

Open Computers

Мод, добавляющий программируемые компьютеры и роботов. Позволяет автоматизировать процессы с помощью Lua-скриптов, управлять машинами, сетью и производством.

- [Магазин в компьютере с БД и управлением с телефона](#)

Магазин в компьютере с БД и управлением с телефона

Надоело бегать к терминалам, чтобы поменять цены? Хотите управлять экономикой сервера прямо с телефона, пока пьете чай? Этот гайд для вас.

Мы построим полностью автоматизированный магазин для Applied Energistics 2, которым управляет компьютер из OpenComputers. **Главная фишка:** вы можете подключить его к облачной базе данных и управлять всеми товарами, балансами игроков и смотреть логи через красивую веб-панель на вашем смартфоне или ПК!

Если вам не нужен сайт и вы хотите просто крутой магазин внутри игры — скрипт умеет работать и в **полностью Offline-режиме**. Выбирайте свой путь!

ШАГ 1: Собираем "Железо" в игре

Для работы магазина нам понадобится правильная физическая сборка.

1. Компоненты

Компонент	Количество	Примечание
Серверная стойка	1 шт	Для сервера
Монитор	6 шт	Уровень 3 (для сборки 3x2)
Видеокарта	1 шт	Уровень 3
Процессор (ЦП)	1 шт	Уровень 3
Память (ОЗУ)	4 шт	Уровень 3.5
Жесткий диск	1 шт	Уровень 3 (4 Мб)

Компонент	Количество	Примечание
EEPROM	1 шт	Прошивка Lua BIOS
Дискета	1 шт	С установщиком OpenOS
Адаптер	1 шт	Для связи с устройствами
Преобразователь энергии	1 шт	Питание компьютера
Клавиатура	1 шт	Для ввода команд
Транспозер	1 шт	Для связи с устройствами
Сервер	1 шт	Уровень 3
Дисковод	1 шт	Для серверной стойки
Интернет карта	1 шт	Для связи с интернетом
Улучшение База данных	1 шт	Уровень 3

2. Левая сторона (Приемка / Скупка / Сканер)

Сюда игроки будут сдавать лут, а админ — класть новые товары для сканирования.

- Ставим **МЭ Интерфейс**.
- Вплотную к нему — **Транспозер** (из OpenComputers).
- На Транспозер ставим буферный сундук.

