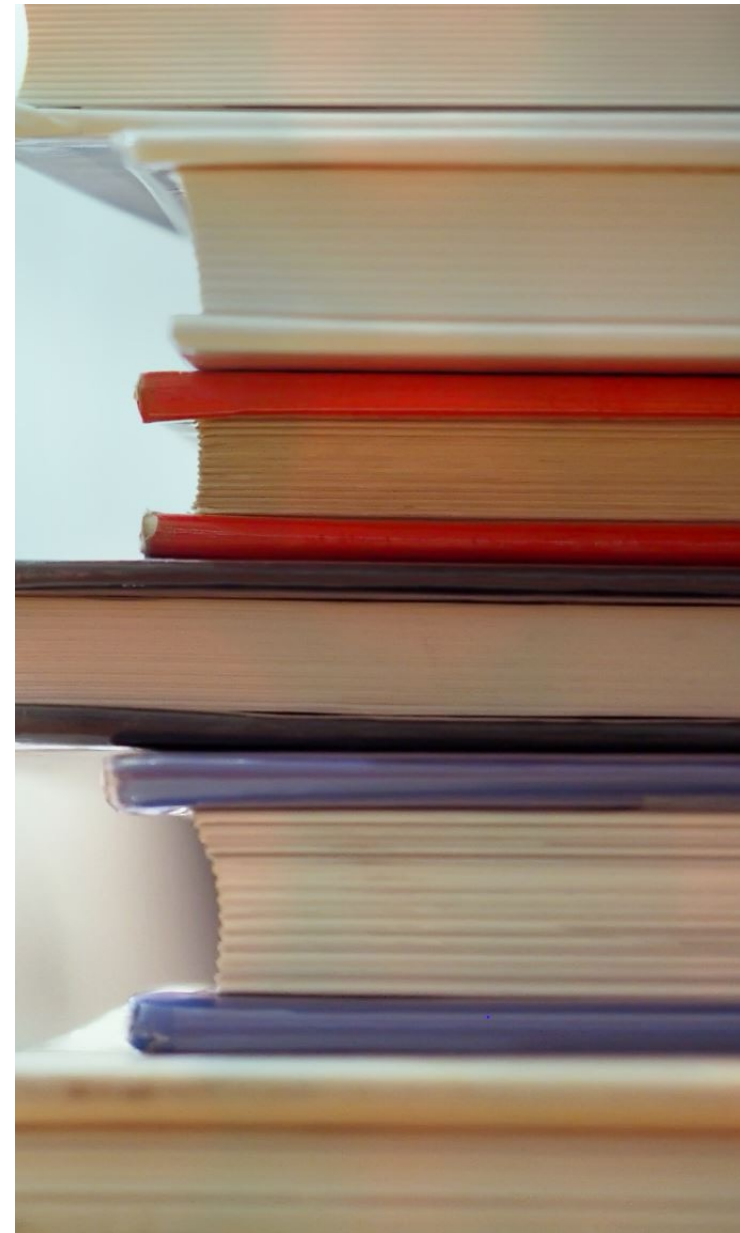


COLLABORATIVE STUDY PLATFORM

SEMESTERIAL WORK

**Z tejto semestrálnej úlohy môžete získať
maximálne 26 bodov**



Úvod

- V tejto semestrálnej práci budeme vyvíjať platformu na kolaboratívne štúdium, ktorá umožní študentom vytvárať študijné skupiny, spravovať úlohy, zdieľať materiály a sledovať aktivitu členov tímu.
- Cieľom projektu je rozvíjať schopnosť návrhu komplexnej aplikácie, ktorá kombinuje:
 - databázovú logiku,
 - webové služby (REST API),
 - real-time komunikáciu,
 - a moderné používateľské rozhranie.
- Projekt podporuje AI asistenciu pri vývoji, no každý študent alebo tím musí preukázať pochopenie návrhu a implementácie.



Tímy a spolupráca

Študenti môžu pracovať **individuálne alebo v tíme dvoch osôb.**

Každý tím bude mať pridelený **spoločný GitHub repozitár**, do ktorého budú commitovať svoj kód. Zazdieľam odkaz na repozitár a na získanie prístupu budete musieť prijať pozvánku.

Očakáva sa **rovnomerné rozdelenie práce** – úlohy musia byť v histórii repozitára rozdelené medzi členov.

AI nástroje (ChatGPT, Copilot, Gemini a pod.) možno používať, ale je nutné uviesť, **ktoré časti boli generované AI** a ako boli prispôbené.

