

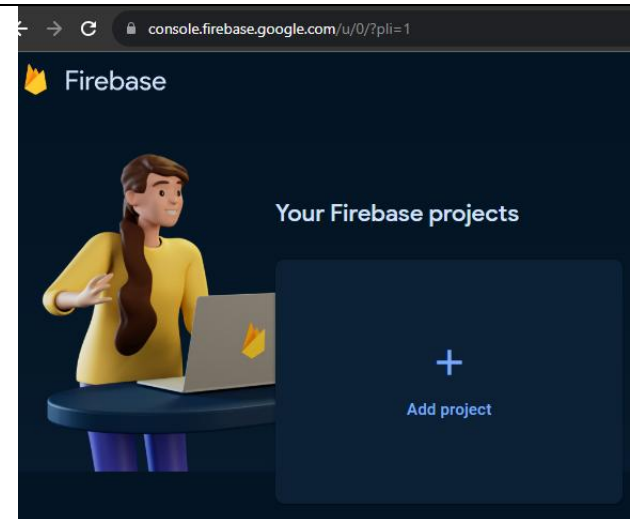
## CLI. Робота з Firestore Database в Firebase

### Add Firebase to your JavaScript project

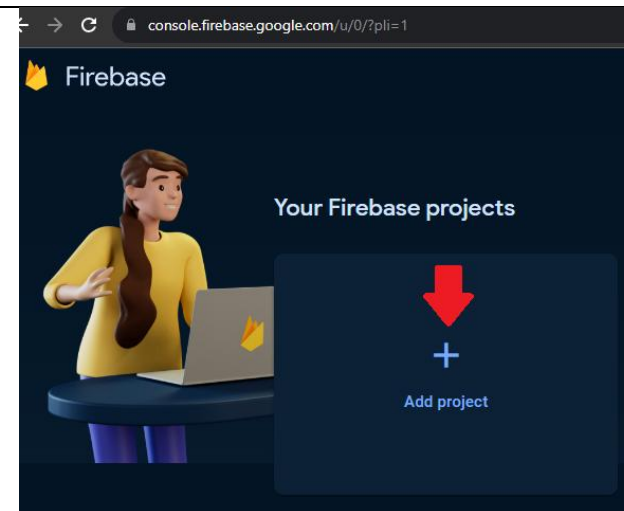
[https://firebase.google.com/docs/web/setup?hl=en&authuser=0&gl=1\\*1as5592\\*ga\\*MjA4NDYzOTg2Ny4xNzAwNjczMzM5\\*ga\\_CW55HF8NVT\\*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA](https://firebase.google.com/docs/web/setup?hl=en&authuser=0&gl=1*1as5592*ga*MjA4NDYzOTg2Ny4xNzAwNjczMzM5*ga_CW55HF8NVT*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA).

Переходимо у консоль firebase

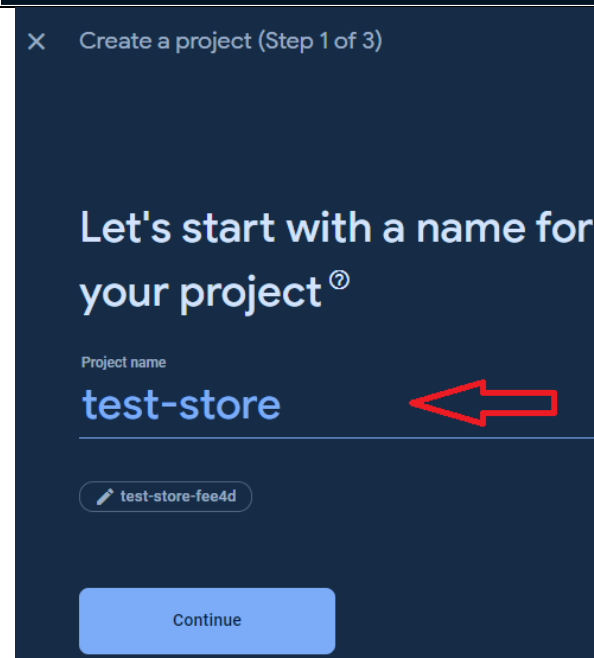
<https://console.firebase.google.com/u/0/?pli=1>



Створюємо проект в Firebase



Задаємо якусь назву проекту



Відключаємо аналітику

X

Create a project (Step 2 of 2)

Google Analytics

for your Firebase project

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, and Cloud Functions.

