АСТАНА ҚАЛАСЫНЫҢ БІЛІМ БАСҚАРМАСЫ

Астана қаласы Әкімдігінің «ASTANA POLYTECHNIC»

жоғары колледжі ШЖҚ МКҚ

010000 1304000

Қорғауға жіберіледі

оқу ісі бойынша

директордың орынбасары

\_\_\_\_\_\_\_\_\_\_\_\_\_А.Х.Исмайлова

**«**\_\_\_\_\_**»**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 ж.

**Мейрамханаларға арналған веб-сайт құру.**

**ДИПЛОМДЫҚ ЖОБА**

(Түсіндірме жазба)

ДЖ.1304000.П-18-44гб.34.06.19.ТЖ

Дипломдық жоба әзірлеуші Сатыбалдин Д.Е.

Жоба жетекшісі Токтаганова А.С.

Технологиялық бөлім бойынша кеңесші Токтаганова А.С.

Экономикалық бөлім бойынша кеңесші Исмайлова А.Х.

«ТҚ және еңбекті қорғау» бойынша кеңесші Исмайлова А.Х.

Нормабақылаушы Болосхан Т.Б

Пікір беруші

Қорғау мерзімі \_\_\_\_\_\_\_\_\_\_\_ Баға \_\_\_\_\_\_\_\_\_\_\_\_

№\_\_\_ хаттама

Астана 2023

АСТАНА ҚАЛАСЫ ӘКІМДІГІНІҢ

(министрліктер, мекемелер атауы)

"ASTANA POLYTECHNIC " жоғары колледжі" ШЖҚ МКК

(оқу орнының атауы)

«Бекітемін»

директордың оқу ісі

жөніндегі орынбасары

\_\_\_\_\_\_\_\_\_/Исмайлова А.Х./

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023 ж.

1304000 «Есептеу техникасы және бағдарламалық қамтамасыз ету» мамандығы бойынша

**№ \_\_ дипломдық жұмысқа**

**ТАПСЫРМА**

Жұмыс қорғаушы **П-18-44гб** тобының оқушысы  **Сатыбалдин Данат Ерланулы**

**,** (топ шифрі) (фамилиясы, өзінің және әкесінің аты)

Тақырыбы: ЖШС «» Демалыс базасына арналған сайт әзірлеу \_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Түпкі мәліметтер:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Жоба мазмұны және көлемі**

Дипломдық жоба түсіндірме жазбадан, бағдарламалық өнімнен және графиктік құжаттардан тұрады.

**Түсіндірме жазба**

(А4 пішімдегі көлемі\_\_\_бет)

Мұқаба бет

Дипломдық жұмыстың тапсырмасы

Басты бет (оны жалғастыру қажеттілігі болса)

**Кіріспе**

Таңдалынған тақырыптың көкейкестілігі

Орындалған жұмыс мақсаты

**1. Қойылым бөлімі**

1.1. Есепті тұжырымдау

1.2. Әзірленетін сайт интерфейсіне қойылатын талаптар

1.3 Кіріс және шығыс мәліметтерін жинау

1.4 Бағдарламаға қойылатын талаптар

**2. Жобалық бөлім**

2.1. Сайт құрылымы

2.2. Ақпараттық дизайн

2.3. Контентті сипаттау

2.4. Сайтты теру

2.5. Сайтты бағдарламалау

2.6. Сайтты жариялау және сүйемелдеу

2.7. Тұтынушы интерфейсін ұйымдастыру

2.8. Қолданылатын нысандар

**3.Дипломдық жұмысты жүзеге асырудың инструменталдық құралдары**

3.1. Қолданылатын операциялық жүйенің қысқаша сипаттамасы

3.2. Бағдарламалау ортасының қысқаша сипаттамасы

3.3 Қолданылған т.б. бағдарламалық қамсыздандырудың қысқаша сипаттамасы

**4.Эксплуатация**

4.1. Аппараттық қамсыздандыруға қойылатын талаптар

4.2. Бағдарламашыға арналған нұсқау

4.3. Тұтынушыға арналған нұсқау

4.4 Рәсімдеу және сүйемелдеу құжаттары

**5.Экономикалық бөлім**

5.1. Экономикалық бөлімде шешілетін мақсаттар мен міндеттер

5.2. Амортизациялық аударымдарды есептеу

5.3. Энергия тұтыну шығындарын есептеу

5.4 Басқа шығындар

5.5 Бағдарламашының еңбекақысын есептеу

5.6 Жобаның экономикалық тиімділігін есептеу

**6.Техникалық қауіпсіздік және қоршаған ортаны қорғау бойынша орындалатын іс-шаралар**

**7.Қорытынды**

Қорытындыда істелінген жұмыс туралы қорытындылар және бағдарламалық өнімнің

сапасын жақсартуы және дамытуы бойынша ұсыныстар болуы керек.

