



## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Извештај за проектна задача по предметите “Имплементација на системи со слободен и отворен код” и “Напреден веб дизајн” на тема:

**“Систем за прибирање и менаџирање на  
одговори од онлајн форм - FormFlow”**

Изработила: Моника Тодорова, индекс: 201082

Ментор: д-р Бобан Јоксимоски



## Апстракт

Секојдневно во светот на вебот се раѓаат стотици нови веб страници со најразлични цели за маркетинг, поврзување со потенцијални клиенти, продажба и уште многу акции кои се тесно поврзани со комуникација. Со цел да се оствари основна комуникација помеѓу две страни на една веб страна потребна е конфигурација која за лица со недоволно техничко познавање е предизвикувачка за поставување. Поттикнува од овој проблем, ја креирам FormFlow апликацијата која овозможува олеснет пристап на конфигурација, а воедно нуди и соодветени опции за прибирање и менаџирање на информациите добиени од онлајн формите кои се поврзани во апликацијата.

## Вовед

FormFlow е веб апликација која е дизајнирана да го поедностави процесот на прибирање и менаџирање на одговори од онлајн форми. Им овозможува на корисниците да ги поврзат нивните форми до соодветен ендпоинт, со што сите одговори се прибираат и организираат. Во FormFlow корисниците можат да видат детали за одговорите, да додават тагови за полесна категоризација, да филтрираат според таг или статус и да групираат во проекти. Ова ја прави апликацијата соодветна алатка за лица со и без техничко познавање, иако е потребно основно познавање за поврзување на формата. Изработена е користејќи технологии како Laravel за backend, Vue3 за frontend и MySQL база на податоци.

Во продолжение детално ќе разработиме за веб апликации, користените технологии како и начинот на изработка и функционирање на апликацијата FormFlow.

## Што е Веб?

Вебот, познат како World Wide Web (WWW) е меѓусебно поврзан систем на јавни веб страници достапни преку Интернет. Вебот е всушност една од многуте апликации изградени врз основа на Интернетот. Системот што ние денес го знаеме како "Веб" се состои од неколку компоненти:

- HTTP протоколот - управува со преносот на податоци помеѓу серверот и клиентот,
- URL - идентификатор кој обезбедува пристап до веб-компонента,
- HTML - најпознат формат за публикување на веб документи,
- Веб сервер - компјутер каде се зачувуваат документи до кои може да се пристапи со користење на HTTP протоколот.

Поврзувањето на различните веб компоненти преку хипер врски е главниот концепт на Вебот. Така всушност е креирана функцијата како збирка од поврзани документи. Вебот е од огромно значење во денешницата. Го поврзува целиот свет и овозможува лесна и непречена комуникација, споделување и пребарување во било кое време.



## Веб страница и Веб сајт

Веб страница е документ поставен на Интернет и достапен за разгледување. Веб документите имаат наставка .html или .htm. Содржината на веб страниците е составена комбинирани текстови, слики, звуци и видео записи. За да биде достапна една веб страница, таа мора да има свој уникатен линк за пристап. Линкот, попознат и како URL.

## Веб апликација

Веб апликациите користат комбинација од скрипти од страна на серверот за да се справат со складирање и пронаоѓање информации и од страна на клиентот за да се презентираат тие информации. Преку нив се овозможува интерактивност помеѓу понудувачи на услуги и нивните корисници. Апликациите овозможуваат креирање документи, споделување информации, непречена комуникација без оглед на локацијата или уредот.

### A. Како работат веб апликациите?

Еден типичен тек на веб апликација ги содржи следниве чекори:

1. Корисникот активира барање до веб серверот преку Интернет, поконкретно преку веб прелистувач.
2. Веб серверот го препраќа барањето до соодветниот сервер за веб апликации.
3. Веб апликативниот сервер ја извршува бараната задача - како што е пребарување во базата на податоци или обработка на податоци и потоа генерира соодветен повратен одговор.
4. Серверот за веб апликации ги испраќа резултатите со бараните податоци до веб серверот.
5. Веб серверот му одговара на клиентот со приказ на бараните информации на екранот.

### B. Предности на веб апликациите

Во последно време веб апликациите заземаат голем замав и влијание во светот на Интернет, а тоа се должи на неколкуте предности со кои тие се истакнуваат:

1. Веб апликациите работат на повеќе платформи без оглед на оперативниот систем или уредот се додека прелистувачот е компатибилен.
2. Сите корисници пристапуваат до истата верзија, елиминирајќи ги сите проблеми со компатибилноста.
3. Тие не се превземаат на уредот, со што се елиминираат ограничувањата на меморија во уредот.
4. Ги намалуваат трошоците за бизнисот и за крајниот корисник бидејќи се полесни за одржување и не бараат специфична хардверска конфигурација од крајниот корисник.



## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

### С. Од што се состои една веб апликација

Споменавме дека веб апликацијата е всушност напредна веб страница, односно веб страница која нуди интерактивност и подобро корисничко искуство на крајните корисници.

За една веб апликација да биде функционална и атрактивна потребно е да се обрати внимание на повеќе фактори. Идеално е апликацијата да има привлечен, интуитивен изглед, а притоа непречено да ги извршува функциите за кои е наменета.

Со оваа цел, за апликацијата FormFlow се изработени backend, frontend и база на податоци. Во продолжение ќе посветиме повеќе внимание на сите споменати.

### База на податоци

База на податоци е збирка или колекција на податоци кои опишуваат одреден елемент. Со цел комуникација со базите на податоци, земање и анализирање на податоци, постојат системи за управување со бази на податоци (DBMSs). Овозможуваат создавање, дефинирање, ажурирање, пребарување и управување со базите. Познати вакви системи се MySQL, PostgreSQL, Oracle, Microsoft SQL Server итн.

За потребите на оваа апликација е искористено MySQL.

За полесен пристап до базата на податоци и информациите во неа е употребено phpMyAdmin и browser.

The screenshot shows the phpMyAdmin interface for a database named 'formflow'. The left sidebar displays a tree view of the database structure, including tables like 'colors', 'failed\_jobs', 'forms', 'migrations', 'oauth\_access\_tokens', 'oauth\_auth\_codes', 'oauth\_clients', 'oauth\_personal\_access\_clients', 'oauth\_refresh\_tokens', 'password\_reset\_tokens', 'payments', 'projects', 'recipients', 'submissions', 'submissions\_tags', 'tags', and 'users'. The main panel shows a table structure view with columns: Table, Action, Rows, Type, Collation, Size, and Overhead. The table 'users' is selected, and its structure is displayed. The table has 19 rows and is of type InnoDB with utf8mb4\_general\_ci collation. The total size of the database is 5,965 KiB.

Table	Action	Rows	Type	Collation	Size	Overhead
colors	Browse Structure Search Insert Empty Drop	34	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
failed_jobs	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
forms	Browse Structure Search Insert Empty Drop	215	InnoDB	utf8mb4_unicode_ci	80.0 KiB	-
migrations	Browse Structure Search Insert Empty Drop	20	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
oauth_access_tokens	Browse Structure Search Insert Empty Drop	63	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
oauth_auth_codes	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
oauth_clients	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
oauth_personal_access_clients	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
oauth_refresh_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
password_reset_tokens	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
payments	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
projects	Browse Structure Search Insert Empty Drop	60	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
recipients	Browse Structure Search Insert Empty Drop	407	InnoDB	utf8mb4_unicode_ci	80.0 KiB	-
submissions	Browse Structure Search Insert Empty Drop	4,610	InnoDB	utf8mb4_unicode_ci	2.7 MiB	-
submissions_tags	Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
tags	Browse Structure Search Insert Empty Drop	523	InnoDB	utf8mb4_unicode_ci	80.0 KiB	-
users	Browse Structure Search Insert Empty Drop	19	InnoDB	utf8mb4_unicode_ci	80.0 KiB	-
18 tables	Sum	5,965	InnoDB	utf8mb4_general_ci	3.3 MiB	0 B

Слика 1 - Приказ на табелите од базата на податоци во phpmyadmin



## Backend

Backend е делот од една веб-апликација кој работи во позадина и е одговорен за обработка на податоци, логика на апликацијата, и комуникација со базата на податоци. Тој не е видлив за корисниците, но е клучен за правилното функционирање на апликацијата. Бекендот ги обработува барањата од корисничкиот интерфејс (фронт-енд) и испраќа назад информации кои се прикажуваат на корисникот. За изработка на бекенд делот во FormFlow е искористен Laravel.

## Што е Laravel?

Laravel е популарен PHP фрејмворк за развој на бекенд делот од веб-апликации. Тој се користи поради својата едноставност, брзина и моќни функции. Laravel нуди структуриран начин на организирање на кодот, што го прави развојот побрз и поефикасен. Нуди различни алатки и библиотеки кои помагаат во автоматизирање на чести задачи, како што се рутирање на барања, автентикација на корисници, управување со бази на податоци.

## Предности на користење на Laravel во апликацијата

Користењето на Laravel во FormFlow, носи многу предности. Laravel ја поедноставува интеграцијата на автентикација и безбедност преку алатки како Laravel Passport, што овозможува безбедно управување со кориснички пристап и заштита на податоците. Потоа, Laravel е многу скалабилен, што значи дека може да се користи за развој на мали апликации, но и за големи и комплексни системи. Исто така има активна заедница и богата документација, што го олеснува процесот на учење и решавање на проблеми. Со користење на Laravel, апликацијата станува посигурна, полесна за одржување, и побрза за развој.

## Работа со Laravel

За да се креира нов Laravel проект се користи командата

```
laravel new formflow-backend
```

или ако се користи Composer

```
composer create-project laravel/laravel formflow-backend
```

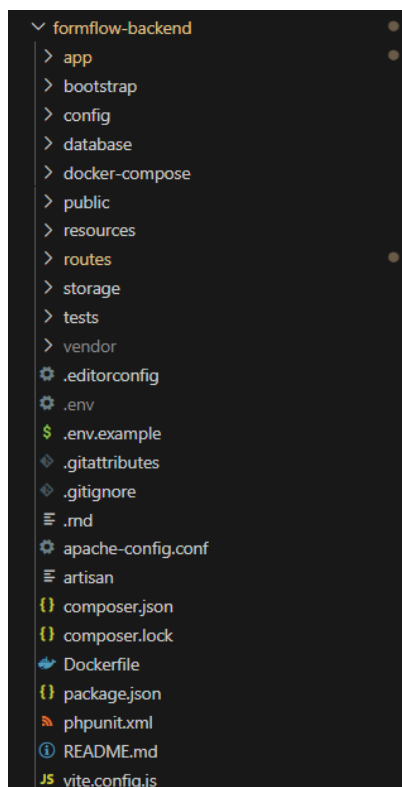
Со ова, ќе се креира нова Laravel апликација во директориумот formflow-backend.



## Структура

Откако ќе се креира проектот, структурата е организирана на следниот начин:

- `app/`: Содржи апликациски код, модели, контролери, и други класи.
- `bootstrap/`: Содржи код за почетно подигање на апликацијата.
- `config/`: Содржи конфигурациски датотеки.
- `database/`: Содржи миграции, factories, seeders.
- `public/`: Јавниот директориум од каде што се служат датотеките.
- `resources/`: Содржи извори на податоци како што се views, JavaScript и CSS.
- `routes/`: Содржи сите дефинирани рути на апликацијата.
- `storage/`: Содржи кеширани датотеки, логови, и сесии.
- `tests/`: Содржи тестови за апликацијата.
- `vendor/`: Содржи библиотеки инсталирани преку Composer.



