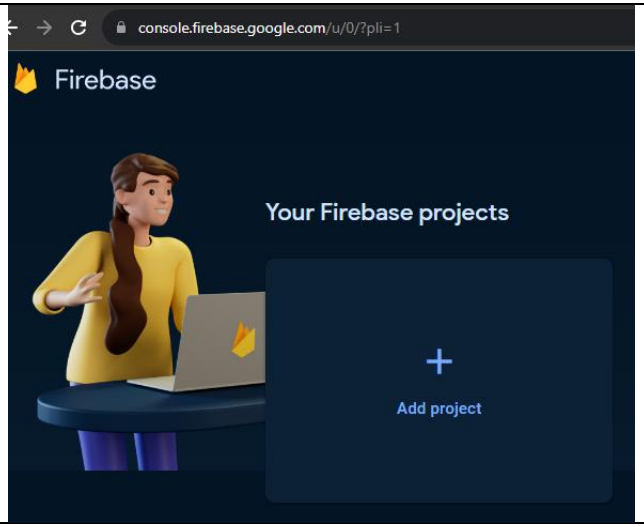
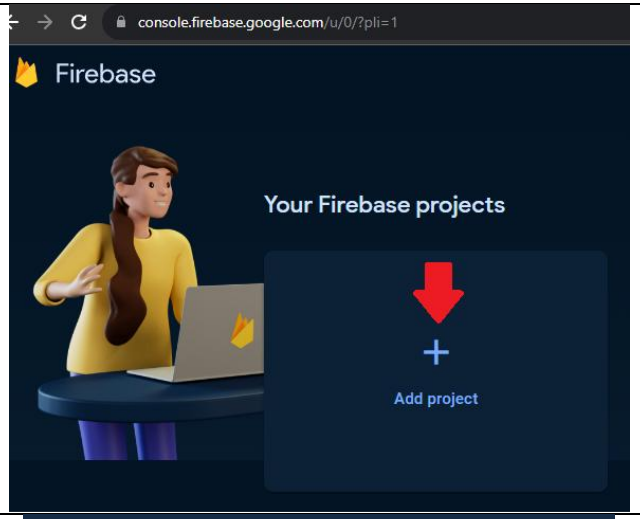
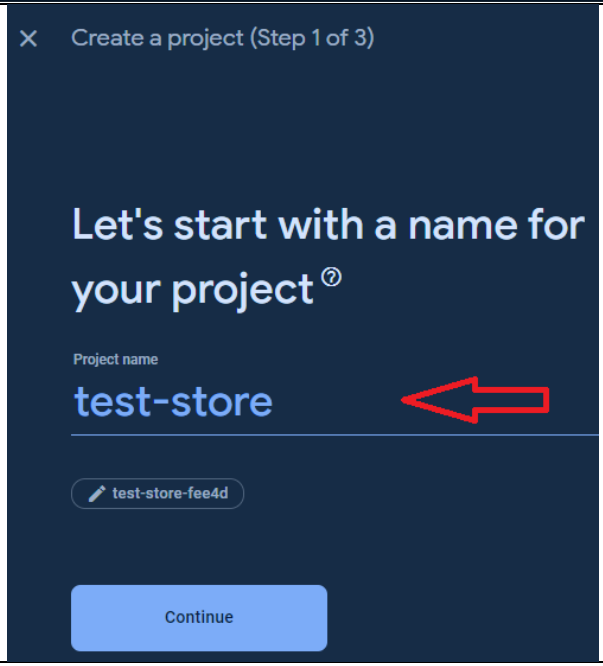
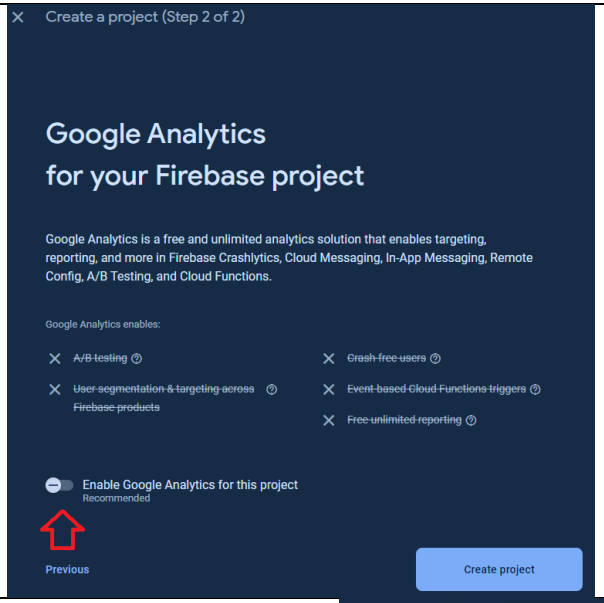
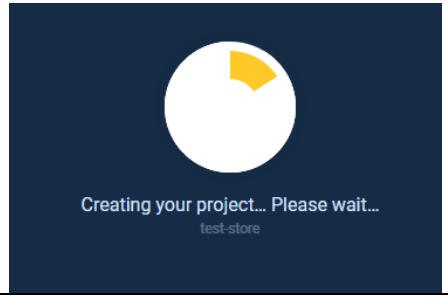
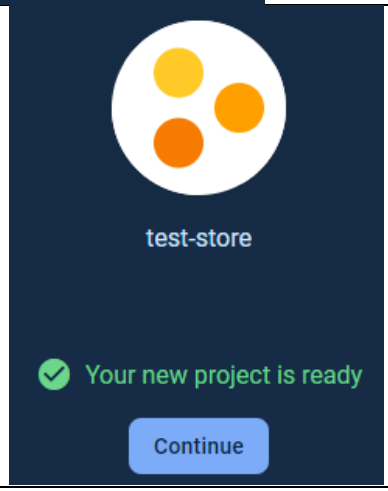
CLI. Робота з Firestore Database в Firebase