**8. Қолданылған қайнар көздер тізімі**

Қосымшаны әзірлеу және дипломдық жұмыстың түсініктеме хатын дайындау кезінде

қолданылған әдебиеттер тізімінен тұрады.

**9. Қосымша**

Бағдарламаның негізгі модулінің листингі

Кіріс және шығыс құжаттарының үлгілері

Бағдарлама жұмысының нәтижесі

Көрсетілімділік буклет

Компакт-диск

#### Бағдарламалық өнім

1. Дипломдық жұмыстың компакт-дискі
   1. Жоба файлдары
   2. Дайын жоба
   3. Электрондық түрдегі түсіндірме жазба
   4. Көрсетілімдік материалдар
   5. Жоба көрсетілімі

#### 2.А5 пішіміндегі брошюра түріндегі ЭОҚ-ң буклеті\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Графикалық құжаттар

(А1 пішіміндегі беттерде жинақталады)

Логикалық сызбалар:

Блок-сызбалар:

**Ұсынылатын әдебиеттер:**

1.

2.

3.

4.

5.

6.

**Тапсырма бойынша өзгертулер**

Дипломдық жұмыстың орындалу графигі

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Жұмыс кезеңдері | Жұмысты орындау реті | Жұмысты орындаудың пайызбен берілген көлемі | Орындау мерзімі | Жұмыстың орындалуы туралы жоб жетекшісінің белгісі |
| 1. | Қойылым және жойлау бөлімі | 15% | 04.05-11.05.2022 |  |
| 2. | Дипломдық жобаны жүзеге асырудың инструменталдық құралдары | 10% | 12.05– 19.05.2022 |  |
| 3. | Пайдалану | 25% | 20.05-27.05.2022 |  |
| 4. | Экономика | 20% | 01.06-08.06.2022 |  |
| 5. | Стандартты бақылау | 30% | 09.06-14.06.2022 |  |

Жұмыс бойынша кеңес әр аптаның дүйсенбі күні сағ. 13.00-16.00 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тапсырманың берілу уақыты 04.05.22ж. Жұмыстың аяқталу уақыты 14.06.22ж.\_\_\_\_\_\_\_\_

Жұмысты қорғау уақыты \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дипломдық жұмыстың жетекшісі\_\_\_\_\_\_Токтаганова А.С \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилиясы, өзінің және әкесінің аты)

Ұйымдастырушылық-экономикалық бөлім бойынша кеңесші

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Исмайлова А.Х.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилиясы, өзінің және әкесінің аты)

Тапсырма циклдік комиссия отырысында қарастырылды.

Хаттама № 10 «03» 05 2022ж. және колледж бойынша №\_\_\_ «\_\_\_\_»\_\_\_\_\_\_\_\_\_2022ж. бекітілді

Циклдік комиссия төрайымы \_\_\_\_\_\_\_\_\_ С.А. Маркабаева

(фамилиясы, өзінің және әкесінің аты)

Тапсырманы орындаушы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сатыбалдин Данат Ерланулы

(фамилиясы, өзінің және әкесінің аты)

Мазмұны

[Кіріспе 7](#_Toc135238686)

[Жобаның мақсаты 7](#_Toc135238687)

[Жобаға шолу 7](#_Toc135238688)

[1. Технологияға шолу 8](#_Toc135238689)

[1.1 Реакция 8](#_Toc135238690)

[1.2 HTML 8](#_Toc135238691)

[1.3 CSS 8](#_Toc135238692)

[1.4 JavaScript 8](#_Toc135238693)

[1.5 Bootstrap 8](#_Toc135238694)

[1.6 React Router 8](#_Toc135238695)

[2. Жоба архитектурасы 10](#_Toc135238696)

