API por **Sanctum/JWT** (definido) consistente.
- Rotación / expiración de tokens (o estrategia definida).
- Logout invalida token (si corresponde).
- Rate limit en login/SSO y endpoints sensibles.

### 4.2 SSO Google / Apple

- Validación real del `id_token` (firma, issuer, audience, exp).
- Manejo Apple: email relay y “nombre solo primera vez”.
- Vinculación de cuentas (misma persona loguea con Google/Apple).

### 4.3 Roles y permisos (RBAC)

- Admin: acceso total.
- Gestor contenidos: solo cursos/bloques/temas/recursos/tests.
- Profesor: **solo** responder Q&A (no ve progreso/actividad).
- Alumno: solo contenido permitido + su progreso.
- Policies/middleware aplicados en **todos** los endpoints (no solo UI).
- Pruebas automáticas por rol (al menos smoke tests).

### 4.4 Seguridad de archivos

- URLs firmadas para descargas (si es privado).
- Validación de tipo/tamaño de archivos (PDF, imágenes, etc.).
- Antivirus/escaneo (si aplica) o al menos mitigación.
- No exponer rutas internas en responses.

### 4.5 OWASP básico

- CSRF (si hay sesión web).
- CORS configurado correctamente.
- Prevención XSS/HTML injection (si renderizan contenido).
- Prevención IDOR (acceso por id a recursos ajenos).

- Logs sin datos sensibles (tokens, contraseñas, payload de pago completo).
- 

## 5) Base de datos y modelo (integridad)

### 5.1 Esquema

- Tablas mínimas y relaciones bien modeladas:
  - courses
  - blocks
  - topics
  - resources (pdf principal, extras, youtube refs)
  - progress (user\_topic status)
  - quizzes/tests, questions, options, attempts, answers
  - enrollments
  - subscriptions
  - payments (Redsys)
  - Q&A: questions, replies
  - (si aplica) battles, rankings, simulacros, achievements
- Foreign keys y constraints (con cascadas razonables).
- Índices en columnas de búsqueda y relaciones (course\_id, user\_id, status, slug).

### 5.2 Migraciones/seeders

- Migraciones reproducibles desde cero.
  - Seeders de demo para QA.
  - Soft deletes donde se definió (cursos, etc.).
  - Auditoría / timestamps coherentes.
- 

## 6) API: diseño, consistencia y DX

### 6.1 CRUD y endpoints especiales

- Cursos: list/get/create/update/delete/duplicate
- Bloques y temas: CRUD + reorder
- Recursos: list/get/download
- Progreso: marcar visto / revisar más tarde; resumen por curso
- Tests: generar intento, responder, submit, resultados
- Suscripciones: estado, ciclo, contenido liberado (drip)

- Pagos: init + notify + confirmaciones seguras
- Q&A: alumno pregunta, profesor responde

## 6.2 Paginación / filtros / orden

- List endpoints paginados (cursor/offset) con límites.
- Filtros (status, search, course, block, topic).
- Orden consistente (por `sort_order`, `created_at`, etc.).

## 6.3 Contratos y documentación

- OpenAPI describe request/response reales.
  - Ejemplos incluidos.
  - Versionado y changelog de API.
- 

# 7) Drag & Drop (reordenamiento) — auditoría específica

- Reorder de bloques **dentro de curso** (estable).
  - Reorder de temas **dentro de bloque**.
  - Endpoint transaccional: o aplica todo o nada.
  - Se valida pertenencia (no mover tema a bloque de otro curso sin permiso).
  - Conurrencia: si dos editan, hay estrategia (`updated_at`, lock optimista, etc.).
  - `sort_order` sin duplicados ni gaps problemáticos (o normalización al guardar).
- 

# 8) Contenido y recursos (PDF, extras, YouTube)

## 8.1 PDF principal versionable

- Existe “PDF principal” por curso con historial de versiones (o estrategia definida).
- Se puede ver online (visor) y descargar.
- Se registran versiones (fecha, autor, notas de cambio).

## 8.2 Extras descargables

- Extras por curso y/o por tema.
- Lectura online + descarga.
- Permisos correctos según matrícula/suscripción.

## 8.3 Videos YouTube

- Se guarda `youtube_video_id` /embed URL (no mp4).
- No hay endpoint “download video”.

- Controles de acceso: aunque sea YouTube, respetar gating del curso en la app.
- 

## 9) Progreso del alumno y “plan de estudio”

### 9.1 Progreso por tema

- Estados mínimos: `completed`, `review_later` (y opcional `in_progress`).
- Barra “X/Y completadas” por bloque/curso.
- Solo el alumno ve su progreso; admin puede ver global; gestor contenidos NO.

### 9.2 Cronogramas (cliente lo considera clave)

- Cronograma cerrado (definido por academia, no editable por alumno).
  - Cronograma editable del alumno:
    - mover temas en calendario
    - marcar “completado” / “necesita repaso”
    - reprogramar y registrar % avance
  - Registro de tiempo estudiado (si se implementó):
    - tiempos por fase (estudio/repaso/repaso rápido)
    - historial por tema
- 

## 10) Tests / Simulacros (motor de evaluación)

### 10.1 Tests estándar

- Banco de preguntas por bloque/tema/dificultad.
- Intentos: crear → responder → submit → resultados.
- Métricas: aciertos/fallos/no contestadas, % por dificultad.
- Explicación por pregunta (si existe).

