



Универзитет „Св. Кирил и Методиј“ во Скопје
**ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И
КОМПЈУТЕРСКО ИНЖЕНЕРСТВО**

Предмет: Напреден веб дизајн

Проектна работа
Систем за управување со автосервис
(Car Service Manager)

Изработиле:

Стефан Колов 223226

Андреј Јакимовски 226086

Професор: Проф. Д-р Бобан Јоксимоски

Мај 2025

Содржина

1. Вовед	2
2. Структура на проектот.....	3
2.1 assets	3
2.2 components	3
2.3 router.....	4
2.4 views.....	4
2.4.1 CarManagement.vue	4
2.4.2 EmployeeManagement.vue.....	6
2.4.3 ServiceManagement.vue.....	7
3. Компонентна организација	8
3.1 Архитектура и организација на апликацијата	8
4. Функционалности на апликацијата.....	10
Управување со возила (CarManagement.vue)	10
Управување со вработени (EmployeeManagement.vue).....	11
Управување со сервисирања (ServiceManagement.vue)	11
5. Технолошки стек.....	13
6. Начини на користење на апликацијата.....	14
6.1 Регистрација и управување со возила	14
6.2 Регистрација и управување со клиенти.....	14
6.3 Закажување и евиденција на сервисни интервенции	14
6.4 Управување со вработени и доделување задачи	14
6.5 Приказ и анализа на историја на сервисирања	14
6.6 Измени и бришење на податоци	14
7. Интерфејс на корисникот	15
7.1 Структура на интерфејс (UI)	15
7.2 Оптимизација на искуството на корисникот (UX)	15
8. Заклучок	16

1. Вовед

Во рамки на оваа проектна работа е изработена веб апликација на тема **Car Service Manager**, користејќи ја современата JavaScript рамка **Vue.js**, која е позната по својата ефикасност, флексибилност и поддршка за компонентно програмирање. Целта на апликацијата е да се овозможи ефикасно, прегледно и централизирано управување со сите аспекти на процесот на сервисирање на возила, вклучувајќи: регистрирање на клиенти, информации за возила, закажување и евидентирање на сервисни интервенции, како и следење на комплетна историја на поправки за секое возило.

Преку едноставен и интуитивен интерфејс, апликацијата овозможува брза навигација и пристап до податоците, што значително ја олеснува работата на вработените во сервисот и на администраторите. Податоците се организирани по ентитети (клиенти, возила, вработени, сервиси), а секој ентитет има соодветни форми за внесување, измена и бришење, со цел да се обезбеди конзистентност и лесно одржување на информациите.

Апликацијата демонстрира примена на повеќе напредни концепти од областа на веб-развојот. Преку компонентно програмирање, секој дел од апликацијата е модуларен и независен, што овозможува повторна искористливост и полесна одржливост на кодот. Реактивноста овозможува автоматско ажурирање на прикажаните податоци при секоја промена, без потреба од освежување на страницата. Дополнително, комуникацијата со надворешен API или база на податоци овозможува зачувување и повлекување на реални податоци, што ја прави апликацијата функционална и подготвена за интеграција во реални деловни процеси.

Овој проект има за цел не само да демонстрира техничка изведба, туку и да прикаже разбирање на начелата на современ веб-дизајн, корисничко искуство (UX), како и основни концепти на архитектурата на веб апликации. На тој начин, **Car Service Manager** претставува корисна, практична алатка, но и силен пример за успешна имплементација на фронтенд технологија во решавање на конкретен проблем од секојдневната пракса.

2. Структура на проектот

Проектот **Car Service Manager** е изграден со употреба на **Vue.js**, популарен JavaScript фрејмворк кој овозможува изработка на динамични кориснички интерфејси. Vue.js е избран заради неговата леснотија за учење и употреба, како и заради неговата модуларност и ефикасност. Апликацијата следи стандардната структура на Vue.js проекти која ја прави апликацијата модуларна, флексибилна и лесна за одржување и проширување.

2.1 assets

Директориумот `assets/` е резервиран за **статички ресурси** кои се користат во апликацијата, како што се **слики, стилови, иконки** и други **мултимедијални фајлови**. Овде се чуваат фајловите што не подлежат на преработка или обработка од страна на апликацијата, како што се логотипи, позадински слики, икони, шеми и други графички елементи кои се користат низ апликацијата. Со одвојување на овие ресурси, се обезбедува чистота на кодот и подобрување на управувањето со датотеки.