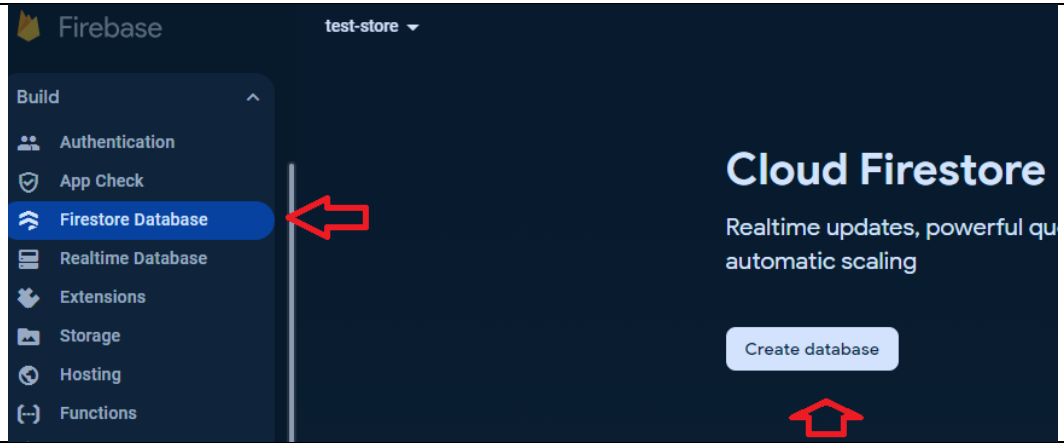
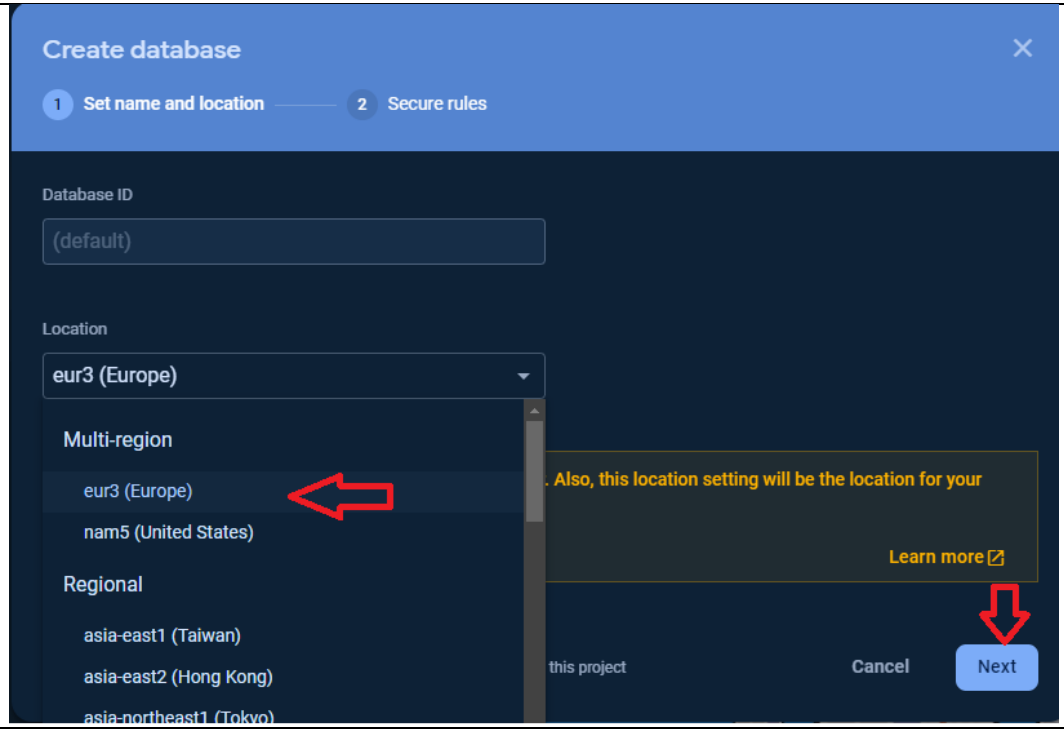
Add Firebase to your JavaScript project