### 10.2 “Test a la carta” (cliente)

- Filtros: acertadas/falladas/no contestadas/nunca vistas; dificultad; temas; nº preguntas.
- “Marcar como duda” y estadísticas de dudas.
- Post-test: favorito, anotación, impugnar, enviar duda (según alcance).

### 10.3 Simulacros (si entró en alcance)

- Simulacros separados de test.
- Simulacros gratuitos para registrados (si se implementó).

- Clasificación/ranking de simulacros.
- 

## 11) Q&A / Tutorías (profesor)

- Alumno crea duda por curso/tema.
  - Profesor responde (y solo eso).
  - Estados: open/answered/closed.
  - Notificación al alumno al responder.
  - Buscador/filtrado por bloque/tema (si se implementó).
- 

## 12) Suscripciones + gating (cerrado vs drip)

- Curso cerrado: acceso a todo desde el inicio.
  - Curso drip mensual: contenido liberado por mes/ciclo.
  - Reglas claras:
    - qué pasa si cancela / expira
    - qué pasa si reanuda
  - En backend se valida gating (no confiar en frontend).
- 

## 13) Pagos Redsys + matrícula manual

### 13.1 Flujo Redsys

- Init pago: crea orden con referencia única.
- Notify callback: valida firma/params (según Redsys) y confirma pago.
- Idempotencia: misma notificación no duplica matrícula.
- Registro en `payments` : status, timestamps, raw safe subset.
- Acceso automático tras pago confirmado.

### 13.2 Matrícula manual

- Admin puede otorgar acceso manual.
- Registro de auditoría: quién lo otorgó y por qué.

### 13.3 Facturas / cupones (si aplica por brief)

- Descargar factura (o placeholder documentado).
  - Cupones/descuentos si entraron en alcance.
-

## 14) Notificaciones y mensajería

- Campana de notificaciones con endpoints (listar, marcar leída).
  - Alertas (triángulo) definidas (qué es alerta vs notificación).
  - Mensajes internos/tutor (si se implementó).
  - Preferencias (silenciar, frecuencia) si aplica.
- 

## 15) Gamificación (si está en alcance)

- Rangos oro/plata/bronce definidos por reglas medibles.
  - Ranking (batallas/tests/horas) con cálculo transparente.
  - Avatar/uniformes desbloqueables (si se implementó).
  - Batallas:
    - solicitud, aceptación, generación de test igual, ventana de tiempo
    - puntos y resolución de empate
    - anti-trampa básico (tiempos, reintentos, etc.)
- 

## 16) Frontend/UX (si aplica)

- Sidebar coincide con lo definido (o mapeo nombre↔sección).
  - Estado “loading/empty/error” en pantallas clave.
  - Buscadores, filtros y paginación.
  - Responsive mínimo (mobile/tablet).
  - Accesibilidad básica (teclado, contraste, labels).
- 

## 17) Rendimiento y escalabilidad

- Objetivo concurrente (p.ej. 50 usuarios simultáneos) testeado.
  - Queries sin N+1 (eager loading).
  - Caching donde corresponda (catálogo, temario, stats).
  - Rate limits por endpoint.
  - Storage eficiente para PDFs/extras.
  - Colas para tareas pesadas (emails/notificaciones/reportes).
- 

## 18) Observabilidad (logs, métricas, auditoría)

- Logs estructurados (request\_id, user\_id).
  - Trazabilidad de errores (Sentry o similar, o estrategia).
  - Métricas básicas: latency, error rate, queue depth.
  - Auditoría de acciones admin (crear/editar/eliminar/duplicar, matrícula manual).
- 

## 19) CI/CD y ambientes

- Tests + lint + static analysis corren en CI.
  - Deploy reproducible (Docker o scripts).
  - Separación de ambientes: dev/staging/prod.
  - Secrets gestionados (no hardcode).
  - Rollback definido y probado (al menos en staging).
- 

## 20) Backups, migración y mantenimiento

- Backups automáticos (DB + storage) y restauración probada.
  - Migración de contenido fácil (import/export) si es requisito.
  - Plan de mantenimiento mensual (qué cubre, SLAs si aplica).
  - Documentación para el cliente (video/guía) si está pactado.
- 

## 21) Testing (mínimos exigibles)

- Unit tests de lógica clave (gating, reorder, scoring).
  - Feature/API tests:
    - permisos por rol
    - CRUD cursos/bloques/temas
    - duplicación profunda
    - progress
    - tests attempts
    - redsys notify idempotente
  - E2E smoke (login, entrar curso, ver tema, marcar visto).
  - Pruebas de seguridad (IDOR, rate limit, roles).
- 

## 22) Auditoría de “cosas típicas que se rompen”

- Duplicar curso duplica también orden y recursos.
  - Reordenar no rompe numeración ni progreso existente.
  - El alumno NO ve contenido fuera de su matrícula/suscripción.
  - Profesor NO puede ver progreso/actividad.
  - Descargas solo para quien corresponde.
  - YouTube no expone descarga.
  - Redsys no genera dobles matrículas por reintentos.
  - Slugs únicos; cambios de título no rompen URLs (redirección o regla).
- 

Si querés, te lo convierto en una plantilla de auditoría (tabla “Requisito → OK/NO → Evidencia → Observaciones → Severidad → Acción”) lista para pegar en Notion/Docs/CSV.

