Slovenská technická univerzita
Fakulta informatiky a informačných technológii
Ilkovičova 2, 841 04 Bratislava

Mobilná aplikácia pre vzdelávacie online kurzy

Mobilné technológie a aplikácie

Autori: Oliver Leontiev, Marek Oravec
Garant predmetu: prof. Ing. Ivan Kotuliak, PhD.
Cvičiaci: Ing. Marek Galinski, PhD.
Akademický rok: 2021/2022
Semester: Letný

Obsah

Github repozitáre

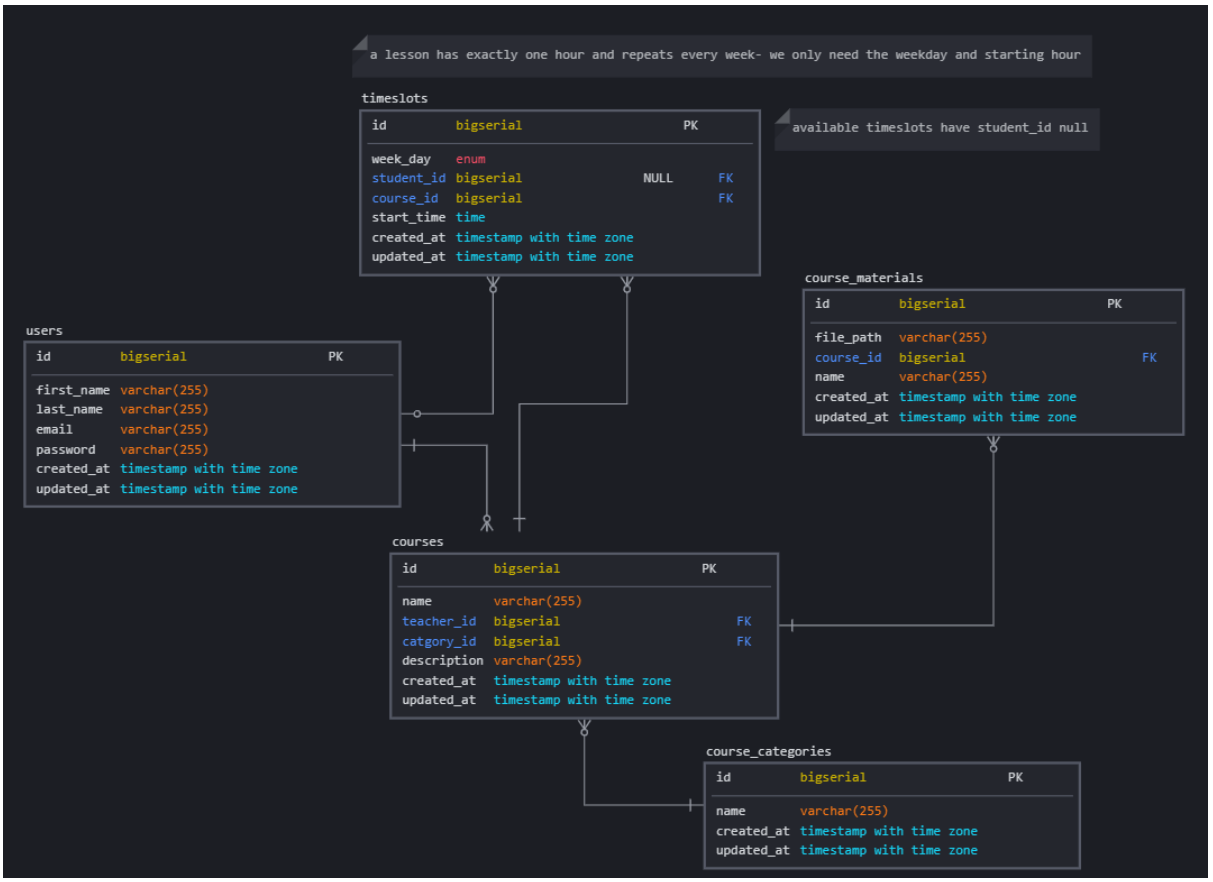
Backend: https://github.com/revilO602/mtaa-backend
Frontend: https://github.com/revilO602/mtaa-frontend

Opis

Táto mobilná aplikácia umožňuje používateľom vytvárať kurzy s hodinovými “one on one” lekciami, ktoré sa opakujú každý týždeň v rovnakom čase. Následne sa iný používatelia môžu prihlásiť na voľný časový blok (timeslot). Používateľ si potom môže zobraziť svoj rozvrh. Súčasťou aplikácie sú “one on one” lekcie formou videohovorov medzi učiteľom a študentom.

Návrh

Dátový model



REST API volania

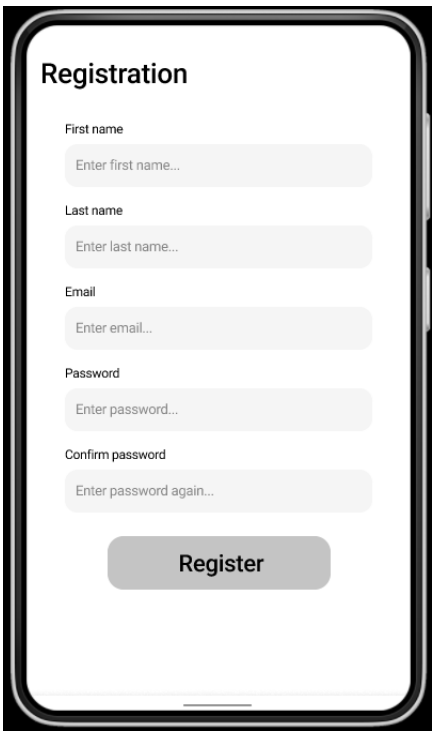
Swagger 2.0 dokumentácia: https://app.swaggerhub.com/apis-docs/revilO6024/MTAA/1.1.0