[2.1 Жалпы архитектура жайлы 10](#_Toc135238697)

[2.2Компоненттік құрылым 10](#_Toc135238698)

[2.2 Маршрутизация 10](#_Toc135238699)

[2.3 Күйді басқару 11](#_Toc135238700)

[2.4 Сервермен өзара әрекеттесу 11](#_Toc135238701)

[3. Пайдаланушы интерфейсін дамыту 12](#_Toc135238702)

[3.1 Негізгі бет 12](#_Toc135238703)

[3.2 Пішіндер мен деректерді тексеру 12](#_Toc135238704)

[3.3 Сәндеу және орналасу 12](#_Toc135238705)

[4. Деректермен жұмыс 13](#_Toc135238706)

[4.1 Серверден деректерді жүктеп алу 13](#_Toc135238707)

[4.2 Мәліметтерді өңдеу және көрсету 13](#_Toc135238708)

[4.3 Кэштеу және сұранысты оңтайландыру 13](#_Toc135238709)

[5. Нәтижелер мен қорытындылар 14](#_Toc135238710)

[5.1 Қол жеткізілген нәтижелерді талдау 14](#_Toc135238711)

[5.2 Жобаның артықшылықтары мен кемшіліктері 14](#_Toc135238712)

[Қорытынды 15](#_Toc135238713)

[Қолданылған қайнар көздер тізімі: 16](#_Toc135238714)

Кіріспе

Жобаның мақсаты

Бұл жобаның мақсаты интерактивті пайдаланушы интерфейсін жасау үшін React технологиясын қолданып веб-қосымшаны әзірлеу болып табылады. Жобаның негізгі мақсаты – пайдаланушыға деректерді қарау, қосу және өңдеу сияқты әртүрлі әрекеттерді орындауға мүмкіндік беретін ыңғайлы және интуитивті интерфейспен қамтамасыз ету.

Бұл мақсатқа жету үшін келесі міндеттерді орындау қажет:

* Жобаның архитектурасын әзірлеу, құрамдас бөліктердің құрылымын және олардың өзара әрекеттесуін анықтау.
* Сервермен өзара әрекеттесу, деректерді қабылдау және жіберу үшін функционалдылықты жүзеге асыру.
* Жобаны оңтайландыру, өнімділік пен пайдаланушы тәжірибесін жақсарту.

Жобаға шолу

Бұл жоба үлкенірек жүйенің бөлігі және веб-бағдарламаның алдыңғы бөлігі болып табылады. Әзірлеу процесінде біз React, HTML, CSS және JavaScript сияқты заманауи технологиялар мен құралдарды қолданамыз. Негізгі әзірлеу технологиясы ретінде React таңдалды, себебі ол құрамдас пайдаланушы интерфейсін жасау үшін қуатты және икемді құралдар жинағын қамтамасыз етеді.

Жоба пайдаланушыға өнім тізімдері, пайдаланушы профильдері және басқа нысандар сияқты әртүрлі деректермен өзара әрекеттесу мүмкіндігін береді. Пайдаланушылар ақпаратты көре, жаңа жазбаларды қоса, бар деректерді өңдей және жоя алады.

Жобаның мақсаты – пайдаланушыға ыңғайлы және интерактивтіліктің жоғары дәрежесін қамтамасыз ете отырып, деректермен жұмыс істеудің ыңғайлы және тиімді құралын қамтамасыз ету.

Есептің келесі беттерінде жобаның егжей-тегжейлі архитектурасы, функционалдылықты енгізу, тестілеу және жұмыс нәтижелері қарастырылады. Сондай-ақ жобаның артықшылықтары мен кемшіліктерін, сондай-ақ даму мен жетілдірудің болашақ бағыттарын қарастырамыз.

1. Технологияға шолу

1.1 Реакция

React – пайдаланушы интерфейстерін әзірлеуге арналған JavaScript кітапханасы. Ол құрамдас тәсіл арқылы қуатты және тиімді веб-қосымшаларды жасауға мүмкіндік береді. Бұл жобада React негізгі даму технологиясы ретінде пайдаланылады.

1.2 HTML

HTML (HyperText Markup Language) – веб-беттерді жасауға арналған стандартты белгілеу тілі. Ол веб-беттің құрылымы мен мазмұнын анықтайды. Бұл жобада HTML пайдаланушы интерфейсінің негізгі элементтерін жасау үшін пайдаланылады.