Google Analytics enables:

X

A/B testing ⓘ

X

Crash-free users ⓘ

X

User segmentation & targeting across Firebase products ⓘ

X

Event-based Cloud Functions triggers ⓘ

X

Free unlimited reporting ⓘ

Enable Google Analytics for this project

Recommended

↑

Previous

Create project

Створюємо проєкт

Creating your project... Please wait...

test-store

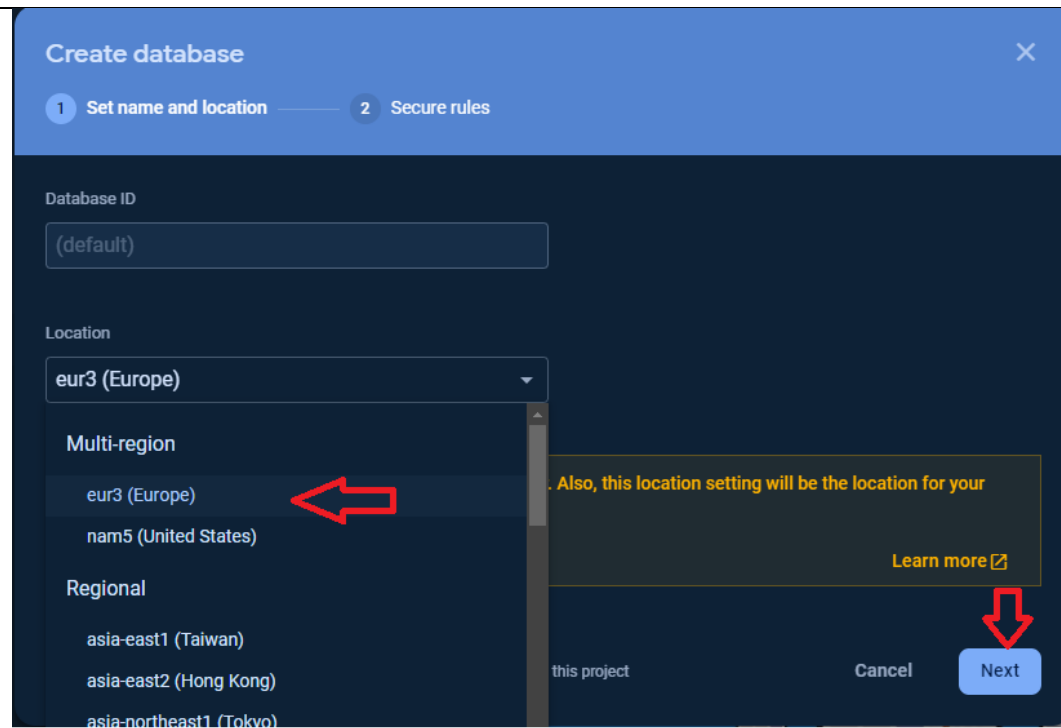
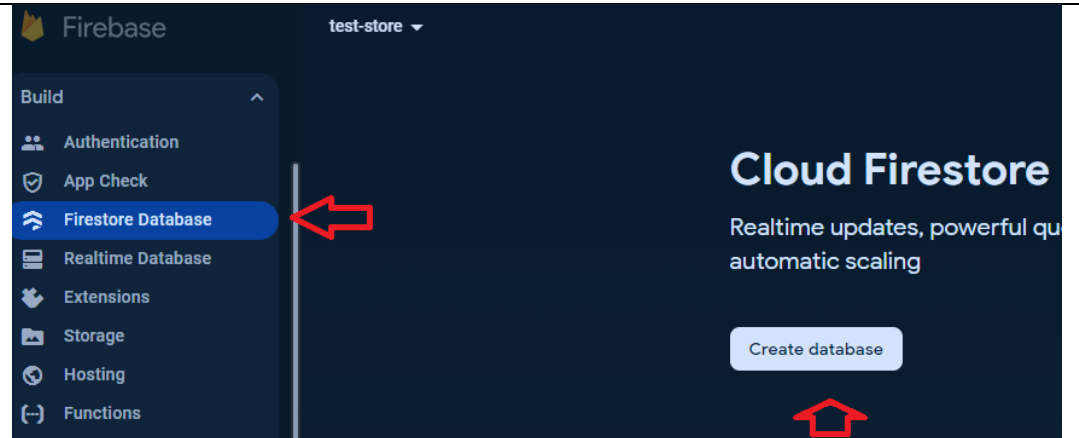
test-store