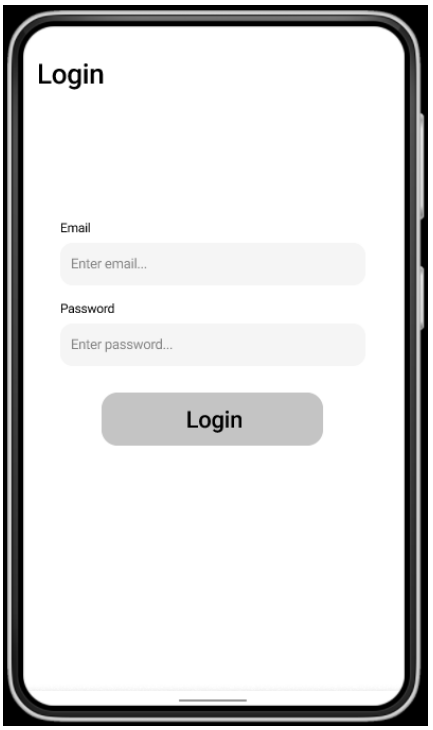
POST **/users/register** - Registrácia
GET **/users/login** - Login - prihlasovacie údaje sa posielajú v headeri requestu
POST **/courses** - Vytvorenie kurzu
GET **/courses** - Vrátenie zoznamu voľných kurzov s vyhľadávaním a filtrovaním
/courses/?categoryId=1
/courses/?q=math
GET **/courses/<:id>** - Detail kurzu
PUT **/courses/<:id>** - Zmeniť informácie kurzu
DEL **/courses/<:id>** - Vymazanie kurzu
DEL **/courses/<:id>/timeslots/<:id>** - Vymazanie timeslotu z kurzu
POST **/courses/<:id>/timeslots** - Pridanie timeslotu ku kurzu
POST **/courses/join** - Pridanie sa ku kurzu (súčasťou request body je pole vybraných timeslot ID - timeslot rovno definuje aj o ktorý kurz sa jedná)
POST **/courses/leave** - Odhlásenie sa z timeslotov kurzu (súčasťou request body je pole vybraných timeslot ID - timeslot rovno definuje aj o ktorý kurz sa jedná)
GET **/courses/categories** - Vrátenie listu kategórií kurzov
GET **/courses/<:id>/students** - Vrátenie listu študentov kurzu
GET **/courses/<:id>/materials** - Zoznam materiálov pre kurz
GET **/media/<:file_path>** - Stiahnutie materiálu
POST **/courses/<:id>/materials** - Upload materiálu pre kurz
GET **/timetable** - List timeslotov, v ktorých má user kurzy
GET **/student/courses** - List kurzov kde user je študent
GET **/student/courses/<:id>** - Detail kurzu kde user je študent
GET **/teacher/courses** - List kurzov kde user je učiteľ
GET **/teacher/courses/<:id>** - Detail kurzu kde user je učiteľ

Návrh obrazoviek

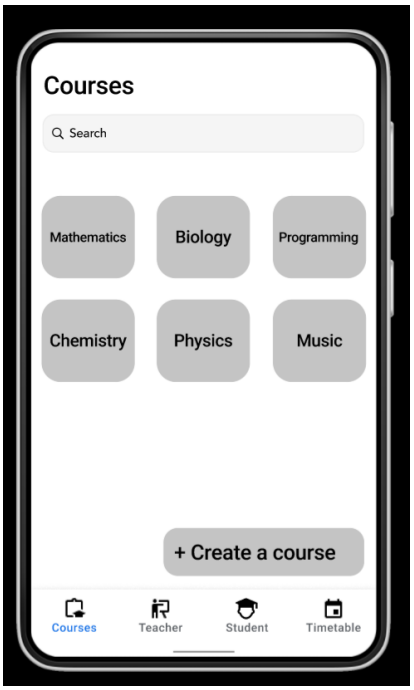
Figma klikateľný prototyp:
https://www.figma.com/proto/UmunG9GPzqjJTq1Qpahqff/Wireframe?node-id=39%3A2&scaling=scale-down&page-id=0%3A1&starting-point-node-id=39%3A2