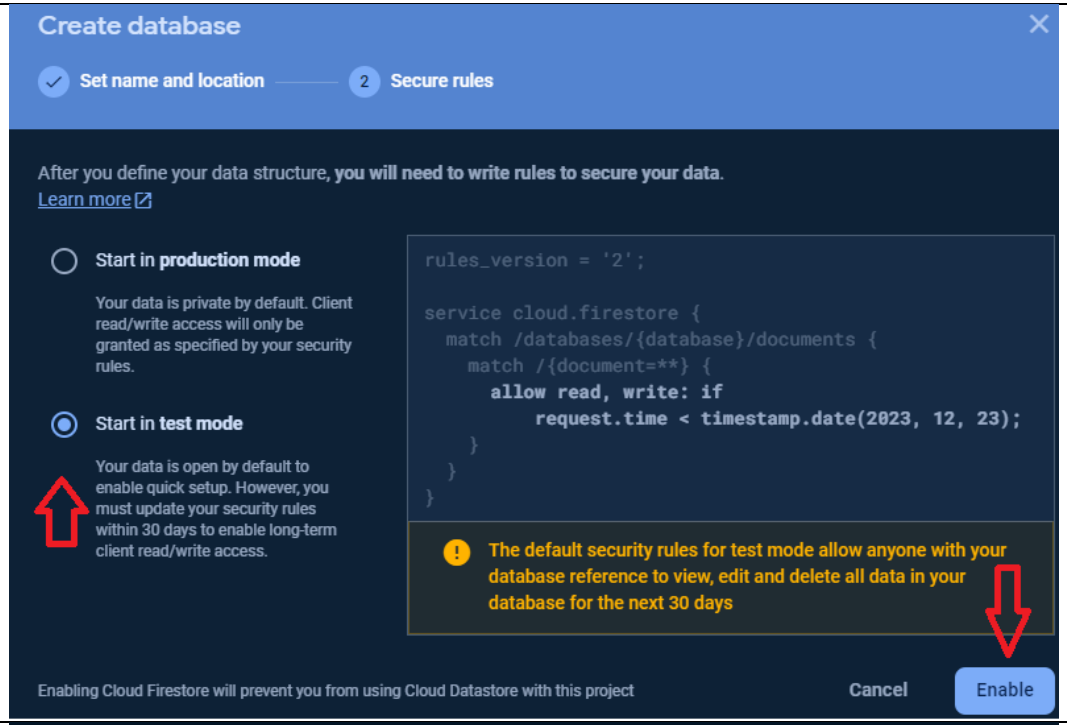
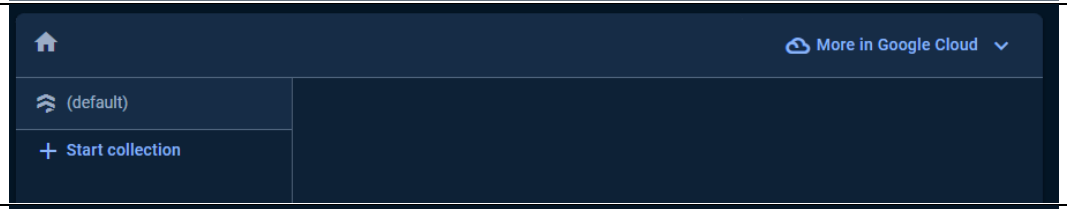
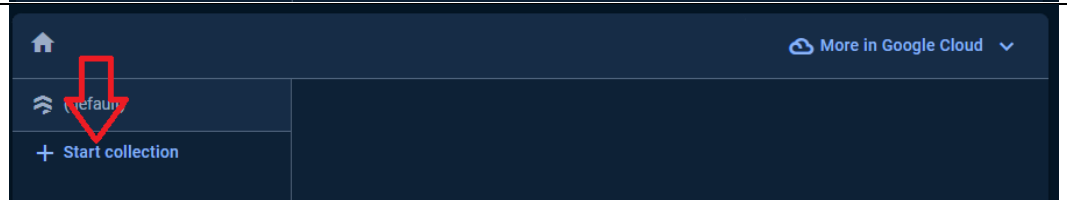
https://firebase.google.com/docs/web/setup?hl=en&authuser=0&_gl=1*1as5592*_ga*MjA4NDYzOTg2Ny4xNzAwNjczMzM5*_ga_CW55HF8NVT*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA.

