# Документација на проектот: Learning Management System

## Вовед

**Learning Management System (LMS)** е софистицирана платформа дизајнирана за управување со процесот на учење. Во оваа апликација, користени се **Vue.js** за фронтенд развој и **Django** за бекенд развој, со цел да се создаде интуитивен и функционален систем кој ги поддржува потребите на три различни типови на корисници: студенти, наставници и администратори.

### Главни компоненти на апликацијата

1. **Кориснички улоги и пристап:**
   - **Студенти (Students):** Може да се пријават, да разгледуваат курсеви и да се запишуваат на тие што ги интересираат. Имаат пристап до панелот за студенти каде можат да управуваат со запишувањето на курсевите.
   - **Наставници (Teachers):** Имаат можност да додаваат нови лекции во курсевите. Тие управуваат со содржината на курсевите и можат да ги уредуваат лекциите.
   - **Администратори (Admins):** Имаат целосен пристап и контрола над платформата. Може да додаваат нови курсеви, управуваат со корисниците и подесуваат основните параметри на системот.
2. **Курсеви и Лекции:**
   - Апликацијата овозможува создавање и управување со курсеви и лекции. Администраторите создаваат курсеви, додека наставниците додаваат лекции во тие курсеви. Студентите можат да се запишат на курсеви кои ги интересираат и да ги следат лекциите.
3. **Функционалности:**
   - **Регистрација и Логирање:** Корисниците можат да се регистрираат и логираат во системот, со што добиваат пристап до нивните специфични панели и функционалности.
   - **Панели за Управување:**
     - Студентскиот панел им овозможува на студентите да видат кои курсеви се запишани и да управуваат со нивните запишувања.
     - Панелот за наставници им овозможува на наставниците да управуваат со лекции и да ги уредуваат.
     - Панелот за администратори им овозможува на администраторите да управуваат со курсеви и корисници.
4. **Кориснички Интерфејс:**
   - Интерфејсот е дизајниран да биде интуитивен и лесен за користење. Користи компоненти на Vue.js за динамичко прикажување на информации и интеракција со корисниците.
5. **Управување со Состојба (Vuex):**
   - Vuex се користи за глобално управување со состојбата на апликацијата, вклучувајќи податоци за корисници, курсеви, лекции и запишувања. Ова

овозможува синхронизација на податоци и управување со состојбата на апликацијата преку различни компоненти.

# Frontend (Vue.js)

Frontend делот на проектот е имплементиран во Vue.js и користи Vue Router за управување со рутите. Еве краток преглед на главните патеки и нивните компоненти:

1. **Главна страница (/)** - Прикажува општи информации за платформата и можности за навигација.
2. **Регистрација (/register)** - Форма за регистрација на нови корисници.
3. **Најавување (/sign-in)** - Форма за најавување на постојни корисници.
4. **Контакт (/contact)** - Форма за контакт со администраторот на системот.
5. **Панел за наставник (/teacher)** - Панел за управување со курсеви и лекции за наставниците.
6. **Панел за студент (/student)** - Панел за преглед на курсеви и лекции за студентите.
7. **Корисници (/users)** - Приказ на список на сите корисници.
8. **Додавање курс (/addCourse)** - Форма за додавање нов курс.
9. **Уредување курс (/editCourse/:id)** - Форма за уредување на постоечки курс.
10. **Уредување лекција (/editLesson/:id)** - Форма за уредување на постоечка лекција.
11. **Додавање корисник (/addUser)** - Форма за додавање нов корисник.
12. **Уредување корисник (/editUser/:id)** - Форма за уредување на постоечки корисник.
13. **Детали за курс (/course/:id)** - Приказ на детали за конкретен курс.
14. **Додавање лекција (/course/:courseId/lessons/add)** - Форма за додавање нова лекција во одреден курс.
15. **Лекции по курс (/courses/:id/lessons)** - Приказ на лекциите за конкретен курс.
16. **Список на курсеви (/allCourses)** - Приказ на сите достапни курсеви.
17. **Запишани курсеви (/enrolledCourses)** - Приказ на курсевите во кои корисникот е запишан.