Registrácia a login

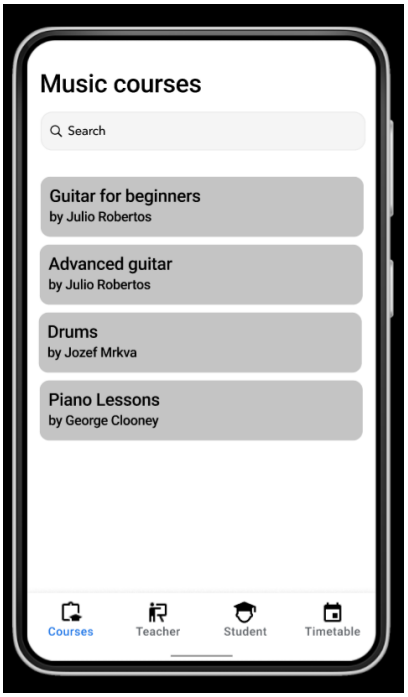




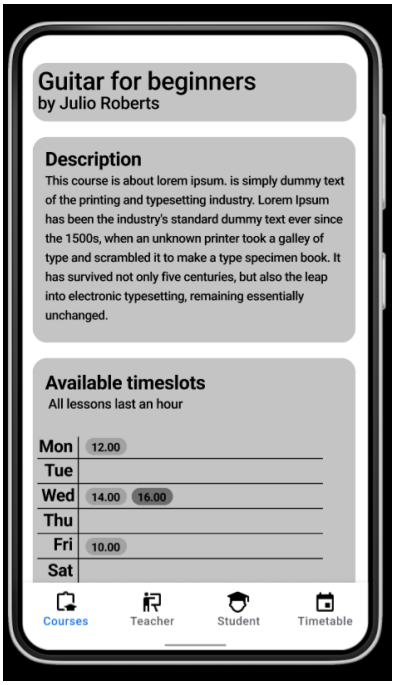
Prehľad kurzov a kategórií

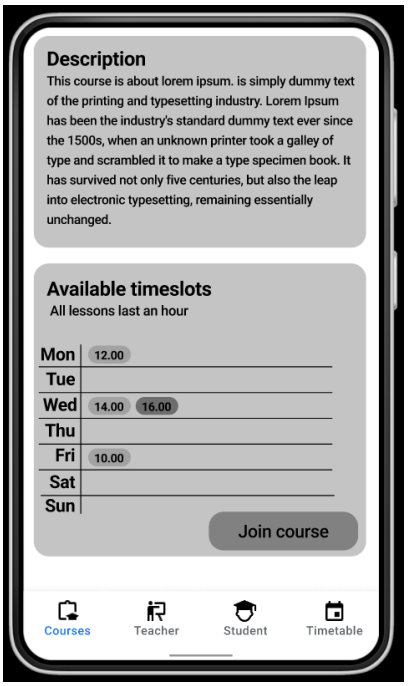


Kurzy vo zvolenej kategórii

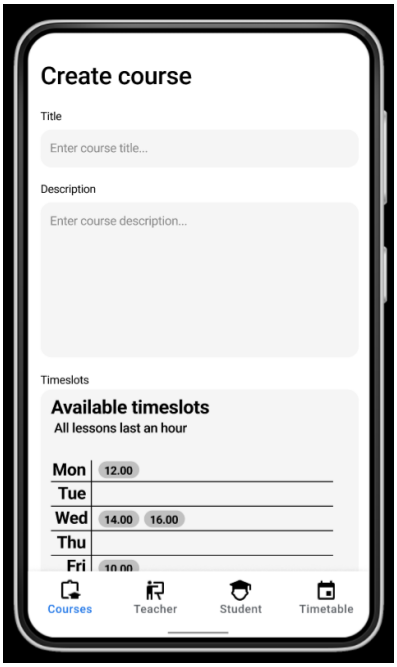


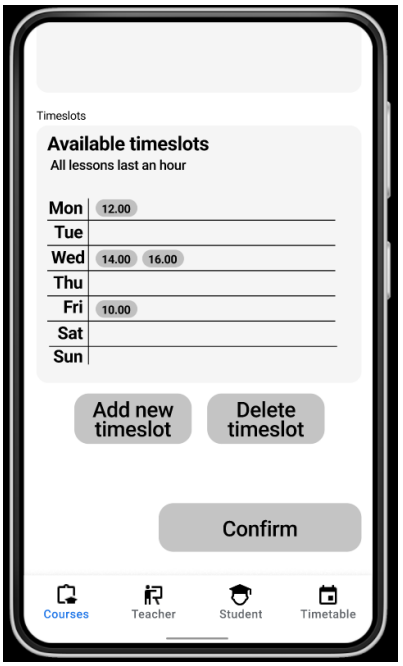
Popis konkrétneho kurzu





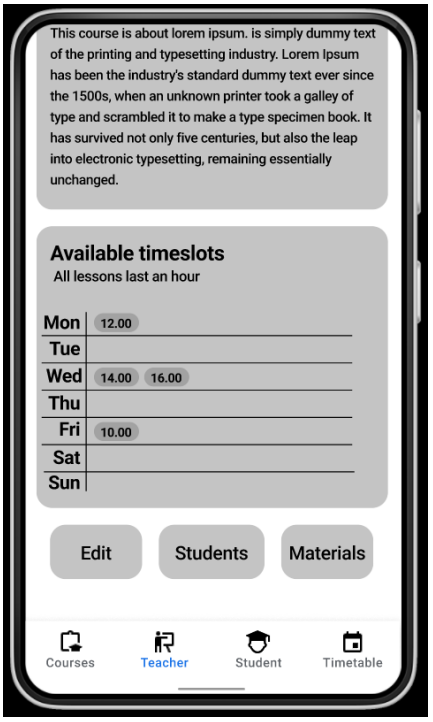
Vytvorenie kurzu



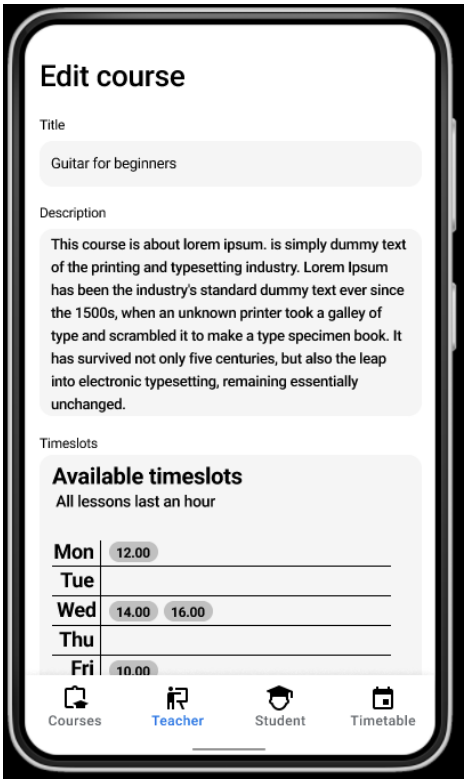


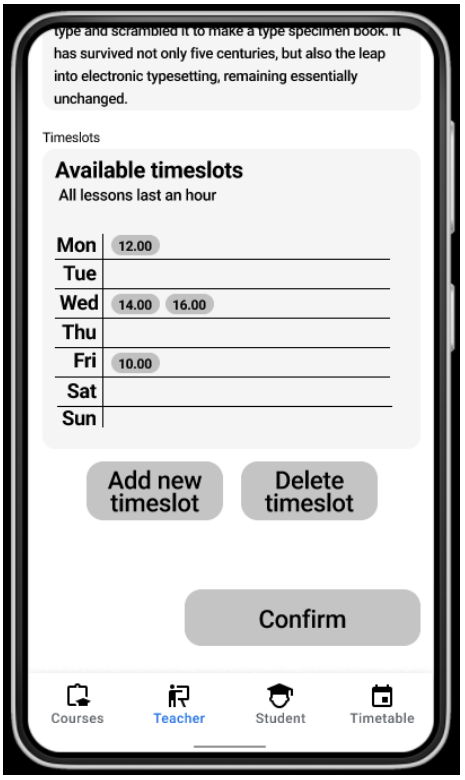
Dostupné aktívne kurzy z pohľadu učiteľa