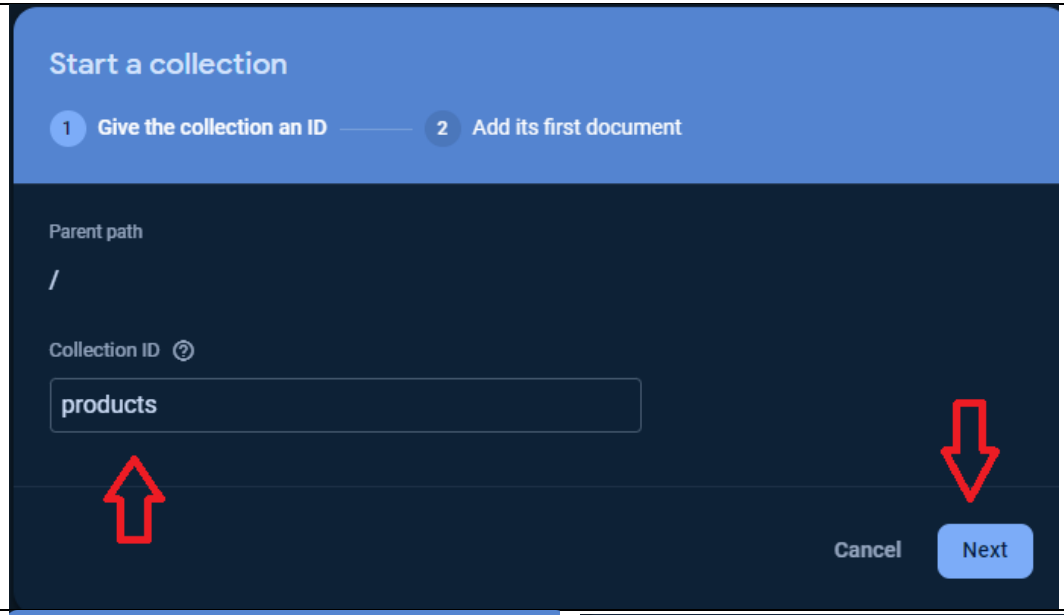
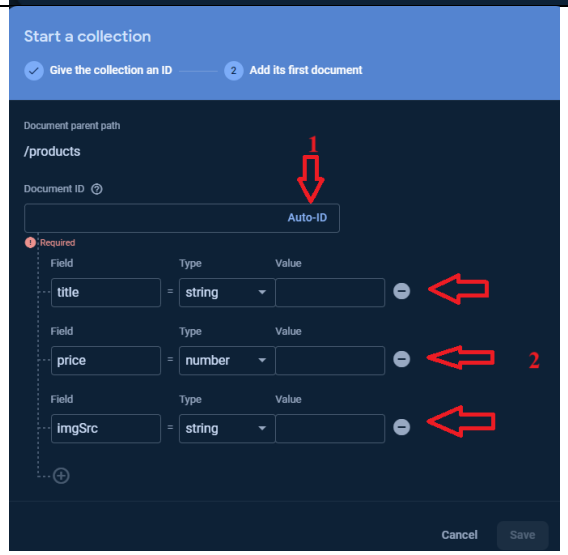
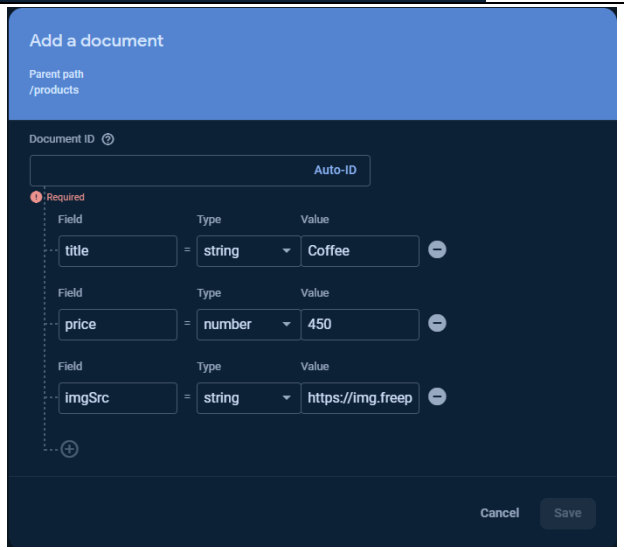
Переходимо у консоль firebase https://console.firebase.google.com/u/0/?pli=1		