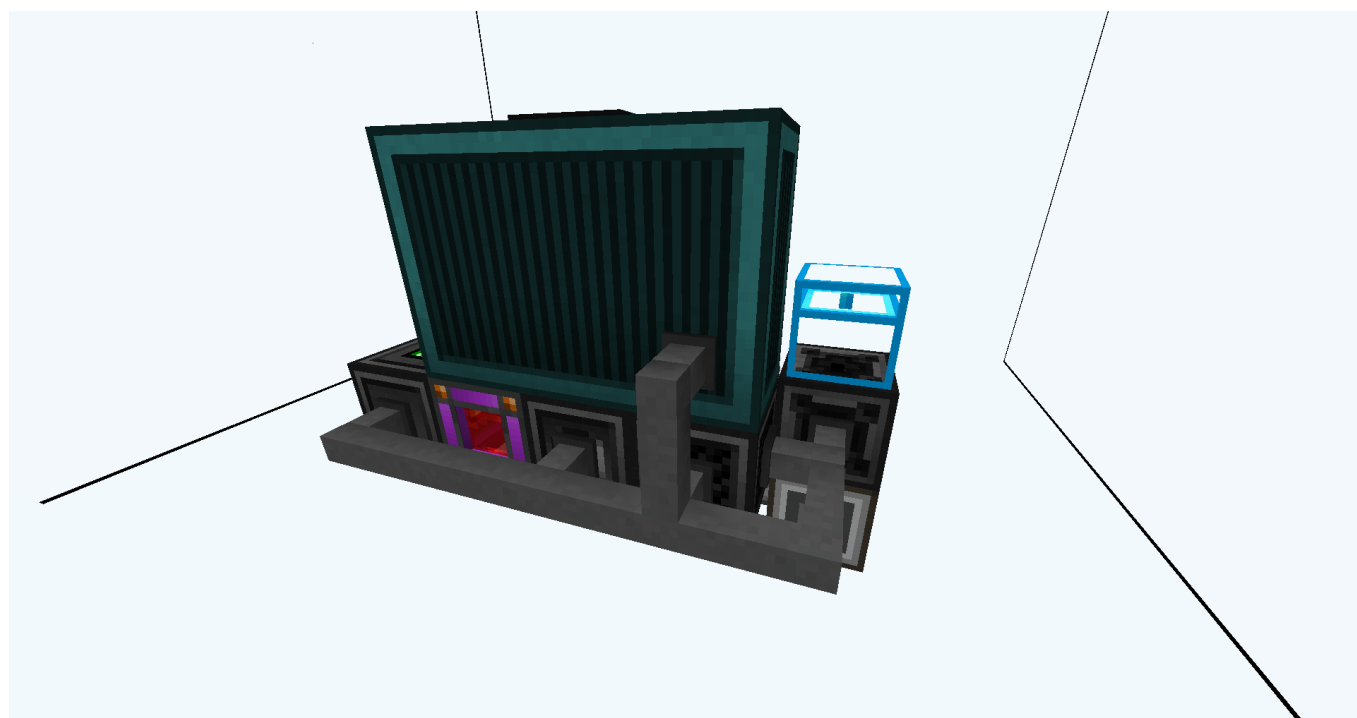
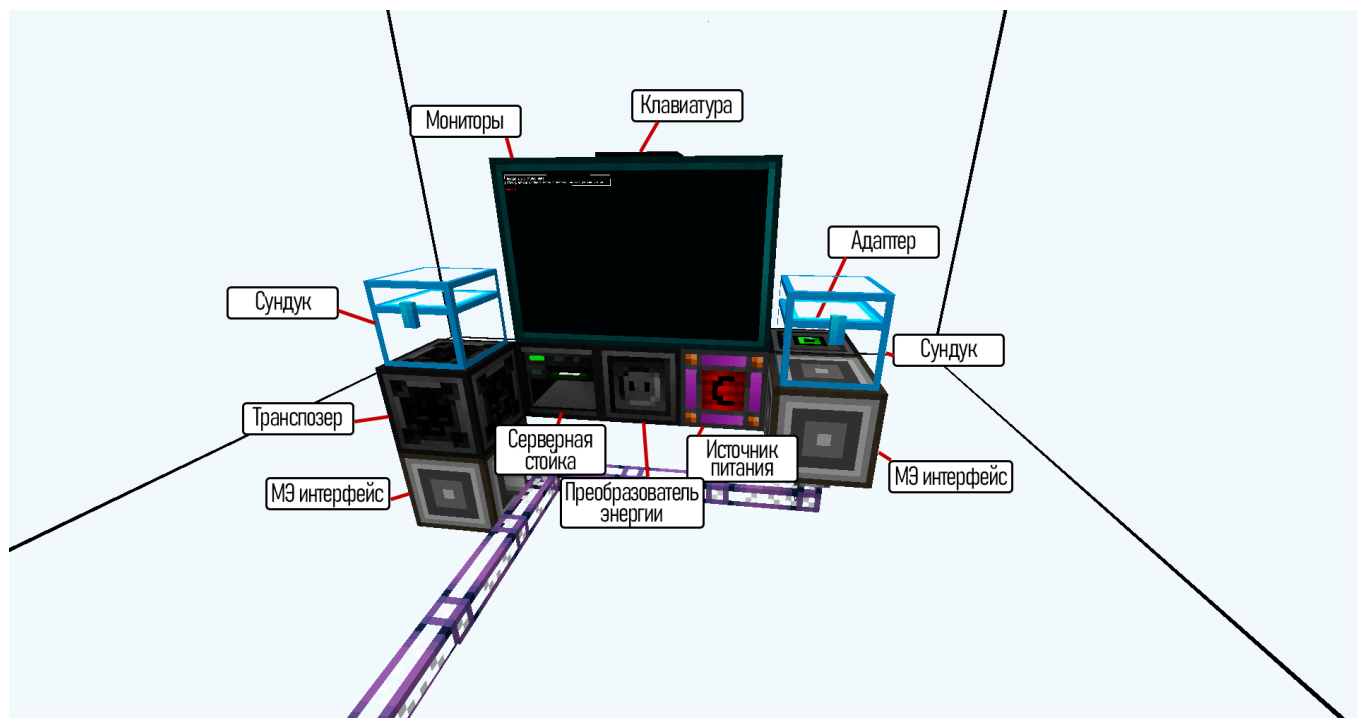
3. Правая сторона (Выдача покупок)