## Vue Router

Vue Router е библиотека која управува со рутирањето во Vue.js апликации и овозможува навигација помеѓу различни страници (патеки) во апликацијата. Оваа конфигурација на Vue Router е дизајнирана за управување со навигацијата помеѓу различни компоненти на апликацијата и обезбедува безбедност преку проверка на аутентификација. Секој патека е поврзана со специфична Vue компонента и се применува логика за контролирање на пристапот базиран на статусот на аутентификација на корисникот.

### 1. Импортирани Модули

- createRouter и createWebHistory од 'vue-router': Овие функции се користат за создавање на инстанца на Vue Router со историја на навигација.
- getAuth и onAuthStateChanged од 'firebase/auth': Овие функции се користат за проверка на статусот на аутентификација на корисниците преку Firebase.

2. **Патеки и Компоненти**

- /: Главната страница (Home.vue)
- /register: Страница за регистрација (Register.vue)
- /sign-in: Страница за логирање (SignIn.vue)
- /contact: Страница за контакт (ContactUs.vue)
- /teacher: Панел за наставници (Teacher-Panel.vue)
- /student: Панел за студенти (StudentPanel.vue)
- /users: Страница за управување со корисници (Users.vue)
- /addCourse: Форма за додавање на курс (CourseAddForm.vue)
- /editCourse/:id: Форма за уредување на курс со специфичен ID (CourseEditForm.vue)
- /editLesson/:id: Форма за уредување на лекција со специфичен ID (LessonEditForm.vue)
- /addUser: Форма за додавање на корисник (UserAddForm.vue)
- /editUser/:id: Форма за уредување на корисник со специфичен ID (UserEditForm.vue)
- /course/:id: Детали за курс со специфичен ID (CourseDetails.vue)
- /course/:courseId/lessons/add: Форма за додавање на лекција во курс со специфичен ID (AddLesson.vue)
- /courses/:id/lessons: Лекции за курс со специфичен ID (CourseLessons.vue)
- /allCourses: Листа на сите курсеви (CourseList.vue)
- /enrolledCourses: Страница за курсеви на кои студентите се запишани (EnrolledCourses.vue)

## Vuex Store

Vuex store се користи за управување со состојбата, Се справува со корисници, курсеви, лекции и автентикација на корисници, меѓу другото. Оваа структура обезбедува централизирано управување со состојбата на апликацијата, овозможувајќи лесен пристап и модификација на податоците.

## 1. Состојба (State)

**Состојбата** е основната структурална единица на Vuex store. Таа ги содржи сите податоци што се следат и управуваат од store-от. Во овој случај, состојбата содржи:

- currentUser: Податоци за моментално најавениот корисник.
- users: Листа на сите корисници во системот.
- courses: Листа на курсеви кои се достапни.
- lessons: Листа на лекции.
- role: Улогата на моментално најавениот корисник.
- currentLesson: Лекција која е моментално избрана за преглед.
- enrolledCourses: Листа на курсеви во кои корисникот е запишан.

## 2. Мутации (Mutations)

**Мутациите** се методи кои се користат за директно менување на состојбата. Тие се синхрони и се единствениот начин за промена на состојбата во Vuex.

- SET_ENROLLED_COURSES: Ажурира листата на запишани курсеви.
- setCurrentUser: Поставува моментално најавен корисник.
- setUsers: Поставува листа на корисници.
- setCourses: Поставува листа на курсеви.
- setLessons: Поставува листа на лекции.
- addUser: Додава нов корисник во листата.
- updateUser: Ажурира податоци за постоечки корисник.
- deleteUser: Отстранува корисник од листата.
- addCourse: Додава нов курс.
- updateCourse: Ажурира постоечки курс.
- deleteCourse: Отстранува курс.
- ADD_LESSON: Додава нова лекција во курс.
- UPDATE_LESSON: Ажурира постоечка лекција.
- DELETE_LESSON: Отстранува лекција.
- addLessonToCourse: Додава лекција во конкретен курс.
- updateLessonInCourse: Ажурира лекција во конкретен курс.
- removeLessonFromCourse: Отстранува лекција од конкретен курс.
- setUserRole: Поставува улога на корисникот.
- SET_CURRENT_LESSON: Поставува моментално избрана лекција.

## 3. Акции (Actions)

**Акциите** се методи кои се користат за извршување асинхрони операции, како што се AJAX повици, и можат да повикаат мутации за менување на состојбата. Тие можат да обработуваат асинхрони податоци и да вршат логика пред да ја променат состојбата.