#### Checklist de entidades, vistas y migraciones

Acá tenés un checklist práctico de entidades (tablas/modelos), migraciones y vistas/pantallas para Academicus (MVP + lo agregado del cliente). Está organizado para que audites “¿existe? ¿está conectado? ¿se usa?”.

---

## 1) Checklist de ENTIDADES (Modelos/Tablas)

### 1.1 Core Académico (obligatorio)

- User ( `users` )
- Role ( `roles` ) + pivots ( `model_has_roles` / `role_user` ) (*según implementación*)
- Course ( `courses` )
- CourseBlock ( `course_blocks` )
- Topic ( `topics` )
- Resource ( `resources` )
- Enrollment ( `enrollments` )

### 1.2 Progreso y estudio (MVP dictado)

- TopicProgress ( `topic_progress` ) (*visto / revisar más tarde / timestamps*)
- CourseProgressSummary (*ideal como query/view, no necesariamente tabla*)
- StudyPlan ( `study_plans` ) (*si hay plan global por usuario*)

- **StudyPlanItem** ( `study_plan_items` ) (*tareas en calendario*)
- **StudySession** ( `study_sessions` ) (*autoestudio / registro tiempo*)

### 1.3 Cronogramas (cliente: doble calendario)

- **AcademySchedule** ( `academy_schedules` ) (*cronograma cerrado*)
- **StudentSchedule** ( `student_schedules` ) (*cronograma editable alumno*)
- **StudentScheduleItem** ( `student_schedule_items` ) (*reprogramación + estado + % avance*)

### 1.4 Evaluaciones (tests / preguntas)

- **Test** ( `tests` )
- **Question** ( `questions` )
- **QuestionOption** ( `question_options` )
- **Attempt** ( `attempts` )
- **AttemptAnswer** ( `attempt_answers` )
- (*opcional MVP*) **QuestionTag/Difficulty** ( `question_tags` , `difficulties` )

### 1.5 Q&A / Tutorías (profesor responde)

- **SupportQuestion** ( `support_questions` ) (*dudas del alumno por curso/tema*)
- **SupportReply** ( `support_replies` )

### 1.6 Pagos y suscripciones

- **Payment** ( `payments` ) (*Redsys + manual*)
- **Subscription** ( `subscriptions` )
- **Plan** ( `plans` ) (*si hay múltiples planes*)
- **Invoice** ( `invoices` ) (*si se descarga factura*)
- **Coupon** ( `coupons` ) (*si hay descuentos*)

### 1.7 Notificaciones y mensajería (header: campana/alertas/mensajes)

- **Notification** ( `notifications` ) (*Laravel notifications o tabla propia*)
- **Alert** ( `alerts` ) (*si es distinto a notification*)
- **MessageThread** ( `message_threads` )
- **Message** ( `messages` )

### 1.8 Gamificación (si entra en alcance)

- **RankTier** ( `rank_tiers` ) (*oro/plata/bronze*)
- **UserRank** ( `user_ranks` ) (*puntos, tier actual*)
- **Battle** ( `battles` )

- **BattleParticipant** ( `battle_participants` )
- **BattleResult** ( `battle_results` )
- **Achievement** ( `achievements` ) (*uniformes desbloqueables*)
- **UserAchievement** ( `user_achievements` )

## 1.9 Media/Archivos (versionado PDF principal)

- **CourseMainPdfVersion** ( `course_main_pdf_versions` ) (*versionado del PDF principal*)
- (*opcional*) **FileAsset** ( `file_assets` ) (*storage genérico para PDFs/extras*)
- (*si perfil avatar descargable*) **AvatarAsset** ( `avatar_assets` )

## 1.10 Auditoría (recomendado)

- **AuditLog** ( `audit_logs` ) (*acciones admin: duplicar, matricular, borrar, etc.*)
- 

# 2) Checklist de MIGRACIONES (qué debe existir sí o sí)

## 2.1 Principales (core)

- `create_users_table`
- `create_roles_table` + pivot (si RBAC)
- `create_courses_table`
  - `title`, `slug` (unique), `status`, `access_type` (closed/drip), timestamps, softDeletes
- `create_course_blocks_table`
  - `course_id`, `title`, `sort_order`, timestamps, softDeletes
- `create_topics_table`
  - `block_id`, `title`, `sort_order`, `release_month` (*si drip*), timestamps, softDeletes
- `create_resources_table`
  - `owner_type/owner_id` (polimórfico: curso o tema), `type` (main\_pdf/extrá/youtube), `path / url`, `mime`, `size`, `is_downloadable`, timestamps
- `create_enrollments_table`
  - `user_id`, `course_id`, `status`, `starts_at`, `ends_at`, `source` (*redsys/manual*)

## 2.2 Progreso (MVP)

- `create_topic_progress_table`
  - `user_id`, `topic_id`, `status` (completed/review\_later), `completed_at`, timestamps

- Unique index: ( user\_id , topic\_id )

