



ESTRUCTURA DEL PROYECTO

Nombre tentativo: HabitWise (¡cámbialo como prefieras)








Concepto:

Una aplicación web que ayuda al usuario a **crear hábitos saludables**, hacer **seguimiento diario** y ver su **progreso con estadísticas visuales**. También cuenta con sistema de puntos y metas para mantener la motivación.





FUNCIONALIDADES PRINCIPALES

Módulo	Funciones
 Usuarios	Registro, login, roles (<code>Admin</code> , <code>Usuario</code>)
 Gestión de Hábitos	CRUD de hábitos con frecuencia y tipo
 Registro Diario	Marcar hábitos cumplidos día a día
 Estadísticas	Porcentaje de cumplimiento, rachas, gráficas
 Metas	Agrupar hábitos bajo objetivos mayores
 Puntos y rachas	Gamificación: puntos por cumplimiento
 Exportación	(Opcional) Informe PDF / Excel
 Sobre mí	Perfil con link a tu CV/LinkedIn
 Tema oscuro	(Extra) Cambio de tema claro/oscurο
 Seguridad	Spring Security con roles y login

BASE DE DATOS (relacional)

Mínimo 5-6 tablas:

- **Usuario** : ID, nombre, email, contraseña, rol
- **Habito** : ID, nombre, descripción, frecuencia, tipo, fecha_inicio, id_usuario
- **RegistroDiario** : ID, id_habito, fecha, completado (fecha incluida )
- **TipoHabito** : ID, nombre
- **Meta** : ID, nombre, descripción, id_usuario
- **PuntosUsuario** : ID, puntos_totales, racha_actual, id_usuario

 Cumple el requisito de:

- 4+ tablas
- Fechas
- Clave primaria en todas

TECNOLOGÍAS SUGERIDAS

Backend:

- Java 17+
- Spring Boot
 - Spring MVC (Web)
 - Spring Data JPA (con Hibernate)
 - Spring Security
 - MySQL o PostgreSQL

Frontend:

- Thymeleaf + Bootstrap (más fácil si haces app web integrada)
- O React (si lo haces separado como SPA)

Extras:

- Lombok (para reducir código repetitivo)
- iText (para PDF)
- Chart.js o ApexCharts (para estadísticas)
- Git + GitHub para control de versiones

¿CÓMO CUMPLE LOS REQUISITOS?

Requisito	Cumplido	¿Cómo?
BBDD relacional	✓	MySQL con 5+ tablas
Al menos una app (Escritorio/Web/Móvil)	✓	App Web con Spring Boot
App funcional	✓	CRUD, estadísticas, login, interfaz usable
Identificadores únicos	✓	Clave primaria en todas las tablas
Fechas	✓	En <code>RegistroDiario</code> y <code>Habito</code>
Registro/Login	✓	Usuarios con roles y contraseña
Usuario Admin	✓	Rol <code>ADMIN</code> con más privilegios
Logotipo original	✓	Puedes crear uno sencillo y ponerlo en la navbar
Sección "Sobre mí"	✓	En el menú, con link a CV o LinkedIn
Documentación	✓	Vas a entregar documentación técnica, de usuario e instalación
Presentación y demo	✓	Tendrás una presentación visual + defensa del proyecto



Consejo Final

Hazlo modular, claro y bien documentado. Con buena presentación, este proyecto puede quedar de sobresaliente. No hace falta que inventes Facebook, pero sí que lo que hagas funcione, tenga sentido, y lo expliques con seguridad.

✖ Funcionalidades finales (versión API REST completa, realista para tu nivel)

Módulo	Funcionalidad REST que harás
👤 Usuarios	<code>POST /api/registro</code> – crear usuario <code>POST /api/login</code> – autenticarse <code>GET /api/usuarios/{id}</code> – datos de usuario
📅 Hábitos	<code>GET /api/habitos</code> – listar hábitos <code>POST /api/habitos</code> – crear hábito <code>PUT /api/habitos/{id}</code> – actualizar <code>DELETE /api/habitos/{id}</code> – borrar
📅 Seguimiento	<code>POST /api/seguimiento</code> – marcar hábito como hecho <code>GET /api/seguimiento/{fecha}</code> – obtener hábitos de ese día
📊 Estadísticas	<code>GET /api/estadisticas/{usuarioId}</code> – datos generales <code>GET /api/estadisticas/racha</code> – rachas <code>GET /api/estadisticas/progreso-metas</code>
🎯 Metas	<code>GET/POST/PUT/DELETE /api/metad</code>
👤 Sobre mí	Página HTML conectada o estática con tu CV/linkedin

✖ MÓDULOS Y FUNCIONALIDADES DETALLADAS

👤 Gestión de Usuarios

Objetivo: Permitir que los usuarios se registren, inicien sesión y tengan diferentes permisos (admin / usuario normal).