## Работа со база и податоци во Laravel

Миграциите се начин за управување со базите на податоци во Laravel. За креирање на нова миграција се користи командата

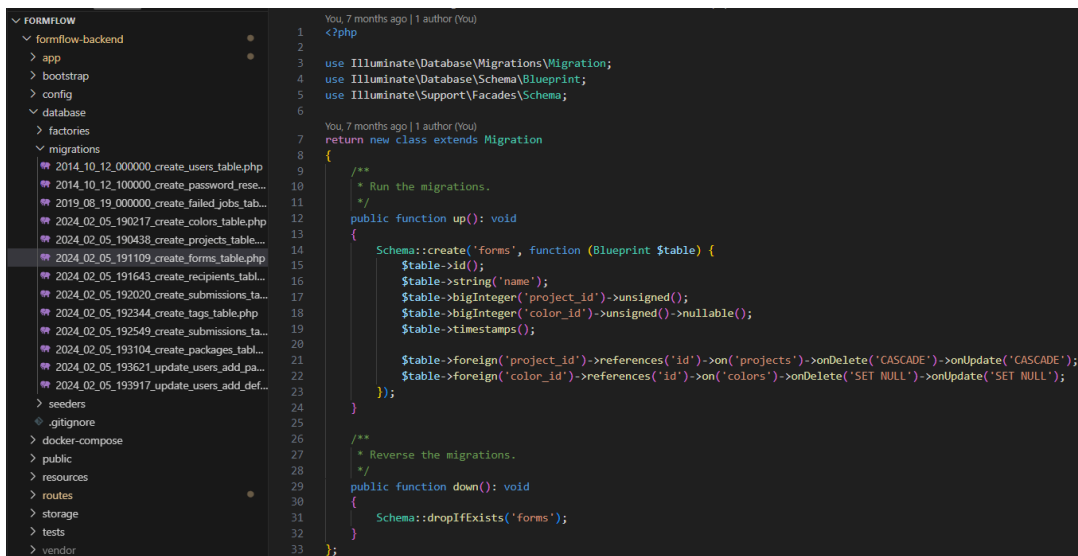
```
php artisan make:migration create_users_table
```

Ова ќе креира нова миграција во папката 'database/migrations'. Во миграцијата се дефинира структура на табелите во база. По дефинирањето на миграциите, со цел да се применат промените во база се извршува командата

```
php artisan migrate
```



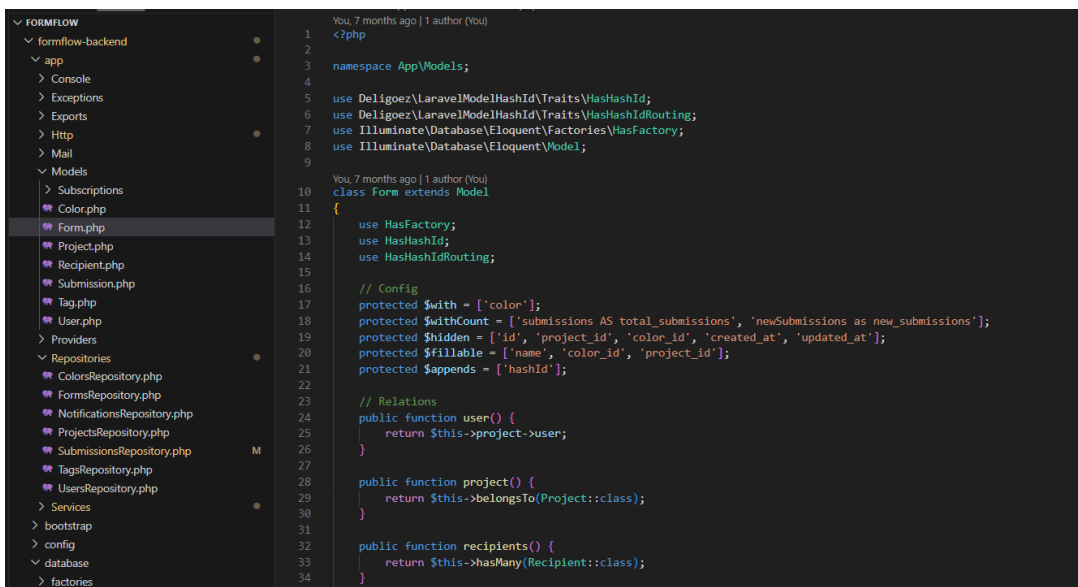
## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО



Слика 2 - Пример миграција за креирање на табелата за форми

### Eloquent

Eloquent е ORM (Object-Relational Mapping) кој го користи Laravel за работа со бази на податоци. Со Eloquent може да се работи со базите на податоци користејќи PHP објекти наместо SQL команди. Eloquent моделите се всушност класи кои ги претставуваат табелите од база и обезбедуваат методи за работа со податоците. Eloquent може да се користи за креирање, читање, ажурирање и бришење на записи во базата.



Слика 3 - Модел за ентитетот форма, приказ на неколку релации



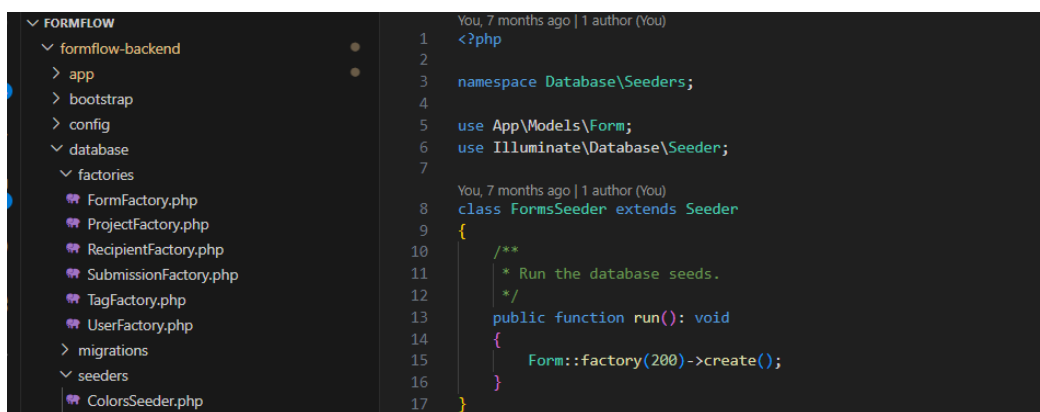
## Database seeders

Откако ќе се дефинираат миграциите и ќе се дефинираат табелите во база, можеме да креираме seeders – класи кои дозволуваат автоматско внесување на податоци во базата на податоци. Со ова, на едноставен начин, ја полниме базата со иницијални или тест податоци.

Нова seeder класа во фолдерот 'database/seeders' креираме, и ги извршуваме истите преку командите

```
php artisan make:seeder FormsTableSeeder
```

```
php artisan db:seed
```

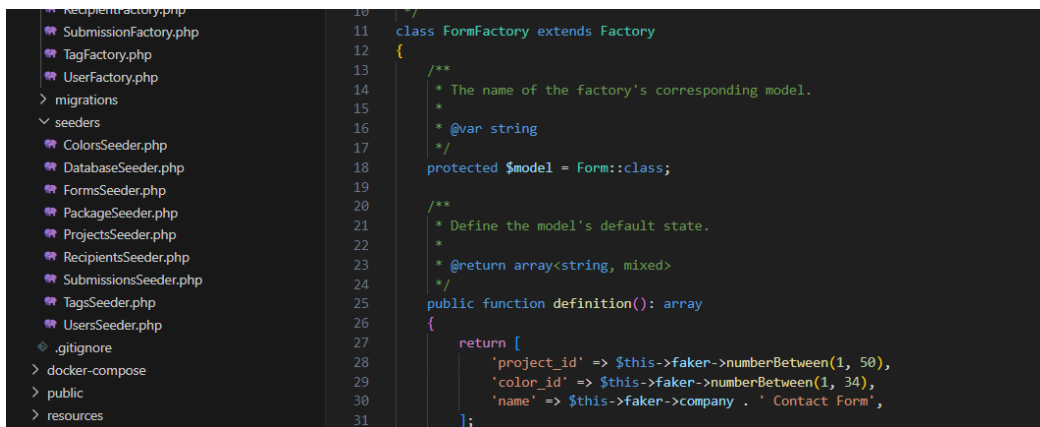


Слика 4 - Seeder за ентитетот Form

## Database factories

Со цел да пополниме податоци во базата на податоци, потребно е претходно да ги дефинираме податоците. Factories се класи кои овозможуваат брзо креирање на објекти според даден модел, за тестирање и полнење на податоци. Factory класа се креира со

```
php artisan make:factory FormFactory --model=Form
```



Слика 5 - Factory класа за моделот Form

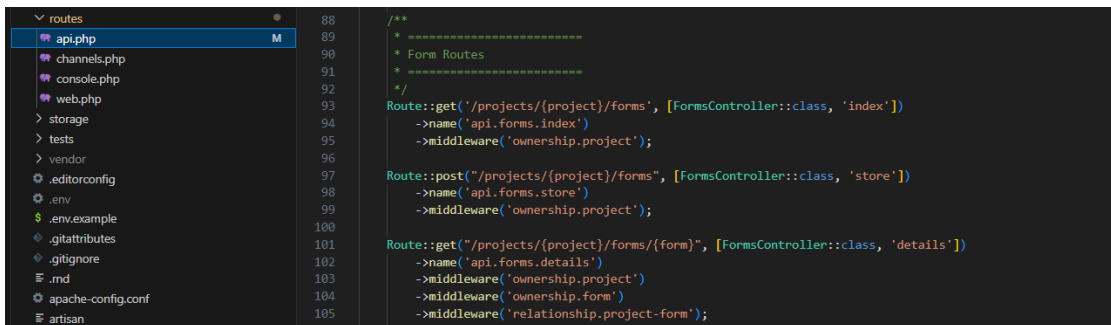




## Дефинирање на рути за API

За да може фронт-енд делот од апликацијата да комуницира со бекенд и базата, потребно е да се креираат API ендоинти.

Во Laravel, рутите се дефинираат во 'routes' папката. Рутите за API обично се наоѓаат во 'routes/api.php'. Овие рути се користат за целосната комуникација, како и за управување со CRUD операции (create, read, update, delete).

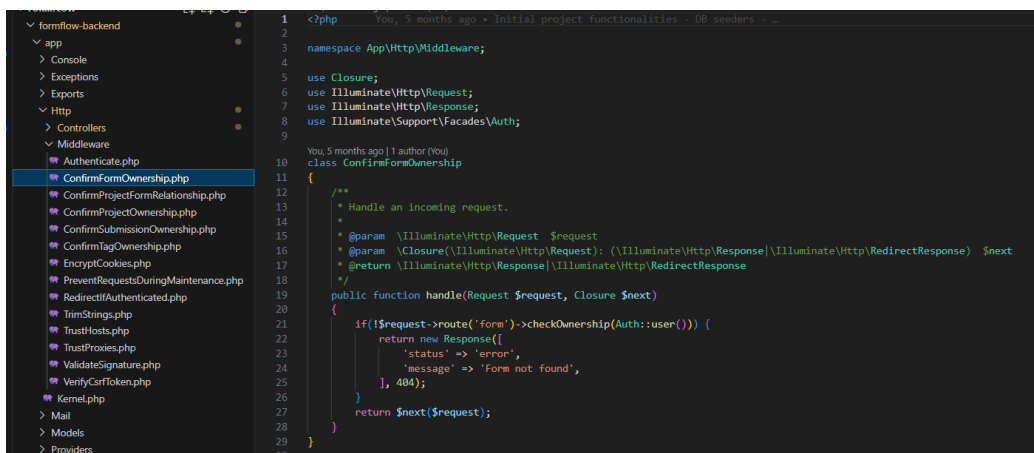


Слика 6 - Рутите за forms ентитетот

## Middleware

Middleware во Laravel се класи кои ја обработуваат HTTP барањето пред да стигне до контролерот. Тие се користат за да извршат различни проверки или операции, како што се автентикација, валидација на податоци, проверка на кориснички дозволи и многу повеќе.