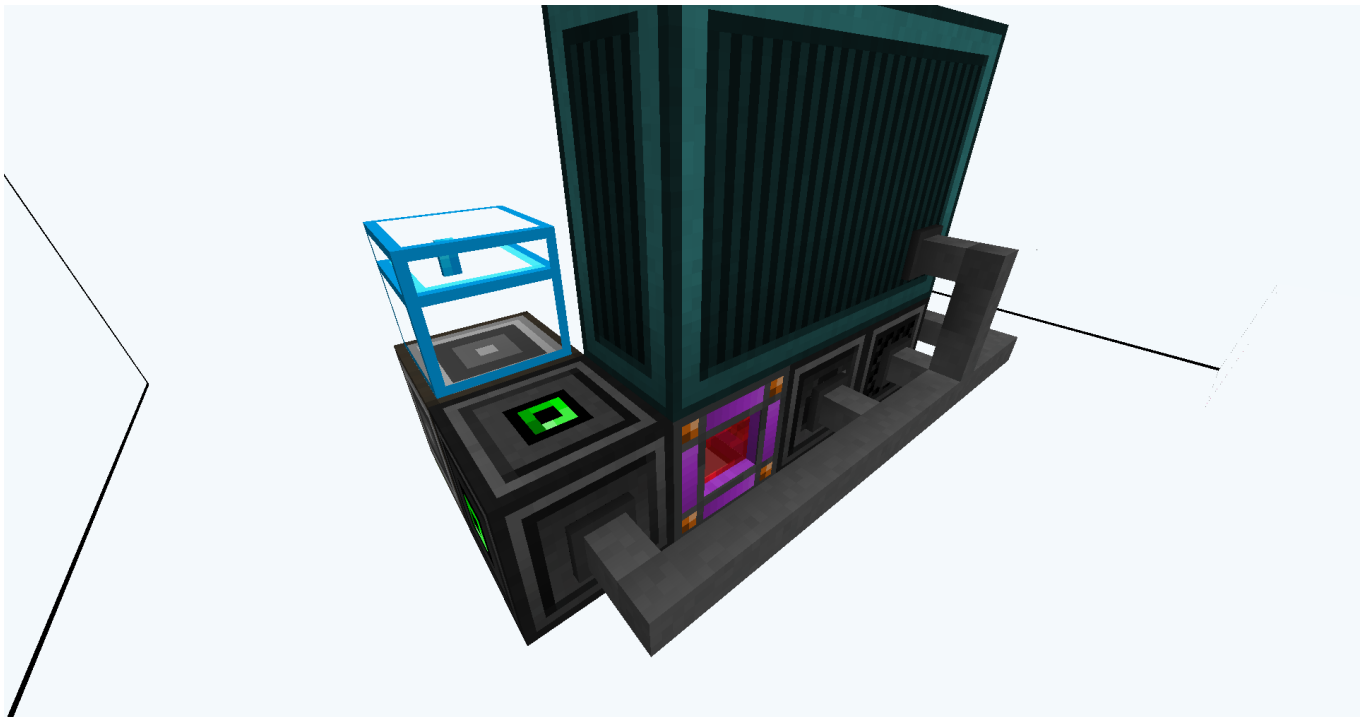
Сюда будут выпадать купленные товары.

