

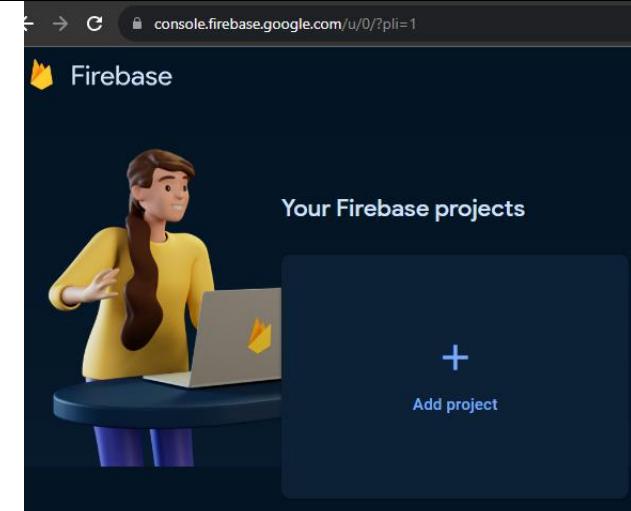
## CLI. Робота з Firestore Database в Firebase

# Add Firebase to your JavaScript project

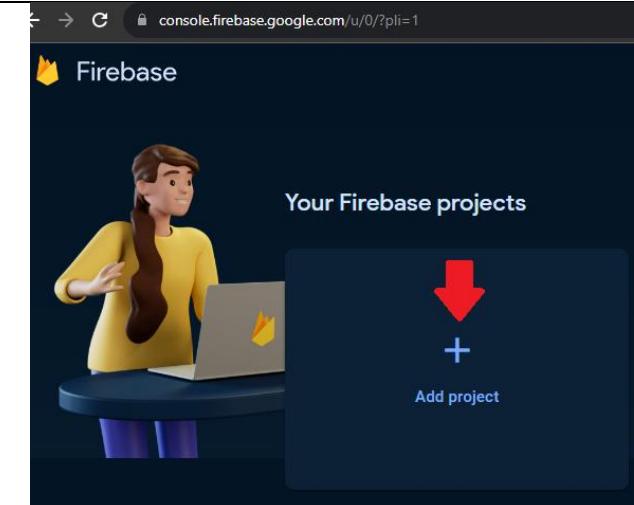
[https://firebase.google.com/docs/web/setup?hl=en&authuser=0& gl=1\\*1as5592\\* ga\\*MjA4NDYzOTg2Ny4xNzAwNjczMzM5\\* ga CW55HF8NVT\\*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA.](https://firebase.google.com/docs/web/setup?hl=en&authuser=0& gl=1*1as5592* ga*MjA4NDYzOTg2Ny4xNzAwNjczMzM5* ga CW55HF8NVT*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA.)

Переходимо у консоль firebase

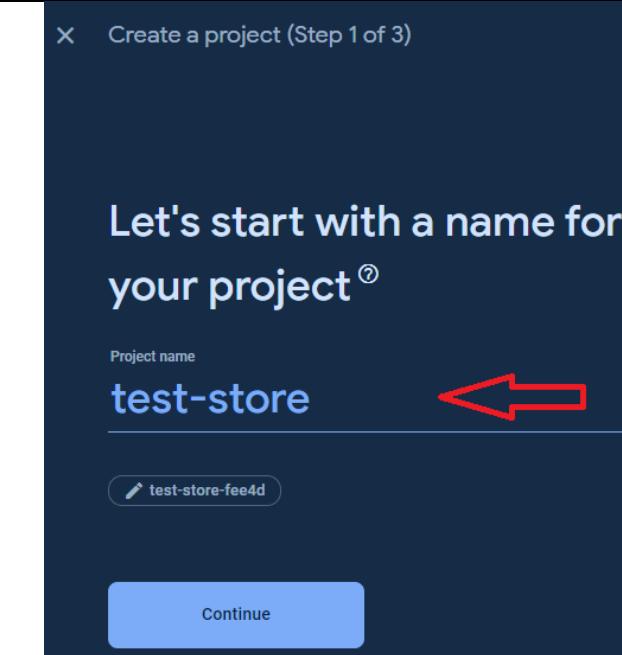
<https://console.firebaseio.google.com/u/0/?pli=1>



Створюємо проект в Firebase



Задаємо якусь назву проекта



Відключаємо аналітику

>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:

- A/B testing ⓘ
- Crash-free users ⓘ
- User segmentation & targeting across Firebase products ⓘ
- Event-based Cloud Functions triggers ⓘ
- Free unlimited reporting ⓘ