Во случајов кај рутите на форма имаме дефинирано три middlewares за спроведување на проверки. Два од нив се однесуваат на ownership, односно проверуваме дали соодветниот проект и форма се во сопственост на моменталниот корисник, а третиот се однесува на проверка дали постои врска меѓу формата и проектот.



Слика 7 - Пример middleware за проверка на сопственост на форма



## Структура на проект

### Модели

Моделите во Laravel ги претставуваат табелите во базата на податоци и овозможуваат интеракција со податоците преку Eloquent ORM. Тие се наоѓаат во `app/Models`.



Слика 8 - Модел за Form

### Контролери

Контролерите се класи кои ги обработуваат HTTP барањата и ги враќаат одговорите. Тие обично се наоѓаат во `app/Http/Controllers`.

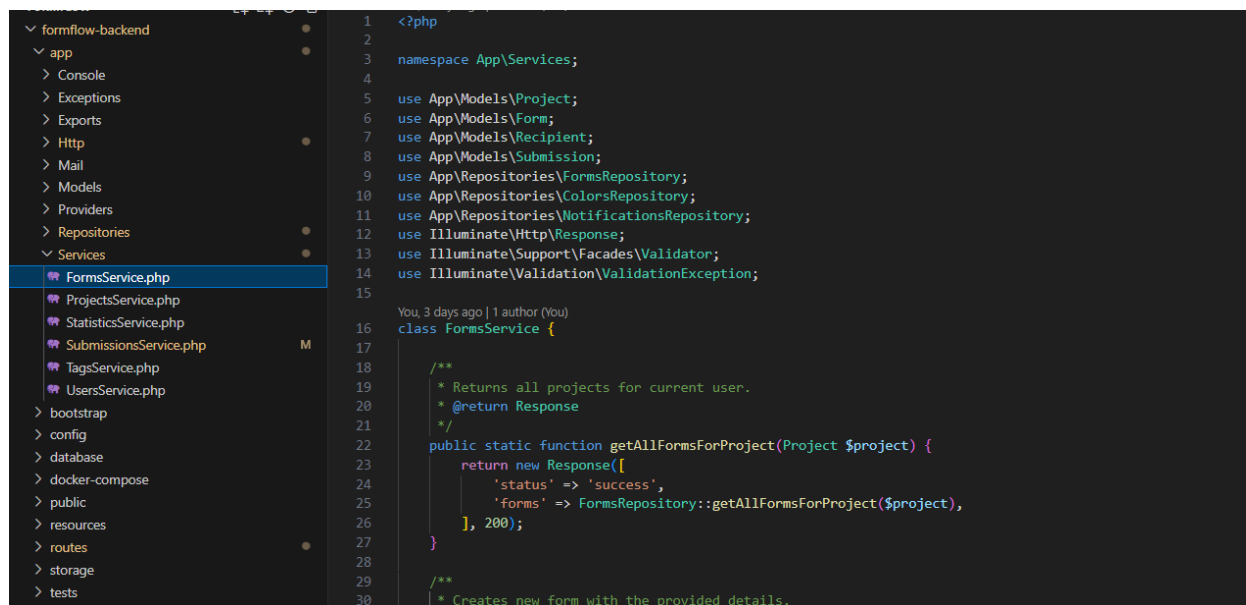


Слика 9 - Контролер за Form



## Сервиси

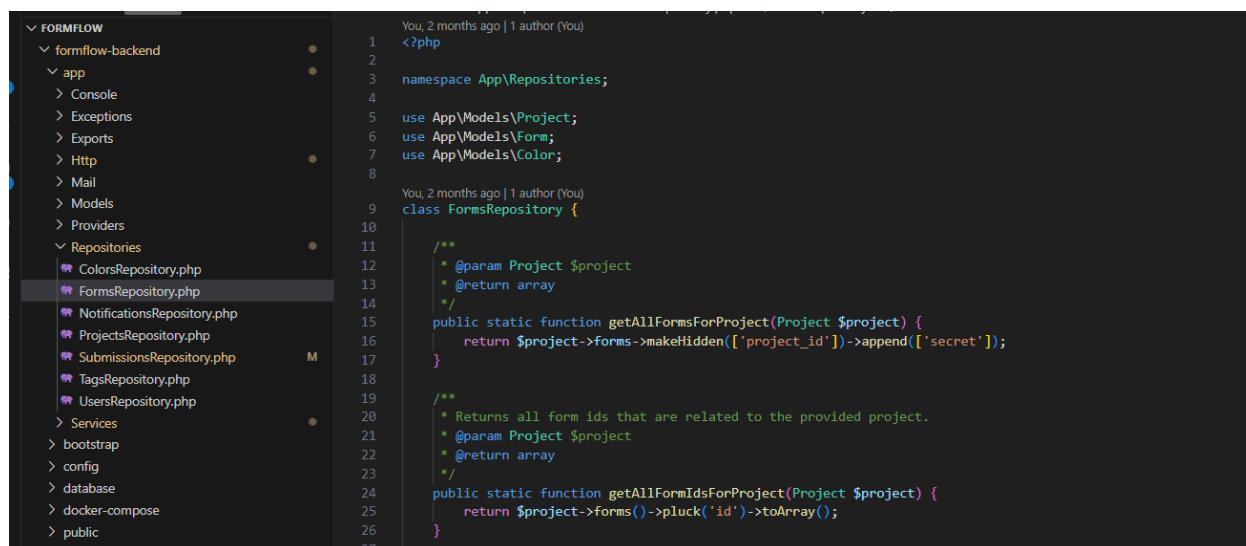
Сервисите се класи кои содржат бизнис логика која е одделена од контролерите за да се подобри организацијата и одржувањето на кодот. Тие се наоѓаат во `app/Services`.



Слика 10 - Сервис за Form

## Репозиториуми

Репозиториумите се класи кои управуваат со директен пристап до податоците. Тие овозможуваат изолација на слојот за пристап до податоците и се наоѓаат во `app/Repositories`.



Слика 11 - Репозиториум за Form



## Како се обработува барање во Laravel

Кога ќе се испрати барање до Laravel апликацијата, се случува следново:

1. **Рутата се препознава:** Laravel ја проверува рутата што одговара на URL-то на барањето (во routes/api.php).
2. **Контролерот се повикува:** Рутата ја повикува соодветната метода од контролерот. На пример, ако барањето е за /forms со GET метод, ќе се повика index() метода од FormController.
3. **Сервисот го обработува барањето:** Ако има сервис, контролерот го користи сервисот за да ја обработи бизнис логиката (на пр. преземање на сите форми).
4. **Моделите и Репозиториумите ја вршат интеракцијата со базата:** Ако се користат репозиториуми, сервисот или контролерот комуницира со базата преку нив за да ги добие или зачува потребните податоци.
5. **Одговорот се враќа:** Контролерот го враќа одговорот (на пример, JSON).

## Frontend

Frontend е делот од веб-апликацијата кој е видлив и интерактивен за корисниците. Тоа е корисничкиот интерфејс (UI) кој корисниците го користат за да комуницираат со апликацијата. Frontend-от е одговорен за прикажување на податоците добиени од backend-от и за прифаќање на кориснички внес.

Frontend делот на **FormFlow** е изработен со користење на Vue.js 3.

## Што е Vue.js 3?

Vue.js 3 е модерен JavaScript фрејмворк за развој на кориснички интерфејси и апликации од една страна (SPA). Тој е лесен за користење, реактивен, и овозможува брз и флексибилен развој.

Vue.js 3 користи компоненти, што овозможува креирање на повторно употребливи делови на кодот кои лесно се интегрираат и управуваат. Ова го прави развојот поорганизиран и полесен за одржување.

Во овој дел, ќе ги разгледаме основите на креирање на проект со Vue.js 3, дефинирање на рутирање, управување со состојба (state) користејќи Pinia, работа со репозиториуми, Axios за HTTP барања, компоненти, страници, настани.



## Креирање на Vue.js 3 проект

За да креирате нов Vue.js 3 проект, можете да се користи Vue CLI или Vite, се извршуваат командите:

```
npm create vite@latest my-vue-app -- --template vue
```

```
cd my-vue-app
```

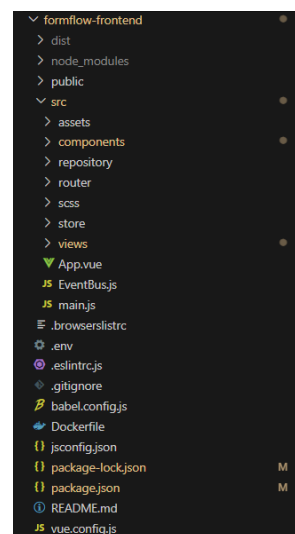
```
npm install
```

```
npm run dev
```

## Структура

По креирање на проектот, се добива основна структура која се состои од повеќе елементи, и ги содржи сите потребни датотеки и документи за стартување на иницијален проект.

- node\_modules - Ги содржи сите зависности на проектот
- public - Статички ресурси
- src - Изворни датотеки
- assets - Статички ресурси како слики
- components - Vue компоненти
- views - Страници на апликацијата
- App.vue - Главна апликациска компонента
- main.js - Влезна точка за апликацијата
- .gitignore - Ги исклучува датотеките од Git
- index.html - Главна HTML датотека
- package.json - Информации за проектот и зависности



## Стартување и пристап до Vue проектот

По креирањето на проектот, потребно е прво да се инсталираат сите зависности, односно да се изврши командата

```
npm install
```

Потоа, за да се стартува проектот, со што истиот ќе може да се пристапи на browser, се извршува

```
npm run dev
```



## Рутирање во Vue

Рутирањето е клучен концепт во развојот на апликации со една страна (SPA) со Vue.js 3. апликациите користат единствена HTML страница и динамички ги вчитуваат и менуваат компонентите, без потреба од целосно освежување на страницата.

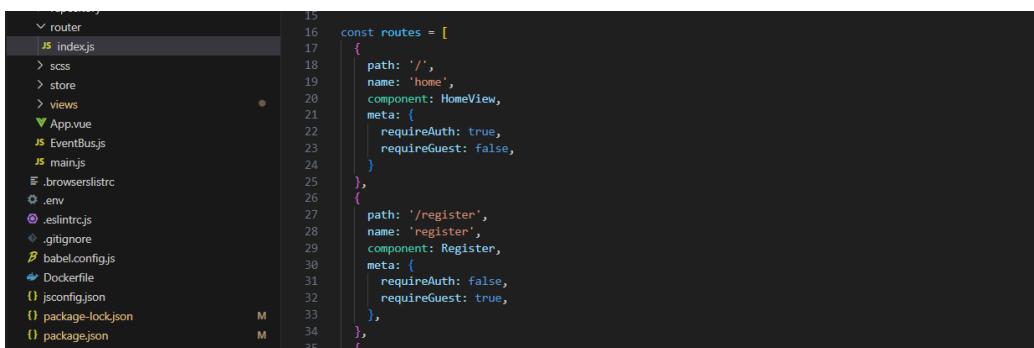
За да се постигне ова, потребно е да се има систем кој управува со различните патеки (URL адреси) во рамките на апликацијата. Рутирањето овозможува навигација помеѓу различни компоненти или "страници" на апликацијата, користејќи го притоа истиот HTML документ.