Zvolený kurz z pohľadu učiteľa



Úprava kurzu učiteľom

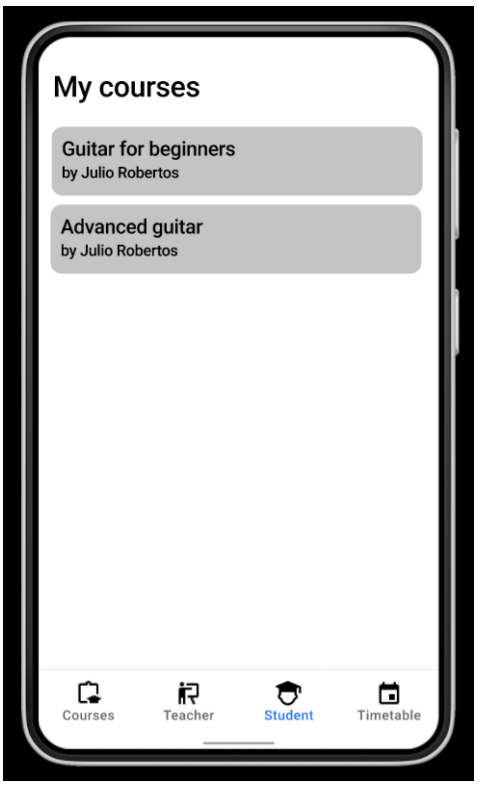




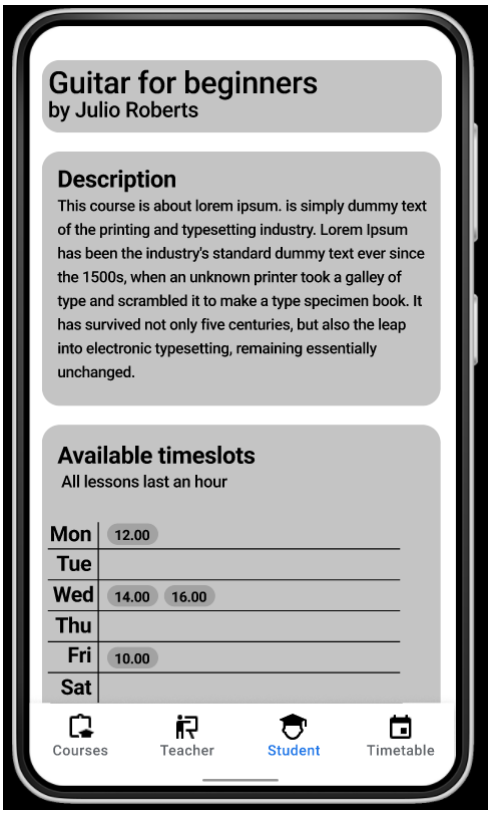
Zoznam študentov kurzu pre učiteľa

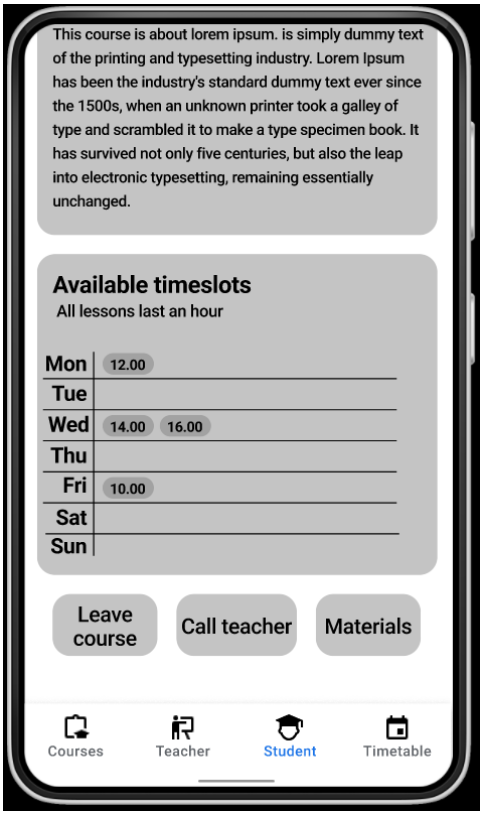
Dostupné materiály pre daný kurz z pohľadu učiteľa

Dostupné aktívne kurzy z pohľadu študenta



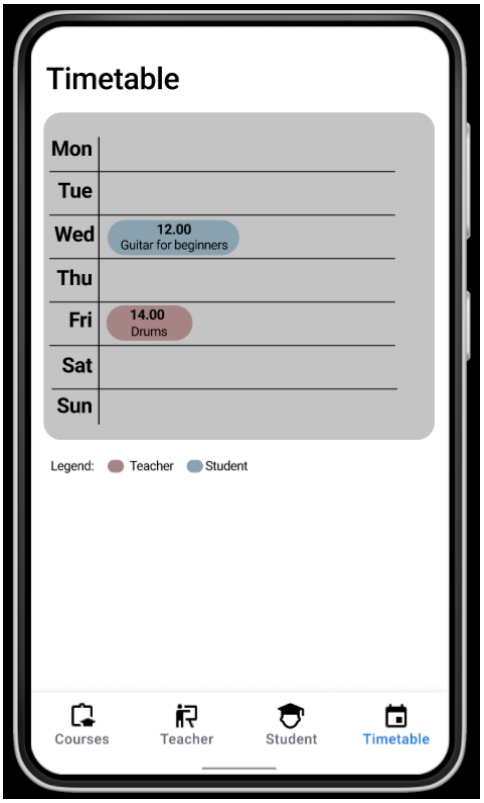
Detail zvoleného kurzu z pohľadu študenta





Dostupné materiály na stiahnutie z pohľadu študenta

Aktuálny rozvrh pre používateľa, kde sú zobrazené aktuálne prebiehajúce kurzy



Akceptačné testy

Backend testy

Test 1: Zobrazenie zoznamu kurzov pre kategóriu mathematics. (positive test)	
Vstupné podmienky	Súčasťou každého volania musí byť autorizačný token.
Výstupné podmienky	Vrátenie HTTP response s dátami kurzov pre matematiku.
Postup	1. Aplikácia zavolá HTTP GET /courses
	2. Volanie vráti 200 OK, pričom vráti list všetkých kurzov.
	3. Zavolá sa HTTP GET /courses/?category=mathematics.
	4. Volanie vráti 200 OK a zobrazia sa kurzy v kategórii "mathematics".
Výsledok: PASS / FAIL	

Test 2: Stiahnutie materiálu. (positive test)	
Vstupné podmienky	Súčasťou každého volania musí byť autorizačný token.
Výstupné podmienky	Úspešný pokus o stiahnutie materiálov použitím danej URL.
Postup	1. Aplikácia zavolá HTTP GET /courses/<:id>/materials. Pričom používateľ je študentom daného kurzu
	2. Volanie vráti 200 OK a zoznam dát dostupných materiálov.
	3. Aplikácia zavolá /<:material_id>, kde material_id je id existujúceho materiálu.
	4. Vráti sa 200 OK a materiál sa stiahne.
Výsledok: PASS / FAIL	

Test 3: Zobrazenie detailu kurzu, ktoré daný používateľ vlastní. (positive test)	
Vstupné podmienky	Súčasťou každého volania musí byť autorizačný token.
Výstupné podmienky	HTTP response s detailami aktuálneho kurzu, ktorého je používateľ vlastníkom.
Postup	1. Aplikácia zavolá HTTP GET /teacher/courses.
	2. Volanie vráti kód 200 OK, pričom sa zobrazia kurzy patriace tomuto používateľovi.
	3. Aplikácia zavolá HTTP GET /teacher/courses/<:id>, kde poskytne id platného kurzu.
	4. Volanie vráti 200 OK, zobrazí sa detail kurzu so zadaným id.
Výsledok: PASS / FAIL	

Test 4: Zobrazenie študentov kurzu, kde prihlásený používateľ je študentom. (negative test)	
Vstupné podmienky	Súčasťou každého volania musí byť autorizačný token.
Výstupné podmienky	Vrátenie chybného kódu 403 Forbidden vrátane chybovej hlášky.
Postup	1. Používateľ je prihlásený ako študent nejakého kurzu. (nie je jeho majiteľom)
	2. Aplikácia zavolá HTTP GET /courses/<:id>/students, kde id je platné id existujúceho kurzu.
	3. Volanie vráti 403 Forbidden.
	4. Zobrazí sa chybová hláška, že študent nie je učiteľom daného kurzu.
Výsledok: PASS / FAIL	

Test 5: Zmazanie existujúceho kurzu. (negative test)	
Vstupné podmienky	Súčasťou každého volania musí byť autorizačný token.
Výstupné podmienky	Vrátenie chybného kódu 403 Forbidden vrátane chybovej hlášky.
Postup	1. Aplikácia zavolá HTTP GET /courses/<:id> , pričom je zadané id existujúceho kurzu.
	2. Volanie vráti 200 OK a v body dáta kurzu.
	3. Aplikácia následne zavolá HTTP DEL /courses/<:id>, kde je zadané id existujúceho kurzu.
	4. Volanie vráti 403 Forbidden, pretože používateľ nemá právo na zmazanie kurzu.
Výsledok: PASS / FAIL	