✔

Your new project is ready

Continue

Додаємо Firestore Database



	<div><div>Create database</div><div><div>✓ Set name and location</div><div>2 Secure rules</div></div><div>After you define your data structure, you will need to write rules to secure your data. <a href="#">Learn more</a></div><div><div><div><div></div><div>Start in production mode</div></div><div>Your data is private by default. Client read/write access will only be granted as specified by your security rules.</div></div><div><div><div></div><div>Start in test mode</div></div><div>Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.</div></div></div><div><pre>rules_version = '2';  service cloud.firestore {   match /databases/{database}/documents {     match /{document=**} {       allow read, write: if         request.time &lt; timestamp.date(2023, 12, 23);     }   } }</pre><div><div>!</div><div>The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days</div></div></div><div><div>Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project</div><div>Cancel</div><div>Enable</div></div></div>
	<div><div><div>Home icon</div><div>More in Google Cloud</div></div><div><div>(default)</div><div>+ Start collection</div></div></div>
Створюємо колекцію даних	<div><div><div>Home icon</div><div>More in Google Cloud</div></div><div><div>(default)</div><div>+ Start collection</div></div></div>

Задаємо ім'я колекції

Start a collection

1 Give the collection an ID — 2 Add its first document

Parent path

/

Collection ID ?

products

Cancel Next

1)Визначаємо спосіб задання id як автоматичний  
2)Додаємо документ (задаємо поля документа(назва, тип, значення)

Можемо задати вручну декілька документів

Start a collection

✓ Give the collection an ID — 2 Add its first document

Document parent path

/products

Document ID ?

Auto-ID

1 Required

Field	Type	Value
title	string	
price	number	
imgSrc	string	

Cancel Save

Add a document

Parent path

/products

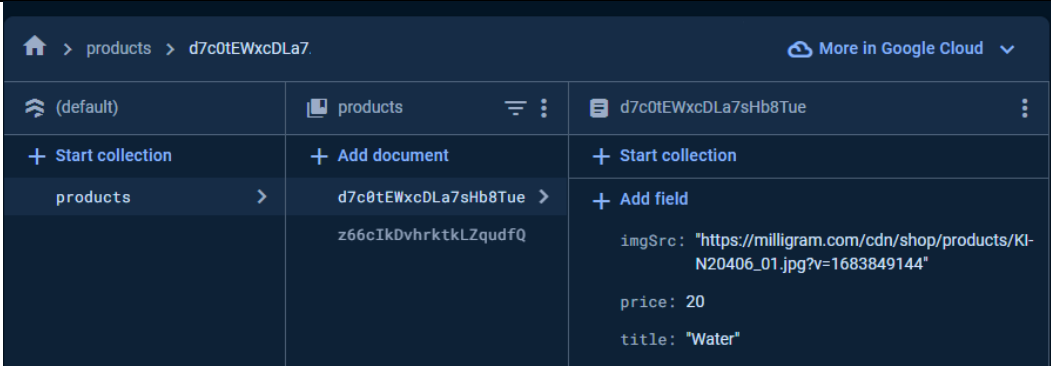
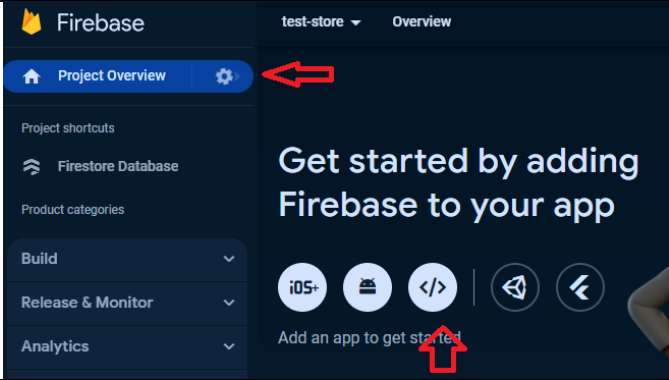
Document ID ?

