**GymBro – SDD (Software Design Document)**

**Dokumentace návrhu softwaru (SDD)**

**2.1 Architektura systému**

GymBro je **jednostránková webová aplikace** (SPA) v technologiích React + TypeScript, přičemž backend je outsourcingovaný do služby Supabase. Architektura se skládá z těchto vrstev:

* **Klientská část (Frontend):** Běží v uživatelově prohlížeči. React komponenty zajišťují uživatelské rozhraní a reagují na akce (zobrazení seznamů, odesílání formulářů). Route (React Router) přepíná mezi stránkami (Dashboard, Goals, Workouts, Calendar, Settings). Komponenty komunikují se Supabase API (REST/Realtime) přímo pomocí Supabase JS klienta.
* **Serverová část (Backend-as-a-Service):** Supabase poskytuje API pro autentizaci a přístup k databázi PostgreSQL. Tabulky jsou definovány takto:
  + auth.users (vestavěná tabulka Supabase Auth) obsahuje základní účet (id, email, hashed\_password).
  + profiles ukládá doplňkové údaje (id uživatele, jméno, roli, email, URL avatara). Role je omezena na 'athlete', 'coach' nebo 'admin'. Cizí klíče mezi profiles.id a auth.users.id zajišťují integritu dat.
  + workouts (tabulka tréninků): každý záznam má user\_id, datum, délku, energie, námahu, popis atd..
  + workout\_exercises: podrobné cviky pro jednotlivé tréninky (každý cvik spojován cizím klíčem na tabulku workouts). Ukládá název cviku, sérií, opakování, zátěže, poznámky.
  + goals: ukládá cíle sportovců s poli title, description, target\_value, current\_value, unit, start\_date, end\_date, popř. název cviku (exercise\_name). Vztahuje se cizím klíčem na auth.users přes user\_id. Obsahuje i kontrolu integrity (např. hodnoty námahy 1–10 v workoutu).
  + coach\_athletes: spojovací tabulka mezi trenéry a sportovci. Každý řádek definuje vztah coach\_id – athlete\_id s atributem status (možné hodnoty: 'pending', 'active', 'rejected'). Jedinečnost dvojice coach\_id, athlete\_id je vynucena unikátním indexem a cizí klíče na profiles zajišťují vazby.
* **Autentizace a session:** Supabase Auth spravuje přihlášení. AuthContext (React) poslouchá na změny session a ukládá je do kontextu. Po přihlášení se do Supabase Auth automaticky doplňuje záznam v profiles s informacemi z registrace (jméno, role).

Celkově je architektura klient-server, kde serverová logika je minimalizována na operace DB (via SDK) a aplikační logiku zajišťuje hlavně frontend.

**2.2 Návrh komponent (Frontend)**

Frontend je rozčleněn do modulárních komponent, odpovídajících logickým částem UI:

* **Layout a navigace:** Komponenta MainLayout obaluje většinu obrazovek. Obsahuje TopNavbar (zobrazující jméno přihlášeného uživatele, tlačítko odhlášení, přepínač tématu) a Sidebar (navigační panel s ikonami a odkazy). Sidebar vybírá položky menu podle user.role (athlete vs coach).
* **Dashboard:**
  + AthleteDashboard – zobrazí statistiky sportovce a možnost výběru trenéra. V komponentě se pomocí Supabase načítají data o všech trenérech (role='coach') a do menu je vloženo pole pro přiřazení trenéra (vloží záznam do coach\_athletes při odeslání). Vykresluje se souhrn tréninků a aktivních cílů.
  + CoachDashboard – zobrazuje počet atletů s daným trenérem (status active), počet žádostí (pending) a základní metriky. Komponenta AthleteCard zobrazuje jméno/email sportovce; EnergyExpenditureChart kreslí graf akumulačního výkonu. Tlačítka pro schválení/odmítnutí spouští aktualizaci tabulky coach\_athletes.
* **Správa cílů:**
  + Goals (stránka) – obsahuje záložky *Active*, *Completed*, *Expired*. Pro každou skupinu cílů používá podkomponentu, která vykreslí mřížku karet (Card) cíle. U každého cíle je zobrazen progress bar vypočtený pomocí funkce calculateGoalProgress z objektů goals a workouts (tj. počítá aktuální vs. cílovou hodnotu).
  + GoalsList – seznam aktivních cílů (karta s tlačítky a lištou). V *Progress* stránce je podobná komponenta zobrazující aktivní cíle v přehledu.
  + GoalForm – formulář pro přidání/úpravu cíle. Validuje vstupy (název, jednotka, cílová/aktuální hodnota). Po odeslání vytvoří nebo aktualizuje řádek v DB tabulce goals.
  + GoalDetail – stránka detailu jednoho cíle. Načítá cíl z DB podle id z URL. Zobrazuje detailní informace (titulek, popis, progress bar, časový rámec) a umožňuje update progress nebo mazání cíle.
* **Správa tréninků:**
  + Workouts (stránka) – načítá všechna data tréninků z DB pro uživatele. Uspořádává je do skupin podle měsíce. Používá komponenty WorkoutHeader (zobrazuje název stránky a tlačítko *Add*), QuickWorkoutTemplates (tlačítkové karty pro přednastavené tréninky typu *Push/Pull/Legs*) a WorkoutTabs či WorkoutList pro seznam záznamů.
  + WorkoutForm – formulář pro přidání/úpravu tréninku. Obsahuje podkomponenty WorkoutBasicInfo (základní informace: název, datum, délka, úroveň námahy, atd.) a ExerciseForm (dynamický seznam cviků s poli name/sets/reps/weight). Při odeslání provede validaci a zavolá službu saveWorkoutToDatabase, která provede INSERT do tabulek workouts a workout\_exercises.
* **Další komponenty:**
  + **CalendarDisplay / DayDetail:** Vizualizuje kalendář a detaily tréninku vybraného dne.
  + **Progress** a další stránky (např. Progress – souhrn aktivit, Settings – profil) – komponenty sloužící specifickým úkolům.
  + **Tematická podpora:** ThemeToggle a ThemeContext pro přepínání světlého/tmavého režimu.
  + **Auth/ProtectedRoute:** AuthContext spravuje přihlášení (signIn, signUp, signOut). Komponenta ProtectedRoute obalí stránku, aby byla dostupná pouze po přihlášení.