## 2.3 Tests

- create\_tests\_table (course\_id/topic\_id, title, config)
- create\_questions\_table (test\_id o banco global + metadata)
- create\_question\_options\_table
- create\_attempts\_table (user\_id, test\_id, started\_at, submitted\_at, score, meta)
- create\_attempt\_answers\_table (attempt\_id, question\_id, option\_id/null, is\_flagged\_doubt, is\_favorite, notes, timestamps)

## 2.4 Tutorías/Q&A

- create\_support\_questions\_table
  - course\_id/topic\_id , user\_id , title , body , status , timestamps
- create\_support\_replies\_table
  - support\_question\_id , user\_id (teacher/admin), body , timestamps

## 2.5 Suscripciones y pagos

- create\_plans\_table (si aplica)
- create\_subscriptions\_table
  - user\_id , course\_id , plan\_id , status , current\_period\_start/end , timestamps
- create\_payments\_table
  - user\_id , course\_id , subscription\_id? , provider (redsys/manual), reference , status , amount , currency , timestamps
  - Index + unique: reference
- (si aplica) create\_coupons\_table
- (si aplica) create\_invoices\_table

## 2.6 Cronogramas (si se implementa doble calendario)

- create\_academy\_schedules\_table + items
- create\_student\_schedules\_table + items
  - campos en item: date , type (video/test/material), course\_id/topic\_id , status , %progress , notes

## 2.7 Notificaciones/Mensajes

- notifications (si Laravel notifications)
- create\_message\_threads\_table
- create\_messages\_table

## 2.8 Gamificación/Batallas (si entra)

- `create_rank_tiers_table`
- `create_user_ranks_table`
- `create_battles_table` + participants + results
- `create_achievements_table` + `create_user_achievements_table`

## 2.9 Versionado PDF principal (muy recomendado)

- `create_course_main_pdf_versions_table`
    - `course_id`, `version`, `path`, `uploaded_by`, `changelog`, timestamps
- 

# 3) Checklist de VISTAS / PANTALLAS (UI)

## 3.1 Layout global alumno

- **Header:** logo empresa + fecha + notificaciones + alertas + mensajes + usuario/avatar + menú
- **Sidebar:** el que definiste (Dashboard, Mi Plan de Estudio..., Temario, etc.)

## 3.2 Alumno (MVP)

- **Dashboard**
  - resumen progreso + "continuar"
  - (si cliente) comparativas/ranking + cronograma semanal
- **Mi Plan de Estudio**
  - Cronograma (cerrado)
  - Planificador editable (alumno)
  - Gestión del estudio (tiempo/tareas)
  - Retos semanales
  - Autoestudio
  - Progreso
- **Temario**
  - lista bloques/temas
  - vista tema: PDF principal + extras + video + acciones (visto / revisar más tarde)
- **VideoClases**
  - listado por curso/bloque/tema + directos/diferidos
  - vista videoclase: player YouTube + materiales adjuntos + enviar duda
- **Tests**
  - listado

- hacer test
- resultados + revisión
- **Progreso y resultados**
  - historial tests + métricas
- **Tutorías**
  - crear duda + ver respuestas
- **Técnicas de estudio**
- **Suscripciones**
  - estado/plan/pagos
- **Perfil y ajustes**
  - datos + seguridad + logout
  - (si cliente) avatar/uniformes + descarga

### 3.3 Backoffice (Admin + Gestor de contenidos)

- Gestión de cursos (listar/crear/editar/eliminar/duplicar)
- Editor de curso con **drag&drop** bloques/temas
- Gestión de recursos (PDF principal versionado + extras + youtube refs)
- Gestión de tests/preguntas
- (admin) matrículas manuales, pagos, suscripciones, usuarios/roles

### 3.4 Profesor

- Panel de **Tutorías/Q&A**
    - listado
    - responder
    - filtros por curso/tema (*si aplica*)
- 

## 4) “Definition of Done” para auditar rápido

Para cada entidad/pantalla:

- Migración existe y se ejecuta limpio
  - Factory/Seeder (mínimo)
  - CRUD API funcionando (Postman)
  - Policies por rol
  - Validaciones request
  - Tests automáticos básicos
  - UI consume la API (sin endpoints “fantasma”)
-

Si querés, te lo dejo aún más utilizable como **lista de verificación por módulo** (en formato CSV con columnas: Módulo, Item, Estado, Evidencia, Severidad) para que lo pegues directo en tu flujo de QA.

Estoy aquí para responder dudas

Genial. Para aprovechar que estás disponible, te tiro las **dudas más importantes (de definición)** que hoy mismo condicionan la arquitectura y el desarrollo. Respondeme con el número + tu decisión (o "A/B/C").