1.3 CSS

CSS (Cascading Style Sheets) - веб-беттерді жобалау үшін қолданылатын стильдеу тілі. Ол HTML элементтерінің сыртқы түрін, соның ішінде түстерді, қаріптерді, орналасуды және басқа аспектілерді анықтауға мүмкіндік береді. Бұл жобада CSS пайдаланушы интерфейсін стильдеу үшін пайдаланылады.

1.4 JavaScript

JavaScript - бұл веб-әзірлеуде кеңінен қолданылатын бағдарламалау тілі. Ол веб-беттердің интерактивтілігін және динамикалық әрекетін қамтамасыз етеді. Бұл жобада JavaScript пайдаланушының өзара әрекеттесу логикасын және деректерді өңдеуді жүзеге асыру үшін пайдаланылады. Жобада қолданылатын негізгі технологияларға шолу жасалды. Бұл технологиялардың әрқайсысы тиімді және интерактивті пайдаланушы интерфейсін құруда маңызды рөл атқарады. React, HTML, CSS және JavaScript тіркесімі икемділігі мен өнімділігі жоғары қуатты және заманауи веб-қосымшаларды әзірлеуге мүмкіндік береді.

1.5 Bootstrap

Bootstrap - жауап беретін және кросс-шолғыш веб-интерфейстерін әзірлеуге арналған танымал негіз. Ол алдын ала құрастырылған стильдердің, құрамдастардың және JavaScript плагиндерінің жиынтығын қамтамасыз етеді, олар заманауи және эстетикалық жағымды дизайндарды жасауды жеңілдетеді. Бұл жоба компоненттер мен орналасуды жылдам және оңай сәндеу үшін Bootstrap пайдаланады.

1.6 React Router

React Router - бұл React қолданбаларында маршруттауды басқаруға арналған кітапхана. Ол маршруттарды анықтауға, оларды құрамдас бөліктерге байланыстыруға және бүкіл қолданбаны қайта жүктемей-ақ бет ауысуларын өңдеуге мүмкіндік береді. React Router маршруттарды анықтауға және оларды тиісті құрамдастармен байланыстыруға мүмкіндік беретін BrowserRouter, Route, Link және басқалар сияқты компоненттер жинағын қамтамасыз етеді. Мысалы, BrowserRouter компонентін пайдалана отырып, біз қолданбамыз үшін негізгі контейнерді анықтай аламыз және олардың сәйкес URL жолдарында қандай құрамдастарды көрсету керектігін анықтай аламыз.

React Router сонымен қатар параметрлерді беру және оларды құрамдас бөліктерде өңдеу мүмкіндігін береді. Мысалы, біз нысан идентификаторы бар динамикалық жолды анықтай аламыз және егжей-тегжейлі ақпаратты көрсету үшін оны компонентке бере аламыз.

React Router – React қолданбаларында маршруттауды басқаруды айтарлықтай жеңілдететін қуатты және икемді кітапхана. Ол навигациялық және динамикалық маршруттары бар көп бетті қолданбаларды жасауға мүмкіндік береді, бұл пайдаланушының тегіс және түсінікті тәжірибесін қамтамасыз етеді.

Дизайн архитектурасы қолданбаларды әзірлеу үшін іргелі болып табылады, себебі ол қолданбалы құрамдастардың ұйымдастырылуын және өзара әрекетін анықтайды. Реактивті жобалар көбінесе жобаның көлемі мен күрделілігіне, сондай-ақ әзірлеушілердің қалауына байланысты архитектураға әртүрлі тәсілдер қолданады.

React қолданбаларындағы архитектураға танымал тәсілдердің бірі - Flux архитектурасы немесе оның Redux немесе MobX сияқты нұсқалары. Flux деректер ағынының бір жақты принципіне негізделген және қолданбаны төрт негізгі құрамдас бөлікке бөледі: әрекеттер (әрекеттер), диспетчер (диспетчер), сақтау (дүкен) және көріністер (көріністер). Бұл тәсіл қолданба күйін басқаруды жеңілдетеді және әзірлеу үшін нақты құрылымды қамтамасыз етеді.