- Ставим **МЭ Интерфейс**.
- Прямо на него (сверху) ставим сундук.
- Вплотную к МЭ Интерфейсу ставим **Адаптер** (из OpenComputers). В адаптер нужно положить Улучшение База данных Зий уровень.

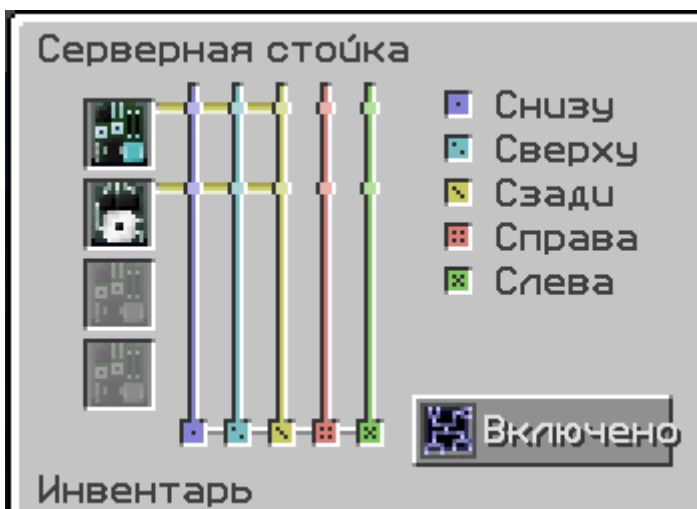
Подключите Транспозер, Адаптер и сам Компьютер кабелями из мода OpenComputers.
Готово!

Схема





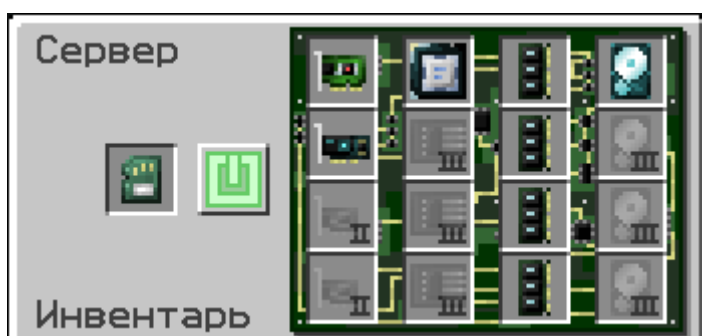
- **Настройка стойки:** Нажмите **ПКМ** по стойке. Вставьте сервер и дисковод. Кликните по желтым точкам на корпусе, чтобы задать направление сторон для подключения кабелей.



- **Установка ОС:** Вставьте дискету OpenOS в дисковод сервера через **Shift + ПКМ**.



- **Установка компонентов:** Откройте интерфейс сервера и вставьте видеокарту, процессор, память, диск, интернет карту, EEPROM (Lua BIOS).



4. Программная часть

Включите сервер и выполните следующие шаги:

```
install
```

```
/home # install
Install OpenOS? [Y/n]
```

Нажимайте **Enter**, подтверждая установку. В конце система предложит перезагрузку - снова нажмите **Enter**.

```
Installation complete!
Reboot now? [Y/n]
```

ШАГ 2: Выбор пути установки

Путь А: Полная установка (С облачной базой данных Firebase и Веб-сайтом).

☐ ПУТЬ А: ПОЛНАЯ УСТАНОВКА (С ВЕБ-ПАНЕЛЬЮ)

Если вы выбрали этот путь, мы создадим базу данных, чтобы связать игру с вашим телефоном.

Часть 1: Настройка Firebase (База Данных)

- Зайдите на [Firebase](#) под своим Google-аккаунтом и нажмите **Get started in console**.

Get started in console

- Нажмите **Create a new Firebase project**, назовите его (например, `me-shop`) и создайте (**Google Analytics** и **Gemini** можно выключить).



Create a new Firebase project

Integrate Firebase products to super-charge your app

Let's start with a name for your project[?]