- 1. Multi-tenant:** ¿Academicus es una sola academia (un cliente) o varias academias/escuelas dentro del mismo SaaS?
  - A) Single-tenant (una academia)
  - B) Multi-tenant por “organización” (muchas academias)
- 2. Tipos de cursos/productos** (por el material del cliente): ¿qué entra en MVP sí o sí?
  - A) Solo “curso mensual por suscripción” + “curso cerrado”
  - B) Además “solo test”, “presencial”, “simulacros” (y gratis para registrados)
- 3. Cronograma doble** (cerrado + editable alumno): ¿entra en MVP?
  - A) Sí, ambos
  - B) Solo cerrado (academia)
  - C) Solo editable alumno (y lo cerrado queda como plantilla)
- 4. Fases de estudio** (estudio/repaso/repaso rápido) + páginas estudiadas:
  - A) MVP incluye registro de páginas y horas por fase
  - B) MVP solo “visto / revisar más tarde” (sin fases)
- 5. Tests “a la carta”:**
  - A) MVP incluye filtros (falladas/nunca vistas/dificultad/temas) + marcar duda + favoritas
  - B) MVP test estándar por tema (sin personalización)
- 6. Batallas (PVP) + ranking + oro/plata/bronce:**
  - A) MVP sí
  - B) Fase 2
- 7. Redsys:** ¿el pago es solo tarjeta o también transferencia/manual desde el inicio?
  - A) Solo tarjeta Redsys
  - B) Tarjeta + transferencia con alta manual
- 8. Acceso YouTube:** ¿los videos son públicos/no listados/privados?
  - A) No listados

- B) Privados (más complejo por permisos)
- C) Da igual, solo embed

## 9. PDF principal versionado:

- A) Historial real de versiones (tabla + listado)
- B) Solo "reemplazar" y guardar última versión

## 10. Profesor: ¿ve todas las dudas o solo las de cursos asignados?

- A) Todas
- B) Solo asignados (requiere "teacher\_course" o similar)

Con esas 10 respuestas, cierro el alcance MVP y te dejo el spec final (entidades + endpoints + reglas).

1. Es una sola academia (un cliente)
2. Todo
3. A
4. A.
5. A.
6. A.
7. B.
8. C.
9. B.
10. B

Perfecto, queda definido así (y esto ya fija arquitectura + backlog):

1. Single-tenant (una sola academia)
2. Entra TODO en MVP  (incluye: suscripción mensual, curso cerrado, solo test, presencial, simulacros, batallas, ranking, gamificación, etc.)
3. Cronograma doble: cerrado + editable alumno
4. Fases + páginas + horas por fase
5. Tests a la carta + duda + favoritas + filtros completos
6. Batallas + ranking + oro/plata/bronze
7. Pagos: Redsys tarjeta + transferencia con alta manual
8. YouTube: embebido (sin preocuparse por si es público/privado, solo referencia)
9. PDF principal: solo última versión (sin historial)
10. Profesor: ve solo dudas de cursos asignados  (requiere relación teacher↔course)

## Impacto inmediato (lo que hay que agregar sí o sí)

- Asignación profesor→curso: entidad/pivot `course_teacher` o similar para filtrar tutorías.
- Doble calendario:
  - `academy_schedule_items` (no editable por alumno)
  - `student_schedule_items` (editable: mover, completar, repaso, % avance)
- Tracking avanzado de estudio:
  - por tema: fases (1/2/3), páginas, horas, pausas, último estudio.

- **Tests a la carta:**
  - reglas de generación de test desde banco de preguntas + filtros.
  - atributos por pregunta: dificultad, origen (oficial), tags/tema, etc.
- **Batallas + ranking + tiers:**
  - flujo solicitud/aceptación, test espejo, puntos, clasificación y tier oro/plata/bronze.
- **Pagos mixtos:**
  - Redsys + "transferencia/manual" con auditoría de admin.
- **Productos:**
  - modelar "tipo de curso"/producto (12m/18m, mensual, presencial, solo test, simulacros, gratis registrados).

Si querés, el siguiente paso que te recomiendo (para que puedas auditar al dev sin piedad) es que te devuelva un **esquema definitivo** con:

- **entidades + campos** (DB)
- **endpoints API v1**
- **reglas de negocio** (gating suscripción, drip, batallas, generación de tests)

Decime "Dale" y lo saco en formato checklist/spec listo para desarrollo y QA.

Dale y adicionalmente Checklist con absolutamente todo para auditar el trabajo de programación y desarrollo web, Checklist de entidades, vistas (frontend también) y migraciones

Pensado durante 17s

Dale. Te dejo dos checklists:

1. **Checklist total (absolutamente todo) para auditar programación + desarrollo web** (backend + frontend + devops + seguridad + QA).
2. **Checklist específico de entidades + migraciones + vistas (frontend también)**, alineado con TODO el alcance definido (single-tenant, doble cronograma, fases, tests a la carta, batallas, ranking, oro/plata/bronze, Redsys + transferencia manual, teacher por curso, YouTube embed, PDF principal solo última versión).

# 1) Checklist TOTAL para auditar programación y desarrollo web (end-to-end)

## 1. Evidencias y trazabilidad (lo primero que pedís)

- Documento “Scope/MVP” firmado (lo que entra/lo que NO entra) y coincide con tus decisiones.
- Matriz Requisito → Implementación → Endpoint → Pantalla → Tests → Evidencia.
- Colección Postman/Insomnia completa y funcional (staging).
- OpenAPI/Swagger (o al menos docs) consistente con la API real.
- Accesos de QA: admin / gestor / profesor / alumno.
- Seeds/datos demo: cursos, bloques, temas, recursos, preguntas, batallas.
- Lista de pendientes + deuda técnica + riesgos.

## 2. Repo / Git / Calidad de entrega

- Ramas + PRs: revisiones reales, no merges directos a main.
- Commits con mensajes útiles, sin “fix” repetido.
- Conventional commits / changelog (ideal).
- Tags/releases para staging y prod.