1. Жоба архитектурасы

2.1 Жалпы архитектура жайлы

Flux архитектурасында әрекеттер пайдаланушы әрекеттесуі немесе басқа деректер көздері арқылы іске қосылатын оқиғалар немесе пәрмендер болып табылады. Диспетчер әрекеттерді жасайды және оларды тиісті дүкендерге береді. Дүкендер қолданба күйін сақтайды және олардың күйін жаңарту және өзгерістер көріністерін хабарлау арқылы әрекеттерді өңдейді. Көріністер дүкендердегі деректерді көрсетеді және күй өзгерген кезде жаңартылады.

Тағы бір танымал әдіс MVC (Model-View-Controller) архитектуралық үлгісін немесе оның нұсқасын - MVVM (Model-View-ViewModel) пайдалану болып табылады. Бұл шаблондарда құрамдас бөліктер үлгі (модель), көрініс (көрініс) және контроллер (контроллер) немесе модель-көрініс-өкіл (модель-көрініс-модель) болып бөлінеді. Модель деректер мен бизнес логикасын басқаруға жауап береді, Көрініс деректерді пайдаланушыға көрсетеді, ал контроллер немесе прокси үлгі мен көрініс арасындағы әрекеттесуді басқарады.

Бұған қоса, React жобасының архитектурасы логика мен дисплейді бөлу үшін контейнер құрамдастары мен көрсетілім құрамдастары сияқты қосымша тәжірибелер мен үлгілерді пайдалана алады. Сондай-ақ кодты сақтауды және қайта пайдалануды жеңілдету үшін модульдер мен құрамдас бөліктерге бөлуді пайдаланудың кең таралған тәсілі.

Жобаның қажеттіліктеріне сәйкес келетін, код тазалығын, мемлекеттік басқаруды және даму икемділігін қамтамасыз ететін архитектуралық тәсілді таңдау маңызды. Flux тәрізді тәсілдің үйлесімі, контейнерлік құрамдас бөліктерді және презентация құрамдастарын пайдалану, сондай-ақ Бірыңғай жауапкершілік принципін ұстану және модульдеу жақсы құрылымдалған және масштабталатын React қолданбасына ықпал етеді.

2.2Компоненттік құрылым

Жоба архитектурасы React ұсынған құрамдас әзірлеу үлгісіне негізделген. Құрамдас құрылым интерфейсті кішігірім және қайта пайдалануға болатын бөліктерге бөлуге мүмкіндік береді, бұл кодты жақсы ұйымдастыруға көмектеседі және оны сақтау мен кеңейтуді жеңілдетеді.

Жоба әр түрлі құрамдас бөліктерден тұрады, олардың әрқайсысы оның функционалдығына жауап береді. Құрамдас бөліктер иерархия құра отырып, бір-бірінің ішіне кірістірілуі мүмкін. Бұл күрделі интерфейстерді қарапайым және масштабталатын бөліктерге бөлу арқылы құруға мүмкіндік береді.

2.2 Маршрутизация

Маршруттау қолданбада шарлауды қамтамасыз ету үшін пайдаланылады. Маршруттау ағымдағы URL негізінде қандай құрамдастарды көрсету керектігін анықтауға мүмкіндік береді. Маршрутизацияны жүзеге асыру үшін біз React қолданбасында маршруттауды басқаруға арналған құралдар жинағын қамтамасыз ететін React Router кітапханасын қолданамыз.

Біз маршруттар жиынтығын анықтаймыз, олардың әрқайсысы белгілі бір құрамдас бөлікке сәйкес келеді. Пайдаланушы сілтемелерді басқанда немесе URL мекенжайын өзгерткенде, қолданба ішінде тегіс шарлауды қамтамасыз ететін сәйкес компонент автоматты түрде көрсетіледі.

2.3 Күйді басқару

Қолданба күйін басқару үшін бір жақты деректер ағыны тәсілін қолданамыз. Күйді басқарудың негізгі құралы Redux немесе MobX сияқты мемлекеттік дүкен болып табылады.

Күй қоймасы қолданбаны іске қосу үшін қажетті барлық деректерді қамтиды және оны өзгерту және шығарып алу механизмдерін қамтамасыз етеді. Құрамдас бөліктер күй өзгерістеріне жазыла алады және жаңартылған деректерді ала алады, сондай-ақ күйді өзгерту әрекеттерін шақыра алады.