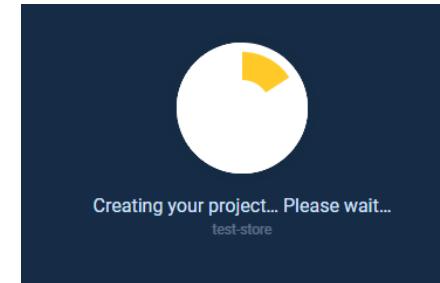
Enable Google Analytics for this project  
Recommended



Previous

Create project

Створюємо проект



test-store

Your new project is ready

Continue

## Додаємо Firestore Database

The screenshot shows the Firebase console interface. On the left, there's a sidebar with a tree view under the 'Build' category. The 'Firestore Database' item is highlighted with a blue background and has a red arrow pointing to it from the top right. To the right of the sidebar is a large banner for 'Cloud Firestore' with the text 'Realtime updates, powerful querying, automatic scaling'. Below the banner is a 'Create database' button and a house icon. The main area is titled 'Create database' and shows two steps: '1 Set name and location' and '2 Secure rules'. Under 'Database ID', '(default)' is selected. Under 'Location', 'eur3 (Europe)' is selected, with a red arrow pointing to it from the bottom left. A note says 'Also, this location setting will be the location for your data in this project.' At the bottom right are 'Cancel' and 'Next' buttons, with a red arrow pointing to the 'Next' button.

Firebase

test-store ▾

Build

- Authentication
- App Check
- Firestore Database**
- Realtime Database
- Extensions
- Storage
- Hosting
- Functions

Cloud Firestore

Realtime updates, powerful querying, automatic scaling

Create database

House icon

Create database

1 Set name and location 2 Secure rules

Database ID  
(default)

Location  
eur3 (Europe)

Multi-region  
eur3 (Europe) **nam5 (United States)**

Regional  
asia-east1 (Taiwan)  
asia-east2 (Hong Kong)  
asia-northeast1 (Tokyo)

Also, this location setting will be the location for your data in this project

Learn more ⓘ

Cancel Next

## Create database

Set name and location

Secure rules

After you define your data structure, **you will need to write rules to secure your data.**

[Learn more](#)

Start in **production mode**

Your data is private by default. Client read/write access will only be granted as specified by your security rules.

Start in **test mode**

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.

```
rules_version = '2';

service cloud.firestore {
    match /databases/{database}/documents {
        match /{document=**} {
            allow read, write: if
                request.time < timestamp.date(2023, 12, 23);
        }
    }
}
```

**! 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**

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project

[Cancel](#)

[Enable](#)

Створюємо колекцію даних

More in Google Cloud ▾

(default)

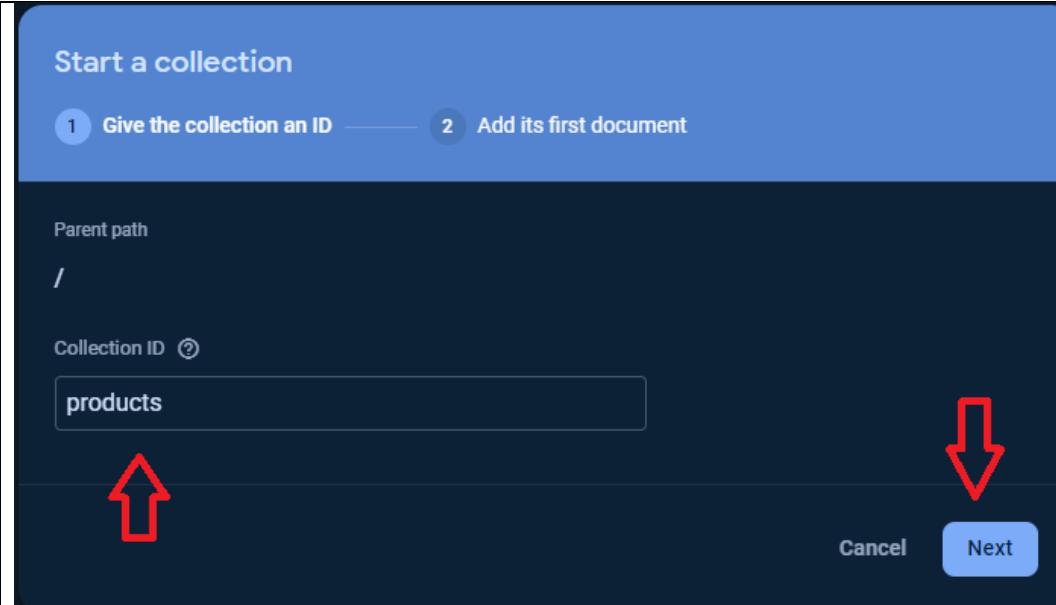
+ Start collection

More in Google Cloud ▾

default

+ Start collection

Задаємо ім'я колекції



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

Можемо задати вручну декілька документів

Start a collection

1 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	Coffee
price	number	450
imgSrc	string	https://img.freep