- setUser: Поставува моментално најавен корисник.
- fetchUsers: Превзема список на корисници од серверот и ги поставува во состојбата.
- fetchCourses: Превзема список на курсеви од серверот.
- fetchLessons: Превзема лекции од серверот.
- createUser: Креира нов корисник и го додава во состојбата.
- getUser: Превзема податоци за конкретен корисник.
- editUser: Ажурира податоци за постоечки корисник.
- deleteUser: Отстранува корисник од серверот и состојбата.
- createCourse: Креира нов курс и го додава во состојбата.
- getCourse: Превзема податоци за конкретен курс.
- getLesson: Превзема конкретна лекција.
- editCourse: Ажурира податоци за курс.
- editLesson: Ажурира лекција.
- fetchLessonById: Превзема конкретна лекција по ID.

- removeCourse: Отстранува курс.
- createLesson: Креира нова лекција за конкретен курс.
- updateLesson: Ажурира лекција.
- deleteLesson: Отстранува лекција.
- fetchCurrentUser: Превзема податоци за моментално најавениот корисник.
- loginUser: Логира корисник и поставува токен за автентикација.
- enrollInCourse: Запишува корисник во курс.
- fetchEnrolledCourses: Превзема курсеви во кои корисникот е запишан.

## 4. Getters

**Getters** се користат за извршување на операции врз состојбата и враќање на резултати. Тие се слични на функции и се користат за добивање на податоци од состојбата.

- lessons: Враќа список на лекции.
- getLessonById: Враќа конкретна лекција по ID.
- currentLesson: Враќа моментално избрана лекција.
- enrolledCourses: Враќа список на курсеви во кои корисникот е запишан.
- isAuthenticated: Проверката дали корисникот е најавен.
- getUserbyEmail: Враќа корисник по email.
- currentUser: Враќа моментално најавен корисник.
- getCourses: Враќа список на курсеви.
- getCoursesByUserId: Враќа курсеви за конкретен корисник по ID.
- getCourseById: Враќа курс по ID.
- getLessonsByCourseId: Враќа лекции за конкретен курс по ID.
- userRole: Враќа улога на корисникот.

## App.vue

Оваа компонента претставува навигациски бар (navbar) за Vue.js апликација и дел од главната содржина. Еве како функционира целокупно:

1. **Навигациски бар**:
    - Компонентата создава навигациски бар што се прикажува на врвот на страницата. Овој бар содржи линкови за навигација помеѓу различни страници (како HOME, REGISTER, LOGIN, USERS) и копче за одјавување ако корисникот е најавен.
    - Прикажува улогата на најавениот корисник, ако таа улога е достапна.
2. **Управување со состојби**:
    - Проверува дали корисникот е најавен користејќи Firebase автентификација. Ако корисникот е најавен, се прикажува копчето за одјавување.
    - За време на процесот на монтирање на компонентата, се проверува статусот на најавата на корисникот и се ажурира состојбата isLoggedIn.
3. **Функции за навигација и одјавување**:
    - Дефинира функција за одјавување на корисникот. По успешната одјава, корисникот се пренасочува на почетната страница.

4. **Прикажување на содржина**:
    - Во делот  се прикажува главната содржина на страницата што зависи од активната рута, благодарение на <router-view/>.
5. **Стилизација**:
    - Компонентата вклучува стилови за дизајн на навигацискиот бар и копчињата, како и визуелни ефекти при наведнување.

Оваа компонента нуди основна навигација за корисниците на апликацијата и управува со прикажување на опции за најавување и одјавување, заедно со прилагодени стилови.

## AddLesson.vue

Оваа компонента овозможува додавање нова лекција за курс.

- **Шаблон (Template)**: Содржи форма со два полиња за внесување информации (наслов и содржина на лекцијата) и копче за додавање на лекцијата.
- **Скрипт (Script)**:
    - Користи Vuex акција createLesson за да ја испрати новата лекција на серверот.
    - По успешно додавање, се пренасочува на страницата со лекциите за соодветниот курс.
- **Стилови (Styles)**: Дизајнира форма која е центрирана на страницата со современ изглед, вклучувајќи транзиции и прилагодливост за различни големини на екрани.