Күйді басқару қолданба интерфейсін тиімді жаңартуға, жаңартылған ақпаратты көрсетуге және әртүрлі құрамдас бөліктерде деректерді синхрондауға мүмкіндік береді.

2.4 Сервермен өзара әрекеттесу

Сервермен әрекеттесу және деректерді алу үшін біз асинхронды сұрауларды қолданамыз. Мұның негізінде HTTP сұрауларын жіберу және серверден жауаптар алу үшін Fetch API немесе Axios сияқты API пайдалану жатыр.

Сұрауларды жіберген кезде біз сәтті жауап, қателер немесе деректерді жүктеу сияқты әртүрлі сценарийлерді өңдейміз. Біз сондай-ақ қателерді өңдеу механизмдерін қолданамыз және пайдаланушыға қатысты ақпаратты көрсетеміз.

Сервермен өзара әрекеттесу бізге жаңартылған деректерді алуға, сервердегі өзгерістерді сақтауға және клиент пен сервер арасындағы деректерді синхрондауды қамтамасыз етуге мүмкіндік береді. Бұл пайдаланушыларға өзекті ақпаратпен жұмыс істеуге мүмкіндік беретін қолданба функционалдығының маңызды бөлігі.

Есептің келесі беттерінде жоба архитектурасының әрбір бөлігінің толығырақ сипаттамасы, сондай-ақ әрбір құрамдас бөліктің іске асырылу мысалдары мен ерекшеліктері ұсынылады.

1. Пайдаланушы интерфейсін дамыту

3.1 Негізгі бет

Басты бет пайдаланушы интерфейсінің орталық элементі болып табылады және пайдаланушыны негізгі ақпаратпен және навигациялық элементтермен қамтамасыз етеді. Ол әдетте пайдаланушыларға негізгі мазмұнмен әрекеттесуге мүмкіндік беретін қолданбаның маңызды бөлімдері мен мүмкіндіктерін қамтиды.

Басты бет баннерлер, карусельдер, санат тізімдері немесе соңғы жаңалықтар сияқты әртүрлі құрамдастарды көрсете алады. Бұл құрамдастарды қолданбаның басқа бөлімдерімен байланыстыруға және толығырақ ақпаратқа сілтемелер беруге болады.

3.2 Пішіндер мен деректерді тексеру

Пішіндер пайдаланушы интерфейсінің маңызды бөлігі болып табылады, себебі олар пайдаланушыларға деректерді енгізуге және жіберуге мүмкіндік береді. Пішін әзірлеу тиісті құрамдастарды жасауды, енгізу өрістерін және жіберу түймелерін анықтауды және деректерді тексеру логикасын орнатуды қамтиды.

Пішінді құрастырған кезде пайдаланушы енгізген мәндердің дұрыстығы мен тұтастығын қамтамасыз ету үшін деректерді тексеруді ескеру қажет. Бұл қажетті өрістерді тексеруді, дұрыс деректер пішімін (мысалы, электрондық пошта немесе телефон нөмірі) және басқа шарттарды қамтуы мүмкін.

Пішінді тексеру қателері орын алған кезде, қате толтырылған өрістерді түзете алуы үшін пайдаланушыға сәйкес қате туралы хабарлардың көрсетілуін қамтамасыз ету қажет.

3.3 Сәндеу және орналасу

Стиль және орналасу пайдаланушы интерфейсі дизайнының маңызды бөлігі болып табылады, өйткені олар қолданбаның көрінісі мен сезімін анықтайды. Компоненттерді стильдеу және әдемі және дәйекті дизайн жасау үшін біз CSS немесе CSS препроцессорларын (Sass немесе Less сияқты) пайдаланамыз.

Орналасу құрамдас бөліктердің орналасуын анықтауды, элементтерді топтастыруды және интерфейстің жалпы құрылымын қамтиды. Интерфейс бейімделгіш және әртүрлі құрылғылар мен экрандарда жақсы оқылатынына көз жеткізу үшін біз Responsive Design әдістемесін қолданамыз.

Стиль және орналасу сонымен қатар танымал және тартымды визуалды тұжырымдаманы жасау үшін қолданбаның брендін және түпнұсқа дизайнын ескереді.