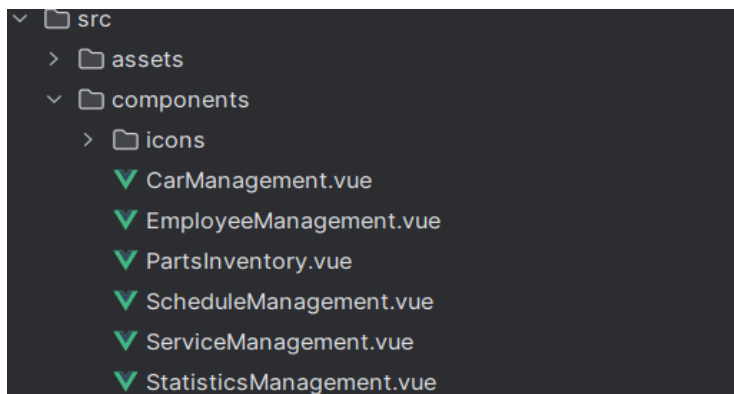


Слика 2.1 Структура на датотеки во директориумот `assets`

2.2 components

Директориумот `components/` ги содржи **Vue компонентите** кои се користат за креирање на независни UI елементи. Овие компоненти се лесни за повторна употреба и можат да бидат користени на повеќе места низ апликацијата. Некои примери на компоненти кои може да се најдат во овој дел се **форми за внесување на податоци, менување на податоци, картички за приказ на информации** и други UI елементи кои помагаат да се изгради целосен интерфејс.

Главната предност на употребата на Vue компоненти е што тие се независни и можат да се изолираат и тестираат, што ја зголемува флексибилноста и стабилноста на кодот. На пример, компонентите за управување со возила и клиенти можат да се имплементираат како посебни компоненти што ќе се повторуваат во различни делови од апликацијата.



Слика 2.2 Структура на датотеки во директориумот components

2.3 router

Во директориумот router/ се наоѓа главниот **JavaScript фајл** кој ја конфигурира навигацијата во апликацијата, односно **index.js**. Овој фајл ја управува целокупната рутирање на страниците и овозможува **динамично преминување** помеѓу различни погледи (views) во апликацијата, без потреба од рефреширање на страницата. Со помош на Vue Router, апликацијата може да се движи низ различни компоненти и страници со безпрекорно корисничко искуство, што ја прави апликацијата поефикасна.

Фајлот index.js дефинира како ќе се навигира помеѓу страниците како што се **страниците за клиенти, термини за сервисирање** и други важни делови од апликацијата. Vue Router овозможува и динамички патеки, со што е можно да се пренесуваат податоци низ различни делови од апликацијата, како што се **податоци за возила или клиенти**.



Слика 2.3 Структура на датотеки во директориумот router

2.4 views

Директориумот views/ претставува еден од најважните делови од Vue апликацијата, бидејќи ги содржи **главните визуелни страници (погледи)** кои се прикажуваат на корисниците при навигација низ различните делови од системот. За разлика од компонентите кои се мали, повторно искористливи елементи, views/ ги дефинира **целосните страници** што одговараат на одредена рута, односно функционалност на апликацијата. Во проектот Car Service Manager, фокусот е ставен на три клучни страници:

2.4.1 CarManagement.vue

Компонентата CarManagement.vue, која се наоѓа во директориумот views/, претставува една од клучните кориснички интерфејсни страници во апликацијата *Car Service Manager*. Како што сугерира и самото име, оваа компонента е специјално наменета за управување со автомобилите кои се внесени во системот од

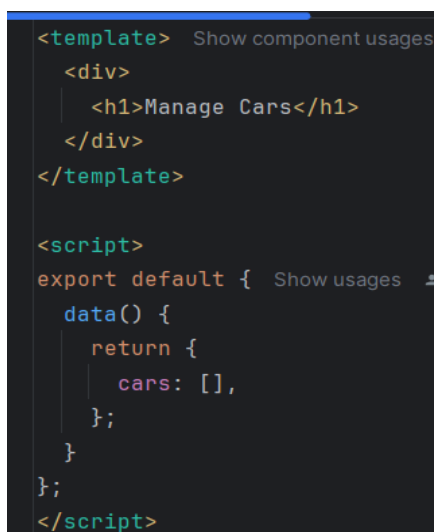
страна на корисниците (клиенти). Таа обезбедува централизирана точка за пристап и манипулација со сите информации поврзани со возилата.

Во својата почетна и основна верзија, компонентата вклучува минимален HTML темплејт кој прикажува заглавие со текстот **"Manage Cars"**, со што визуелно го дефинира делот од апликацијата каде ќе се врши целосна контрола врз возилата. Во скриптскиот дел (`<script>`), дефинирана е `data()` функција која враќа објект со празна низа `cars`. Оваа низа претставува основната структура во која ќе се чуваат сите информации за автомобилите кои ќе бидат внесени, прикажани или изменети во системот.