## CourseAddForm

Оваа компонента е форма за додавање нов курс во системот. Еве краток преглед на нејзината функција:

- **Шаблон (Template)**:
    - **Форма**: Содржи полиња за внесување на наслов и опис на курсот.
    - **Избор на наставници**: Користи select со повеќекратен избор за додавање наставници.
    - **Избор на студенти**: Користи select со повеќекратен избор за додавање студенти.
    - **Префрлање на слики**: Овозможува прикачување слика (thumbnail) за курсот.
    - **Копче за испраќање**: Прачува податоци од формата и ги испраќа.
- **Скрипт (Script)**:
    - **Податоци за формата**: Користи ref за управување со податоците на курсот.
    - **Податоци од серверот**: По вчитувањето на компонентата, добива информации за наставниците и студентите од серверот.
    - **Обработка на слики**: Управува со прикачувањето на слики.
    - **Испраќање на форма**: Кога формата се испраќа, податоците се испраќаат на серверот преку Vuex акција.
- **Стилови (Styles)**:

- **Општ изглед**: Дизајнира форма со чист и модерен изглед, со внимание на флексибилност за различни големини на екрани.

Во целина, компонентата овозможува администраторите или наставниците да создадат нов курс, да го дефинираат неговото име, опис, наставници и студенти, и да додадат слика.

## CourseDetails.vue

Оваа компонента се користи за прикажување на детали за одреден курс, вклучувајќи список на лекции.

- Шаблон (Template): Прикажува наслов и опис на курсот, список на лекции ако тие постојат, и две копчиња - едно за додавање нова лекција и едно за враќање на страницата со сите курсеви.

**- Скрипт (Script):**

- Користи `vuex` за добивање на информации за курсот и лекциите.

- По вчитувањето на компонентата, го зема курсот и лекциите од серверот.

- Обезбедува функција за навигација назад на страницата со курсевите.

- Стилови (Styles): Дизајнира изглед на страницата со фокус на естетски пријатен изглед и функционални стилови, адаптирани за различни големини на екрани.

## CourseEditForm.vue

Оваа компонента е форма за додавање или уредување на курс. Ако има id во URL-то, компонента ќе се користи за уредување на постоечки курс; инаку, ќе се користи за создавање нов курс. Кога се прикачува форма, се користат Vuex акциите за создавање или уредување курс и корисникот се пренасочува на списокот на курсеви. Копчето „Cancel“ го враќа корисникот на списокот на курсеви без да прави промени.

## CourseLessons.vue

Оваа компонента служи за управување со лекциите на одреден курс. Еве што прави:

1. **Прикажува Лекции**: Прикажува листа на лекции поврзани со курсот. Секој елемент на листата покажува наслов и ID на лекцијата, како и опции за уредување и бришење на лекцијата.

2. **Генерира Акции**: Дава опции за:
    - **Уредување** на лекција (пренасочување на форма за уредување).
    - **Бришење** на лекција (прикажува модален прозорец за потврда пред да се избрише).
3. **Навигација**: Обезбедува копчиња за:
    - **Враќање на детали за курсот**.
    - **Додавање на нова лекција**.
4. **Модален Прозорец за Потврда**: Прикажува модален прозорец кога корисникот сака да избрише лекција. Овој прозорец обезбедува опции за потврда или откажување на бришењето.
5. **Прикажува Статус на Вчитување**: Прикажува порака „Loading lessons...” ако нема лекции за прикажување.

**Процеси:**

- **Прикажување на лекции**: На почеток, компонента ја повлекува листата на лекции од Vuex store по ID на курсот.
- **Обработка на Акции**: Управува со кликовите на копчињата за уредување и бришење, и го управува модалниот прозорец за потврда.
- **Навигација**: Употребува Vue Router за навигација помеѓу различни страници.

**Дизајн**: Компонента има стилови за визуелно привлекување и интеракција, со акценти на градиенти, преливи и реакции на ховер.

Оваа компонента е важна за администраторите и наставниците да управуваат со лекциите на курсот, и да имаат контрола над нивната содржина и организација.