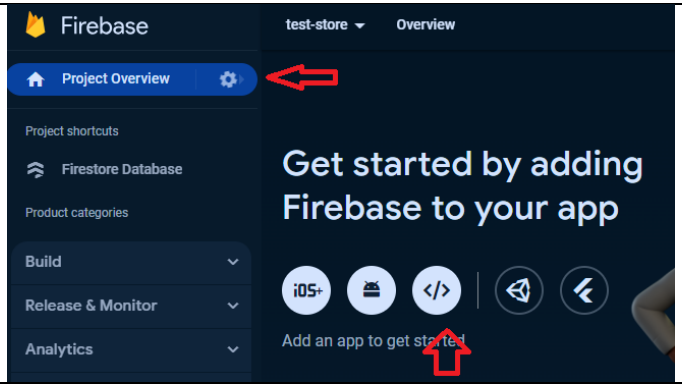
Створюємо проект в Firebase		
Задаємо якусь назву проєкта		

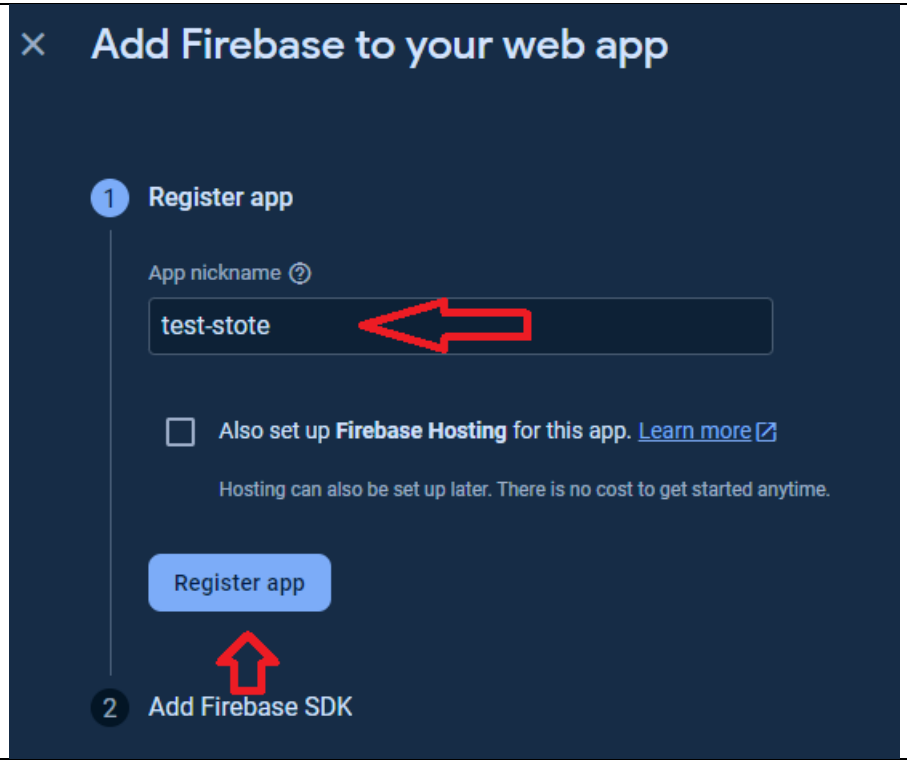
Відключаємо аналітику	
Створюємо проєкт	 

Додаємо Firestore Database	
	

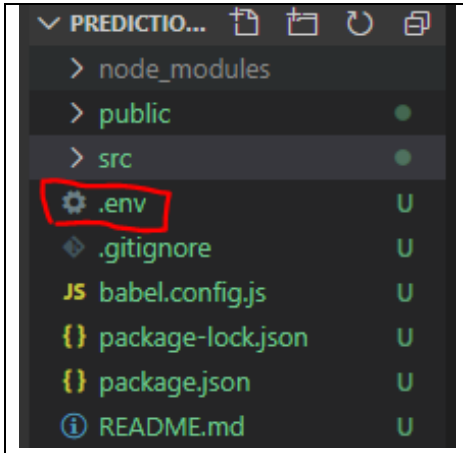
	
	
Створюємо колекцію даних	

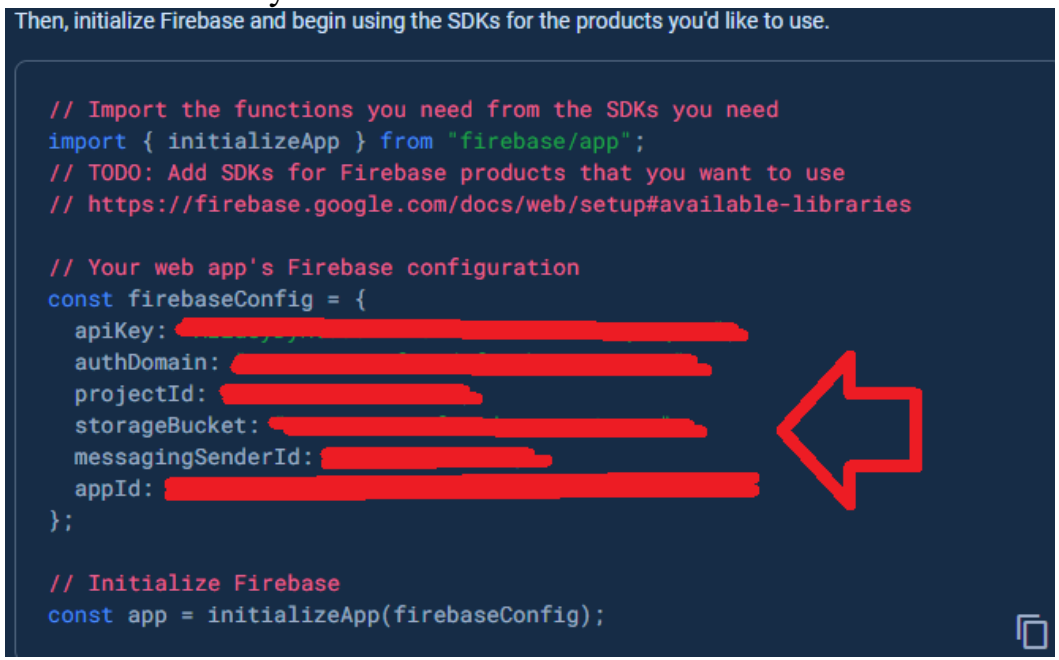
Задаємо ім’я колекції	
1)Визначаємо спосіб задання id як автоматичний 2)Додаємо документ (задаємо поля документа(назва, тип, значення) Можемо задати вручну декілька документів	 

<table>
<tr><td></td><td> 