Во понатамошната надградба на оваа компонента, ќе се имплементираат следниве функционалности:

- **Додавање ново возило:** Преку интерактивна форма, корисниците ќе можат да внесат ново возило во базата на податоци, при што ќе се собираат информации како што се марка, модел, година на производство, број на шасија и други релевантни податоци.
- **Приказ на сите возила:** Ќе се прикаже табела или листа со сите внесени возила, која ќе овозможува лесен преглед и навигација низ постоечките записи.
- **Ажурирање/уредување на возило:** Корисниците ќе имаат можност да ги изменат податоците за веќе внесени возила, на пример, доколку има грешка или е потребна промена на сопственикот или други атрибути.
- **Бришење возило:** Компонентата ќе овозможи бришење на возила од системот кога тие повеќе не се релевантни или се внесени по грешка.
- **Поврзување со клиенти и интервенции:** Како дел од целокупната логика на апликацијата, ќе се овозможи поврзување на секое возило со неговиот сопственик (клиент), како и евиденција на сите сервисни интервенции што се извршени над него.

Со други зборови, `CarManagement.vue` ќе претставува централно место каде што ќе се реализираат сите активности поврзани со управување на автомобилите. Оваа компонента ќе биде тесно интегрирана со другите делови од апликацијата, како што се управување со клиенти и управување со сервисни услуги, со што ќе придонесе за создавање на комплетна, функционална и кориснички ориентирана системска целина.



```
<template> Show component usages
  <div>
    <h1>Manage Cars</h1>
  </div>
</template>

<script>
export default { Show usages
  data() {
    return {
      cars: [],
    };
  }
};
</script>
```

Слика 2.4.1 Изглед на Vue компонента за управување со автомобили

2.4.2 EmployeeManagement.vue

Компонентата EmployeeManagement.vue, лоцирана во директориумот views/, е одговорна за управување со човечките ресурси, односно со вработените во рамките на апликацијата *Car Service Manager*. Таа претставува една од најзначајните функционални целини, бидејќи овозможува прецизно следење, ажурирање и администрација на податоците за персоналот кој работи во сервисот.

Во нејзината почетна верзија, компонентата содржи минимален темплејт со заглавие „Manage Employees“, со што се означува дека оваа страница е посветена на менаџирање со вработените. Скриптскиот дел вклучува data() функција што иницијализира празна низа employees, која е наменета за чување на објекти што ги претставуваат индивидуалните вработени. Оваа структура ќе се користи за внес, приказ, уредување и бришење на податоци поврзани со секој вработен.

Оваа компонента ќе биде надградена со богата функционалност, вклучувајќи:

- **Додавање нов вработен:** Корисниците со администраторски пристап ќе можат преку специјализирана форма да внесат нови вработени. При внесот, ќе се собираат податоци како име и презиме, работна позиција, контакт информации, плата, и други релевантни детали.
- **Приказ на вработени:** Сите внесени вработени ќе бидат прикажани во табеларен приказ или листа, што ќе овозможи лесен преглед, брза навигација и подобро управување со човечкиот капитал.
- **Филтрирање и пребарување:** Корисниците ќе имаат можност да пребаруваат по име или презиме, како и да филтрираат вработени според нивната работна позиција, статус (активен/неактивен), или други критериуми.
- **Ажурирање на податоци:** Компонентата ќе поддржува уредување на личните и професионалните информации за секој вработен, со што ќе се овозможи одржување на ажурна и точна база на податоци.
- **Бришење на вработени:** Администраторите ќе имаат опција за отстранување на вработен од системот, на пример поради заминување од работа или грешно внесување.
- **Доделување задачи и поврзување со интервенции:** Како дел од пошироката логика на апликацијата, ќе биде овозможено секој вработен да биде поврзан со специфични задачи или сервисни интервенции, при што ќе се овозможи подобро распределување на обврските и следење на извршените активности.

Понатаму, оваа компонента ќе биде интегрирана со останатите модули во апликацијата, како што се модулите за сервисни интервенции и управување со возила. Со ова ќе се постигне комплетна функционална поврзаност, која ќе придонесе за поефикасно управување со целиот процес во сервисниот центар.