Пример, во апликација како FormFlow, токму рутирањето овозможува имање на различни страници за прикажување на сите форми, преглед на поединечни одговори, менаџирање на проекти итн.

Во Vue.js 3, рутирањето се имплементира со помош на **Vue Router**, кој се инсталира со помош на командата:

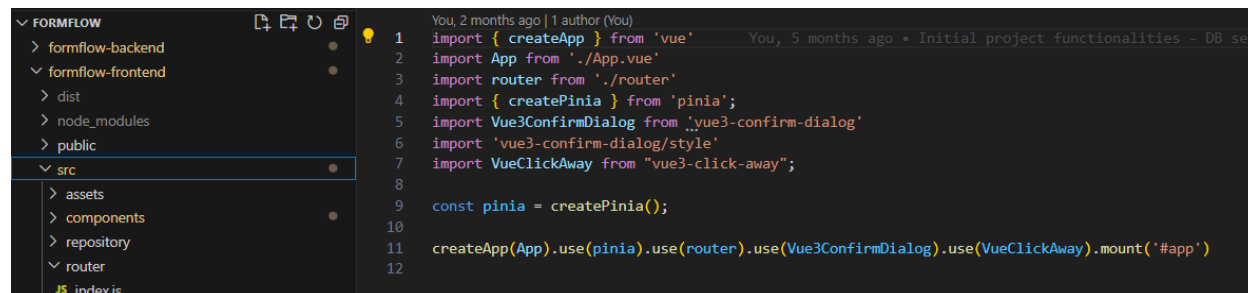
```
npm install vue-router
```

Потоа, се креира посебна датотека за дефинирање на сите рути, најчесто `src/router/index.js`. каде се дефинираат сите рути кои апликацијата ќе ги користи.



Слика 12 - Дефиниција на рути

По дефинирањето на рутерот, потребно е истиот да се интегрира во проектот. Во `main.js` датотеката. И потоа со `router-link` се поставуваат линковите за рутирање.



Слика 13 – Интеграција на vue router



```
24
25
26     <li class="nav-item">
27       <router-link to="/submissions" class="nav-link" exact>
28         
29         <span>Submissions</span>
30       </router-link>
31     </li>
32
33     <li class="nav-item">
34       <router-link to="/tags" class="nav-link" exact>
35         
36         <span>Tags</span>
37       </router-link>
38     </li>
```

Слика 14 – Употреба на vue router

## Управување со состојба во Vue

Управување со состојба (state management) е процес на следење и управување со податоците што ги користи апликацијата. Во веб-апликациите, "состојба" се сите информации кои апликацијата ги чува и менува додека корисникот ја користи, како што се внесените податоци, статусот на формите, активниот корисник, и слично.

Управување со состојба е потребно за да се осигура дека сите делови од апликацијата имаат точни и актуелни податоци. На пример, ако корисникот се логира, состојбата ја чува информацијата за логираниот корисник и ја користи за да одлучи кои податоци и функционалности се достапни за него.

## Pinia

Во Vue.js 3, Pinia е библиотека која помага да се управува со оваа состојба на еден централен и организиран начин, што го прави кодот полесен за разбирање и одржување.

За инсталирање на Pinia се користи командата

```
npm install pinia
```

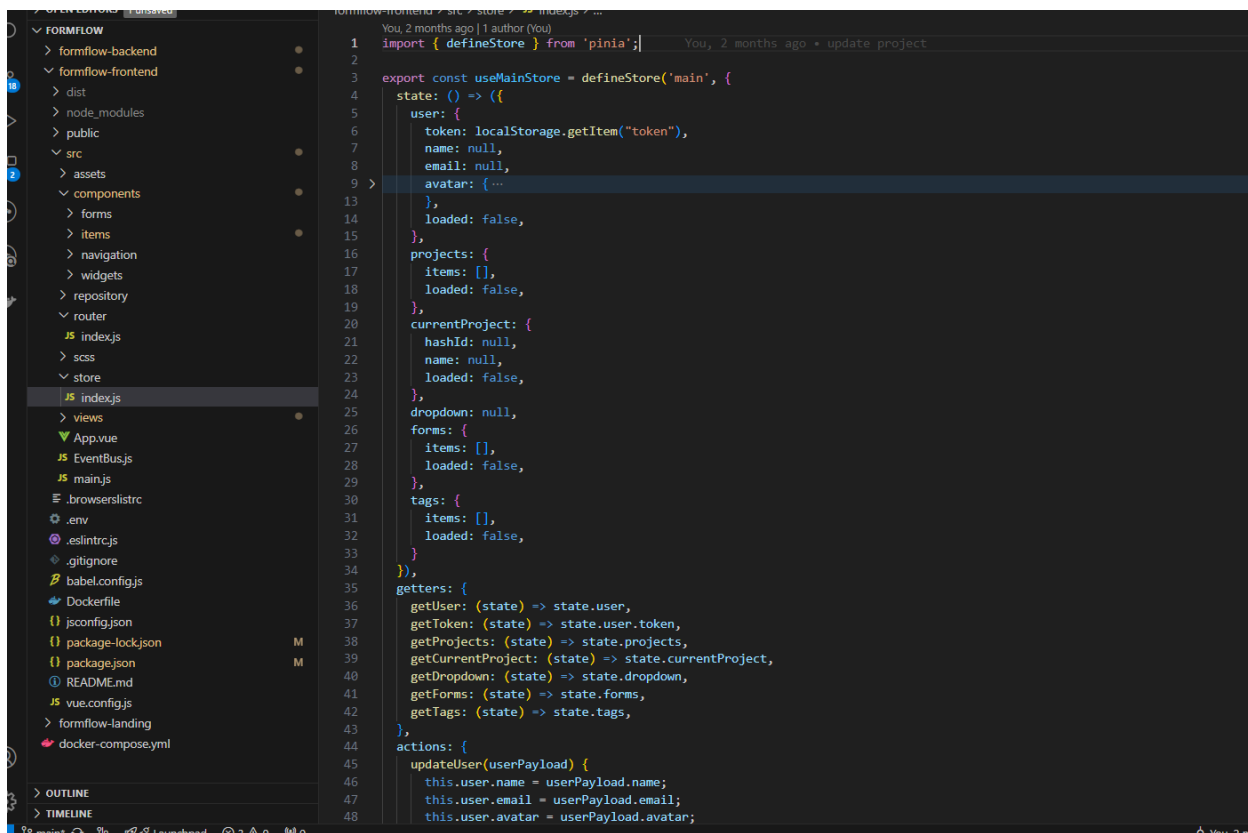
Откако ќе се инсталира, потребно е да се направи интеграција, слично како во примерот погоре во main.js документот.

По интеграцијата се креира Store со Pinia, каде што се дефинираат:

- **state:** Дефинира објект кој ги содржи сите податоци (состојби) кои ќе бидат користени во Store-от
- **getters:** Пресметани вредности кои овозможуваат да се пристапат податоците
- **actions:** Методите кои овозможуваат да се менува состојбата

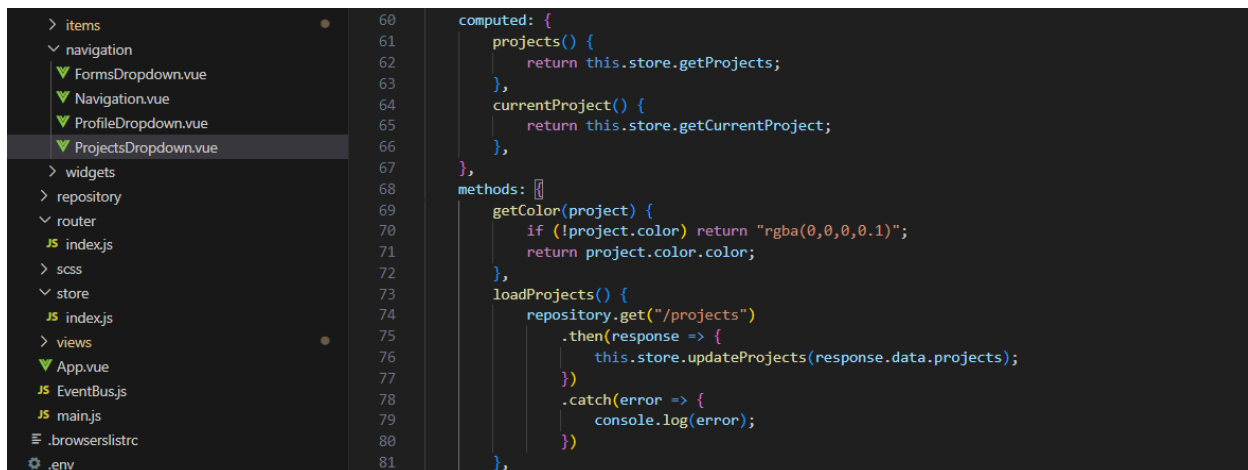


## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО



Слика 15 – Store во Vue

Откако ќе биде дефиниран, истиот може да се вклучи во компонентите и да се користи со повикување на акциите од него.



Слика 16 – Употреба на store





## Axios

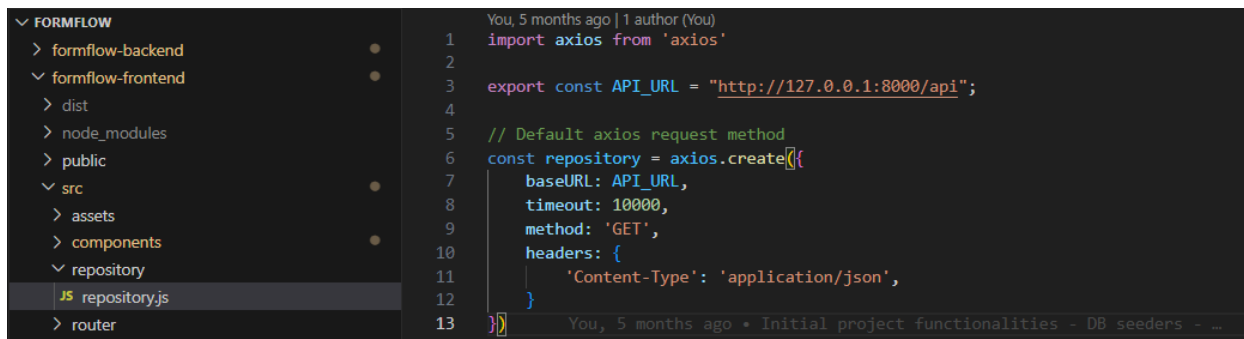
Axios е популарна JavaScript библиотека што се користи за правење HTTP барања од клиентската страна (фронт-енд) до серверската страна (бекенд). Со други зборови, Axios овозможува комуникација помеѓу фронт-ендот на апликацијата и серверот, каде што се наоѓа бекендот.