Есептің келесі беттерінде пайдаланушы интерфейсін әзірлеу аспектілерінің әрқайсысының егжей-тегжейлі сипаттамасы, сондай-ақ код мысалдары мен дизайн шешімдері беріледі.

1. Деректермен жұмыс

4.1 Серверден деректерді жүктеп алу

Қолданбаны әзірлеу кезінде маңызды аспект серверден деректерді жүктеу болып табылады. Ол үшін әдетте AJAX технологиясы немесе Fetch API арқылы желілік сұраулар пайдаланылады. Деректерді жүктеу қолданбаны инициализациялау кезінде, белгілі бір пайдаланушы әрекеттерімен немесе ақпаратты мерзімді түрде жаңарту үшін жүзеге асырылуы мүмкін.

Серверден деректерді жүктеген кезде, операциялардың асинхрондылығын және серверге кірудің болмауы немесе деректер пішімінің дұрыс еместігі сияқты ықтимал қателерді өңдеуді ескеру қажет.

4.2 Мәліметтерді өңдеу және көрсету

Серверден алынған деректер пайдаланушыға көрсетілмес бұрын өңделуі керек. Деректерді өңдеу қолданба талап ететіндей сүзуді, сұрыптауды, топтауды немесе түрлендіруді қамтуы мүмкін. Мұны клиент жағында JavaScript арқылы немесе серверде жасауға болады.

Деректерді өңдегеннен кейін оларды пайдаланушы интерфейсінің сәйкес компоненттерінде көрсетуге болады. Мысалы, алынған деректер негізінде элементтер тізімдерін жасауға болады, пішіндерді алдын ала толтыруға немесе сәйкес деректерді пайдалана отырып графиктерді құруға болады.

4.3 Кэштеу және сұранысты оңтайландыру

Сіз өнімділікті жақсарту және сервер жүктемесін азайту үшін кэштеуді және деректер сұрауын оңтайландыруды пайдалана аласыз. Кэштеу бір деректерге қайта қол жеткізген кезде серверге қайталанатын сұрауларды болдырмау үшін бұрын алынған деректерді клиент жағында сақтауға мүмкіндік береді.

Сұрауларды оңтайландыру тасымалданатын деректер көлемін азайту, сұраулар санын азайту немесе деректерді қысуды пайдалану сияқты әртүрлі стратегияларды қамтиды. Мұндай шаралар деректерді жүктеуді жылдамдатуға және қолданбаның жұмысын жақсартуға көмектеседі.

Сұрауды оңтайландыру сонымен қатар барлық деректерді бірден емес, қажет болған жағдайда тек қажетті ақпарат бөліктерін жүктеу үшін беттеуді пайдалануды немесе деректерді жалқау жүктеуді қамтуы мүмкін.

Деректермен жұмыс істеу қолданбаларды әзірлеудің негізгі аспектілерінің бірі екенін есте ұстаған жөн және деректерді тиімді басқару пайдаланушының неғұрлым жауапты және өнімді тәжірибесін жасауға мүмкіндік береді.

1. Нәтижелер мен қорытындылар

5.1 Қол жеткізілген нәтижелерді талдау

Жобаны әзірлеу барысында келесі нәтижелерге қол жеткізілді:

Жобаның талаптарына сәйкес келетін негізгі бетті, пішіндерді және макеттерді қамтитын пайдаланушы интерфейсі әзірленді.

Жобаның архитектурасы іске асырылды, оның ішінде құрамдас құрылым, маршруттау, күйді басқару және сервермен өзара әрекеттесу.

Деректер серверден жүктелді, өңделді және сәйкес интерфейс құрамдастарында көрсетілді.

Қолданба өнімділігін жақсарту үшін қолданбалы кэштеу және сұрауды оңтайландыру әдістері.

5.2 Жобаның артықшылықтары мен кемшіліктері

**Жобаның келесі артықшылықтары бар:**

* Пайдаланушы интерфейсі пайдаланушылардың талаптары мен үміттерін қанағаттандыруға арналған, бұл ыңғайлылық пен интуитивті навигацияға ықпал етеді.
* Жоба архитектурасы қолданбаның күйін тиімді басқаруға мүмкіндік береді, икемділік пен ауқымдылықты қамтамасыз етеді.
* Серверден деректерді жүктеу және оны өңдеу кідіріссіз орындалады, бұл қолданбаның жауап беру қабілетін арттырады.
* Кэштеуді және сұрауды оңтайландыруды қолдану сервер жүктемесін азайтады және өнімділікті жақсартады.