</td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>Додаємо firebase до проєкта</td><td></td><td> 
</td><td></td></tr>
</table>

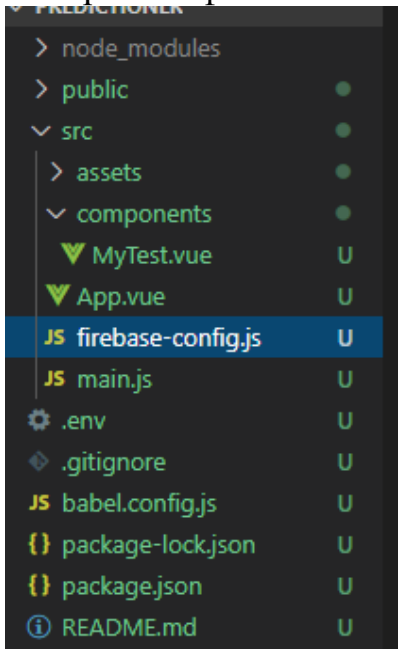
<table>
<tr><td>Задаємо назву і реєструємо</td><td> 
</td></tr>
<tr><td>Встановлюємо необхідні модулі
https://firebase.google.com/docs/web/setup?hl=en&authuser=0&_gl=1*1as5592*_ga*MjA4NDYzOTg2Ny4xNzAwNjczMzM5*_ga_CW55HF8NVT*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA.</td><td><code>npm install firebase</code></td></tr>
<tr><td>Створюємо файл .env</td><td>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</td></tr>
</table>