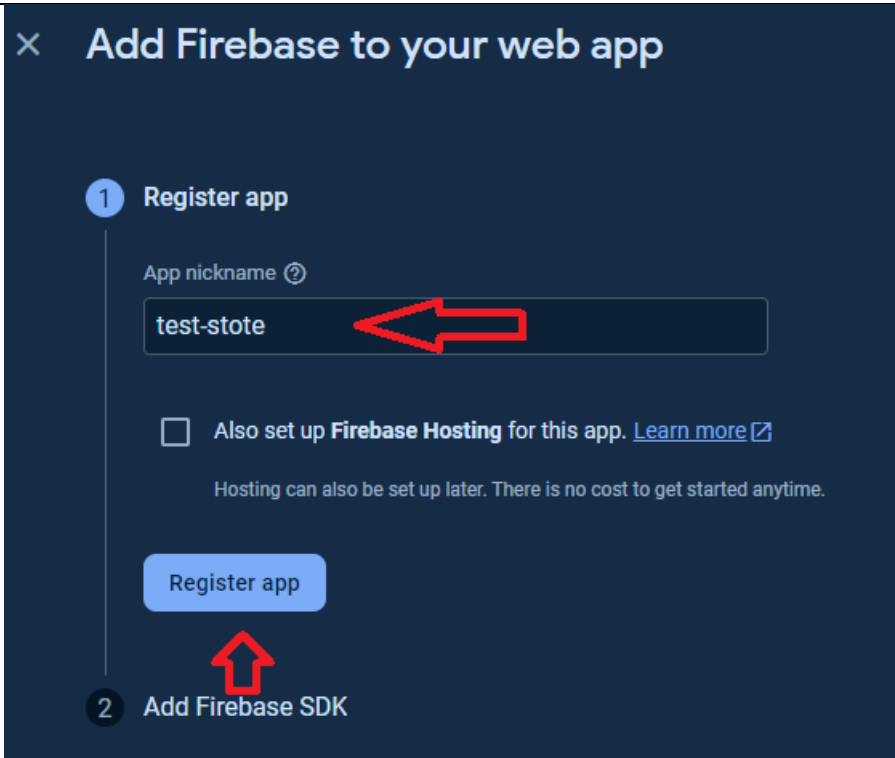
Auto-ID

1 Required

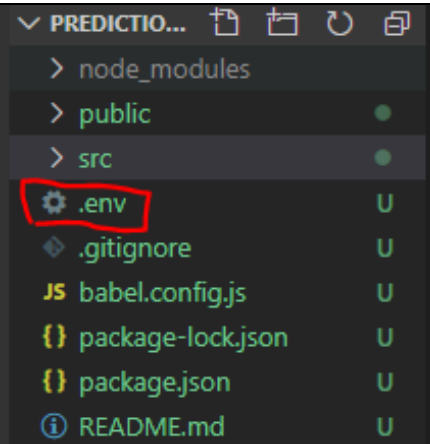
Field	Type	Value
title	string	Coffee
price	number	450
imgSrc	string	https://img.freep