Funcionalidades:

- `POST /api/registro` – Registrar nuevo usuario.
- `POST /api/login` – Iniciar sesión (recibirás token JWT).
- `GET /api/usuarios/me` – Obtener perfil del usuario autenticado.
- `PUT /api/usuarios/{id}` – Editar perfil.
- `DELETE /api/usuarios/{id}` – Borrar cuenta.
- Roles: Usuario, Administrador (el admin puede ver todos los usuarios, por ejemplo).
- Validaciones: Email único, contraseñas seguras.
- (Opcional) Recuperación de contraseña por email.

Tecnologías:

- Spring Security + JWT
- BCrypt para cifrado de contraseñas

CRUD de Hábitos

Objetivo: Permitir crear, ver, editar y eliminar hábitos personales.

Funcionalidades:

- `GET /api/habitos` – Listar hábitos del usuario.
- `POST /api/habitos` – Crear nuevo hábito.
- `PUT /api/habitos/{id}` – Editar hábito.
- `DELETE /api/habitos/{id}` – Eliminar hábito.
- Atributos: título, descripción, frecuencia (diaria/semanal), color, prioridad, fecha de inicio.

Seguimiento de Hábitos

Objetivo: El usuario puede marcar qué hábitos ha cumplido cada día.

Funcionalidades:

- `POST /api/seguimiento` – Registrar que un hábito fue completado hoy.
- `GET /api/seguimiento/{fecha}` – Ver hábitos marcados ese día.
- `GET /api/seguimiento/actual` – Ver hábitos pendientes de hoy.
- `DELETE /api/seguimiento/{id}` – Desmarcar un hábito.

Ideas adicionales:

- Bloquear edición de días pasados (excepto admin).
- Enviar recordatorios (notificación o correo, opcional).



Metas y Objetivos

Objetivo: Permitir crear metas personales, con vinculación a hábitos que ayudan a lograrlas.

Funcionalidades:

- `GET /api/metast` – Listar metas.
- `POST /api/metast` – Crear una meta.
- `PUT /api/metast/{id}` – Editar meta.
- `DELETE /api/metast/{id}` – Eliminar meta.
- Vincular hábitos a una meta (`habito.meta_id`).
- Visualización del progreso hacia la meta según hábitos cumplidos.

Estadísticas y Gráficas

Objetivo: Motivar al usuario mostrando su progreso y rachas.

Funcionalidades:

- `GET /api/estadisticas/resumen` – Total hábitos completados, porcentaje cumplimiento.
- `GET /api/estadisticas/racha` – Racha actual y máxima.
- `GET /api/estadisticas/por-habito` – Datos por hábito específico.
- `GET /api/estadisticas/progreso-metas` – Progreso hacia cada meta.
- `GET /api/estadisticas/por-dia` – Hábitos cumplidos por fecha.

Visualización:

- Se consume desde el frontend (gráficas con Chart.js)
- Representación de:
 - Barras por hábito
 - Circular de metas cumplidas
 - Calendario de hábitos completados (heatmap estilo GitHub)

Exportación de Datos

Objetivo: Que el usuario pueda descargar un informe PDF.

Funcionalidades:

- `GET /api/exportar/pdf` – Descargar informe general.
- Contenido: resumen de hábitos, metas, progreso, estadísticas, fechas clave.
- Generado con iText o JasperReports.

Área de Administración (Admin)

Objetivo: Gestión superior del sistema por parte de un rol administrador.

Funcionalidades:

- Ver todos los usuarios (`GET /api/admin/usuarios`)
- Ver estadísticas globales
- Eliminar usuarios inactivos
- Modificar cualquier dato del sistema (si fuera necesario)



Área Sobre Mí

Objetivo: Cumplir el requisito obligatorio del proyecto de mostrar tu perfil profesional.

Contenido:

- Página accesible desde `/sobre-mi` (o desde API con `GET /api/sobremi`)
- Tu nombre, formación, link a CV PDF o LinkedIn
- Breve presentación

Frontend / Interfaz

Objetivo: Interfaz clara, accesible, y usable por cualquier tipo de usuario.

- Vistas modernas, responsive (Bootstrap)
- Menú de navegación con:
 - Inicio
 - Mis Hábitos
 - Seguimiento
 - Metas
 - Estadísticas
 - Mi perfil
 - Sobre mí
 - Cerrar sesión
- Formulario validado para cada operación