## 3. CI/CD y ambientes

- CI corre: lint + static analysis + tests.
- Deploy reproducible (Docker o scripts) + rollback.
- Staging y Prod separados (DB, storage, secrets).
- Config segura: `.env.example` completo; secrets fuera del repo.
- Migraciones se ejecutan en pipeline (con control).

## 4. Arquitectura (Laravel API-First / “microservicio”)

- API versionada `/api/v1`.
- Controllers finos + Services/Actions (negocio fuera del controller).
- DTO/Request validation consistente.
- Responses normalizadas (Resources/Transformers).
- Manejo de errores centralizado (códigos + mensajes + validaciones).
- Encolado (jobs) para tareas pesadas (notificaciones, generación de tests/batallas si aplica).

## 5. Seguridad (crítico)

### 5.1 Auth + SSO Google/Apple

- Login social: valida id\_token (issuer/aud/exp/firma) correctamente.

- Vinculación de cuentas (misma persona con 2 proveedores).
- Rate limit en login / auth endpoints.
- Logout invalida token (si corresponde).

## 5.2 RBAC (roles y permisos)

- Admin = todo.
- Gestor = cursos/bloques/temas/recursos/tests (NO pagos, NO actividad de alumnos).
- Profesor = SOLO tutorías/Q&A de cursos asignados (NO progreso, NO tests globales).
- Alumno = solo sus cursos + su progreso.
- Policies/middleware aplicados en API, no solo UI.
- Tests automáticos por rol (smoke al menos).

## 5.3 Protección de datos / IDOR

- Un alumno no puede acceder por ID a recursos de otro curso o usuario.
- Descargas: URLs firmadas o control de permisos servidor-side.
- Logs sin tokens/contraseñas/datos de pago sensibles.

## 6. Base de datos y consistencia

- Integridad referencial (FKs) donde corresponda.
- Índices clave: `user_id`, `course_id`, `topic_id`, `status`, `slug`, `created_at`.
- Soft deletes donde aplica (cursos, bloques, temas, recursos opcional).
- Transacciones en operaciones críticas: duplicado, reorder, confirmación de pago, cierre de batalla.

## 7. API: calidad, consistencia, DX

- Paginación y filtros en listados.
- Sorting consistente (`sort_order`, fechas).
- Errores de validación consistentes.
- Idempotencia (pagos, notificaciones, batallas).
- Documentación actualizada y usable.

## 8. Módulos funcionales (auditoría por feature)

### 8.1 Cursos (CRUD + duplicar)

- Listar/crear/editar/eliminar (soft) cursos.
- Duplicar profundo: curso + bloques + temas + recursos + tests config (y opcional preguntas si corresponde).

- Slug único; cambios controlados.

## 8.2 Bloques/Temas + Drag&Drop

- CRUD bloques/temas.
- Reorder transaccional (no queda a medias).
- Validación de pertenencia (no mover tema a otro curso sin permiso).
- Conurrencia (lock optimista: updated\_at/version) o estrategia clara.

## 8.3 Recursos (PDF principal + extras + lectura + descarga)

- PDF principal: solo última versión (reemplazo), pero con auditoría mínima (quién y cuándo lo subió).
- Extras: descargables + visualización en plataforma.
- Visibilidad por gating (suscripción/curso).
- No se exponen rutas internas.

## 8.4 Videos YouTube (sin descarga)

- Guardar youtube\_video\_id o URL embed.
- Sin endpoints de descarga de video.
- Control de acceso: aunque sea embed, solo usuarios autorizados ven/reciben el link.

## 8.5 Progreso y fases de estudio (A)

- Por tema: fases (estudio/repaso/repaso rápido).
- Registro: horas, páginas, pausas, último estudio.
- Estados mínimos (completado / necesita repaso / en curso).
- Resúmenes: por curso/bloque/tema, métricas "mi escritorio".

## 8.6 Doble cronograma (A)

- Cronograma cerrado (academia): editable solo admin/gestor.
- Cronograma editable alumno: mover temas, completar, reprogramar, % avance.
- Home muestra: "directos esta semana" + "qué estudiar esta semana".

## 8.7 Tests a la carta (A)

- Banco de preguntas bien indexado (tema/bloque/dificultad/origen).
- Generación de test por filtros: falladas, acertadas, no contestadas, nunca vistas, exámenes oficiales, dificultad, temas, nº preguntas.
- Marcar pregunta como duda.
- Post-test: resultados por dificultad + favorito + impugnar + enviar duda (si entró).
- Anti-trampa básico (tiempos, reintentos, consistencia).

## 8.8 Simulacros (incluye “gratis registrados” si aplica en tu alcance)

- Simulacros separados de tests regulares.
- Acceso gratuito: solo simulacros + clasificación (sin contenido premium).
- Ranking de simulacros.

## 8.9 Batallas (A) + ranking + oro/plata/bronze

- Flujo: solicitar → aceptar → generar test espejo → resolver → puntuar.
- Ventana de tiempo/estado de batalla.
- Reglas de puntuación, empate, abandono.
- Ranking global + por materias/bloques.
- Tier oro/plata/bronze: reglas transparentes (horas, puntos, tiempo en academia, etc.).