Во заклучок, EmployeeManagement.vue не само што претставува визуелна точка за управување со персоналот, туку и логичка основа за поврзување на човечките ресурси со останатите компоненти во системот. Таа ќе биде клучна алатка за секој корисник кој е одговорен за кадровската администрација во сервисниот менаџмент.

```

<template> Show component usage
<div>
  <h1>Manage Employees</h1>
</div>
</template>

<script>
export default { Show usages
  data() {
    return {
      employees: [],
    };
  }
}
</script>

```

Слика 2.4.2 Изглед на Vue компонента за управување со вработени

2.4.3 ServiceManagement.vue

Компонентата ServiceManagement.vue, која се наоѓа во директориумот views/, е наменета за управување со сервисните интервенции во рамките на апликацијата *Car Service Manager*. Таа има централна улога во системот бидејќи овозможува структурирано и ефикасно следење, креирање, уредување и анализа на сите сервисни активности поврзани со возилата пријавени од страна на клиентите.

Во својата почетна верзија, компонентата содржи едноставен, но јасен HTML template со наслов „Manage Services“, кој служи како визуелен индикатор за корисникот дека се наоѓа на страницата посветена на сервисните интервенции. Во скриптскиот дел, дефинирана е функцијата data() која иницијализира празна низа services. Оваа низа претставува основна структура во која ќе се складираат сите податоци за постоечките или новокреирани сервисни интервенции – било да се работи за прегледи, поправки, замена на делови, дијагностика или други технички активности извршени врз возилата.

Во идните верзии, компонентата ќе биде проширена со комплетна функционалност која ќе вклучува:

- **Креирање на нова сервисна интервенција:** Корисниците (на пример вработени техничари или администратори) ќе можат преку интерактивна форма да креираат нова интервенција, внесувајќи релевантни детали како тип на сервис, датум и време, опис на проблемот, идентификација на возилото и вработениот кој ќе ја изврши интервенцијата.
- **Приказ на сите интервенции:** Компонентата ќе прикажува табеларен или листен приказ на сите закажани, тековни или завршени интервенции, со можност за брза навигација и пристап до поединечни ставки.
- **Филтрирање и пребарување:** Ќе се овозможи лесно пребарување на сервисни записи според датум, регистрација на возило, тип на интервенција, одговорен вработен или статус (во тек, завршено, откажано).
- **Ажурирање на постоечка интервенција:** Секој запис ќе може да се уреди во согласност со промените во реалната ситуација – на пример, промена на датум, додавање на дополнителни детали, или назначување на друг техничар.
- **Бришење на интервенции:** Ќе се овозможи бришење на погрешно внесени или откажани интервенции од системот, со што се одржува точноста на базата на податоци.

- **Поврзување со други ентитети:** Секоја сервисна интервенција ќе биде поврзана со конкретен автомобил и со вработен кој е назначен за нејзино извршување, како и со клиентот кој го пријавил проблемот. Ова ќе овозможи целосна тракинг логика и аналитичка контрола над целиот процес.

Дополнително, компонентата може да биде збогатена со известувања (notifications) за надминати термини, автоматска генерација на фактури или извештаи за извршените услуги, и интеграција со календар или надворешни API сервиси.

Во поширок контекст на апликацијата, ServiceManagement.vue ќе игра клучна улога во зголемувањето на оперативната ефикасност, транспарентноста и организираноста на сервисниот центар, овозможувајќи му на персоналот да ги следи сите активности во реално време и да донесува информирани одлуки за понатамошни интервенции.



```

<template> Show component usage
  <div>
    <h1>Manage Services</h1>
  </div>
</template>

<script>
export default { Show usages
  data() {
    return {
      services: [],
    };
  },
};
</script>

```

2.4.3 Изглед на Vue компонента за управување со сервиси

3. Компонентна организација

Во современите веб-апликации, особено оние развиени со фрејмворкот Vue.js, компонентната архитектура претставува основен пристап за структурирање и организирање на кодот. Преку користење на јасно дефинирани и независни компоненти, се овозможува поголема повторна искористливост, полесна одржливост и подобра прегледност на апликацијата.

Во рамки на оваа апликација за управување со автомобилски сервис, секој клучен сегмент од функционалноста е инкапсулиран во посебна компонента, која има сопствени податоци, логика и визуелен приказ. Компонентите меѓусебно комуницираат преку пренос на податоци (props) и емитување на настани (custom events), што овозможува чист и контролирачки начин на размена на информации помеѓу различните делови од системот.

Овој пристап не само што ја зголемува читливоста и одржливоста на кодот, туку и овозможува поефикасен развој, тестирање и надградба на апликацијата, особено во контексти каде што функционалноста се проширува или надградува со тек на време.

3.1 Архитектура и организација на апликацијата

Во рамки на оваа апликација за управување со автомобилски сервис, применета е компонентна организација базирана на архитектурата на Vue.js. Компонентите овозможуваат подобра модуларност, полесна одржливост и прегледност на кодот. Секоја компонента има јасно дефинирана одговорност и комуницира со родителската компонента преку props и custom events, што придонесува за чиста и ефикасна размена на податоци.

