Projektplan – Checklisten-Verwaltung

****Projektidee:****

Eine Web-App, mit der Nutzer Checklisten anlegen, Items abhaken, Vorlagen speichern und Listen (optional) teilen können.

****Technologie-Stack:****

- Frontend: HTML, CSS, JavaScript, React
- Backend: Java, Spring Boot, Spring Data JPA, Spring Web, Spring Security
- Datenbank: PostgreSQL oder H2 (Testumgebung)

****Kernfunktionen (MVP):****

- User-Registrierung und Login (Auth mit JWT oder Session)
- Checklisten anlegen, umbenennen, löschen
- Items hinzufügen, bearbeiten, löschen und abhaken
- Filter- und Sortierfunktionen (z. B. nach Fälligkeit oder Status)
- Vorlagen erstellen und daraus neue Listen generieren

****Datenmodell (vereinfacht):****

• users(id, email, password_hash, created_at)
• checklists(id, owner_id, title, created_at, updated_at)
• items(id, checklist_id, title, description, is_done, due_at, priority, position)
• templates(id, owner_id, title, payload_json, created_at)

****REST-Endpunkte (Beispiele):****

- POST /api/auth/register – Nutzer registrieren
- POST /api/auth/login – Login und Token erhalten
- GET /api/checklists – Eigene Checklisten abrufen
- POST /api/checklists – Neue Checkliste anlegen
- PATCH /api/items/{id} – Item bearbeiten (z. B. abhaken)
- GET /api/templates – Vorlagen abrufen

****Frontend-Struktur:****

- /login, /register – Auth-Seiten
- /app – Dashboard mit Listenübersicht
- /app/checklists/:id – Detailansicht einer Checkliste
- /app/templates – Vorlagenverwaltung

****Sprint-Vorschlag (4 Phasen):****

- Sprint 1: Projekt-Setup, Entities, erste CRUD-Endpunkte, React-Skeleton
- Sprint 2: Checklisten & Items vollständig umsetzen
- Sprint 3: Templates, Drag-and-Drop, Optimistic UI
- Sprint 4: Tests, Feinschliff, Fehlerbehandlung und UI-Politur

****Optionale Erweiterungen:****

- Sharing-Funktion (Checklisten mit anderen Nutzern teilen)
- Tags und Filter für Items
- Benachrichtigungen/Reminder
- Offline-Modus oder Export (JSON/CSV)
- Statistiken (z. B. Erledigungsquote)