Frontend testy

Test 1: Používateľ sa prihlási na kurz. (positive test)	
Vstupné podmienky	Používateľ s prístupom na internet, ktorý je prihlásený do aplikácie.
Výstupné podmienky	Aktuálne prihlásený používateľ bude prihlásený na kurz v konkrétnom čase.
Postup	1. Používateľ sa prihlási.
	2. Z ponuky dostupných kurzov sa vyberie požadovaný kurz.
	3. Zvolený kurz bude zobrazovať meno školiteľa a náplň kurzu.
	3. Používateľ vyberie čas začiatku lekcie.
	4. Používateľ klikne na možnosť "Prihlásiť sa na kurz".
	5. Používateľ uvidí kurz v študentskej sekcii.
Výsledok: PASS / FAIL	

Test 2: Používateľ sa prihlási ako učiteľ/lektor a pokúsi sa vytvoriť kurz. (positive test)	
Vstupné podmienky	Používateľ s prístupom na internet sa prihlási do aplikácie.
Výstupné podmienky	Aktuálne prihlásený používateľ bude autorom práve vytvoreného kurzu.
Postup	1. Používateľ sa prihlási.
	2. Používateľ prejde do sekcie "Kurzy".
	3. Používateľ vyberie možnosť "Vytvoriť kurz".
	4. Zobrazí sa obrazovka formuláru pre vytvorenie kurzu.
	5. Vytvorený kurz sa priradí do zvolenej kategórie a je možné sa naň prihlásiť.
Výsledok: PASS / FAIL	

Test 3: Možnosť nahratia materiálov pre kurz. (positive test)	
Vstupné podmienky	Používateľ s prístupom na internet prihlásený do aplikácie.
Výstupné podmienky	V konkrétnom kurze je viditeľný nahratý študijný materiál pre študentov.
Postup	1. Prihlásený používateľ prejde do učiteľskej sekcie.
	2. Používateľ prejde do aktuálne prebiehajúcich kurzov.
	3. Vyberie kurz, pre ktorý chce nahrať materiály a nahrá ich.
	4. Používateľ prejde do zložky "Materiály" a pozrie, či sú materiály nahraté.
Výsledok: PASS / FAIL	

Test 4: Pokus o vymazanie materiálu pre kurz. (negative test)	
Vstupné podmienky	Používateľ s prístupom na internet, ktorý nie je autorom žiadneho kurzu.
Výstupné podmienky	Neschopnosť vymazať nahratý študijný materiál z akéhokoľvek kurzu.
Postup	1. Používateľ sa prihlási, nie je učiteľom žiadnych kurzov.
	2. Používateľ prejde do ľubovoľného kurzu kde je študentom.
	3. Používateľ sa pokúsi pridať materiál.
	4. Nie je možné pridať materiál pre kurz, pokiaľ používateľ nie je jeho autorom.
Výsledok: PASS / FAIL	

Test 5: Vybratie hodiny na lekciu s lektorom, ktorý v danom čase už má inú lekciu. (negative test)	
Vstupné podmienky	Používateľ s prístupom na internet, ktorý má prístupný ľubovoľný kurz.
Výstupné podmienky	Nemožnosť si vybrať čas pre lekciu, kde už konkrétny lektor lekciu má.
Postup	1. Používateľ prejde do sekcie "Prebiehajúce kurzy".
	2. Používateľ sa pokúsi vybrať hodinu pre lekciu, kedy vie že jeho lektor je nedostupný.
	3. Používateľ zistí, že si nemôže zvoliť konkrétnu požadovanú hodinu.
Výsledok: PASS / FAIL	

Implementácia

Backend

REST API pre mobilnú aplikáciu bolo vytvorené v **Javascripte** s použitím webového servera **Node.js** a rámca **express.js**. Databázu sme zvolili **postgresql** a použili sme **ORM** sequelize na komunikáciu serveru s databázou a validáciu dát poslaných na server. Databáza aj

webový server beží iba na lokálnom stroji a dá sa na server pripojiť z iného zariadenia. Na autentifikáciu používateľov sme použili knižnicu **express-basic-auth**.

/database
Vytvorenie prepojenia s databázou pre sequelize ORM a vloženie predvolených dát.

/helpers
Predstavuje pomocné funkcie používané na viacerých miestach. Najmä tu je riešené testovanie prekrývajúcich sa timeslotov - takéto timesloty nepovoľujeme používateľom tvoriť.

/media
Používateľmi uploadnuté súbory.

/middleware
Vlastný middleware

/models
ORM reprezentácie databázových tabuliek.

/routes
Spracovanie požiadaviek na jednotlivé URL adresy.

/server.js
Smerovanie požiadaviek do správnych súborov v priečinku */routes.*

Pre spustenie backendu je potrebné najprv spustiť príkaz *npm install* a potom *npm start*. Backend beží na lokalnej adrese http://localhost:3000 . Pri spustení na Wi-Fi pripojení sa vypíše do konzoly adresa dostupná externému zariadeniu na tej istej Wi-Fi.

Frontend

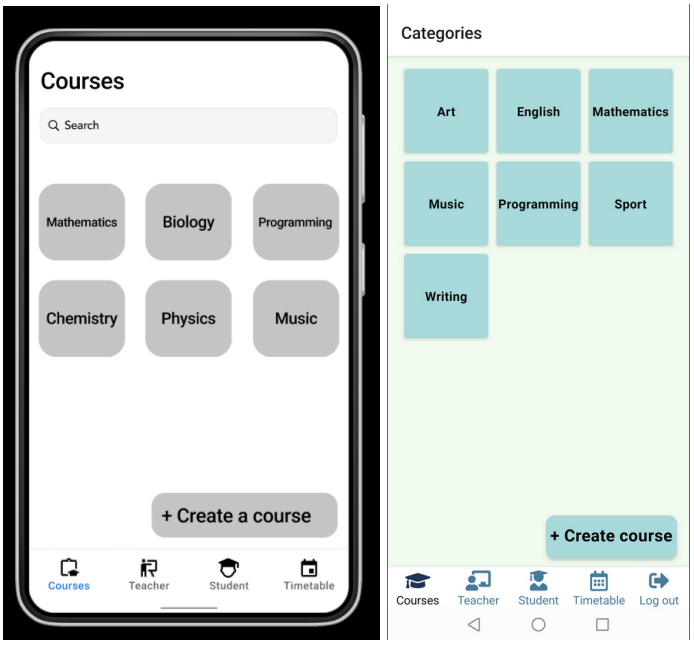
V tejto časti dokumentácie sa zameriame na nami vytvorený frontend a jeho realizáciu. Veľká časť nami navrhnutých obrazoviek ostala v takej podobe, v akej bol aj návrh vo Figme. Spomenieme najmä tie, kde k nejakým drobným zmenám došlo a následne aj tieto zmeny bližšie popíšeme. Rovnako tak sa aj vyjadríme k navrhnutým akceptačným testom, či sme ich splnili a ak nie, k akým zmenám došlo.