Апликацијата е организирана преку три главни компоненти:

3.1.1 *CarManagement.vue*

Опис на компонентата:

Компонентата *CarManagement.vue* е одговорна за управување со сите податоци поврзани со возилата. Таа овозможува интуитивно додавање, уредување и бришење на автомобили, како и визуелизација на сите внесени податоци преку табеларен приказ.

Функционалности:

- Додавање на нов автомобил преку формулар.
- Уредување на постоечки автомобил (со вчитување на податоците во формата).
- Бришење на автомобил од системот.
- Приказ на табела со сите внесени автомобили.

Комуникација со родителска компонента:

- При додавање нов автомобил, се емитува настанот `addCar (this.$emit('addCar', car))`, кој родителската компонента го пресретнува и ја ажурира централната листа на автомобили (`cars`).
- Компонентата прима `props: ['cars']`, преку кои ја добива тековната листа на автомобили од родителската компонента (на пример *App.vue*).

3.1.2 *EmployeeManagement.vue*

Опис на компонентата:

Компонентата *EmployeeManagement.vue* управува со податоците за вработените лица во сервисот. Таа нуди функционалности за додавање, модифицирање и отстранување на вработени, со транспарентен преглед на сите внесени податоци.

Функционалности:

- Додавање на нов вработен преку форма.
- Уредување на постоечки податоци за вработен.
- Бришење на вработен од системот.
- Приказ на табела со сите активни вработени.

Комуникација со родителска компонента:

- При креирање на нов вработен се емитува настанот `addEmployee (this.$emit('addEmployee', employee))`, кој родителската компонента го пресретнува за да ја ажурира главната листа.
- Компонентата прима `props: ['employees']`, што овозможува добивање на тековната листа на вработени од родителот.

3.1.3 *ServiceManagement.vue*

Опис на компонентата:

Компонентата *ServiceManagement.vue* е фокусирана на евиденција и управување со сервисните интервенции

што се изведуваат врз возилата. Таа претставува клучен дел од апликацијата, овозможувајќи креирање и обработка на информации поврзани со сервисите.

Функционалности:

- Додавање на нов сервис, вклучувајќи опис на интервенцијата, цена, избор на автомобил и вработен, како и датум.
- Уредување на податоци за постоечки сервис.
- Бришење на сервисна интервенција.
- Приказ на табела со сите извршени или закажани сервиси.

Комуникација со родителска компонента:

- Компонентата прима props: ['cars', 'employees'], со што се овозможува избор од достапни автомобили и вработени при внес на нов сервис.
- Листата на сервиси се чува локално во рамки на компонентата (services), без централизирана размена со родителската компонента. Во иднина, ова може да се прошири со дополнителна комуникација за синхронизација на податоците.

4. Функционалности на апликацијата

Апликацијата претставува веб-систем за интерно управување со возила, вработени и сервисирања во рамки на автосервис. Таа е развиена со користење на Vue.js и овозможува едноставен и интуитивен кориснички интерфејс за администрација на трите главни ентитети: возила, вработени и сервиси.

- Целта на апликацијата е да овозможи:
- лесно регистрирање и управување со податоци за автомобилите кои се сервисираат,
- одржување на ажурна листа на вработени кои ги извршуваат сервисите,
- и поврзување на секој сервис со соодветно возило и вработен.

Апликацијата поддржува основни CRUD операции (Create, Read, Update, Delete) за сите три ентитети, овозможува динамичка интеракција меѓу компонентите и обезбедува транспарентно управување со податоците.

Во продолжение се дадени детални описи на функционалностите по компоненти:

Управување со возила (CarManagement.vue)

Овој модул овозможува основни CRUD (Create, Read, Update, Delete) операции за автомобили.

Главни функционалности:

- Додавање на возило: Корисникот пополнува форма со следни полиња: марка, модел, мотор, година на производство и VIN број. По клик на "Add Car", возилото се додава во листата.
- Уредување на возило: Со клик на копчето "Edit", податоците за возилото се појавуваат во формата, каде може да се изменат и со клик на "Update Car" се ажурираат.
- Бришење на возило: Со клик на "Delete", избраното возило се отстранува од листата.
- Динамичка форма: Форма која автоматски менува изглед во зависност дали се додава или уредува возило.

- Синхронизација со родителски компоненти: Ново возило се емитира преку @emit на родителската компонента, што овозможува централно чување на податоци.

Управување со вработени (EmployeeManagement.vue)

Овој модул управува со листа на вработени, каде исто така се имплементирани CRUD операции.