# CourseList.vue

Оваа компонента служи за управување и прегледување на курсевите во системот. Основната функционалност е да овозможи корисниците да пребаруваат курсеви, да се запишуваат на курсеви, да ги гледаат лекциите на курсевите, да ги уредуваат и да ги бришат курсевите. Еве како функционира компонента:

## 1. Приказ на курсеви

- **Преглед на Курсеви**: Прикажува листа на курсеви со информации како што се слика на курсот, наслов, и ID на курсот.
- **Филтрирање на Курсеви**: Им овозможува на корисниците да пребаруваат курсеви според терминте внесени во пребарувачот. Курсевите се филтрираат на база на ударноста на терминот со насловите на курсевите.

## 2. Функционалности на Курсевите

- **Запишување на Курс**: Корисниците можат да се запишат на курсеви со клик на копчето „Enroll“. Ова ја повикува функцијата enrollInCourse, која се грижи за процесот на запишување и пренасочува на страницата со запишани курсеви.
- **Преглед на Лекции**: Со клик на копчето „View Lessons“, корисниците можат да ја видат листата на лекции поврзани со курсот. Ова ја повикува функцијата viewLessons, која пренасочува на страницата за лекции на курсот.
- **Уредување на Курс**: Корисниците можат да уредуваат курсеви со клик на копчето „Edit“. Ова ја повикува функцијата editCourse, која пренасочува на форма за уредување на курсот.

## 3. Управување со Курсевите

- **Бришење на Курс**: Корисниците можат да изберат да го избришат курсот со клик на копчето „Delete“. Ова ја прикажува модалната форма за потврда пред да се избрише курсот. Ако корисникот потврди, курсот ќе биде избришан со повикување на функцијата confirmDeletion.

## 4. Добавување на Нов Курс

- **Додавање Курс**: Корисниците можат да додадат нов курс со клик на копчето „Add New Course“. Ова ја повикува функцијата addNewCourse, која пренасочува на форма за додавање нов курс.

## 5. Стил и Дизајн

- **Визуелна Презентација**: Компонентата користи стилови за да го направи интерфејсот визуелно привлечен и лесен за користење. Стиловите вклучуваат употреба на фонтови, бои, и анимации за интеракции со копчињата и картите на курсевите.

## 6. Интеракција со Податоци

- **Vuex Store**: Компонентата користи Vuex за управување со податоците за курсевите и корисниците. Податоците се повлекуваат кога компонентата се монтира, и се обновуваат како што корисниците извршуваат акции.
- **Computed Properties**: Се користат за филтрирање на курсевите и за добивање на информациите за тековниот корисник.

## 7. Модален Прозорец

- **Потврда за Бришење**: Кога корисникот сака да избрише курс, се прикажува модален прозорец за потврда. Овој прозорец овозможува потврда или откажување на акцијата за бришење.

Со оваа компонента, корисниците можат да управуваат со курсевите во системот, да се запишуваат на курсеви, и да ја подобрат својата корисничка интеракција со софтверот.

## Home.vue

Оваа Vue.js компонента е создадена за homepage на LMS. Содржи неколку секции кои го претставуваат системот и нудат информации и опции за корисниците.

## Основни Функции

1. **Херо Секција**:
   - Прикажува приветна порака и краток опис на LMS.
   - Вклучува копче „Explore Courses“ за упатување на корисниците кон страницата со сите курсеви.
2. **Секција со Карактеристики**:
   - Прикажува три клучни карактеристики на системот, како што се интерактивно учење, експертски инструктури и флексибилни распореди.
   - Секцијата користи слики и текстови за да ги истакне овие карактеристики.
3. **Контакт Секција**:
   - Пружа информации за контакт и опција за испраќање прашања или барања за поддршка.
   - Вклучува копче „Contact Us“ за упатување на корисниците на страницата за контакт.

## Структура

- **HTML**: Структурирана во template со различни секции за херо, карактеристики, отзиви и контакт.
- **CSS**: Стиловите се примарно наменети за современ изглед со внимание на анимации, заднини и интерактивност на копчињата и сликите.
- **JavaScript**: Не содржи посебна логика во script делот; се користи само Vue.js template за структурата на страницата.

Оваа компонента служи како основна страница за привлечност на нови корисници и обезбедување на информации за она што го нуди LMS.