**2.3 Datový model**

Databázový model je definován v Supabase přes SQL DDL. Hlavní entitami a jejich vztahy jsou:

* **User / Profile:** auth.users(id, email, password, …) a profiles(id, name, avatar\_url, role, coach\_id, email…). Vztah 1:1 – každý záznam profiles.id odkazuje na auth.users.id. Pole role určuje roli (CHECK omezení: pouze 'athlete', 'coach', 'admin').
* **Workout & WorkoutExercise:** Každý trénink (workouts.id) je spojen s jedním uživatelem (user\_id). Podrobné cviky jsou v tabulce workout\_exercises s FK na workouts(id). Například sloupce duration, energy\_expended, effort\_level definují základní metriku tréninku.
* **Goal:** Tabulka goals obsahuje cíle uživatelů. Má FK user\_id na autora cíle (sportovce). Klíčová pole jsou target\_value, current\_value, jednotka unit a časový interval (start\_date, end\_date). Volitelné pole exercise\_name může specifikovat, ke kterému cviku se cíl vztahuje (např. bench-press).
* **CoachAthleteRelation:** Spojovací tabulka coach\_athletes s poli coach\_id a athlete\_id (oba FK na profiles.id) a se stavem status (pending/active/rejected). Unikátní kombinace (každý sportovec může mít max. jednoho trenéra) je zajištěna constraintem (unikátní klíč).
* **Vazby a omezení:** Cizí klíče a kontrolní omezení (CHECK) zajišťují integritu (např. úroveň námahy mezi 1–10). Přidání a mazání jsou kaskádní tam, kde má logiku (např. při smazání uživatele se smažou také jeho tréninky a cíle).

**2.4 Toky dat a procesní toky**

**Přihlášení a registrace:** Uživatel vyplní email, heslo (a jméno/roli při registraci). Voláním Supabase Auth signUp/signIn se vytvoří účet. Při úspěšném přihlášení AuthContext zjistí session, získá user.id, a načte profil z profiles. Pokud profil ještě neexistuje, vytvoří ho s daty z session (jméno, role). Poté aplikace přesměruje na Dashboard.

**Přidání tréninku (athlete flow):** Uživatel vstoupí na stránku *Add Workout*. Zadá údaje do WorkoutForm (titulek, parametry, cviky) a odešle. Komponenta vytvoří objekt WorkoutWithExercises a zavolá službu saveWorkoutToDatabase, která provede INSERT do workouts a pro každý cvik do workout\_exercises (s vazbou na workout\_id). Po úspěšném uložení naviguje zpět na seznam tréninků.

**Aktualizace cíle (athlete flow):** Uživatel na stránce detailu cíle zadá novou hodnotu do pole „Update Progress“ a klikne **Update**. Spustí se funkce handleUpdateProgress, která provede UPDATE goals SET current\_value = nový\_stav, updated\_at = now(). Poté se klientská část přenačte a ukáže novou hodnotu a progress bar.

**Přiřazení trenéra:** Sportovec vybere z roletky trenéra (z tabulky profiles kde role='coach') a potvrdí. Klient provede INSERT do coach\_athletes s vlastnostmi coach\_id, athlete\_id, status='pending'. Trenér se následně po přihlášení podívá do své dashboardové sekce *Pending Approvals* a může použít tlačítko **Approve** nebo **Reject**. To vyvolá dotaz UPDATE coach\_athletes SET status = 'active' (nebo 'rejected') WHERE coach\_id = ... AND athlete\_id.

**Zobrazení dashboardu:** Po načtení hlavní stránky (Index routa) komponenta zjistí roli uživatele (isCoach). Poté na pozadí zpracuje datové požadavky (asynchronně načte výkony a cíle, či seznam atletů). Až je data připravena, komponenta vykreslí příslušný Dashboard (Athlete nebo Coach). Při chybných dotazech se zobrazí chybová hláška v konzoli a prázdná data.