Главни функционалности:

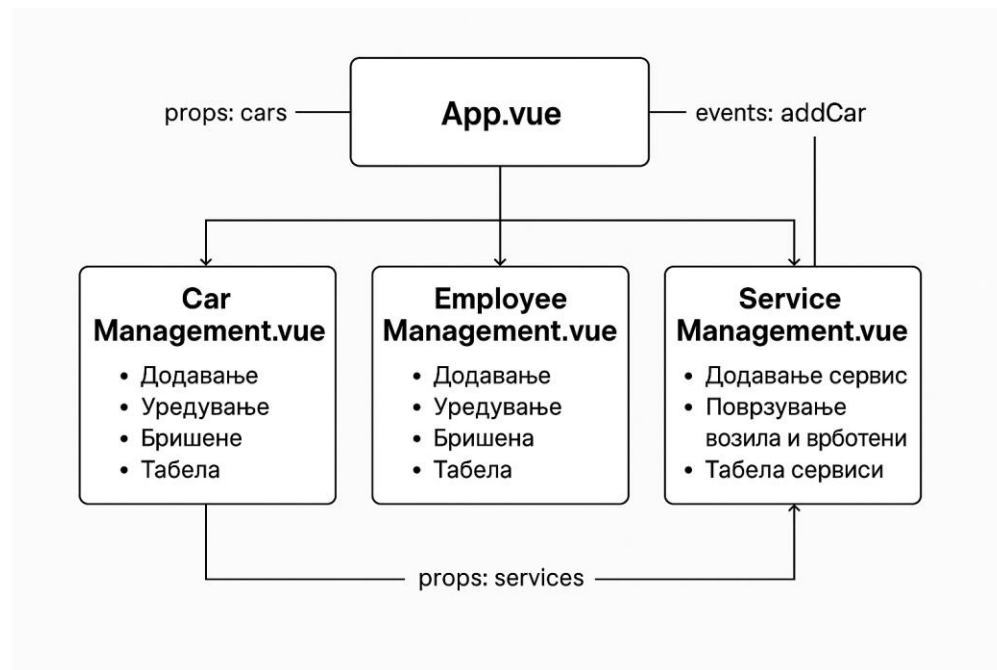
- Додавање на вработен: Преку форма корисникот внесува име на вработениот и со клик на "Add Employee" тој се додава во листата.
- Уредување на вработен: Со клик на "Edit", податоците на избраниот вработен се пренесуваат во формата и може да се уредат. Промените се зачувуваат со клик на "Update Employee".
- Бришење на вработен: Избраниот вработен се отстранува со клик на "Delete".
- Динамична обработка на податоци: Вработените се манипулираат во истата компонента преку splice() и се емитираат промени до родител.

Управување со сервисирања (ServiceManagement.vue)

Овој модул овозможува креирање и управување на сервисирања, поврзани со конкретен автомобил и вработен.

Главни функционалности:

- Додавање на сервис: Формата бара опис, цена, избор на возило, избор на вработен и датум. Се користат dropdown полиња за избор на автомобил и вработен (се поврзуваат преку props).
- Уредување на сервис: Избор на "Edit" овозможува уредување на постоечки сервис и по завршување, ажурирање на листата.
- Бришење на сервис: Избраниот сервис се отстранува преку "Delete".
- Поврзаност на податоци: Сервисите се поврзани со конкретен автомобил и вработен, што овозможува следење кој вработен работел на кој автомобил.
- Локално чување на сервисите: Сервисите се чуваат во data() објект и не се проследуваат надвор од компонентата (освен ако не се додаде emit во иднина).



Слика 4 Дијаграм на компоненти за системот за управување со авто-сервис (Car Service Manager)