<table>
<tr>
<td> 

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</td>
<td>Копіюємо з сайту
 
</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</table>

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`)
```

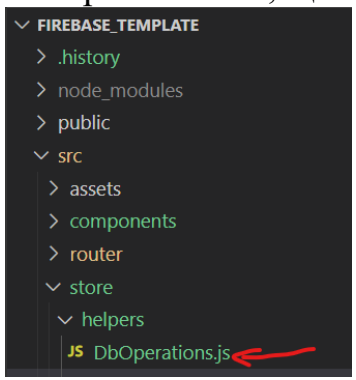
Зчитуваня даних
https://firebase.google.com/docs/web/setup?hl=en&authuser=0&_gl=1*1as5592*_ga*MjA4NDYzOTg2Ny4xNzAwNjczMzM5*_ga_CW55HF8NVT*MTcwMDczNDY2My40LjEuMTcwMDczNjIwOS42MC4wLjA.

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




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

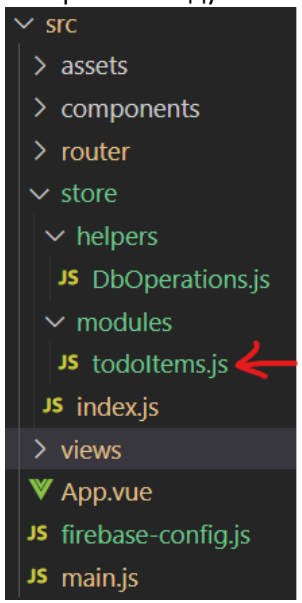
```
            const oldDocRef = doc(this.dbCollection, itemId)
            updateDoc(oldDocRef, data)
                .then(() => {
                    resolve(true)
                })
                .catch((error) => {
                    reject(error)
                })
        })
    }
    loadFilteredData(fieldTitle, 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




```
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
                .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)
```

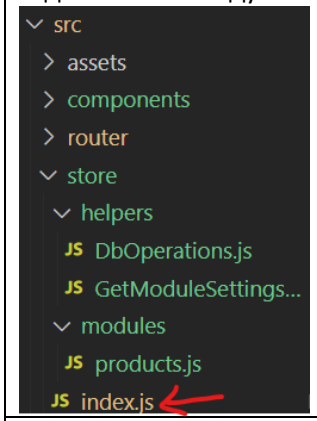
```
            commit('setLoading', true)

            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)
```

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

Підключаємо модуль

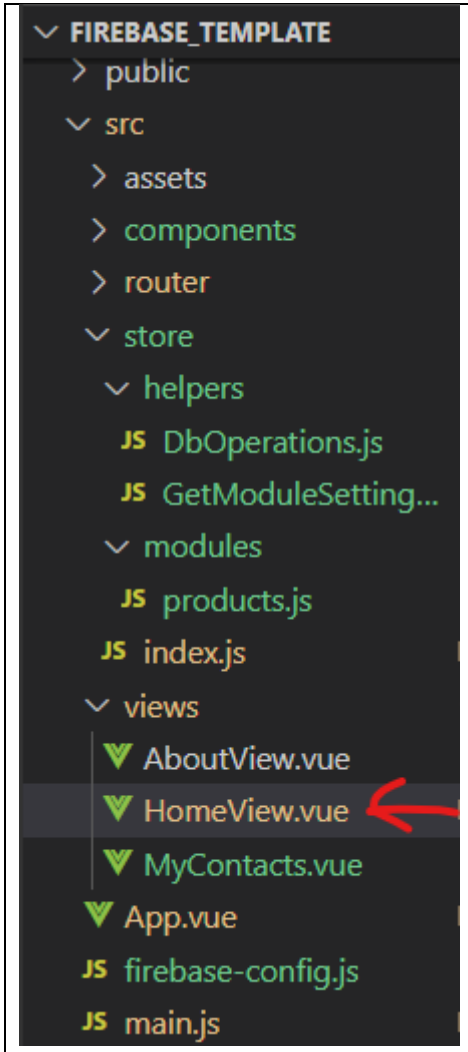


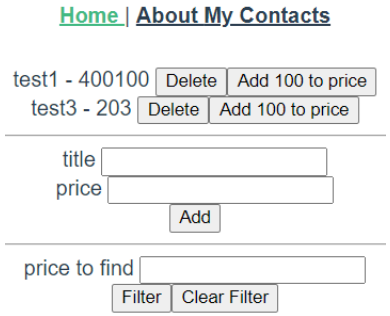

```
import { createStore } from 'vuex'
import todoItems from './modules/todoItems'
export default createStore({
    namespaced: true,

    modules: {
        todoItems,
    },
})
```

Приклад використання стора

```
<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 />
```






```
        <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>
```