Add a document

Parent path /products

Document ID ? Auto-ID

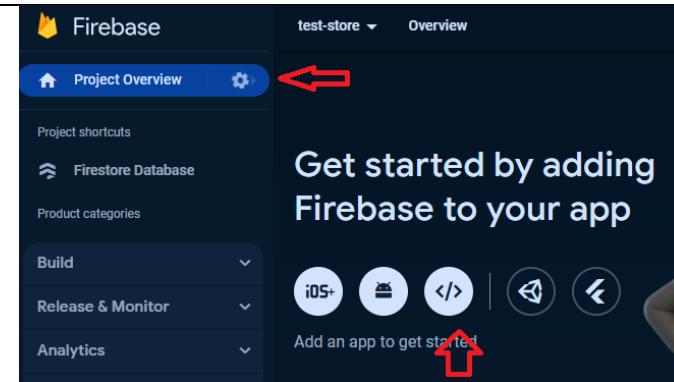
1 Required

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

Cancel Save

<a href="#">More in Google Cloud</a>
(default)
+ Start collection
products
d7c0tEWxcDLa7sHb8Tue
+ Add document
d7c0tEWxcDLa7sHb8Tue >
z66cIkDvhrtkLZqudfQ
+ Start collection
+ Add field
imgSrc: "https://milligram.com/cdn/shop/products/KI-N20406_01.jpg?v=1683849144"
price: 20
title: "Water"

Додаємо firebase до проекта



Задаємо назву і реєструємо

## X Add Firebase to your web app

### 1 Register app

App nickname ?

test-stote



Also set up **Firebase Hosting** for this app. [Learn more ↗](#)

Hosting can also be set up later. There is no cost to get started anytime.

Register app

### 2 Add Firebase SDK



Встановлюємо необхідні модулі

[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).

npm install firebase

Створюємо файл .env

Додаємо наш додаток в Firebase проект

дані з Вашого проекту

VUE\_APP\_FIREBASE\_AUTH\_DOMAIN= дані з Вашого проекту  
= дані з Вашого проекту

VUE\_APP\_FIREBASE\_PROJECT\_ID= дані з Вашого проекту

VUE\_APP\_FIREBASE\_STORAGE\_BUCKET= дані з Вашого проекту

VUE\_APP\_FIREBASE\_MESSAGE\_SENDER\_ID= дані з Вашого проекту

VUE\_APP\_FIREBASE\_APP\_ID= дані з Вашого проекту

✓ PREDICTIO...	File	Folder	Open	Save
> node_modules				
> public				
> src				
⚙️ .env	U			
↳ .gitignore	U			
JS babel.config.js	U			
{ } package-lock.json	U			
{ } package.json	U			
ⓘ README.md	U			

[#%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%BD%D1%8F](https://cli.vuejs.org/ru/guide/mode-and-env.html)

## Копіюємо з сайту

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

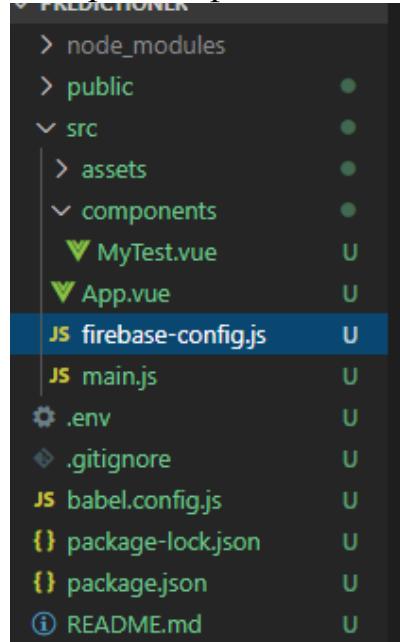
```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: [REDACTED],
  authDomain: [REDACTED],
  projectId: [REDACTED],
  storageBucket: [REDACTED],
  messagingSenderId: [REDACTED],
  appId: [REDACTED]
};

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



## Створюємо файл налаштувань firebase



```
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
```

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

## ----- 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,
```

[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)

```
        ...doc.data(),
      })
    })
  .then(list =>
    resolve(list)
  )
  .catch((error) => {
    reject(error)
  })
}
```