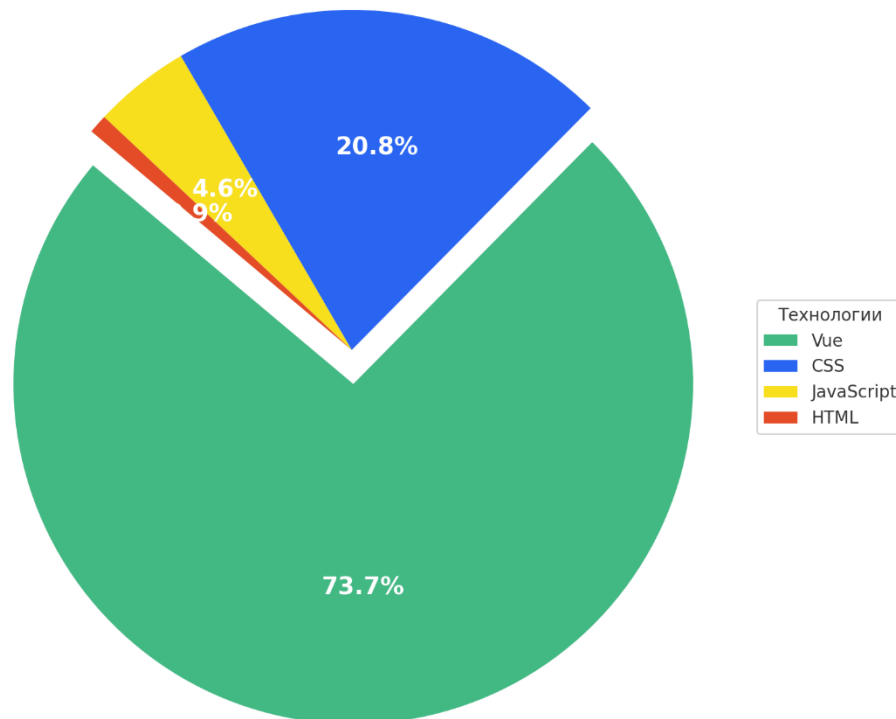
5. Технолошки стек

Во реализацијата на проектот **Car Service Manager** е користен модерен технолошки стек фокусиран на развој на веб-апликации со користење на компонентно базиран пристап. Дијаграмот подолу ја прикажува распределбата на користените технологии:

- **Vue.js (73.7%)**: Главната рамка користена за развој на фронтенд апликацијата. Vue овозможува модуларна организација преку компоненти, што придонесе за подобра читливост, одржливост и проширливост на кодот.
- **CSS (20.8%)**: Користен за стилизирање на компонентите и изгледот на апликацијата. Преку CSS се обезбедува пријателски кориснички интерфејс, прилагодлив на различни уреди.
- **JavaScript (4.6%)**: Употребен за додавање интерактивност и логика во компонентите, особено за ракување со настани, валидација на форми и манипулација со податоци.
- **HTML (9%)**: Служи како основа за структурирање на компонентите и дефинирање на содржината.

Овој стек овозможува развој на динамична, интерактивна и визуелно пријатна веб-апликација, со силен акцент на корисничкото искуство и функционалност.

Процентуална застапеност на јазици во проектот



6. Начини на користење на апликацијата

Апликацијата **Car Service Manager** е наменета за употреба од различни категории на корисници поврзани со автомобилскиот сервис, вклучувајќи административен персонал, техничари и менаџери. Таа овозможува организирана и ефикасна интеракција со сите релевантни информации за сервисирање на возила. Начините на користење на апликацијата се следните:

6.1 Регистрација и управување со возила

Првиот чекор во користењето на апликацијата е внесување на информации за нови возила. Корисникот може да додаде податоци како марка, модел, година на производство, регистарски таблички и сопственик на возилото. Секое возило е поврзано со клиент, што овозможува следење на историјата на сервисирање по индивидуален клиент или возило.

6.2 Регистрација и управување со клиенти

Апликацијата овозможува додавање нови клиенти преку едноставна форма за внес на податоци како име, презиме, контакт информации и други релевантни детали. Секој клиент може да биде поврзан со повеќе возила, што е особено корисно за корпоративни клиенти или семејства кои поседуваат повеќе автомобили.

6.3 Закажување и евиденција на сервисни интервенции

Клучна функционалност на апликацијата е можноста за закажување термин за сервис. Вработените можат да внесат точен датум и време, опис на потребната интервенција и да го поврзат терминот со конкретно возило и клиент. Секоја сервисна интервенција може да вклучува повеќе активности (на пример, промена на масло, дијагностика, замена на делови), а по завршувањето, се зачувува како дел од историјата на возилото.

6.4 Управување со вработени и доделување задачи

Апликацијата овозможува додавање и управување со профили на вработени лица, кои се одговорни за извршување на сервисните активности. На секој сервисен термин може да му се додели еден или повеќе вработени, со што се овозможува подобра организација и следење на работниот ангажман.

6.5 Приказ и анализа на историја на сервисирања

Секое возило има комплетна историја на сите претходни интервенции, вклучувајќи опис на проблемите, извршените работи, вклучените вработени и потрошени ресурси. Оваа функционалност овозможува подобро информирано донесување одлуки за идно одржување, како и обезбедување транспарентност кон клиентите.

6.6 Измени и бришење на податоци

Сите внесени информации во апликацијата може да се уредуваат или бришат во зависност од потребата. Ова е овозможено преку лесно достапни контроли во интерфејсот, со што се обезбедува флексибилност и одржување на точноста на податоците.