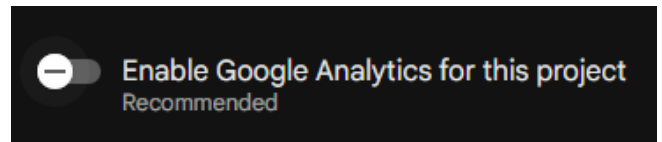
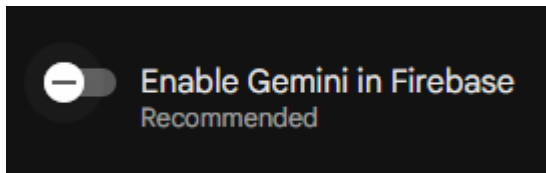
Project name

me-shop

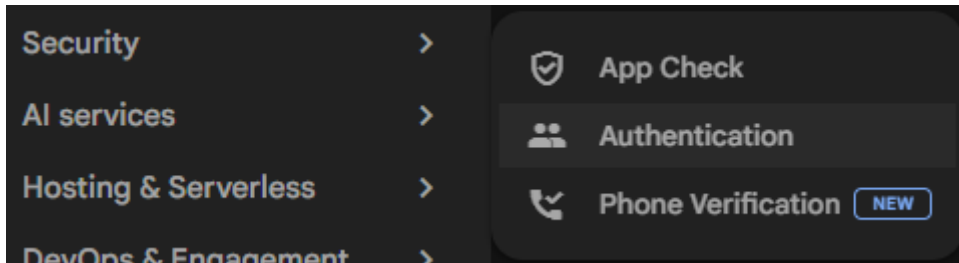
 me-shop-bf0a7

Already have a Google Cloud project?
[Add Firebase to Google Cloud project](#)

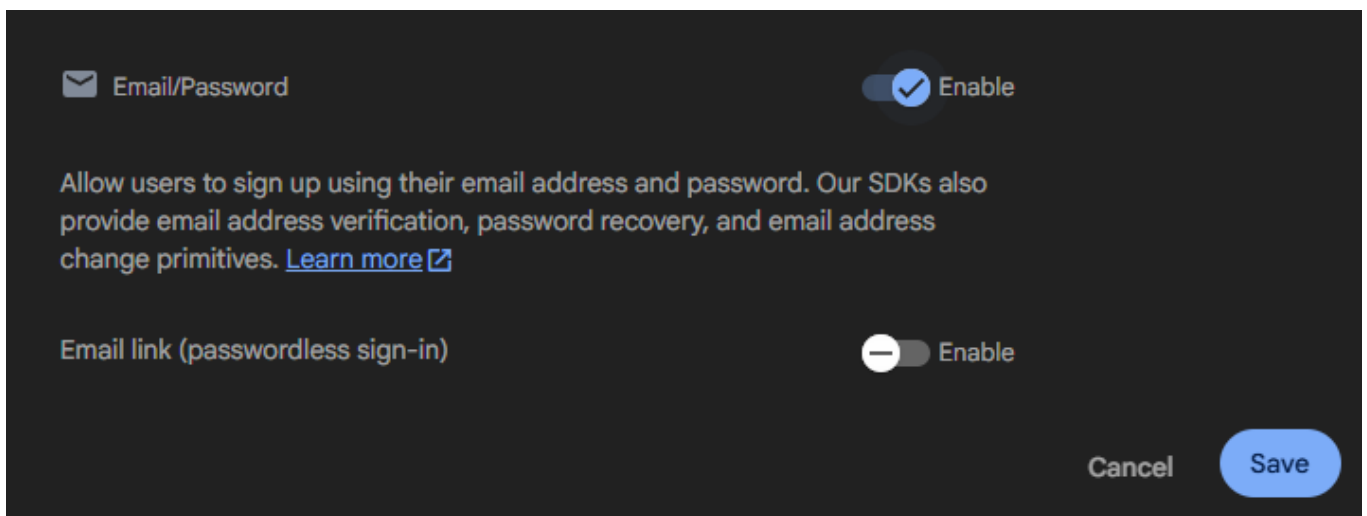
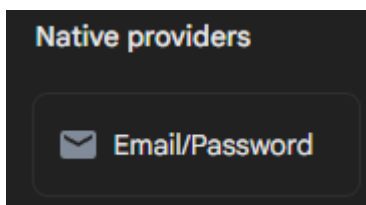
Continue