# LessonEditForm.vue

Оваа Vue.js компонента е формулар за уредување лекција во вашиот LMS. Нумерирани се неколку главни делови и нивната функција:

## Основни Функции

1. **Форма за уредување**:
   - Содржи полиња за внесување на наслов (title) и содржина (content) на лекцијата.
   - Вклучува два копчиња: „Save Changes“ за зачувување на промените и „Cancel“ за откажување на уредувањето.
2. **Логика за податоци**:
   - Во script делот, компонентата користи Vue.js ref за управување со податоците на лекцијата и Vuex за комуникација со серверот.

- По иницијализацијата на компонентата (onMounted), се повикува акција од Vuex (fetchLessonById) за да се преземат податоците на лекцијата според ID-то кое се добива од URL параметрите.
- Функцијата updateLesson испраќа ажурирани податоци на серверот и насочува кон страницата со лекции за конкретен курс.
- Функцијата cancelEdit насочува кон страницата со лекции ако уредувањето се откаже.

3. **Стилови**:
   - Компонентата е стилизирана со CSS за да изгледа современо и пријатно за корисниците. Содржи стилови за распоред на формата, полињата за внесување, и копчињата.
   - Погодности како што се linear-gradient позадини, анимации при движење на копчињата и транзиции за фокусирање на полињата се вклучени за подобрување на корисничкото искуство.
   - Постојат и медиумски упатства за одржување на реагирачки дизајн на помали екрани.

## Како Работи

- По отворањето на компонентата, се преземаат податоците на лекцијата и се прикажуваат во формата.
- Корисниците можат да ги изменат податоците и да ги зачуваат или да го откажат уредувањето.
- При зачувување на промените, податоците се ажурираат на серверот, а корисникот се усмерува на страницата со лекции.

Оваа компонента е клучен дел од администраторскиот интерфејс или интерфејсот на наставниците во вашето LMS, овозможувајќи им на корисниците да управуваат со лекциите ефективно.

## SignIn.vue

Оваа Vue.js компонента е дизајнирана за процесот на влез на корисник во системот со користење на e-mail и лозинка. Еве што прави компонентата на општ начин:

1. **Функционалност**:
   - **Логирање на корисник**: Компонентата користи Firebase Authentication за да изврши пријавување на корисник со e-mail и лозинка.
   - **Справување со грешки**: Ако пријавувањето не успее, компонента покажува соодветна порака за грешка.
   - **Доделување улоги**: По успешна пријава, компонентата ги проверува податоците за корисникот во Vuex складиштето и го пренасочува корисникот на база на неговата улога (наставник, студент или друга улога).
2. **Шаблон (Template)**:
   - **Форма за пријавување**: Постојат полиња за внес на e-mail и лозинка, како и копче за поднесување на формата. Секој повратен елемент за грешка се прикажува доколку има таква.
3. **Стилизирање (Styles)**:

- **Дизајн**: Компонентата има модерен дизајн со позадинска слика, стилски прилагодени внесни полиња и копчиња. Пораките за грешка се прикажуваат во црвена боја за да бидат лесно видливи.

4. **Скрипт (Script)**:
   - **Операции**: Содржи функција за пријавување (register) која се повикува кога корисникот ќе го притисне копчето за пријавување. Оваа функција се повикува на Firebase Auth методите за пријавување и, по успешна пријава, ги користи Vuex акциите за да ја постави улогата на корисникот и да го пренасочи на соодветната страница.

Оваа компонента комбинира функционалност и стил за да обезбеди корисничко искуство за пријавување на корисникот, истовремено управувајќи со различни сценарија за грешки и пренасочување.

## UserAddForm.vue

Оваа Vue.js компонента е форма за додавање нов корисник во системот. Еве краток преглед на тоа што прави:

1. **Форма за Внес**:
   - Компонентата овозможува внесување на различни информации за новиот корисник, вклучувајќи име, презиме, корисничко име, e-mail, лозинка, улога, биографија, дата на раѓање и телефонски број.
   - Полето за улога има три опции: студент, наставник и администратор, што овозможува избор на улога за новиот корисник.
2. **Скрипта**:
   - **Функција за Поднесување**: Кога формата ќе биде поднесена, функцијата handleSubmit се активира. Таа ја користи Firebase Authentication за да создаде нов корисник со e-mail и лозинка. Потоа, ја користи Vuex акцијата за да ги чува податоците за новиот корисник во складиштето.
   - **Пренасочување**: По успешно додавање на корисникот, компонентата го пренасочува корисникот на страница за управување со корисници.
3. **Дизајн**:
   - **Стилизирање**: Компонентата има стилови кои создаваат современ изглед со позадинска слика, прозирни области и стилски прилагодени полиња и копчиња.
   - **Форма**: Секое поле во формата има стилови за подобрување на корисничкото искуство, како што се анимации на етикетите и контрола на изгледот на полето при фокусирање.