7. Интерфејс на корисникот

На основа на проектот **CarServiceManager**, интерфејсот за корисникот (UI) и корисничкото искуство (UX) играат клучна улога во создавање ефективна и удобна апликација за крајните корисници, како вработени, клиенти и администрација на сервисот. Ќе ја разгледаме структурната и визуелната организација, како и тековниот фокус на оптимизација на искуството на корисниците.

7.1 Структура на интерфејс (UI)

Корисничкиот интерфејс на **CarServiceManager** е внимателно дизајниран за да биде едноставен и логички организиран. Со користење на Vue.js, интерфејсот е поделен на различни компоненти кои ги обработуваат специфичните функционалности на апликацијата, како што се управувањето со возила, вработени, сервисни интервенции, и клиентски податоци.

Главното мени на апликацијата обезбедува лесен пристап до сите основни функционалности, преку јасно разграничени секции:

- **Управување со возила** – Каде корисникот може да додава нови возила, да ги уредува и брише.
- **Управување со клиенти** – За регистрирање, уредување и бришење на клиентски податоци.
- **Закажување и евиденција на сервисни интервенции** – Овозможува закажување нови сервисни интервенции и следење на тековните.
- **Управување со вработени** – Вработените можат да бидат додавани, уредувани и доделувани задачи во зависност од потребите на сервисот.
- **Историски податоци и анализа на сервисирања** – Дава пристап до извештаи и историја на сервисирањата, што го олеснува анализирањето на податоците.

7.2 Оптимизација на искуството на корисникот (UX)

За да се постигне оптимално корисничко искуство, интерфејсот е проектиран така што ќе го олесни користењето на апликацијата, ќе ги минимизира потешкотиите при навигација и ќе го зголеми задоволството на корисниците. Некои од основните аспекти кои влијаат на UX на апликацијата се:

7.2.1 Лесна навигација

Користењето на апликацијата е многу едноставно благодарение на интуитивната навигација. Главното мени и сите поврзани компоненти се лесно достапни со само неколку кликови, што овозможува корисниците да ги извршат потребните задачи без дополнителни напори. Важните информации се изложени на видливо место, а функционалностите се групирани по категории што го олеснува нивното користење.

7.2.2 Повратни информации и потврди

При секоја извршена акција, корисникот добива јасни повратни информации. На пример, кога ќе се додаде ново возило или ќе се закаже сервисна интервенција, корисникот ќе добие потврда дека акцијата е успешно извршена. За случај на грешка, ќе се прикаже појасна порака која го објаснува проблемот и како да се реши.

7.2.3 Мобилна оптимизација

Со оглед на тоа што многу корисници може да пристапуваат до апликацијата преку мобилни уреди, интерфејсот е целосно адаптиран за работа на различни големини на екрани. Интерфејсот останува функционален и пријатен за употреба без разлика дали се користи на мобилен телефон, таблет или десктоп компјутер.

7.2.4 Дизајн за пријатно визуелно искуство

Интерфејсот е дизајниран да биде визуелно привлечен, со внимателно избрани бои и типографија кои овозможуваат лесно читање и навигација. Иконите се едноставни и јасни, што ги прави лесни за разбирање и користење.

7.2.5 Интерактивни елементи

Апликацијата користи интерактивни елементи како што се падачки менија, чек-боксови и прогресивни барови кои го прават искуството попријатно и поефикасно. Овие елементи обезбедуваат брзо интеракција со апликацијата, без потреба од оптоварувачки постапки.

8. Заклучок

Изработката на апликацијата Car Service Manager овозможи практична примена на стекнатото знаење од предметот Напреден Веб Дизајн преку развој на реална веб-апликација базирана на компоненти. Апликацијата успешно ги опфаќа сите неопходни функционалности за ефикасно управување со еден автомобилски сервис, вклучувајќи додавање, уредување, преглед и бришење на возила, вработени и сервисни интервенции. Со тоа се овозможува подобра организација на работата, следење на историјата на сервисирање и олеснета комуникација помеѓу клиентите и сервисот. Користењето на Vue.js како фронтенд фрејмворк овозможи изградба на динамичен, интуитивен и лесен за користење кориснички интерфејс. Компонентниот пристап придонесе за подобра структура на кодот, полесно одржување и можност за понатамошна надградба. Воедно, преку имплементирање на CRUD-операции и работа со формулари, беа зајакнати вештините за валидација на податоци, менаџирање на состојбата на апликацијата и поврзување на податоците помеѓу различни компоненти. Овој проект претставува конкретен пример како современите веб-технологии можат да се искористат за решавање на реални проблеми, а воедно придонесе и за продлабочување на техничките и логичките вештини потребни за развој на покомплексни апликации.