## 8.10 Tutorías/Q&A (profesor por curso)

- Alumno crea duda por tema/curso.
- Profesor responde **solo** en cursos asignados.
- Notificación al alumno al responder.
- Buscador/filtros por bloque/tema.

## 8.11 Suscripciones + productos (TODO entra)

- Modelado de productos: 12m, 18m, mensual, presencial, solo test, simulacros.
- Gating: curso cerrado vs drip mensual (si aplica por producto).
- Cambio de plan, cancelación, expiración, reactivación.

## 8.12 Pagos (B): Redsys + transferencia manual

- Redsys: init → notify/callback → confirmación → otorgar acceso.
- Validación firma/parametría Redsys.
- Idempotencia: notify repetida no duplica matrícula.
- Transferencia: alta manual con auditoría (quién, cuándo, evidencia).
- Facturas (si aplica): descarga desde panel.

# 9. Frontend / UX / Integración

- Sidebar y header como definido (notifs/alertas/mensajes + avatar menu).
- Estados UI: loading/empty/error para cada vista.
- Reglas de visibilidad por rol en frontend (sin reemplazar seguridad backend).
- Calendarios usables (cerrado vs editable).
- Motor de tests: UX robusta (marcar duda, cambiar respuesta, submit seguro).
- Batallas: notificaciones y estados claros.

- Accesibilidad básica (teclado, labels) + responsive mínimo.

## 10. Performance / escalabilidad / seguridad operacional

- Sin N+1 (eager loading).
- Cache en listados y stats pesadas.
- Jobs para cálculos (rankings, stats comparativas).
- Rate limit + protección de endpoints costosos.
- Objetivo concurrencia probado (y documentado).

## 11. Observabilidad y auditoría

- Logs estructurados (request\_id, user\_id).
- Monitoreo errores (Sentry u otro) o estrategia.
- Auditoría: duplicar curso, matrícula manual, cambios de roles, cambios de contenido.
- Backups automáticos + restauración probada.

## 12. Testing (mínimo exigible)

- Unit: scoring, gating, reorder, generación test a la carta, battle rules.
  - Feature/API: permisos por rol, pagos idempotentes, cronogramas, Q&A asignación profe-curso.
  - E2E smoke: login SSO, entrar curso, ver tema, marcar progreso, hacer test, ver ranking.
- 

## 2) Checklist de ENTIDADES + MIGRACIONES + VISTAS (Frontend incluido)

### 2.1 ENTIDADES / TABLAS (con lo que sí o sí debería existir)

#### A) Identidad y roles

- users
- roles + pivots (RBAC) ( model\_has\_roles o role\_user )
- social\_accounts (google/apple)
- course\_teachers (pivot profesor↔curso)  (por tu decisión 10.B)

#### B) Cursos y contenido

- courses (incluye tipo/producto + modo acceso)
- course\_blocks (bloques)

- `topics` (temas)
- `resources` (PDF principal, extras, links, youtube)
- `enrollments` (acceso a curso por pago/manual/suscripción)

### C) Progreso y estudio (fases + horas + páginas)

- `topic_study_progress` (por usuario/tema: fase, páginas, % avance, último estudio)
- `study_sessions` (registro de sesiones y tiempos/pausas)
- `study_phase_logs` (opcional, si querés historial por fase)

### D) Cronogramas (doble calendario)

- `academy_schedule_items` (cerrado por academia)
- `student_schedule_items` (editable por alumno)

### E) Tests y banco de preguntas (a la carta)

- `question_bank` (preguntas globales) o `questions` globales
- `question_options`
- `question_meta` (dificultad, origen oficial, tags) o columnas directas
- `tests` (definiciones “plantilla” o generados)
- `test_attempts`
- `attempt_answers`
- `user_question_stats` (acertada/fallada/no contestada/nunca vista) clave para filtros
- `question_favorites` (o columna en stats)
- `question_flags` (duda/impugnación/anotación) (puede unificarse)

### F) Simulacros

- `mock_exams` (simulacros)
- `mock_exam_attempts`
- `mock_exam_rankings` (o ranking calculado)

### G) Batallas + ranking + tiers

- `battles`
- `battle_participants`
- `battle_attempts` (o link a `test_attempts`)
- `rank_tiers` (oro/plata/bronze)
- `user_ranks` (puntos, tier, métricas)
- `leaderboards` (materializado opcional) o cálculo on-demand + cache

## H) Tutorías / Q&A

- `support_questions` (dudas)
- `support_replies`
- `support_attachments` (si adjuntan archivos)
- `support_categories` (opcional: por bloque/tema)

## I) Pagos, suscripciones y productos (TODO entra)

- `products` (12m/18m/mensual/presencial/solo test/simulacros)
- `plans` (si hay variantes)
- `subscriptions`
- `payments` (Redsys + transferencia)
- `invoices` (si descargan factura)
- `coupons` (descuentos)

## J) Notificaciones y mensajería

- `notifications` (o Laravel notifications)
- `alerts` (si las separan)
- `message_threads`
- `messages`

## K) Perfil “avatar/uniformes” (si entra en MVP “todo”)

- `avatars` / `user_avatars` (o `users.avatar_meta`)
- `uniforms`
- `user_uniforms` (desbloqueo por tier/puntos)