Cancel Save

		
Додаємо firebase до проєкта		

<p>Задаємо назву і реєструємо</p>	
<p>Встановлюємо необхідні модулі</p> <p><a href="https://firebase.google.com/docs/web/setup?hl=en&amp;authuser=0&amp;_gl=1*_ga*MjA4NDYzOTg2Ny4xNzAwNjczMzM5*_ga_CW55HF8NVT*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA">https://firebase.google.com/docs/web/setup?hl=en&amp;authuser=0&amp;_gl=1*_ga*MjA4NDYzOTg2Ny4xNzAwNjczMzM5*_ga_CW55HF8NVT*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA</a>.</p>	<pre>npm install firebase</pre>
<p>Створюємо файл .env</p>	<p>Додаємо наш додаток в Firebase проект</p> <p>дані з Вашого проекту</p> <p>VUE_APP_FIREBASE_AUTH_DOMAIN= дані з Вашого проекту</p> <p>= дані з Вашого проекту</p> <p>VUE_APP_FIREBASE_PROJECT_ID= дані з Вашого проекту</p> <p>VUE_APP_FIREBASE_STORAGE_BUCKET= дані з Вашого проекту</p> <p>VUE_APP_FIREBASE_MESSAGE_SENDER_ID= дані з Вашого проекту</p> <p>VUE_APP_FIREBASE_APP_ID= дані з Вашого проекту</p>





<https://cli.vuejs.org/ru/guide/mode-and-env.html#%D0%BF%D0%B5%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5-%D0%BE%D0%BA%D1%80%D1%83%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F>

## Копіюємо з сайту

Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

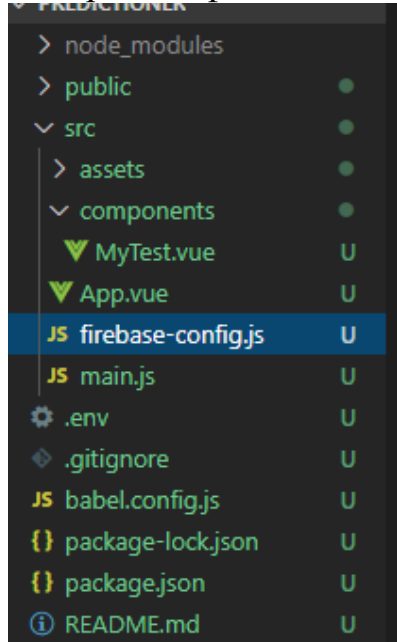
```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyA...",
  authDomain: "...",
  projectId: "...",
  storageBucket: "...",
  messagingSenderId: "...",
  appId: "..."
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```



## Створюємо файл налаштувань firebase



Або для початку можна просто скопіювати firebaseConfig з сайту без створення файлу .env

```
import { initializeApp } from 'firebase/app'
import { getFirestore } from 'firebase/firestore/lite'

const firebaseConfig = {
  apiKey: process.env.VUE_APP_FIREBASE_API_KEY,
  authDomain: process.env.VUE_APP_FIREBASE_AUTH_DOMAIN,
  projectId: process.env.VUE_APP_FIREBASE_PROJECT_ID,
  storageBucket: process.env.VUE_APP_FIREBASE_STORAGE_BUCKET,
  messagingSenderId: process.env.VUE_APP_FIREBASE_MESSAGE_SENDER_ID,
  appId: process.env.VUE_APP_FIREBASE_APP_ID
};

const app = initializeApp(firebaseConfig)
const db = getFirestore(app)
export default db
```

## ----- ОПЕРАЦІЇ З БАЗОЮ ДАНИХ -----

```
//---- отримуємо посилання на базу даних
import firebaseDB from '@/firebase-config'
//---- імпортуємо методи для роботи з даними
import { doc, collection, getDocs, addDoc, deleteDoc, updateDoc, query, where } from 'firebase/firestore/lite'

//---- отримуємо посилання на колекцію
this.dbCollection = collection(firebaseDB, `назва колекції`)
```

### Зчитування даних

[https://firebase.google.com/docs/web/setup?hl=en&authuser=0&\\_gl=1\\*1as5592\\*\\_ga\\*MjA4NDYzOTg2Ny4xNzAwNjczMzM5\\*\\_ga\\_CW55HF8NVT\\*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA](https://firebase.google.com/docs/web/setup?hl=en&authuser=0&_gl=1*1as5592*_ga*MjA4NDYzOTg2Ny4xNzAwNjczMzM5*_ga_CW55HF8NVT*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA).