Ова е многу важно за веб-апликации, каде што корисниците внесуваат податоци и сакаат да ги испратат до серверот за обработка и чување, или пак сакаат да добијат информации од серверот за прикажување.

За да се користи axios потребно е најпрво да се инсталира со командата:

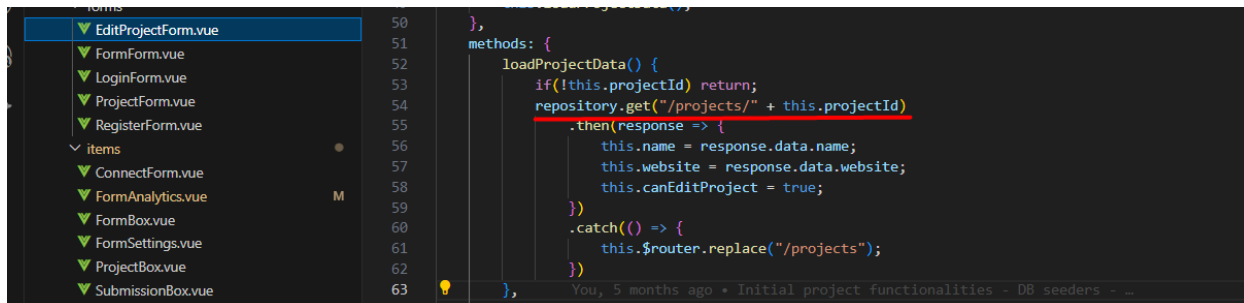
```
npm install axios
```

Откако ќе се инсталира, потребно е да се конфигурира во проектот. Овде се дефинираат основните параметри како API\_URL со цел да не мора на секој повик да се пишува целиосниот url.

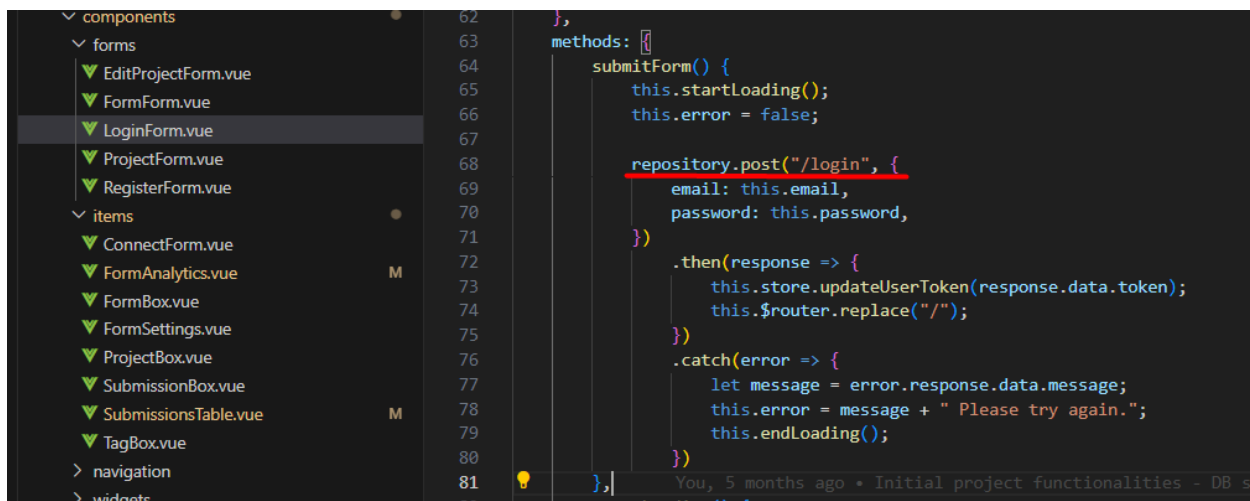


Слика 17 – Конфигурација на axios

Откако е завршена конфигурацијата, може да се испраќаат барања до бекендот и понатаму да се обработуваат добиените одговори.



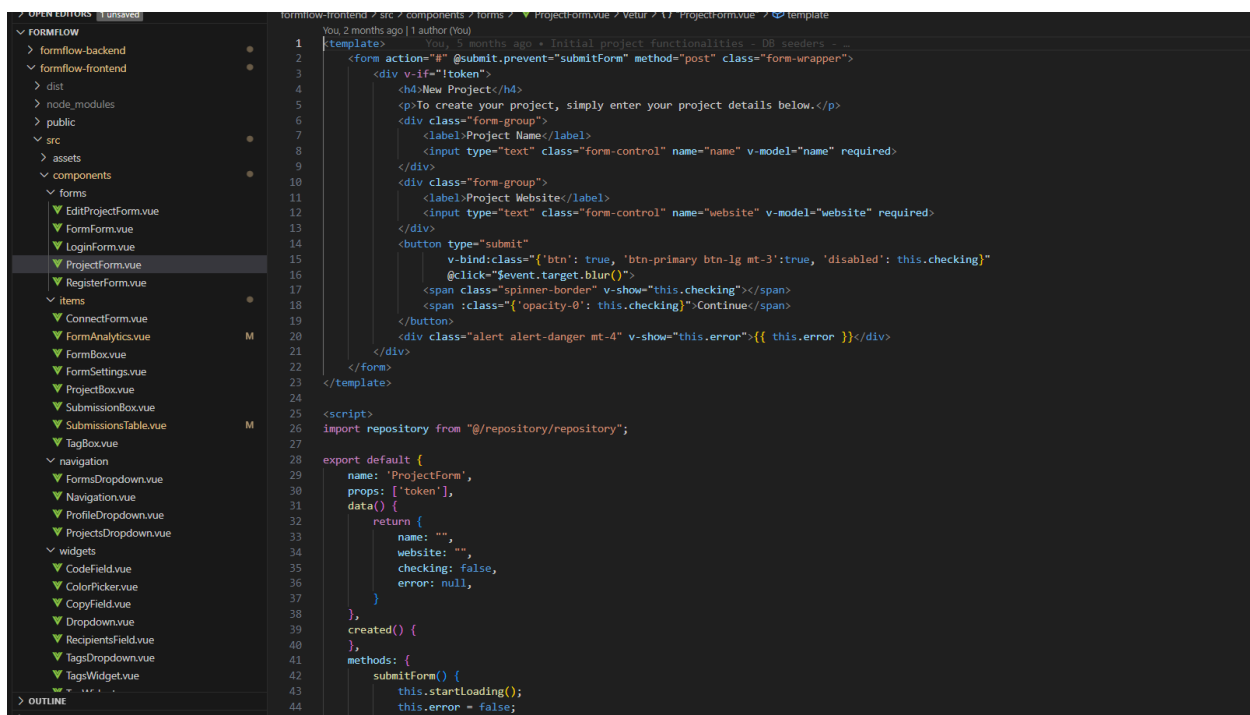
Слика 18 – Употреба на axios за правење на повици за читање до бекенд



Слика 19 – Употреба на axios за правење на повици за запишување до бекенд

## Компоненти

Компоненти се затворен дел од код кој може да се реупотребува на различни делови низ апликацијата. Се состои од html, css и javascript кои се потребни за нејзино функционирање. Како компоненти може да се креираат најразлични елементи како форми, табели, копчиња, па дури и цели страници.



Слика 20 – Пример компонента



## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

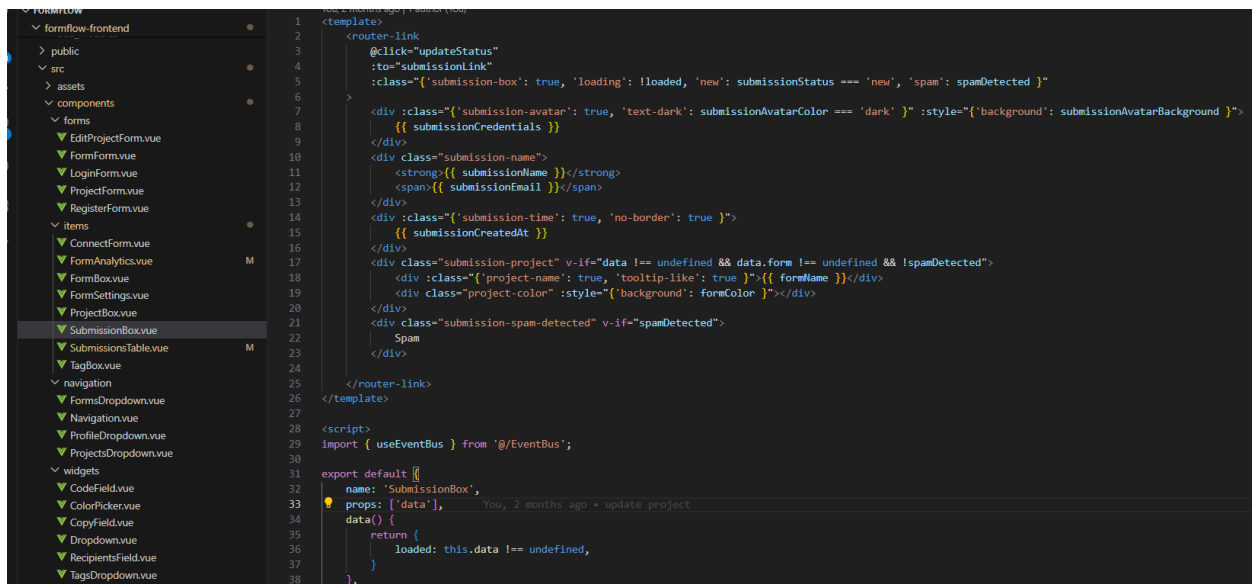
Во примеров е прикажана компонента за форма за креирање проект насловена ProjectForm, која понатаму е интегрирана во одредена страница и се прикажува според одредени услови. Компонентава целосно се справува со процесот на додавање на нов проект во апликацијата.

Во дел од компонентата во script областа може да се забележат повеќе елементи како name, props, data, methods итн.

### Props

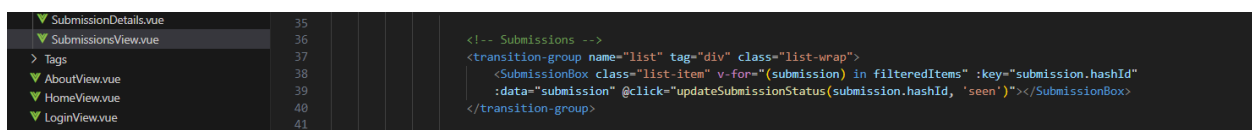
Props се кратенки за “properties” и се користат за пренесување на податоци од родител компонентата до компонентата дете. Ова овозможува компонентите да бидат пофлексибилни и да се употребуваат со различни вредности.

За комуникација пак помеѓу компоненти кои не се во релација родител – дете е искористен event bus. Преку него се емитираат настани и се реагира за соодветни настани исто така. Ова овозможува повеќе компоненти да реагираат на исти настани, и да споделуваат податоци без да се испраќаат како props.



Слика 21 – Употреба на event bus и props

Во примеров како prop е испратен параметарот data, кој содржи податоци за еден одговор на форма. Тоа е искористено при листање на сите одговори, така што за секој одговор се користи истата компонента, но со различни податоци соодветно кои се испраќаат како prop со :data="submission".