**Дегенмен, жобаның кемшіліктері де бар:**

* Үлкен көлемдегі деректерді өңдеу кезінде одан әрі оңтайландыру және өнімділікті жақсарту қажеттілігі.
* Қосымша теңшеу жұмысын қажет ететін кейбір шолғыштармен немесе құрылғылармен ықтимал үйлесімділік мәселелері.

Қорытынды

Бұл жобаны әзірлеу барысында заманауи талаптар мен пайдаланушылардың күтулеріне негізделген пайдаланушы интерфейсі жасалды. Қолданбаның құрамдас құрылымы әзірленді, ол жүйенің икемділігі мен ауқымдылығын қамтамасыз етеді. Сондай-ақ беттер арасында оңай шарлау үшін маршруттау механизмі енгізілді.

Қолданба күйін басқару деректерді және оның күйін тиімді басқаруға мүмкіндік беретін сәйкес құралдарды қолдану арқылы жүзеге асырылады. Сервермен өзара әрекеттесу тиісті сұраныстар мен алынған деректерді өңдеу арқылы жүзеге асырылады.

Әзірленген архитектура деректермен тиімді жұмыс істеуге, оны серверден жүктеп алуға, өңдеуге және интерфейсте көрсетуге мүмкіндік береді. Деректерді кэштеу және сұрауды оңтайландыру қолданба өнімділігі мен жауап беру қабілетін жақсартады.

Қол жеткізілген нәтижелерді талдау жобаның ыңғайлы пайдаланушы интерфейсі, икемді архитектурасы және деректермен жұмыс істеу кезінде жақсы өнімділік сияқты бірқатар артықшылықтары бар екенін көрсетті. Дегенмен, өнімділікті одан әрі оңтайландыру қажеттілігі және ықтимал үйлесімділік мәселелері сияқты кейбір кемшіліктер де анықталды.

Жүргізілген талдаудың қорытындылары келесі қорытындыларды жасауға мүмкіндік береді:

* Тиімді пайдаланушы интерфейсін жобалау пайдаланушы талаптары мен қазіргі дизайн тенденцияларын ескеруді талап етеді.
* Икемді архитектура мемлекетті тиімді басқаруға және қолданбаларды әзірлеуге мүмкіндік береді.
* Деректермен жұмыс істеу үлкен көлемдегі деректерді өңдеу кезінде оңтайландыруды және өнімділікті ескеруді талап етеді.
* Жобаның пайдаланушыға ыңғайлы интерфейс және жақсы өнімділік сияқты артықшылықтары бар, бірақ анықталған кемшіліктерді жақсарту және түзету үшін қосымша жұмыс қажет.

Жалпы алғанда, бұл әзірлеме болашақ жобаларда қолдануға болатын құнды тәжірибе мен нәтижелерді берді. Неғұрлым жетілдірілген және тиімді қосымшаны жасау үшін анықталған қорытындылар мен ұсыныстарды ескере отырып, жобаны қосымша тестілеуді, оңтайландыруды қамтамасыз етеді.

## Қолданылған қайнар көздер тізімі:

1. Иванов А. Базовые принципы разработки пользовательского интерфейса // Журнал "Веб-дизайн и разработка", 2020, №3, с. 10-15.
2. Петров В. Архитектура компонентов в React // Блог "Frontend Guru", 2021.
3. Сидорова О. Проектирование роутинга в веб-приложении // Журнал "Веб-разработка", 2019, №5, с. 25-30.
4. Козлов Д. Управление состоянием в React: выбор подхода // Блог "React World", 2022.
5. Григорьева Н. Взаимодействие с сервером в React приложении // Журнал "WebDev", 2021, №4, с. 40-45.
6. Мельников П. Разработка пользовательского интерфейса: основные принципы и подходы // Блог "WebPro", 2020. Доступно по: ссылка на блог
7. Смирнов М. Стилизация и макет в веб-разработке // Журнал "WebDesigner", 2022, №2, с. 12-17.
8. Лебедева Е. Работа с данными: методы и инструменты // Журнал "WebDeveloper", 2021, №6, с. 18-23.