```
getDocs(this.dbCollection)
  .then((querySnapshot) => {
    const list = []
    querySnapshot.docs.forEach((doc) => {
      list.push({
        id: doc.id,
```

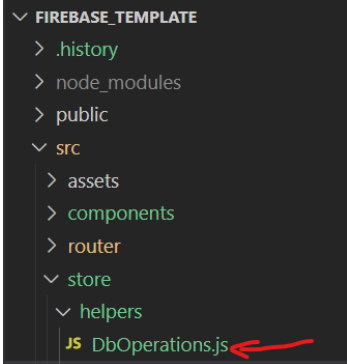
<a href="https://firebase.google.com/docs/reference/js/firestore_lite.md?hl=ru#getdocs">https://firebase.google.com/docs/reference/js/firestore_lite.md?hl=ru#getdocs</a>	<pre>         ...doc.data(),     })     })     resolve(list)   })   .catch((error) =&gt; {     reject(error)   }) </pre>
<p>Додавання даних</p> <a href="https://firebase.google.com/docs/reference/js/firestore_lite.md?hl=ru#adddoc">https://firebase.google.com/docs/reference/js/firestore_lite.md?hl=ru#adddoc</a>	<pre> addDoc(this.dbCollection, item)   .then(() =&gt; {     resolve(true)   })   .catch((error) =&gt; {     reject(error)   }) </pre>
<p>Видалення даних</p> <a href="https://firebase.google.com/docs/reference/js/firestore_lite.md?hl=ru#deletedoc">https://firebase.google.com/docs/reference/js/firestore_lite.md?hl=ru#deletedoc</a>	<pre> deleteDoc(doc(this.dbCollection, id))   .then(() =&gt; {     resolve(true)   })   .catch((error) =&gt; {     reject(error)   }) </pre>
<p>Модифікація даних</p> <a href="https://firebase.google.com/docs/reference/js/firestore_lite.md?hl=ru#updatedoc_2">https://firebase.google.com/docs/reference/js/firestore_lite.md?hl=ru#updatedoc_2</a>	<pre> updateDoc(oldDocRef, data)   .then(() =&gt; {     resolve(true)   })   .catch((error) =&gt; {     reject(error)   }) </pre>
<p>Фільтрація</p> <a href="https://cloud.google.com/firestore/docs/query-data/queries">https://cloud.google.com/firestore/docs/query-data/queries</a>	<pre> loadFilteredData(fieldTitle, compareOperator, valueToCompare) {   const q = query(this.dbCollection, where(fieldTitle, compareOperator, valueToCompare))   return new Promise((resolve, reject) =&gt; { </pre>

```

    getDocs(q)
      .then((querySnapshot) => {
        const list = []
        querySnapshot.docs.forEach((doc) => {
          list.push({
            id: doc.id,
            ...doc.data(),
          })
        })
        resolve(list)
      })
      .catch((error) => {
        reject(error)
      })
    })
  }
}

```

Створюємо клас, що містить основні операції з базою



```

import firebaseDB from '@/firebase-config'
import { doc, collection, getDocs, addDoc, deleteDoc, updateDoc } from
'firebase/firestore/lite'

```

```

class DbOperations {
  constructor(collectionTitle) {
    this.dbCollection = collection(firebaseDB, `/${collectionTitle}`)
  }
  getListFromSnapshot(snapshot) {
    const list = []
    snapshot.docs.forEach((doc) => {
      list.push({
        id: doc.id,
        ...doc.data(),
      })
    })
    return list
  }
}