**2.5 Správa stavu a autentizace**

Aplikace používá **React Context** pro globální stav uživatele (AuthContext) a pro téma (ThemeContext). Pro ostatní data (seznamy tréninků, cílů, atd.) se využívá lokální stav pomocí hooků (useState v jednotlivých komponentách) a provádějí se fetch dotazy do Supabase v useEffect. Detailní popis:

* **AuthContext:** Obsahuje informace o přihlášeném uživateli (user, session), funkce signIn, signUp, signOut a příznak loading. Context poskytuje wrapper komponenta AuthProvider. Po úspěšném přihlášení/registraci je user nastaven a dětem je zprostředkován. Kontext také reaguje na změny stavu přihlášení (listener Supabase) a v případě potřeby načte resp. vytvoří profil uživatele.
* **ProtectedRoute:** Hodnotí, zda je user v kontextu nenulový. Pokud není přihlášen, přesměruje na přihlašovací stránku. Díky tomu jsou soukromé stránky a komponenty chráněny.
* **State komponent:** Např. stránka Workouts má workouts ve svém state a načte je z DB. Podobně Goals page má stavy goals a workouts, Dashboard trenéra má athletes apod. Data se získávají jednorázově při montáži komponenty nebo na změně závislostí (např. user.id). Po změně některé entity se data znovu načtou (např. volání onRefresh v CoachDashboard po změně statusu atleta).
* **Formuláře:** Používají knihovnu React Hook Form s integrací Zod pro validaci. Lokální stav formuláře se spravuje skrze hook useForm, avšak po odeslání formuláře jsou data odeslána přímo do backendu (Supabase) a případně vracíme zpět návratový kód pro zpracování (toastová hlášení).

**2.6 Flow diagramy (procesní toky)**

*Poznámka: Diagramy nelze graficky znázornit zde, ale níže popíšeme hlavní scénáře v krocích.*

1. **Uživatel – registrace a login:**
   * Uživatel navštíví /auth a vyplní registrační formulář (email, jméno, role, heslo). Aplikace zavolá supabase.auth.signUp. Po úspěšném potvrzení emailu a přihlášení se do session uloží nový uživatel. AuthContext zaregistruje změnu stavu a zavolá vložení profilu do profiles (pokud neexistuje) s daty z registrace. Nakonec se uživatel přesměruje na / (Index), kde se vykreslí příslušný Dashboard.
2. **Sportovec – přidání tréninku:**
   * Na stránce *Add Workout* vyplní formulář. Po odeslání se zavolá back-endová funkce saveWorkoutToDatabase. Ta nejdříve vloží řádek do workouts (INSERT) a poté pro každý cvik vloží odpovídající řádky do workout\_exercises. Po ukončení serverové operace front-end naviguje zpět na /workouts a zobrazí nově vložený trénink v seznamu.
3. **Sportovec – správa cílů:**
   * Na stránce *Goals* vytváří nový cíl přes GoalForm. Po uložení se v DB vytvoří záznam v goals. Frontend přesměruje zpět na přehled cílů. Při otevření detailu cíle lze pomocí handleUpdateProgress (zadání nového current\_value) provést SQL UPDATE goals SET current\_value = ....
4. **Sportovec – žádost trenérovi:**
   * V Dashboardu atleta vybere trenéra ze seznamu. Kliknutím na „Assign Coach“ komponenta provede INSERT do coach\_athletes s atributem status='pending'. V uživatelském profilu se trénérovi aktualizuje stav (po přihlášení uvidí novou žádost).
5. **Trenér – schvalování:**
   * Po přihlášení trenéra na jeho Dashboard se automaticky načtou vztahy coach\_athletes pro jeho coach\_id (vyberou se athlete\_id s jejich statusy). V sekci „Pending Approvals“ trenér vidí seznam čekajících atletů. Kliknutím na „Approve“ komponenta provede UPDATE coach\_athletes SET status='active'. Podobně „Reject“ nastaví status='rejected'. Po změně se znovu načtou data a atlet se přesune z „Pending“ do seznamu „Your Athletes“.
6. **Trenér – sledování výkonu:**
   * Trenér může kliknout na konkrétního atleta (např. jeho jméno v seznamu) a zobrazit detaily výkonu: například přehled jeho tréninků a cílů. Častěji však nahlíží na statistiky z Dashboardu (počet tréninků za týden, průměrné intenzity), vycházející z dat uložených Supabase. Grafy agregují data z tabulky workouts (energetický výdej dle data).

Každý z výše uvedených toků je řešen kódem komponent a služeb, jak je uvedeno výše. Komunikace s databází využívá asynchronní volání Supabase klienta; chybové stavy jsou zachyceny a uživateli zobrazovány pomocí toasů/notifikací.

**Shrnutí:** Díky rozdělení do modulů (komponent, služeb, kontextů) je GymBro snadno rozšiřitelný. Při zavedení nových funkcionalit (např. skupinových tréninků, sdílení progresu) lze postupovat v rámci stávající architektury a rozšířit existující entity či přidat nové komponenty.