ARCHITECTURE OVERVIEW



MODEL KLIENT-SERVER



ÚLOHY KLIENTA A SERVERA

Architektúra klient-server

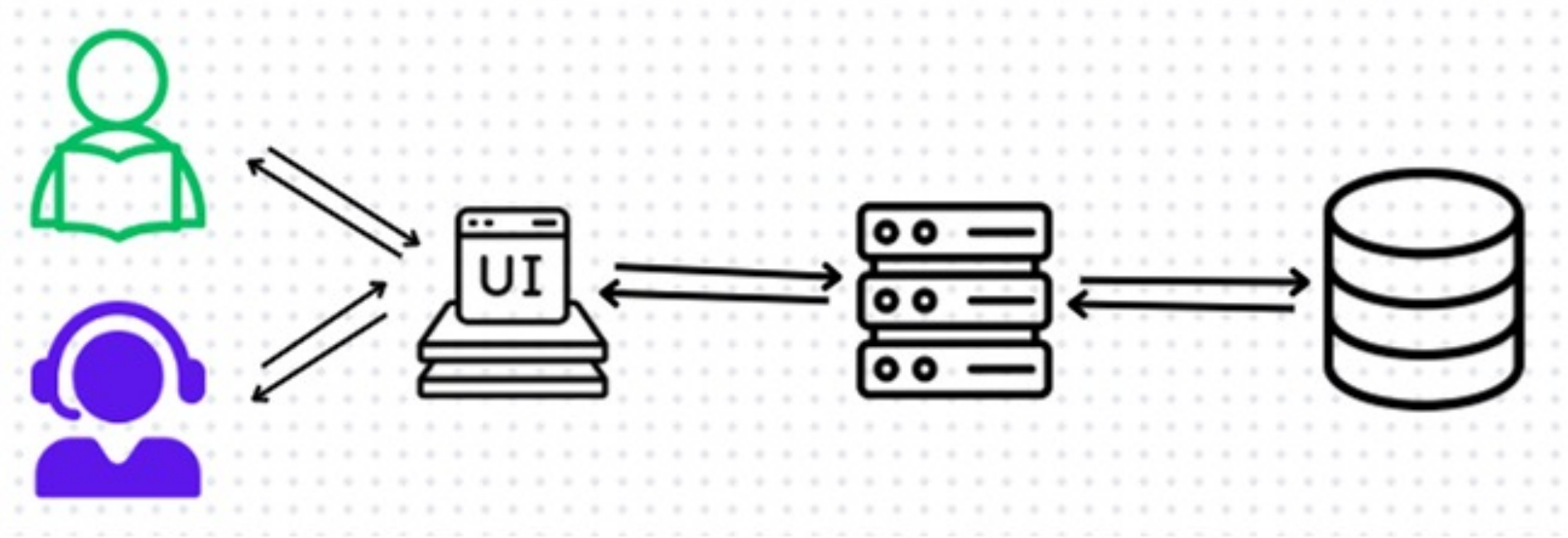
- Architektúra sa riadi modelom klient-server, kde je klient zodpovedný za používateľské rozhranie a komunikuje so serverom pri vykonávaní operácií.
- Klientská aplikácia vytvorená pomocou JavaFX poskytuje používateľom interaktívne rozhranie na prihlásenie, registráciu a správu svojich knižných zbierok.

Klient (JavaFX)

- Handles user interactions, such as logging in, registering users, adding books, and viewing available books.
- Komunikuje so serverom na vyžiadanie a odoslanie údajov pomocou rozhraní RESTful API.

Server (Spring Boot)

- Spracováva požiadavky klientov, spravuje obchodnú logiku a interaguje s databázou.
- Zverejňuje koncové body prostredníctvom rozhraní RESTful API, aby umožnil klientovi vykonávať operácie CRUD (CREATE, READ, UPDATE a DELETE) s knihou a používateľskými údajmi.



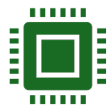
TECHNOLOGY STACK



Client side: JavaFX

Používa sa na vytvorenie používateľského rozhrania aplikácie.

Poskytuje moderné komponenty používateľského rozhrania a podporuje interakcie používateľov.



Server side: Spring Boot

Používa sa na strane servera, spracováva prichádzajúce požiadavky od klienta



Database: SQLite

relačný databázový systém pre ukladanie používateľov, skupín, úloh a materiálov.



Komunikácia: RESTful API a WebSocket

Umožňuje efektívnu komunikáciu klient-server pomocou štandardných metód HTTP.

Podporuje operácie, ako je načítanie (GET), vytváranie (POST), aktualizácia (PUT) a odstraňovanie (DELETE) zdrojov.

WebSocket na obojsmernú komunikáciu v reálnom čase, používa sa napríklad pre notifikácie o nových úlohách alebo zmenách v skupine.



Maven / Gradle

build systémy pre správu závislostí, kompiláciu a spúšťanie aplikácie. Umožňujú jednoducho spravovať knižnice a moduly projektu.

KLÚČOVÉ FUNKCIE APLIKÁCIE

Používateľský prístup a autentifikácia

- **Registrácia a prihlásenie používateľov** – vytváranie účtov, bezpečné ukladanie hesiel (bcrypt).
- **Správa profilov** – úprava mena, e-mailu a základných informácií o používateľovi.

Skupiny a úlohy

- **Vytváranie a správa študijných skupín** – zakladanie, úprava a mazanie skupín, pridávanie členov.
- **Riadenie úloh v skupine** – pridávanie, upravovanie a sledovanie stavu úloh (OPEN / IN PROGRESS / DONE).
- **Termíny a upozornenia** – definovanie deadlinov, pripomienky na blížiace sa úlohy.