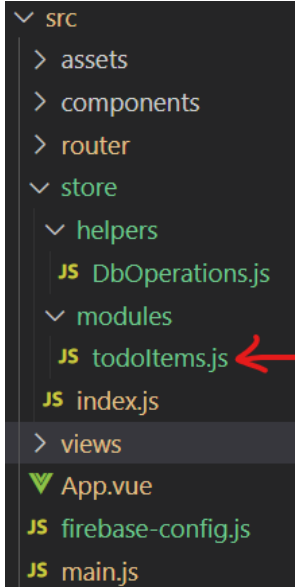
```

	<pre>loadItemsList() {   return new Promise((resolve, reject) =&gt; {     getDocs(this.dbCollection)       .then((querySnapshot) =&gt; {         resolve(this.getListFromSnapshot(querySnapshot))       })       .catch((error) =&gt; {         reject(error)       })   }) }</pre>
	<pre>addItem(item) {   return new Promise((resolve, reject) =&gt; {     addDoc(this.dbCollection, item)       .then(() =&gt; {         resolve(true)       })       .catch((error) =&gt; {         reject(error)       })   }) }</pre>
	<pre>deleteItem(id) {   return new Promise((resolve, reject) =&gt; {     deleteDoc(doc(this.dbCollection, id))       .then(() =&gt; {         resolve(true)       })   }) }</pre>

	<pre>        })         .catch((error) =&gt; {             reject(error)         })     }) } updateItem(itemId, data) {</pre>
	<pre>    return new Promise((resolve, reject) =&gt; {</pre>
	<pre>        const oldDocRef = doc(this.dbCollection, itemId)         updateDoc(oldDocRef, data)             .then(() =&gt; {                 resolve(true)             })             .catch((error) =&gt; {                 reject(error)             })     }) } loadFilteredData(fieldTitle, compareOperator, valueToCompare) {     const q = query(this.dbCollection, where(fieldTitle, compareOperator, valueToCompare))     return new Promise((resolve, reject) =&gt; {         getDocs(q)             .then((querySnapshot) =&gt; {                 resolve(this.getListFromSnapshot(querySnapshot))             })             .catch((error) =&gt; {                 reject(error)             })     }) })</pre>

```
}  
}  
  
export default DbOperations
```

Створюємо модуль



```
import DbOperations from '../helpers/DbOperations'  
const collectionDB = new DbOperations('todo')  
export default {  
  namespaced: true,  
  state: () => ({  
    todoList: [],  
    loading: false,  
    error: null,  
  }),  
  getters: {  
    isLoading: (state) => state.loading,  
    hasError: (state) => state.error,  
  
    getItemList: (state) => state.todoList,  
    getItemById: (state) => (itemId) => state.todoList.find((item) =>  
item.id == itemId),  
  },  
  mutations: {  
    setItemList(state, itemsList) {  
      state.todoList = itemsList  
    },  
  
    setLoading(state, value) {  
      state.loading = value  
    },  
  },  
}
```

```
      setError(state, error) {
        state.error = error
      },
    },
    actions: {
      loadList({ commit }) {
        commit('setError', null)
        commit('setLoading', true)
        collectionDB
          .loadItemsList()
          .then((list) => {
            commit('setItemsList', list)
          })
          .catch((error) => {
            commit('setError', error)
          })
          .finally(() => {
            commit('setLoading', false)
          })
      },
      addItem({ commit, dispatch }, item) {
        commit('setError', null)
        commit('setLoading', true)
        collectionDB
          .addItem(item)
          .then(() => {
            dispatch('loadList')
          })
          .catch((error) => {
            commit('setError', error)
          })
          .finally(() => {
            commit('setLoading', false)
          })
      },
      deleteItem({ commit, dispatch }, itemId) {
        commit('setError', null)
      },
    },
  },
}
```



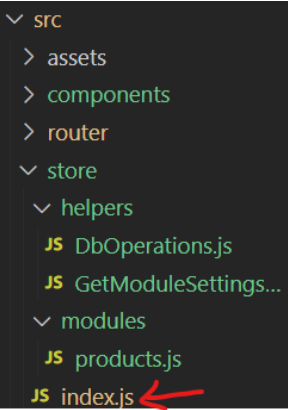
```

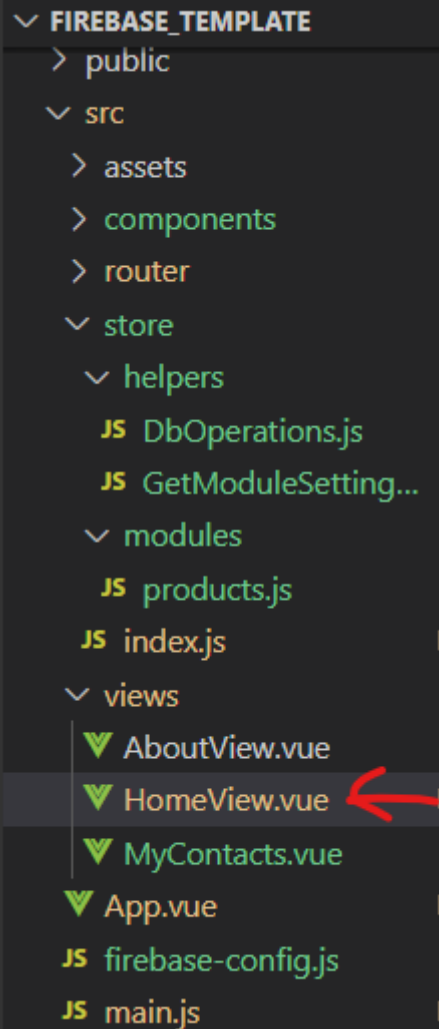
        commit('setLoading', true)