Слика 22 – Испраќање на параметар како prop



## Методи / Акции

Методи во Vue.js се функции кои се дефинираат во скрипт делот на компонентата и може да се користат за обработка на податоци или за манипулирање со состојба. Тие обично се користат како реакција на настани (events), како што се кликови на копчиња или испраќање на форми.

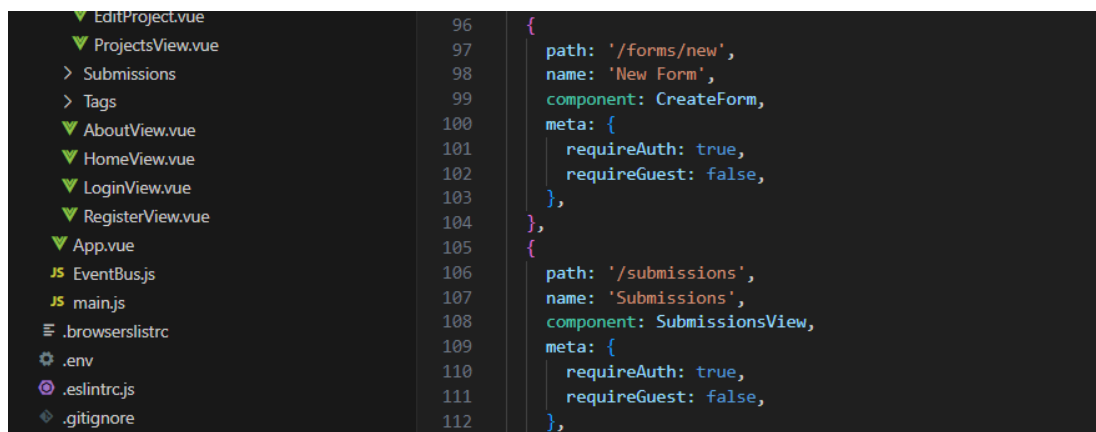


Слика 23 – Приказ на дефинирани методи

Во примеров се неколку акции како читање и листање на таговите, додавање на одреден таг за одреден одговор, справување со клик при додавање на таг.

## Страници

Страниците се специјални типови на компоненти кои претставуваат цели делови од апликацијата, обично врзани за рути (routes). На пример, во апликацијата имаме страници за почетен изглед, регистрација, најава, проекти, форми, одговори, тагови. Страниците се компоненти кои се вчитуваат врз основа на рутата дефинирана во Vue Router.



Слика 24 – Страници поврзани со рути



## Преглед на апликацијата FormFlow

### Креирање профил и најава

Апликацијата нуди едноставни и интуитивно дизајнирани екрани за креирање на корисички профил и најава.



#### Register

Enter your details below to continue

Your Name

Monika Todorova

Your Email

monikatodorova17@yahoo.com

Your Password

\*\*\*\*\*

Repeat Password

\*\*\*\*\*

Continue

Already have an account? [Sign In](#)

Слика 25 – Екран за креирање на профил



#### Login

Please enter your credentials to continue

Your Email

monikatodorova17@yahoo.com

Your Password

\*\*\*\*\*

Continue

Do not have an account? [Register](#)

Слика 26 – Екран за најава



# ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

## Иницијален проект и пример форма

Со креирање на профилот на корисникот се креираат и иницијален проект, кој се поставува како стандарден / селектиран. За истиот проект се креираат и две форми. Една од формите (Test form) содржи и пример одговори, со цел корисникот лесно да добие увид во тоа како неговите податоци ќе се прикажуваат во апликацијата. Другата форма е празна, со спремен код за интеграција, по што ќе може да добие одговори за неа.

The dashboard displays project statistics and a list of forms. The left sidebar shows navigation options: My Project, Dashboard, Forms, Submissions, and Tags. The main content area includes 'Project Statistics' with four cards: Submissions Today (1), Last 7 days (5), Last 30 days (10), and Total Submission (10). Below this is 'Your Forms' section with two cards: 'Test Form (demo)' with 10 submissions and 'Contact Form' with 0 submissions. Each card has a 'View Form' button. The user profile at the bottom left shows 'Monika Todorova' with email 'monikatodorova17@yahoo.com'.

Слика 27 – Иницијален екран

The 'Test Form (demo)' page shows a table of all submissions. The table has columns: Submission, Name, Email, Phone, City, Country, and Status. There are 10 submissions listed. The page also includes a 'Submissions' button, 'Connect', 'Statistics', and 'Settings' links. A dropdown menu for 'All Submissions' and an 'Export submissions' button are present. The user profile at the bottom left shows 'Monika Todorova' with email 'monikatodorova17@yahoo.com'.

Submission	Name	Email	Phone	City	Country	Status	
AS	Alysson Stollenberg	Eise Vandervort	king75@vonrueden.com	+1-432-216-5543	Port Patienceborough	Taiwan	Active
CV	Cecile Veum	Joel Bayer	peichmann@mraz.com	469.564.6620	Alisonfort	Panama	Active
OS	Ocie Stokes	Savanna Stokes	pherman@gmail.com	442.790.8955	North Daniella	Philippines	Active
LC	Laurianne Crist	Alexanne Nader	grant.pouros@johns.biz	+1 (541) 759-3099	West Emmanuel	San Marino	Active
JO	Jerrell Olson	Delbert Heller	sprohaska@marquardt.com	224-203-2883	Port Jayson	Aruba	Active
PC	Prof. Cassandra Hirt...	Freda Jones	eshanahan@yahoo.com	+1 (928) 558-4600	Breitenbergburgh	Libyan Arab Jamahiriya	Active
BH	Bridgette Haag	Dr. Martine Jerde	hamill.torey@hotmail.com	+1-737-289-8088	South Edwardo	Turks and Caicos Islands	Active

Слика 28 – Приказ на тест формата



## Креирање и менаџирање на проекти

Корисникот покрај иницијалниот проект може да креира и други проекти. Проектот всушност претставува опција за групирање на формите.

Идејата за групирање по проект би се користела на пример во случај кога корисникот има две различни веб страни во кои соодветно има форми, би можел да креира различен проект за секоја од страните, и во секој од проектите, да ги креира и конектира соодветните форми.

Креирањето на проект се одвива со внесување на информации во едноставна форма. По успешно креирање на новиот проект, тој се појавува во менито од проекти, како и во листата проекти наменета за нивно менаџирање. Секој од проектите може соодветно да се едитира и / или избрише.

Доколку селектираме некој од излистаните проекти во менито од навигацијата, истиот ќе се постави како главен / селектиран проект, и останатите податоци ќе се ажурираат соодветно на тоа.

**formflow**

My Project ▾

Dashboard

Forms ▾

Submissions

Tags

### New Project

To create your project, simply enter your project details below.

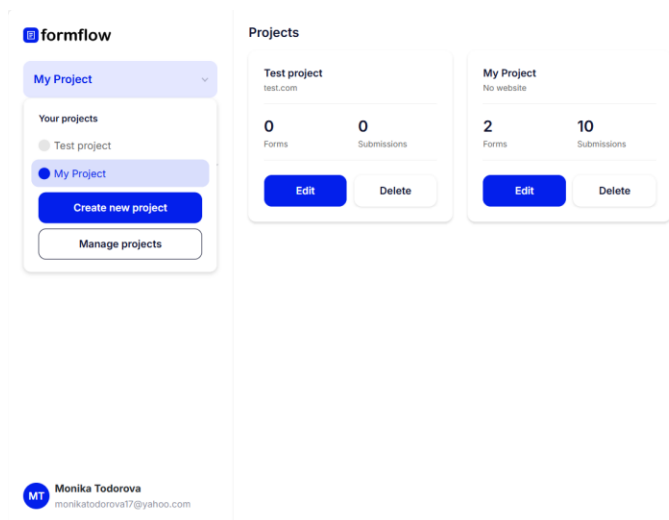
Project Name  
Test project

Project Website  
test.com

**Continue**

**MT** Monika Todorova  
monikatodorova17@yahoo.com

Слика 29 – Екран за креирање на нов проект

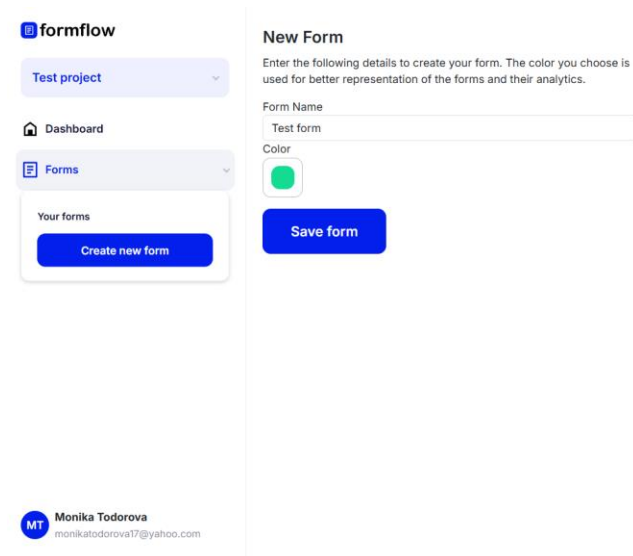


Слика 30 – Екран за приказ на креираните проекти

### Креирање и менаџирање на форми

Покрај креирањето на проекти, корисникот може да креира и нови форми. Формата за креирање на нови форми е достапна преку избирање на опцијата Create new form во менито на форми во навигацијата.

За да се креира една форма, потребно е само да се внесе нејзино име и може да се селектира боја, со единствена цел полесно распознавање на формата и понатаму одговорите поврзани со неа.



Слика 31 – Екран за креирање на нова форма





## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

formflow

My Project

Dashboard

Forms

Submissions

Tags

MT Monika Todorova  
monikatodorova17@yahoo.com

### Project Statistics

1 Submissions Today

5 Last 7 days

10 Last 30 days

10 Total Submission

### Your Forms

Test Form (demo)

10 Submissions

View Form

Contact Form

0 Submissions

View Form

Слика 32 – Екран за приказ на креираните форми

За новокреираната форма да може да добива одговори поврзани со неа во апликацијата, потребно е истата да се конектира. За тоа е креиран посебен екран кој ги содржи инструкциите за таа акција. Имено во моменталната верзија на апликацијата, за формата да се конектира потребно е во action полето на формата од веб страната на корисникот да се внесе дадениот endpoint. Со тоа, формата ќе биде поврзана во апликацијата и одговорите од неа ќе се зачувуваат и прикажуваат овде.

Дополнително на екранов е дадена пример форма со соодветно поврзан ендпоинт која може да се искористи за креирање на минимална функционална онлајн форма.

formflow

Test project

Dashboard

Forms

Submissions

Tags

MT Monika Todorova  
monikatodorova17@yahoo.com

### Test form

Submissions Connect Statistics Settings

Connect your website form to start receiving submissions for your project.

To set up your form correctly, please use this link as the **action** attribute and set the **method** attribute to **POST**. Additionally, ensure that all fields in your form have a **name** attribute, and that at least one field is named **email**.

#### Form Endpoint:

[http://localhost/FormFlow/formflow-landing/f/for\\_A9kVvDeJ07LYr](http://localhost/FormFlow/formflow-landing/f/for_A9kVvDeJ07LYr)

Copy link

Please note that there is a limit of 20 input fields per form. If you submit more than 20 fields in a single form, only the first 20 fields will be saved.