Frontend bol implementovaný v **Javascripte** s použitým softvérového rámca **React Native** a platformy **Expo** pre tento rámec. Pre nainštalovanie potrebných balíčkov je treba spustiť príkazy *npm install*, *yarn install*, *expo install*. Samotná aplikácia je zostavená pomocou Expo Application Services a jej build sa da stiahnuť na adrese https://expo.dev/accounts/revilo602/projects/1on1-courses/builds/f8893e9d-4f42-440f-81c2-275215ae2204
Po stiahnutí treba v konzole spustiť development client pomocou príkazu *expo start --dev-client* . Aplikácia funguje správne iba ak je spustený aj backend a je na rovnakej sieti ako frontend. Adresu, na ktorej je backend dostupný je treba pridať do súboru *Server.tsx* v priečinku *constants.*

BottomTabNavigator

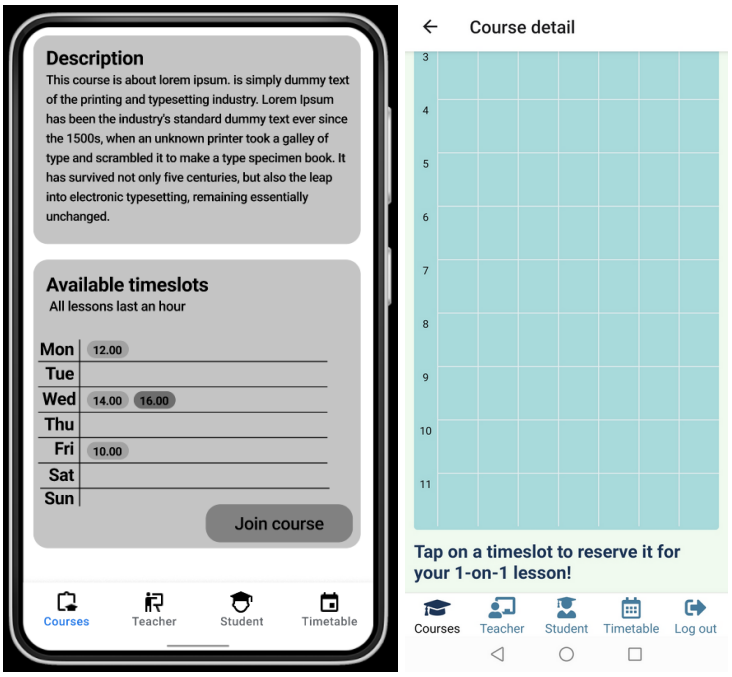
Táto časť aplikácie nie je obrazovkou, avšak používateľ ju neustále vidí pri prechádzaní po jednotlivých častiach aplikácie. V pôvodnom návrhu sa nenachádzal ***Log out*** button, my sme ho tam však pridali, pretože v dnešnej dobe každá aplikácia umožňuje odhlásiť sa zo svojho účtu. Zároveň sme si týmto spôsobom aj overili funkcionalitu, či sa pri zmene používateľa zmení aj obsah, ktorý vidí.

Courses



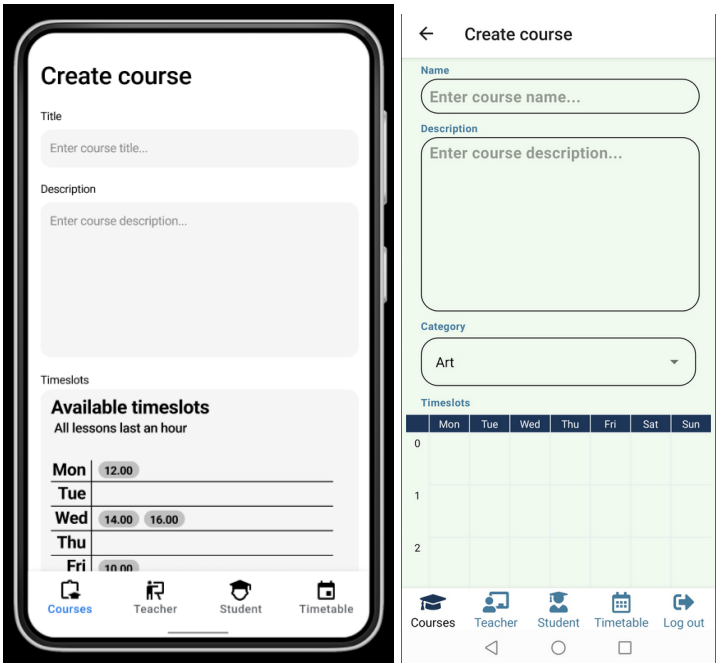
Na tejto obrazovke sa zachovalo všetko okrem farebnej schémy (kombinácia zelených a modrých odtieňov pôsobí živšie) a Search baru. Dôvodom odstránenia Search baru bol fakt, že tento element tam pôsobil zbytočne, keďže máme limitovaný počet kategórii, ktoré je všetky vidieť na obrazovke, a ak by aj používateľ potreboval v aplikácii niečo vyhľadávať, určite by to nerobil na tejto obrazovke.

Course Detail



Na tejto obrazovke je zmenená orientácia rozvrhu, pretože sme použili existujúci TimeTableView komponent, ktorý sa nám ľahko implementoval a vedeli sme si ho prispôsobiť podľa našej potreby. Rovnako tak sa na obrazovke nenachádza tlačidlo Join course z pôvodného návrhu, pretože je rýchlejšie a intuitívnejšie si rovno vybrať konkrétnu hodinu iba kliknutím na dostupný timeslot v rozvrhu, ktorý následne priradí prihlásenému používateľovi zvolenú lekciu.