## L) Auditoría / Operación

- `audit_logs`
  - `failed_jobs` / `jobs` (si usan queue)
  - `password_resets` / security logs (según auth)
- 

## 2.2 MIGRACIONES (checklist por archivo/estructura)

Para cada tabla anterior:

- FK + índices.
- `created_at/updated_at` y `deleted_at` donde corresponda.
- Uniques clave:
  - `courses.slug`

- (`user_id`, `topic_id`) en progreso
  - `payments.reference`
  - (`teacher_id`, `course_id`) en asignación docente
  - Enum/constraints (status, fase, tipo recurso, tipo producto).
  - Migraciones corren en limpio en CI y en staging.
- 

## 2.3 VISTAS / PANTALLAS FRONTEND (por rol)

### Alumno (sidebar final + agregado cliente)

#### Header

- Logo academia + fecha + notificaciones + alertas + mensajes
- Avatar + menú: Mis cursos / Mi perfil / Cerrar sesión

#### 1. Dashboard / "Mi escritorio"

- Comparativas "cómo voy respecto a los demás"
- Métricas: tests hechos, ranking batallas, páginas estudiadas
- Progreso por fases (3)
- "Directos esta semana"
- "Qué estudiar esta semana"

#### 2. Mi Plan de Estudio

- 2.1 Cronograma (cerrado academia)
- 2.2 Gestión del estudio (tiempos/pausas)
- 2.3 Retos semanales
- 2.4 Autoestudio
- 2.5 Progreso (del plan)
- Cronograma editable alumno (mover, completar, repaso, % avance)

#### 3. Temario

- Bloques/temas
- Vista tema: PDF principal (última versión), extras, video embebido, acciones de progreso

#### 4. VideoClases

- Directos vs diferidos
- Por bloque/tema + "subidos recientemente"
- Vista clase: player + materiales adjuntos + enviar duda

#### 5. Tests

- Catálogo por bloque/tema
- Test a la carta (filtros completos)
- Ejecución de test (UX robusta)
- Resultados + review + favoritas + dudas + impugnar + anotación (según alcance)

## 6. Progreso y resultados

- Historial tests/simulacros
- Métricas por bloque/tema + evolución

## 7. Tutorías

- Crear duda por curso/tema
- Ver respuestas

## 8. Técnicas de estudio

- Biblioteca de contenido + videoclases técnicas + materiales

## 9. Suscripciones

- Plan/producto activo
- Pagos (Redsys/transferencia)
- Estado/renovación/cancelación (si aplica)

## 10. Perfil y ajustes

- Perfil, contraseña
- Avatar descargable + uniformes desbloqueables (oro/plata/bronce)
- Preferencias (opcional)

## Extras cliente (si realmente entran como pantallas separadas)

- Simulacros (incluye gratis para registrados si aplica)
- Batallas (solicitar, aceptar, jugar, resultados)
- Estadísticas comparativas
- Materiales (repositorio por tipo)

## Profesor

- Panel Tutorías/Q&A (solo cursos asignados)
- Ver preguntas, responder, adjuntar recursos si aplica
- Notificaciones de nuevas dudas

## Gestor de contenidos (backoffice)

- Gestión de cursos (CRUD + duplicar)
- Editor de curso: bloques/temas + drag&drop reorder
- Gestión recursos (PDF principal reemplazo + extras + youtube refs)

- Gestión tests/preguntas/banco
- Gestión cronograma cerrado (academia schedule)

## Admin

- Todo lo de gestor +
  - Usuarios + roles + asignación profesor↔curso
  - Matrículas manuales
  - Pagos (Redsys + transferencia)
  - Productos/planes/cupones/facturas
  - Auditoría/logs (al menos listado de acciones)
- 

Si querés, en el próximo mensaje te lo convierto a **formato CSV** listo para tu auditoría (columnas: Área, Item, OK/NO, Evidencia, Severidad, Responsable, Fecha ) y te lo dejo ya para copiar/pegar a tu flujo (Notion/Sheets/Jira).

<https://github.com/selvaggiesteban/planning-as-code>  
Planifica y entrega las plantillas para descargar . Aprovecha las plantillas del repositorio de GitHub y reemplaza toda la información en esos tres documentos ¿tienes dudas?

Pensado durante 3m 38s

Pensado durante un par de segundos

Listo, entendí y no tengo dudas. Ya planifiqué y armé las 3 plantillas tipo “planning-as-code” (product-overview, plan, resume) **rellenas con Academicus**, y además agregué los 2 checklists que pediste (auditoría total + entidades/vistas/migraciones).

Descargas:

- [Academicuss\\_Planning-as-Code\\_Templates.zip](#)

Incluye:

- templates/product-overview/product-overview.md + .json
- templates/plan/plan.md + .json
- templates/resume/resume.md + .json
- Academicus\_Audit\_Checklist.md
- Academicus\_Entities\_VIEWS\_Migrations\_Checklist.md

Archivos sueltos (por si los querés sin el zip):

- [Academicus\\_Audit\\_Checklist.md](#)
- [Academicus\\_Entities\\_VIEWS\\_Migrations\\_Checklist.md](#)

Cuando me pegues la transcripción final restante / me indiques qué parte exacta querés "reemplazar" con más fidelidad al repo, lo ajusto para que quede 1:1 con tu formato preferido.