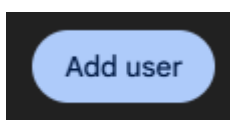
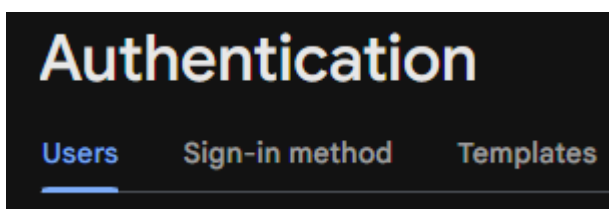
- В левом меню перейдите в **Security -> Authentication**, нажмите **Get Started**.



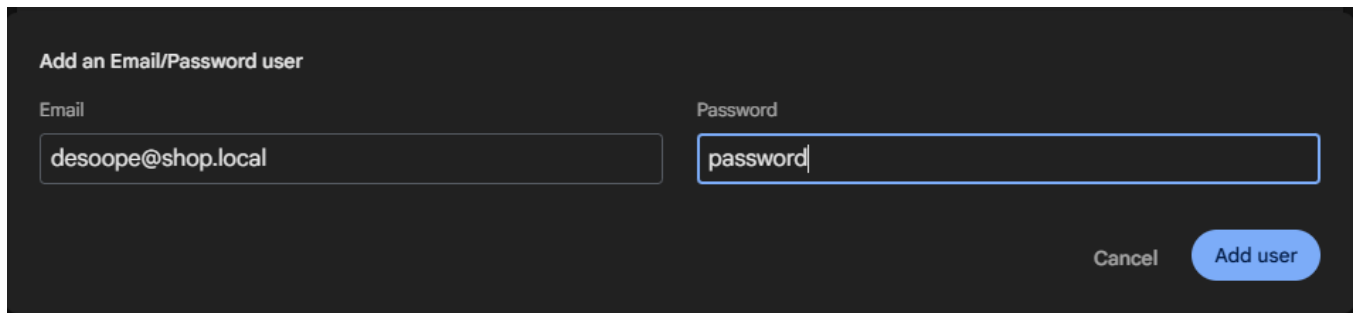
- Выберите **Email/Password**, включите первый ползунок и сохраните.



- Перейдите во вкладку **Users**, нажмите **Add user**.



- **ВАЖНО:** Введите свой ник с приставкой `@shop.local` (например: `Des0ope@shop.local`) и пароль. Этот аккаунт нужен для входа в панель!



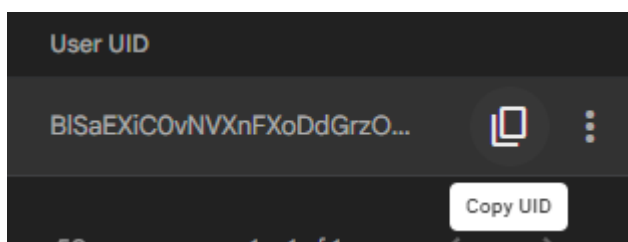
Add an Email/Password user

Email:

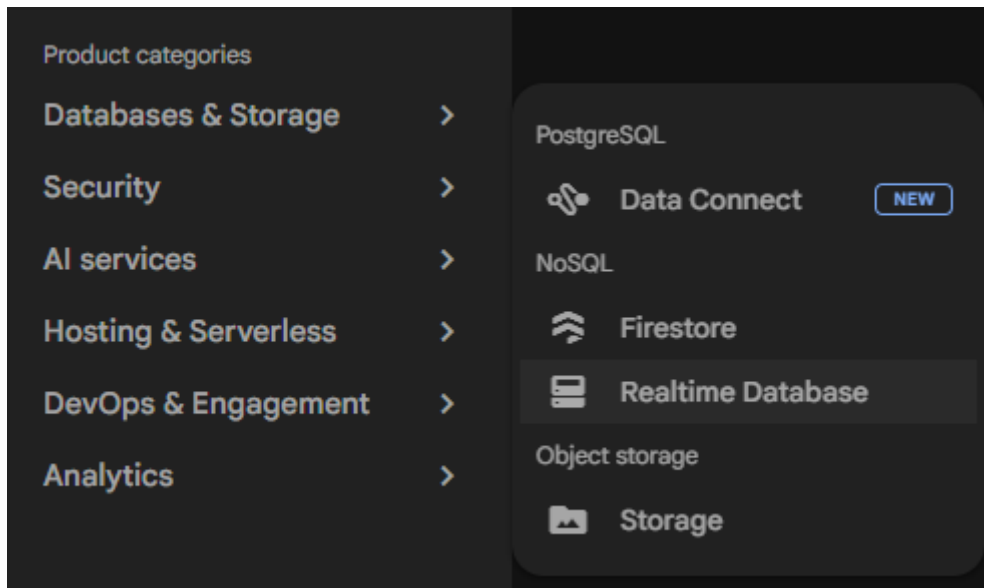
Password:

Cancel Add user

- Скопируйте длинный **User UID** вашего нового пользователя и сохраните в блокнот и подпишите его как **User UID**.



- В левом меню перейдите в **Databases & Storage -> Realtime Database**, нажмите **Create Database** (выберите локацию `Europe`, режим **Start in test mode**).



Set up database



1 Database options — 2 Security rules

Your location setting is where your Realtime Database data will be stored.

Realtime Database location

Belgium (europe-west1)

Cancel

Next

Set up database



Database options — 2 Security rules

Once you have defined your data structure you will have to write rules to secure your data.

[Learn more](#)

☐ Start in locked 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": {
    ".read": "now < 1776459600000", // 2026-4-18
    ".write": "now < 1776459600000", // 2026-4-18
  }
}
```



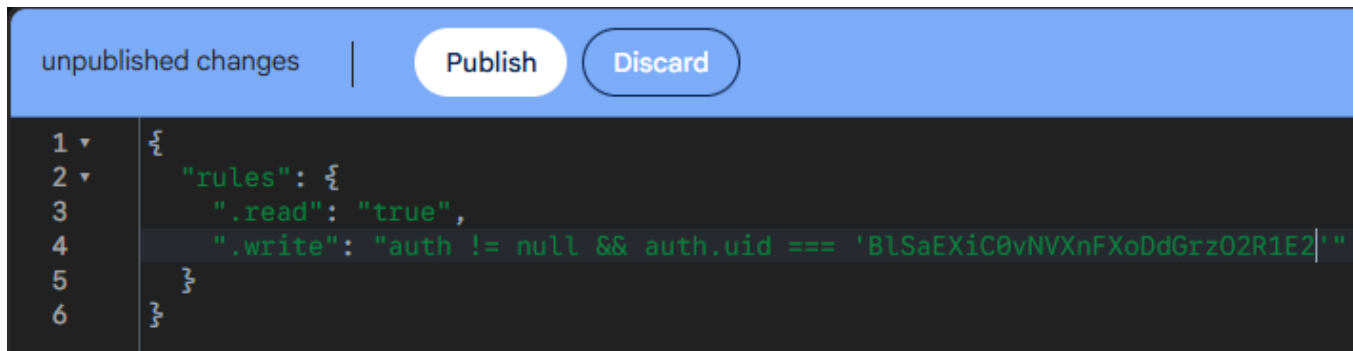
The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days

Cancel

Enable

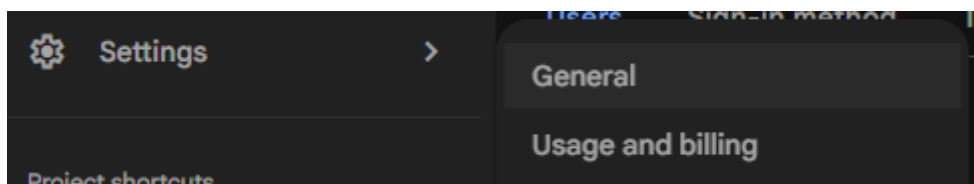
- Скопируйте ссылку на вашу базу (сверху над данными, вида `https://...firebase.database.app`). **Удалите слеш / на конце!** Сохраните её в блокнот и подпишите её как "ссылка на БД".
- Перейдите во вкладку **Rules** и замените весь код на этот (вставьте свой UID):