Zdieľanie a komunikácia

- **Zdieľanie študijných materiálov** – pridávanie odkazov alebo súborov pre členov skupiny.
- **Real-time notifikácie (WebSocket)** – okamžité informovanie o nových úlohách, príspevkoch alebo členoch.
- **Diskusia / správy (voliteľné rozšírenie)** – možnosť komunikovať v rámci skupiny.

Analytika a bezpečnosť

- **Vizualizácia štatistík** – prehľady aktivity, grafy splnených úloh, podiel zapojenia členov.
- **Bezpečné spracovanie dát** – kontrola vstupov, hashovanie hesiel, ochrana pred SQL injection.

Rozšírené funkcie (bonus)

- **OAuth2 prihlásenie cez Google** – jednoduché prihlasovanie pomocou externého účtu.
- **Docker nasadenie** – spustenie celej aplikácie v kontajneri.
- **Automatizované testy a CI/CD pipeline** – voliteľný prídavok pre pokročilých študentov.

NÁVRH DATABÁZY



USERS TABLE



RESOURCES
TABLE



MEMBERSHIPS
TABLE



TASKS TABLE



GROUPS
TABLE



ACTIVITY_LOG

USERS TABLE

Columns

- user_id, name, email, password_hash

Description

- Uchováva základné informácie o používateľoch a ich prihlasovacie údaje. Heslá musia byť hashované (napr. bcrypt).

RESOURCES TABLE

Columns

- resource_id, group_id, uploaded_by, title, type, path_or_url, uploaded_at

Description

- Ukladá zdieľané študijné materiály – buď ako odkazy (link), alebo ako nahrané súbory.

MEMBERSHIPS TABLE

Columns

- membership_id, user_id, group_id, role, joined_at

Description

- Zaznamenáva členstvo používateľov v skupinách a ich rolu (napr. bežný člen alebo administrátor skupiny).

TASKS TABLE

Columns

- task_id, group_id, created_by, title, description, status, deadline, created_at

Description

- Obsahuje úlohy priradené k skupinám. Pomáha organizovať prácu, sledovať pokrok a termíny.

GROUPS TABLE

Columns

- group_id, name, description, created_by, created_at

Description

- Reprezentuje študijné skupiny. Každá skupina má zakladateľa a môže obsahovať viacerých členov, úlohy a materiály.

ACTIVITY_LOG

Columns

- log_id, user_id, action, timestamp, details

Description

- Slúži na sledovanie aktivít používateľov – napr. prihlásenie, vytvorenie úlohy, odovzdanie súboru. Možno použiť na analytiku aktivity členov.

Hodnotenie a kritériá

Bodovanie (primárne body, na ktoré sa treba zamerať):

- Architektúra a návrh DB – 6 bodov
- REST API + WebSocket – 6 bodov
- Používateľské rozhranie – 5 bodov
- Analytika a štatistiky – 5 bodov
- Dokumentácia, README, bezpečnosť – 4 body

Total – 26 bodov

HODNOTENIE

Projekt odošlete na GitHub a po jeho schválení ho budete môcť obhájiť prezentáciou a ak nebude schválený, nebudete ho obhajovať a **nedostanete body**. To znamená, že nahraná verzia by mala byť skontrolovateľná a mala by fungovať.

Po schválení ho predložíte, budú vám položené otázky. Na základe vášho výkonu získate body.

Ak získate **menej ako 50 percent** (menej ako 13 bodov z 26), nebudete pripustení na **SKÚŠKU** a vaše body získané počas hodiny sa nezapočítajú.

README.md

Stručný popis projektu a cieľov aplikácie: Stručne vysvetlite, čo aplikácia robí, komu je určená a aký problém rieši.

Architektúra systému (diagram + vysvetlenie vrstiev): Graficky znázornite štruktúru aplikácie (frontend, backend, databáza) a stručne popíšte úlohu jednotlivých častí.

Databázový model (ER diagram): Priložte diagram s tabuľkami, väzbami a typmi dát. Uvedte, ako databáza podporuje funkcie aplikácie.

Dokumentácia REST API a WebSocket endpointov: Uvedte zoznam všetkých endpointov (GET, POST, PUT, DELETE) s popisom ich funkcie, parametrov a príkladov odpovede.

Ukážky používateľského Rozhrania: Vložte snímky obrazovky (JavaFX okná) a popíšte základný tok práce s aplikáciou.

Popis výziev a riešení (napr. validácia, autentifikácia): Uvedte technické problémy, s ktorými ste sa stretli, a opíšte spôsob, akým ste ich vyriešili.

Zhodnotenie práce s AI – čo pomohlo, čo muselo byť manuálne doladené: Uvedte, ktoré časti kódu boli generované pomocou AI (napr. ChatGPT, Copilot), ako ste ich prispôbili a čo ste sa z toho naučili.