Оваа компонента е наменета за креирање на нови корисници со специфични податоци и улоги, и го комбинира процесот на внес, валидација и чување на податоци во интегриран кориснички интерфејс.

### Users.vue

Оваа Vue.js компонента е страница за преглед на списокот на корисници во системот. Еве краток преглед на тоа што прави:

1. **Преглед на Корисници**:
    - Компонентата прикажува табела која го содржи списокот на корисници. Табелата ги прикажува корисничките информации како корисничко име, e-mail, улога, име, презиме и биографија.
2. **Функции за Управа**:
    - **Уредување**: Кога корисникот ќе кликне на копчето "Edit", компонентата го пренасочува корисникот на форма за уредување на корисничките детали.
    - **Бришење**: Кога корисникот ќе кликне на копчето "Delete", се прикажува потврда за бришење, и ако корисникот потврди, се покренува акција за бришење на корисникот од базата на податоци.
3. **Интеграција со Vuex и Vue Router**:
    - Компонентата користи Vuex за добивање на списокот на корисници од складиштето и за извршување на операции како што се бришење на корисникот.
    - Кога компонентата се создава, автоматски ја повикува акцијата за добивање на корисниците од складиштето.
    - За управување со навигацијата, се користи Vue Router за пренасочување кон страницата за уредување на корисници.

Оваа компонента е корисна за администратори или менаџери кои треба да ги управуваат корисниците во системот, да прават преглед на нивните информации и да ги уредуваат или бришат корисниците по потреба.

# Backend (Django)

Овој backend дел претставува основна структура на Learning Management System (LMS) користејќи Django и Django Rest Framework (DRF).Аархитектурата овозможува основна функционалност за управување со курсеви, лекции и корисници, како и запишување на студентите во LMS систем.

### Models.py

Ова е основен модел за Learning Management System (LMS) изграден со Django. Еве објаснување за секоја од класите и нивната функција:

1. **UserLMS (Корисник)**:
    - Оваа класа претставува корисник на системот. Има различни улоги (student, teacher, admin) дефинирани преку ROLE_CHOICES. Секој корисник има основни информации како име, презиме, корисничко име (уникатно), е-пошта (уникатна), биографија, телефонски број и датум на раѓање.
2. **Course (Курс)**:

- Курсевите се дефинирани со наслов, опис, и поле за слика (thumbnail) како опција за визуелен приказ.
- Наставниците (teachers) можат да предаваат курсеви, а студентите (students) можат да се запишат на нив. Ова е овозможено преку ManyToManyField врски со корисниците.

3. **Lesson (Лекција)**:
   - Секој курс може да има повеќе лекции, каде лекцијата има наслов, содржина (текст), и опционален линк до видео. Лекциите се поврзани со курсевите преку ForeignKey, со што секоја лекција е асоцирана со еден курс.
4. **Enrollment (Запишување)**:
   - Оваа класа претставува запишувањето на студентите на одредени курсеви. Таа ја содржи релацијата помеѓу корисникот (студент) и курсот, заедно со времето на запишување. Полето unique_together осигурува дека еден студент може да се запише на еден курс само еднаш.

Овие модели овозможуваат основната функционалност за корисници, курсеви, лекции и запишување, што претставува основа за еден LMS систем.

## Serializers.py

Овие се **Django Rest Framework (DRF) серијалајзери** за моделите во LMS системот. Серијалзерите се користат за конвертирање на моделите во JSON формат, кој потоа може да се користи во API одговори, или пак за валидација и обработка на податоци при барања.