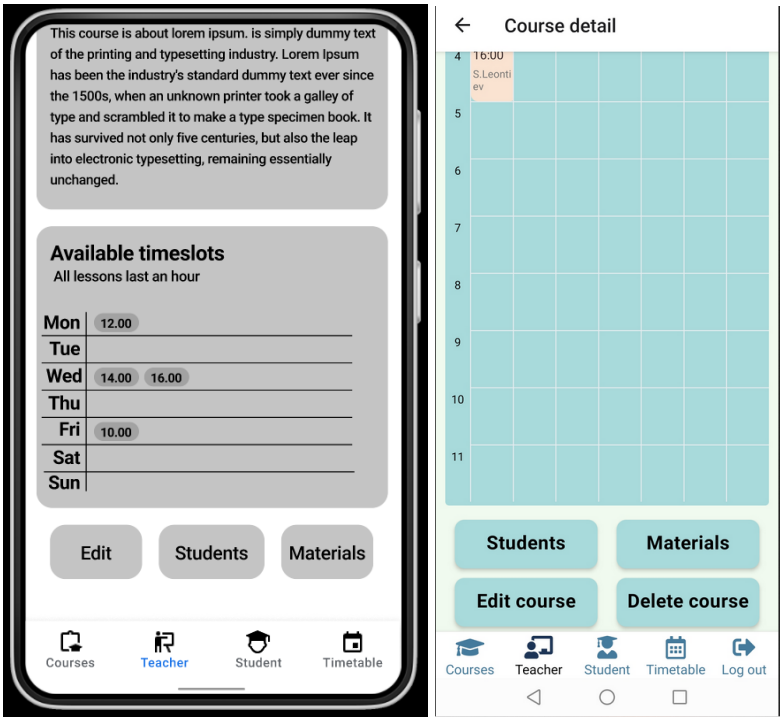
Create Course



Pri obrazovke ***Create course*** došlo len k malej zmene, pretože v našom pôvodnom návrhu sme nevolili kategóriu, do ktorej má kurz patriť. Vo finálnej podobe aplikácie si ešte

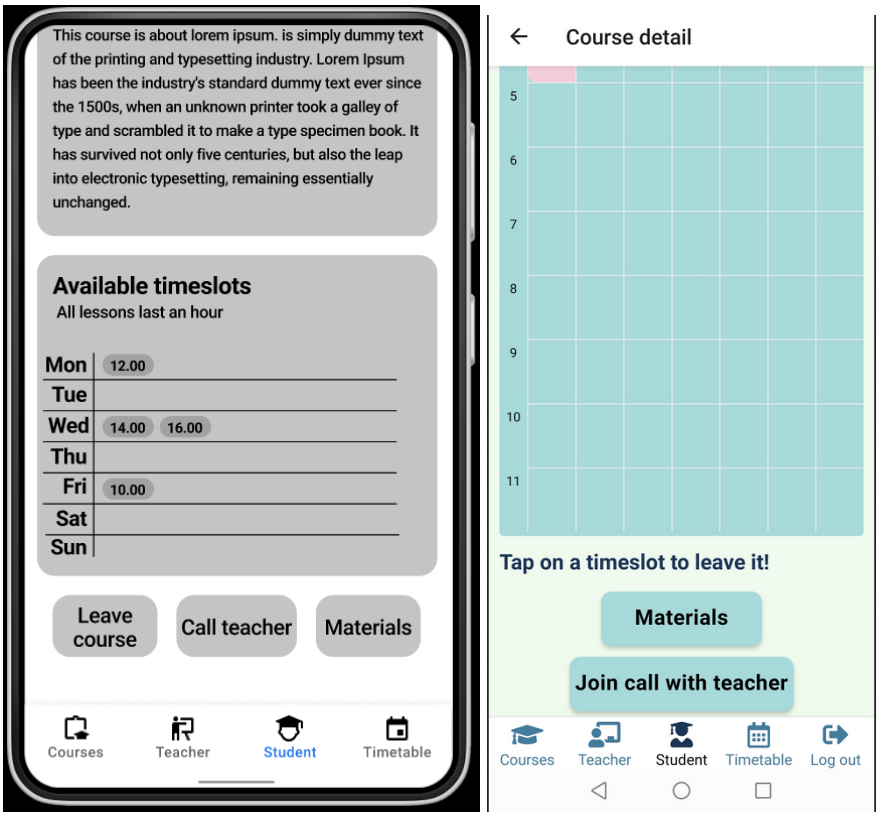
používateľ dodatočne volí kategóriu, v ktorej bude kurz viditeľný. ***Add timeslot*** a ***Delete timeslot*** fungujú na princípe dialógového okna a pomocou komponentu ***Picker*** si používateľ vyberie začiatok a koniec kurzu.

Course Detail (Teacher)



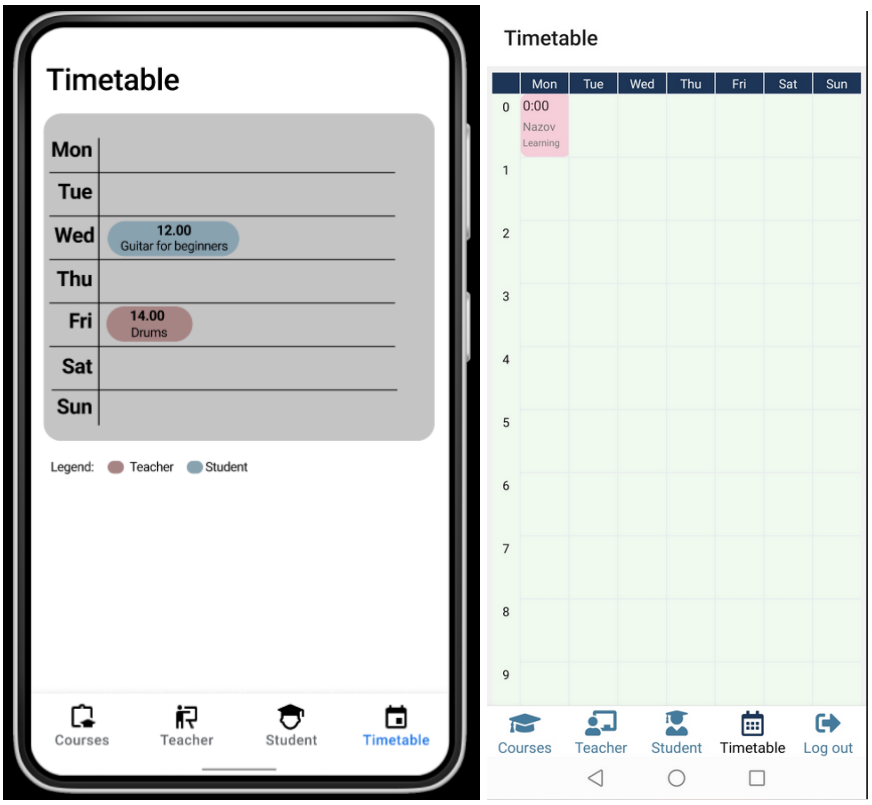
Táto obrazovka dostala len jednu zmenu, a to pridanie tlačidla ***Delete course***, kde ak user je učiteľ daného kurzu, môže tento kurz vymazať. Študentom, ktorí sa na tento kurz prihlásili, následne kurz aj zmizne z ponuky ich kurzov.