        collectionDB
            .deleteItem(itemId)
            .then(() => {
                dispatch('loadList')
            })
            .catch((error) => {
                commit('setError', error)
            })
            .finally(() => {
                commit('setLoading', false)
            })
    },
    updateItem({ commit, dispatch }, { itemId, data }) {
        commit('setError', null)
        commit('setLoading', true)

        collectionDB
            .updateItem(itemId, data)
            .then(() => {
                dispatch('loadList')
            })
            .catch((error) => {
                commit('setError', error)
            })
            .finally(() => {
                commit('setLoading', false)
            })
    },
    loadFilteredData({ commit }, { fieldTitle, compareOperator,
valueToCompare }) {
        commit('setError', null)
        commit('setLoading', true)
        collectionDB
            .loadFilteredData(fieldTitle, compareOperator,
valueToCompare)

```

	<pre>         .then((list) =&gt; {             commit('setItemsList', list)         })         .catch((error) =&gt; {             commit('setError', error)         })         .finally(() =&gt; {             commit('setLoading', false)         })     }, }, } </pre>
<p>Підключаємо модуль</p> 	<pre> import { createStore } from 'vuex' import todoItems from './modules/todoItems' export default createStore({     namespaces: true,      modules: {         todoItems,     }, }) </pre>
<p>Приклад використання стору</p>	<pre> &lt;template&gt;   &lt;div v-if="isLoading"&gt;Loading .....&lt;/div&gt;   &lt;div v-else-if="hasError"&gt;Вибачте. Сталась помилка&lt;/div&gt;   &lt;div v-else&gt;     &lt;div v-for="item in getItemsList" :key="item.id"&gt;       {{ item.title }} - {{ item.price }}       &lt;button @click="deleteItem(item.id)"&gt;Delete&lt;/button&gt;       &lt;button @click="onAddPrice(item)"&gt;Add 100 to price&lt;/button&gt;     &lt;/div&gt;   &lt;/div&gt; &lt;/template&gt; </pre>



[Home](#) | [About](#) | [My Contacts](#)

test1 - 400100

test3 - 203

---

title

price

---

price to find

```
<div>
  <div>
    title
    <input v-model="product.title" type="text" />
  </div>
  <div>
    price
    <input v-model="product.price" type="number" />
  </div>
  <!-- <div>
    image src
    <input v-model="product.imgSrc" type="text" />
  </div> -->
  <button @click="addItem(product)">Add</button>
</div>
<hr />
<div>
  price to find
  <input v-model="targetPrice" type="number" />
</div>
<button @click="loadFilteredDataByPrice">Filter</button>
<button @click="loadList">Clear Filter</button>
</div>
</template>

<script>
import { mapGetters, mapActions } from 'vuex'

export default {
  name: 'HomeView',
  data() {
    return {
      product: {},
      targetPrice: null,
    }
  },
  computed: {
```

```
        ...mapGetters('todoItems', ['getItemsList', 'isLoading',
'hasError'])),
    },
    created() {
        this.loadList()
    },
    methods: {
        ...mapActions('todoItems', ['loadList', 'addItem', 'deleteItem',
'updateItem', 'loadFilteredData']),
        onAddPrice(item) {
            this.updateItem({
                itemId: item.id,
                data: {
                    price: item.price + 100,
                },
            })
        },
        loadFilteredDataByPrice() {
            this.loadFilteredData({ fieldTitle: 'price', compareOperator:
'==', valueToCompare: this.targetPrice })
        },
    },
}
</script>
```