1. **UserLMSSerializer**:
   - Овој серијалајзер ги претставува податоците за корисниците (UserLMS) во JSON формат. Полето fields наведува кои атрибути од корисничкиот модел ќе бидат достапни во API (ID, име, презиме, корисничко име, е-пошта, и улога).
2. **CourseSerializer**:
   - Овој серијалајзер ги претставува податоците за курсевите (Course). Тој содржи логика за:
     - **create()**: При креирање на курс, постои опција за прикачување на thumbnail. Ако сликата постои, таа се зачувува.
     - **update()**: Слично на create(), се овозможува ажурирање на информациите за курсот и сликата.
3. **LessonSerializer**:
   - Овој серијалајзер се користи за лекциите (Lesson). Содржи стандардни методи за:
     - **create()**: Креира нова лекција користејќи ги внесените податоци.
     - **update()**: Ажурира податоци на постоечка лекција.

Со овие серијалајзери, API-то може да испраќа и прима податоци за корисниците, курсевите, и лекциите во JSON формат, како и да ги валидира при внесување нови податоци.

# Views.py

## UserViewSet

- Ова е viewset за управување со корисници на LMS системот.
- Ги поддржува стандардните CRUD операции (create, read, update, delete) за корисници.
- Методите create и update се прилагодени за да логираат и процесираат податоци, како и да го користат serializer-от за валидација.

## CourseViewSet

- Ги управува операциите поврзани со курсевите.
- Методот create:
  - Прима ID за учители и студенти преку барањето, ги обработува, и ги додава во курсот.
  - Го креира курсот со валидираните податоци и ги поврзува со учителите и студентите.
- Методот update:
  - Се користи за ажурирање на постоечките курсеви.

## LessonViewSet

- Се користи за управување со лекции.
- Методот create:
  - Прима ID за курсот во кој се додава лекцијата.
  - Го верификува курсот и го додава ID-то на лекцијата.
- Методот perform_destroy:
  - Принта порака кога лекцијата е избришана.

## EnrolledCoursesList

- API кој ги прикажува сите курсеви во кои е запишан тековниот корисник.
- Ги враќа запишаните курсеви користејќи го Enrollment моделот и ги враќа курсевите преку CourseSerializer.

## EnrollUserView

- API за запишување на корисници во курс.
- Методот post:
  - Прима user_id и course_id, ги верификува соодветните корисници и курсеви.
  - Додава или обновува запишување и враќа соодветна порака ако е успешно или корисникот е веќе запишан.

Овие views овозможуваат основна функционалност за LMS системот за креирање и управување со корисниците, курсевите, лекциите и запишувањата, како и за добивање на податоци преку API.

# Urls.py

Ова е конфигурација за **URL рутирање** во Django Rest Framework (DRF), која ги поврзува различните view-ови со соодветните URL рути за **LMS проектот**. Еве го објаснувањето на деловите:

## Router Configuration

- DefaultRouter() автоматски ги генерира CRUD рутите за viewsets:
  - **users**: CRUD операции за корисници преку UserViewSet.
  - **courses**: CRUD операции за курсеви преку CourseViewSet.
  - **lessons**: CRUD операции за лекции преку LessonViewSet.

## Lesson Management

- За креирање и управување со лекции, има посебни рути за лекции поврзани со курсевите:
  - **lesson_list**: Овозможува добивање на листа од сите лекции за одреден курс (GET) и креирање на нова лекција (POST).
  - **lesson_detail**: Овозможува добивање на деталите за една лекција (GET), ажурирање (PUT), и бришење на лекција (DELETE).

## API Endpoints

1. **/api/users/current/**: Го враќа тековниот најавен корисник преку CurrentUserView.
2. **/api/courses/<int:course_id>/lessons/**: Листа и креира лекции поврзани со одреден курс (користејќи LessonViewSet).
3. **/api/courses/<int:course_id>/lessons/<int:pk>/**: Достап до поединечни лекции поврзани со курсеви.
4. **/api/lessons/**: Листа на сите лекции и нивно креирање (ако не се поврзани со курсеви).
5. **/api/enrolled_courses/**: Ги прикажува курсевите во кои е запишан тековниот корисник преку EnrolledCoursesList.
6. **/api/enroll/**: За запишување на корисници во курсеви преку EnrollUserView.

Оваа конфигурација овозможува флексибилно управување со корисници, курсеви, лекции, и запишувања, и го прави API лесен за користење и интеграција.