#### Example Code:

```
<!-- Simple HTML form that accepts Name, Email and Phone number -->
<form action="http://localhost/FormFlow/formflow-landing/f/for_A9kVvDeJ07LYr" method="POST">
  <div class="form-group">
    <label class="form-label" for="name">Your Name</label>
    <input class="form-control" type="text" name="name" id="name" required>
  </div>
  <div class="form-group">
    <label class="form-label" for="email">Your Email</label>
    <input class="form-control" type="email" name="email" id="email" required>
  </div>
  <div class="form-group">
    <label class="form-label" for="phone">Your Phone</label>
    <input class="form-control" type="text" name="phone" id="phone">
  </div>
  <button type="submit" class="btn btn-primary">Submit form</button>
</form>
```

Слика 33 – Екран со инструкции за конектирање на формата



## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

```
<!DOCTYPE html>
<html>
<head>
<title>Test</title>
</head>
<body>

<!-- Simple HTML form that accepts Name, Email and Phone number -->
<form action="http://localhost/FormFlow/formflow-landing/f/for_A9kVDeJ07LYr"
method="POST">
  <div class="form-group">
    <label class="form-label" for="name">Your Name</label>
    <input class="form-control" type="text" name="name" id="name" required>
  </div>
  <div class="form-group">
    <label class="form-label" for="email">Your Email</label>
    <input class="form-control" type="email" name="email" id="email" required>
  </div>
  <div class="form-group">
    <label class="form-label" for="phone">Your Phone</label>
    <input class="form-control" type="text" name="phone" id="phone">
  </div>
  <button type="submit" class="btn btn-primary">Submit form</button>
</form>

</body>
</html>
```

Your Name	Monika Todorova
Your Email	monikatodorova2001@hotmail.com
Your Phone	011222333
<input type="button" value="Submit form"/>	

Слика 34 – Приказ на пример формата

### Прибирање и менаџирање на одговори

Откако корисникот ќе ја поврзе својата онлајн форма со ендпоинтот од апликацијата. Сите одговори кои ќе бидат испратени таму ќе пристигаат во апликацијата. По испраќањето на одговорот, на соодветната страна се покажува приказ за успешно испраќање.

```
<!DOCTYPE html>
<html>
<head>
<title>Test</title>
</head>
<body>

<!-- Simple HTML form that accepts Name, Email and Phone number -->
<form action="http://localhost/FormFlow/formflow-landing/f/for_A9kVDeJ07LYr"
method="POST">
  <div class="form-group">
    <label class="form-label" for="name">Your Name</label>
    <input class="form-control" type="text" name="name" id="name" required>
  </div>
  <div class="form-group">
    <label class="form-label" for="email">Your Email</label>
    <input class="form-control" type="email" name="email" id="email" required>
  </div>
  <div class="form-group">
    <label class="form-label" for="phone">Your Phone</label>
    <input class="form-control" type="text" name="phone" id="phone">
  </div>
  <button type="submit" class="btn btn-primary">Submit form</button>
</form>

</body>
</html>
```

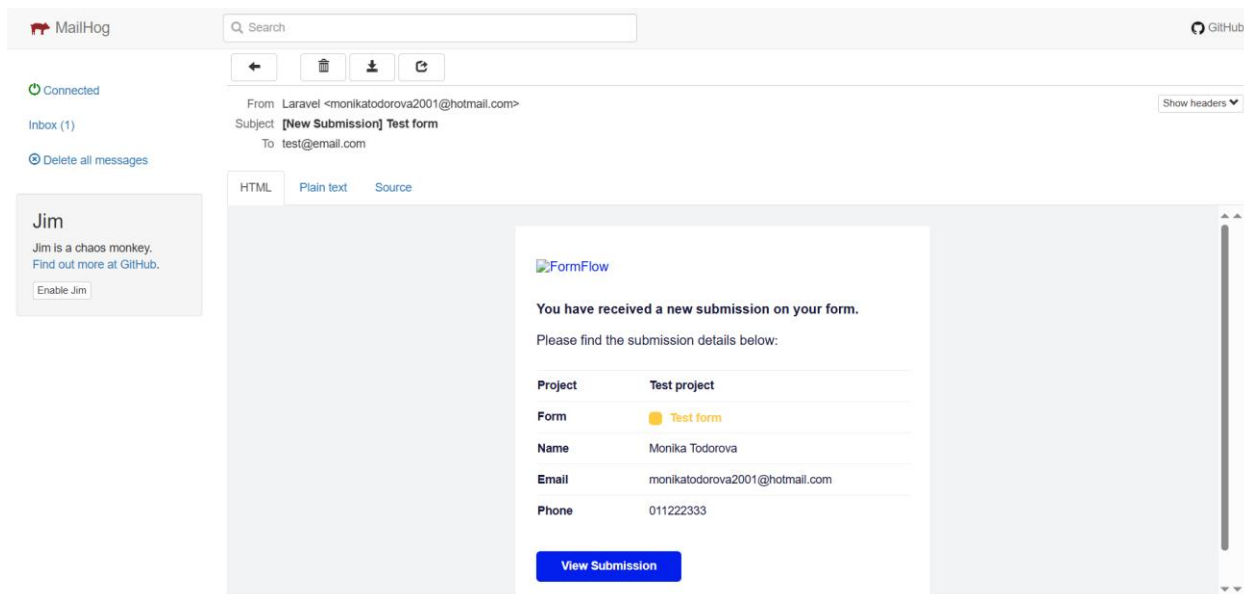
  
**Thank you!**  
Submission has been saved

Слика 35 – Успешно испраќање на одговор преку креираната форма

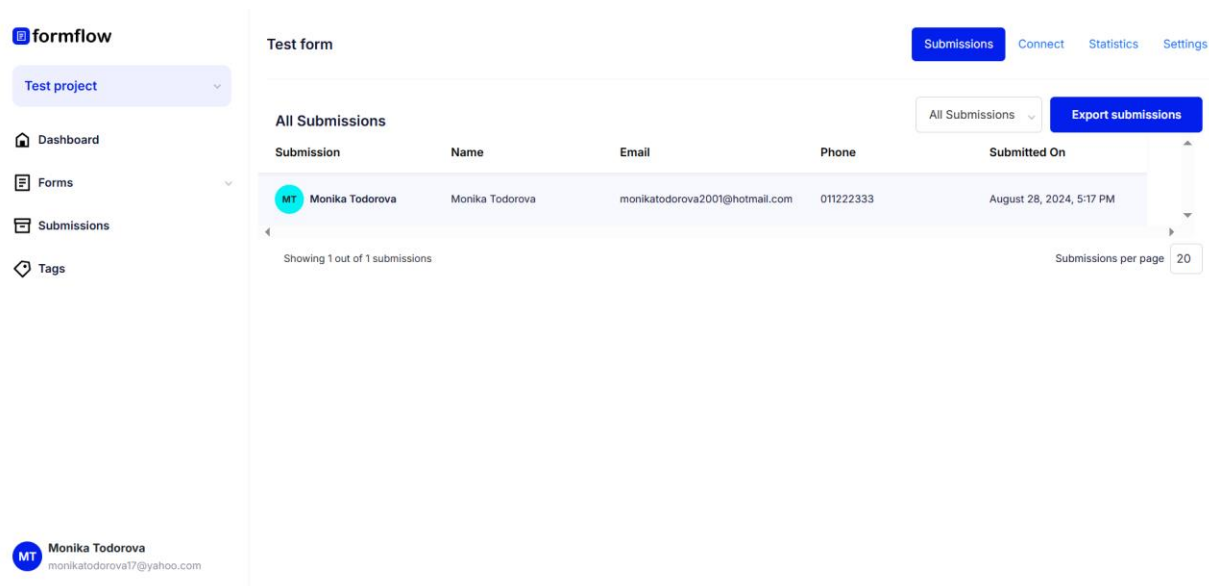


## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Во апликацијата, во главниот екран за формата може веднаш да се види новата порака. Притоа се испраќа и нотификација на емаил за добивањето на новиот одговор од формата. Локално тоа е истестирано со MailHog.



Слика 36 – Емаил известување за новодобиениот одговор

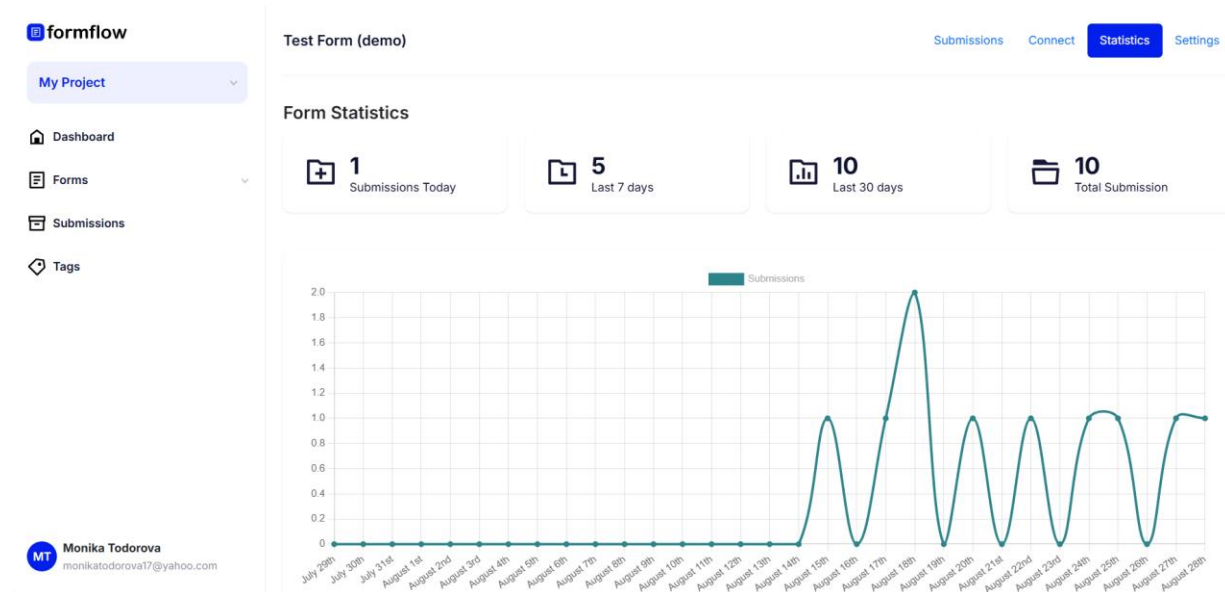


Слика 37 – Приказ на новодобиениот одговор во апликацијата



## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

За секоја од формите е обезбеден дел со аналитика каде што може да се види по текот на претходниот месец, по ниво на денови, колкав е бројот на добиени одговори.



Слика 38 – Приказ на екран со аналитики

### Приказ на одговори и менаџирање со тагови и статус

Освен приказот на одговорите поврзани со една форма, овозможен е и приказ на одговорите поврзани со целиот проект. Во делот на submissions се излистани сите одговори кои се поврзани со некоја од формите во моментално селектираниот проект.