Course Detail (Student)



Na obrazovke Course detail u študenta sme sa rozhodli vynechať tlačidlo Leave course, pretože v iných častiach aplikácie (konkrétne pri rezervácii lekcie) sa študent prihlási pomocou kliknutia na hodinu v rozvrhu. Z tohto dôvodu, pre zachovanie “konvencie”, sme podobne riešili aj odhlasovanie z kurzu kliknutím na konkrétnu hodinu v rozvrhu (študent vidí len jemu priradené). Ďalšou zmenou je tlačidlo ***Call teacher***, ktoré pôsobí tak, že študent môže zavolať učiteľovi a “prizvať” ho do hovoru. V našej finálnej implementácii však vidíme miesto toho tlačidlo ***Join call with teacher***, kde sa študent pridá do hovoru, ktorý medzitým učiteľ pripraví.

Timetable



Táto obrazovka je pomerne jednoduchá, zobrazuje len rozvrh konkrétne prihláseného používateľa. Malou detailnou zmenou je len to, že nemáme dole legendu farebne rozlišujúcu, či daný používateľ je v danom čase študentom alebo učiteľom. Miesto toho sme pridali detailný popis konkrétneho timeslotu priamo do rozvrhu, pretože komponent ***TimeTableView*** poskytuje možnosť dodatočných informácii ako ***extra_descriptions***, ***location*** a podobne.

WebRTC

Volania s použitím technológie WebRTC sú umožnené medzi učiteľom kurzu a konkrétnym študentom. Učiteľ založí “miestnosť” s ID v tvare course<id kurzu>-user<id študenta> a študent sa vie na toto volanie pripojiť. Na frontende sme implementovali riešenie dostupné tu: https://github.com/DipanshKhandelwal/react-native-webrtc-firebase . Na backend sme použili službu **Cloud Firestore** poskytovanú v rámci **Google Firebase**. V tejto databáze dokumentov ukladáme pre každú “miestnosť” SDP údaje zakladateľa spojenia ako “offer” a SDP údaje druhej strany ako “answer”. Zároveň tu ukladáme zoznam ICE kandidátov pre obe strany (“callerCandidates” a “calleeCandidates”) pričom využívame Google STUN serveri. To nám umožňuje nadviazať peer-to-peer spojenie.

FE akceptačné testy

Test 1
Používateľ sa prihlási na kurz. (positive test)
Tento akceptačný test patrí medzi pozitívne testy, a je rovnako aplikovaný aj v našom projekte. Prihlásený používateľ sa prihlasuje na kurz prostredníctvom kliknutia na timeslot v

rozvrhu (čas, kedy bude realizovaná lekcia v danom kurze). Pri prihlasovaní zobrazuje popis kurzu aj meno školiteľa, tak ako bolo stanovené.

Test 2
Používateľ sa prihlási ako učiteľ/lektor a vytvorí kurz. (positive test)
Pri tomto teste nedošlo ku žiadnym zmenám. V sekcii ***Courses*** na ***BottomTabNavigator*** sa v pravom dolnom rohu nachádza tlačidlo + Create course a po kliknutí naň sa objaví formulár pre vytváranie kurzov. Pomocou komponentu Picker používateľ môže zvoliť čas, kedy bude lekcia realizovaná, zvolí si kategóriu, do ktorej má kurz patriť a pridá názov a popis. Vytvorený kurz je podľa testu aj následne viditeľný v príslušnej kategórii.

Test 3
Možnosť nahratia materiálu pre kurz. (positive test)
V tomto teste sa nič nezmenilo a používateľ ako učiteľ kurzu dokáže nahrať materiály pre kurz.

Test 4
Pokus o vymazanie materiálov pre kurz. (negative test)
Tento test v zásade prešiel, pretože UI aplikácie v študentskej sekcii ani neumožňuje nijakým spôsobom manipulovať s materiálmi. Jediné čo študent môže spraviť je daný materiál si stiahnuť, nie ho však vymazať, poprípade nejaký pridať.

Test 5
Vybratie hodiny na lekciu s lektorom, ktorý v danom čase už má inú lekciu. (negative test)
Pri tomto teste opäť UI aplikácie neposkytuje možnosť rezervovať si lekciu v čase, v ktorom už je dohodnutá schôdzka, pretože akonáhle si študent vyberie lekciu v danom kurze, táto lekcia následne vymizne z rozvrhu dostupných hodín a tým pádom si ju už nikto iný zvoliť nemôže.

Changelog oproti návrhu

Dátový model

- Všetky tabuľky v databáze majú nové stĺpce created_at a updated_at
    - Dôvod: Je to zaužívaná prax a ORM na backende to robí automaticky. Bolo by zložitejšie to odstrániť ako to nechať a napriek tomu, že to v rámci projektu nevyužijeme, tak v praxi by to tam vždy bolo.
- V tabuľke timeslots *start_hour* premenované na *start_time* a namiesto typu integer má typ time
    - Dôvod: Kurz môže začať kedykoľvek nie len na začiatku hodiny
- V tabuľke timeslots typ zmenený na enum. Predstavuje psql typ.
    - Dôvod: Dovolujeme iba validný názov dňa (“Monday”, “Tuesday”...)

REST API volania

- Namiesto filtrovania kategórii cez query parameter “category” s názvom kategórie (/?category=mathematics) sa filtruje cez query parameter “categoryId” s ID kategórie (/?categoryId=1).
    - Dôvod: ID umožňuje rýchlejšie vyhľadávanie v databáze bez potreby parsovania reťazcov
- /login a /register zmenené na /users/register /users/login
    - Dôvod: Jasnejšie rozdelenie URL ciest
- Pridané endpointy **POST /courses/<:id>/timeslots/** na pridanie timeslotu a **DEL /courses/<:id>/timeslots/<:id> na** vymazanie timeslotu z kurzu
    - Dôvod: Pôvodne zamýšľané ako súčasť PUT requestu na kurz, ale timesloty sú samostatné entity a mali by mať vlastné volania.
- Zmenený endpoint na prihlásenie na kurz na **POST /courses/join** kde sa do request body vloží pole ID timeslotov - pridaný endpoint **POST /courses/leave** s rovnakou logikou na opustenie timeslotov
    - Dôvod: Timesloty majú unikátne ID, čiže netreba v URL ID kurzu.

Zmeny po Milestone 2

- Odstránený endpoint /call na volanie medzi používateľmi.
    - Dôvod: WebRTC sme implementovali cez Firebase, takže nie je súčasťou nášho backendu.
- Endpoint na sťahovanie materiálov zmenený z GET /materials/<:material_id> na /media/<:file_path>
    - Dôvod: Nahraté materiály vnímame ako statické media súbory a stahujeme ich priamo z URL, na ktorej sa nachádzajú. Je to efektívnejšie ako v databáze hladať file_path na zakláde material_id.

Backend akceptačné testy

Zmeny po Milestone 2

- Akceptačný test 4 sa pôvodne zaoberal zabezpečením, že si študenti nemôžu navzájom volať. Keďže ale WebRTC nakoniec na backende neimplementujeme tak sme ho zmenili na zabezpečenie toho, že študent kurzu si nemôže zobraziť ostatných študentov kurzu a zistiť ich ID - čo má rovnaký efekt.