Додавання даних

[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)

```
addDoc(this.dbCollection, item)
  .then(() => {
    resolve(true)
  })
  .catch((error) => {
    reject(error)
  })
}
```

Видалення даних

[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)

```
deleteDoc(doc(this.dbCollection, id))
  .then(() => {
    resolve(true)
  })
  .catch((error) => {
    reject(error)
  })
}
```

Модифікація даних

[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)

```
updateDoc(oldDocRef, data)
  .then(() => {
    resolve(true)
  })
  .catch((error) => {
    reject(error)
  })
}
```

Фільтрація

<https://cloud.google.com/firestore/docs/query-data/queries>

```
loadFilteredData(fieldTitle, compareOperator, valueToCompare) {
  const q = query(this.dbCollection, where(fieldTitle, compareOperator,
  valueToCompare))
  return new Promise((resolve, reject) => {
```

```
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

```
~ FIREBASE TEMPLATE
> .history
> node_modules
> public
< src
  > assets
  > components
  > router
  < store
    < helpers
      js DbOperations.js ←
```

```
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
    }
}
```

```
loadItemsList() {
    return new Promise((resolve, reject) => {
        getDocs(this.dbCollection)
            .then((querySnapshot) => {
                resolve(this.getListFromSnapshot(querySnapshot))
            })
            .catch((error) => {
                reject(error)
            })
    })
}
```

```
addItem(item) {
    return new Promise((resolve, reject) => {
        addDoc(this.dbCollection, item)
            .then(() => {
                resolve(true)
            })
            .catch((error) => {
                reject(error)
            })
    })
}
```

```
deleteItem(id) {
    return new Promise((resolve, reject) => {
        deleteDoc(doc(this.dbCollection, id))
            .then(() => {
                resolve(true)
            })
            .catch((error) => {
                reject(error)
            })
    })
}
```

```
        })
        .catch((error) => {
            reject(error)
        })
    })
}
updateItem(itemId, data) {
    return new Promise((resolve, reject) => {
```

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

```
        }
    }

export default DbOperations
```

#### Створюємо модуль

```
< src
  > assets
  > components
  > router
  < store
    < helpers
      JS DbOperations.js
    < modules
      JS todoltems.js ←
      JS index.js
    > views
      V App.vue
      JS firebase-config.js
      JS main.js
```

```
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,
        getItemsList: (state) => state.todoList,
        getItemById: (state) => (itemId) => state.todoList.find((item) =>
            item.id == itemId),
    },
    mutations: {
        setItemsList(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
            .loadItemList()
            .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)
```

```
.then((list) => {
    commit('setItemsList', list)
})
.catch((error) => {
    commit('setError', error)
})
.finally(() => {
    commit(' setLoading', false)
})
},
},
}
```

#### Підключаємо модуль

```
<src
  > assets
  > components
  > router
  <store
    <helpers
      JS DbOperations.js
      JS GetModuleSettings...
    <modules
      JS products.js
    JS index.js<img alt="Red arrow pointing to index.js" data-bbox="58 675 135 685"/>
```

```
import { createStore } from 'vuex'
import todoItems from './modules/todoItems'
export default createStore({
    namespaced: true,
    modules: {
        todoItems,
    },
})
```

#### Приклад використання сторінки

```
<template>
  <div v-if="isLoading">Loading .....</div>
  <div v-else-if="hasError">Вибачте. Сталась помилка</div>
  <div v-else>
    <div v-for="item in getItemsList" :key="item.id">
      {{ item.title }} - {{ item.price }}
      <button @click="deleteItem(item.id)">Delete</button>
      <button @click="onAddPrice(item)">Add 100 to price</button>
    </div>
    <hr />
```

```

FIREBASE_TEMPLATE
  > public
  <!-- src -->
    > assets
    > components
    > router
    <!-- store -->
      <!-- helpers -->
        JS DbOperations.js
        JS GetModuleSetting...
      <!-- modules -->
        JS products.js
        JS index.js
    <!-- views -->
      <!-- AboutView.vue -->
      <!-- HomeView.vue --> ←
      <!-- MyContacts.vue -->
      <!-- App.vue -->
      JS firebase-config.js
      JS main.js

```

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

test1 - 400100	<a href="#">Delete</a>	<a href="#">Add 100 to price</a>
test3 - 203	<a href="#">Delete</a>	<a href="#">Add 100 to price</a>
title <input type="text"/> price <input type="text"/> <input type="button" value="Add"/>		
price to find <input type="text"/> <input type="button" value="Filter"/> <input type="button" value="Clear Filter"/>		

```

<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>
```