**Project Submissions**

Select tag to filter

User	Email	Time Ago
Alysson Stollenberg	faustino.pagac@hamill.net	10 days ago
Cecile Veum	jones.rogelio@corkery.com	3 days ago
Ocie Stokes	edickinson@schowalter.net	17 hours ago
Laurianne Crist	nhoppe@hotmail.com	7 days ago
Jerrell Olson	oconnell.stan@pollich.com	13 days ago
Prof. Cassandra Hirthe II	joaquin.runoffsdotir@gmail.com	3 hours ago
Bridgette Haag	ritchie.hermina@goodwin.info	10 days ago
Merritt Fritsch	agoodwin@lowe.net	6 days ago
Miller Koelpin	webster.haley@gmail.com	3 days ago
Miss Golda Jast MD		

**Select a submission to view more details**

Discover additional details by clicking on a submission of your choice

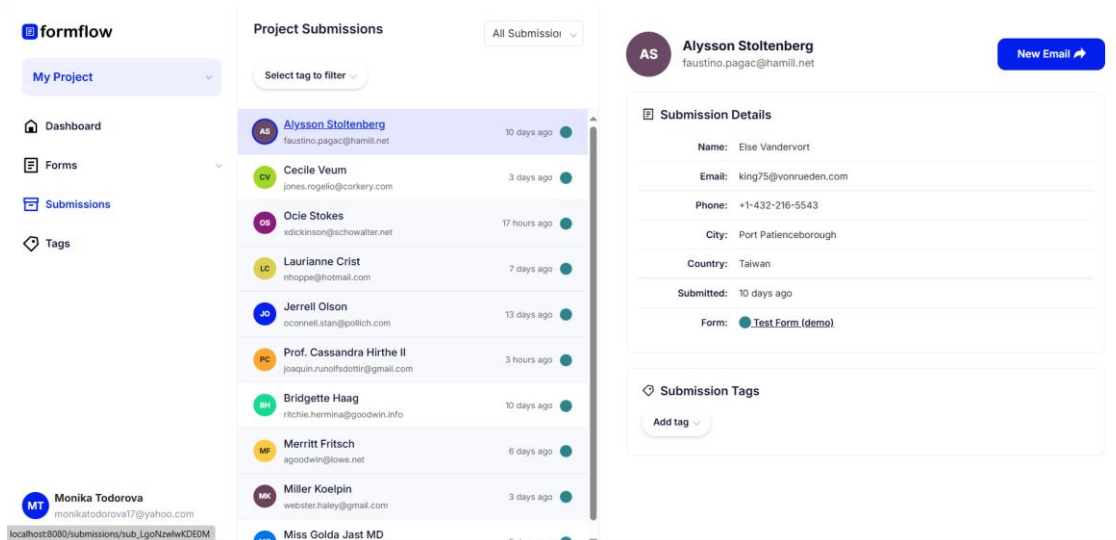
Слика 39 – Приказ на сите одговори од даден проект



## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

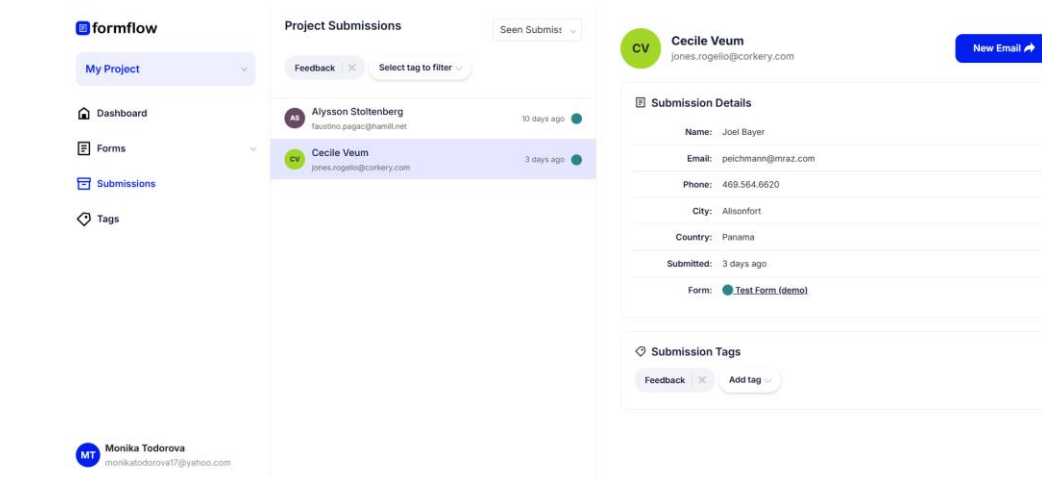
Сите одговори се соодветно означени со бојата на нивната форма, со цел полесно распознавање. Дополнително во зависност од статусот на одговорот, односно дали истиот е виден, позадината на истиот е во соодветна боја.

Со селектирање на некој од одговорите, од десна страна се покажуваат деталите поврзани со него, можност за одговор на истиот што пренасочува со `mailto` линк и опција за додавање на тагови.



Слика 40 – Приказ на делати за одговор

Доколку некој од одговорите го поврземе со тагови, овозможено ни е филтрирање по истите. Односно за првите два одговори од сликата погоре ставаме таг Feedback, што понатаму ни овозможува филтрирање. Филтрирањето дополнително функционира и според статус, односно може да се прикажуваат само новите одговори, само прочитаните одговори или сите заедно.



Слика 41 – Филтрирање со таг



## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Доколку отстранеме некој од таговите во моменталниот приказ, листата автоматски се обновува.

The screenshot shows the Formflow interface. On the left is a sidebar with the 'formflow' logo and navigation links: 'My Project', 'Dashboard', 'Forms', 'Submissions', and 'Tags'. The main area is titled 'Project Submissions' and includes a 'Seen Submissions' dropdown, a 'Feedback' button, and a 'Select tag to filter' dropdown. Below this, a submission by Alysson Stoltenberg is shown. To the right, the 'Submission Details' for Cecile Veum are displayed, including fields for Name, Email, Phone, City, Country, Submitted date, and Form. A 'New Email' button is also present. At the bottom, there is a 'Submission Tags' section with an 'Add tag' button.

Слика 42 – Управување со таговите при филтрирање

### Генерирање и преземање на документ

При приказот на сите одговори од дадена форма, овозможена е акција за презмање на одговорите во документ во csv формат.

The screenshot shows the Formflow interface with a list of submissions. The top bar includes 'Submissions', 'Connect', 'Statistics', and 'Settings'. The main area is titled 'Test Form (demo)' and shows a table of 'All Submissions'. The table has columns for Submission, Name, Email, Phone, City, Country, and Status. The 'Export submissions' button is highlighted with a red box. Below the table, it says 'Showing 10 out of 10 submissions' and 'Submissions per page 20'.

Submission	Name	Email	Phone	City	Country	Status	
AS	Alysson Stoltenberg	Eise Vandervort	king75@vonrueden.com	+1-432-216-5543	Port Patienceborough	Taiwan	At
CV	Cecile Veum	Joel Bayer	peichmann@mraz.com	469.564.6620	Alisonfort	Panama	At
OS	Ocie Stokes	Savanna Stokes	pherman@gmail.com	442.790.8955	North Daniella	Philippines	At
LC	Laurianne Crist	Alexanne Nader	grant.pouros@johns.biz	+1 (541) 759-3099	West Emmanuel	San Marino	At
JO	Jerrell Olson	Delbert Heller	sprohaska@marquardt.com	224-203-2883	Port Jayson	Aruba	At
PC	Prof. Cassandra Hirt...	Freedra Jones	eshanahan@yahoo.com	+1 (928) 558-4600	Breitenbergburgh	Libyan Arab Jamahiriya	At
BH	Bridgette Haag	Dr. Martine Jerde	hamill.torey@hotmail.com	+1-737-289-8088	South Edwardo	Turks and Caicos Islands	At

Слика 43 – Поле за преземање на документ



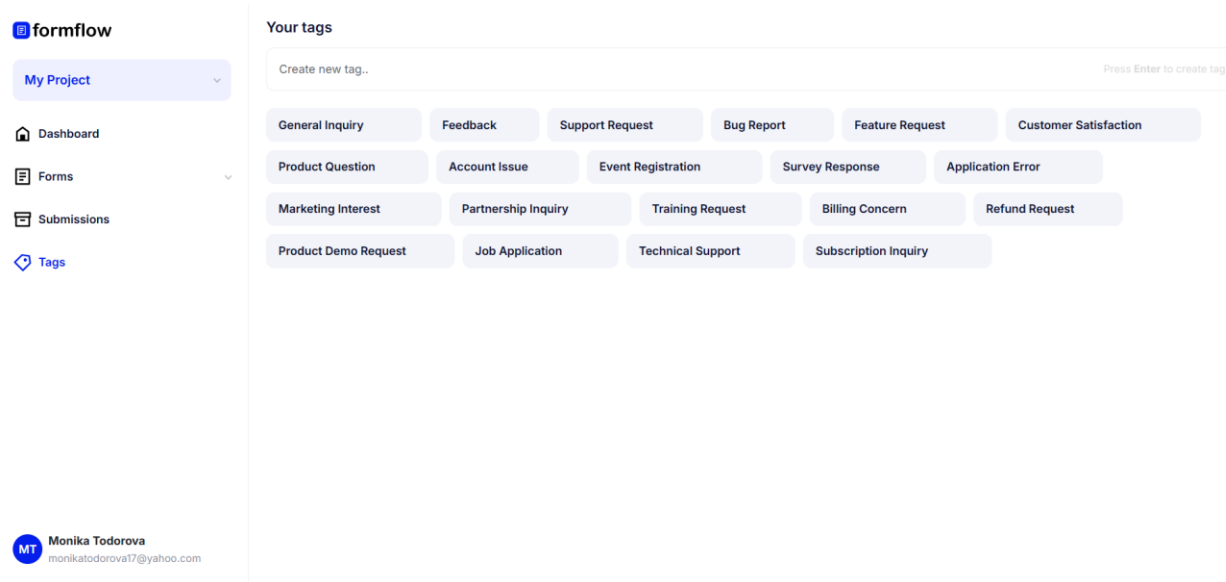
## ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	name	email	phone	city	country									
2	Miss Dayne pfannerstil		-9128	Wildermar	Korea									
3	Mr. Webster oankundin		-8377	East Cryste	Senegal									
4	Donnell Wi asa.roob@	(312) 809-		New Esme	Israel									
5	Dr. Martine hamill.tore		-9113	South Edw.	Turks and Caicos Islands									
6	Freedra Joneshanahar	+1 (928) 55	Breitenber	Libyan Arab	Jamahiriya									
7	Delbert He sprohaska	224-203-2		Port Jaysor	Aruba									
8	Alexanne N grant.pour	+1 (541) 75	West Emm	San Marino										
9	Savanna St pherman@	442.790.8		North Dani	Philippines									
10	Joel Bayer peichman	469.564.6		Alisonfort	Panama									
11	Else Vandé king75@vc		-6190	Port Patier	Taiwan									
12														
13														

Слика 44 – Приказ на преземаниот документ

### Креирање и менаџирање на тагови

Кога станува збор за таговите, нивното менаџирање се одвива во посебна страна, каде може да се додаваат и бришат тагови на ниво на корисник. Улогата на таговите не е ништо друго освен едноставен начин за групирање и означување на добиените одговори.



Слика 45 – Екран за додавање и менаџирање на тагови



## **Заклучок**

Оваа апликација ќе овозможи на сите лица кои имаат потреба од прибирање на одговори и иницизирање на онлајн комуникација преку онлајн форми, истото да го направат на едноставен и интуитивен начин.

Изработката на апликацијата со користење на технологии како Laravel, Vue, MySQL ги опфаќа сите изучени единици според предметите Имплементација на системи со слободен и отворен код и Напреден веб дизајн. Покриени се сите основни концепти за работа со наведените технологии, и притоа е креирана целосно функционална апликација.