```
{
  "rules": {
    ".read": "true",
    ".write": "auth != null && auth.uid === 'ТВОЙ_USER_UID_СЮДА'"
  }
}
```

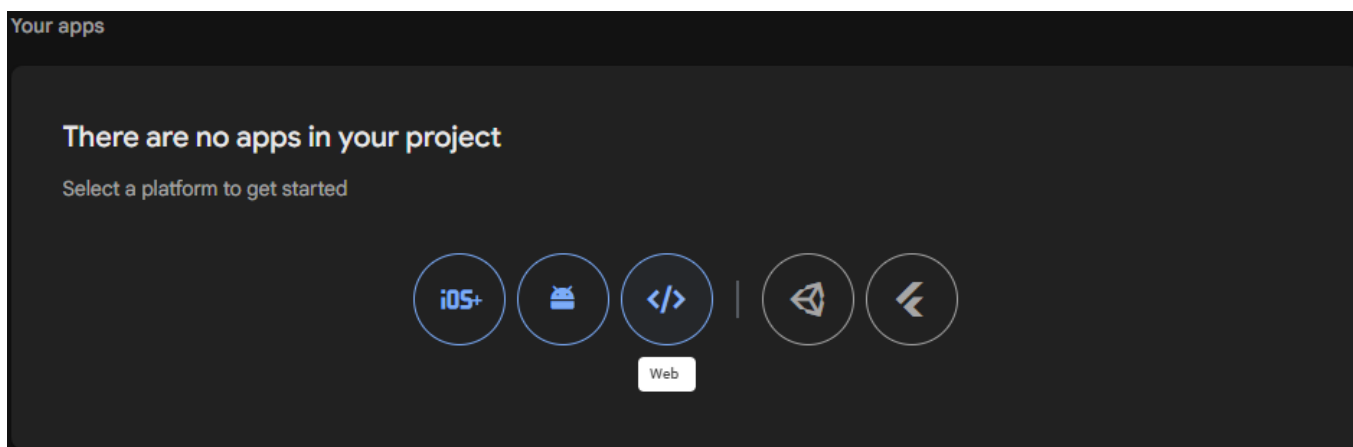


Обязательно жмем **Publish**.

- Идем за ключами: Наведите на **Setting** (слева сверху) -> **General**.



- Прокрутите вниз до "Your apps", нажмите на иконку  (Web).



- Назовите приложение (Например shop_app) и нажмите Register app.

1

Register app

App nickname ?

shop_app

☐

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

- Скопируйте все от **const firebaseConfig = {** до закрывающей скобки **};** и сохраните в блокнот.
- Нажмите на кнопку в самом низу **Continue to console**.

2 Add Firebase SDK

☒ Use npm ☐ Use a <script> tag

If you're already using [npm](#) and a module bundler such as [webpack](#) or [Rollup](#), you can run the following command to install the latest SDK ([Learn more](#)):

```
$ npm install firebase
```

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: "AIzaSyBsc5Mz9eooW1wgp2JXrNvIfaHFewXNAzA",
  authDomain: "me-shop-bf0a7.firebaseio.com",
  databaseURL: "https://me-shop-bf0a7-default-rtdb.europe-west1.firebaseio.com",
  projectId: "me-shop-bf0a7",
  storageBucket: "me-shop-bf0a7.appspot.com",
  messagingSenderId: "816771532922",
  appId: "1:816771532922:web:401b081c4623f0f3b153c3"
};

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

- Перейдите во вкладку **Service accounts** (сверху) -> слева **Database secrets**. Нажмите Show и скопируйте ваш **Секретный ключ**. Сохраняем в блокнот и подписываем "**Секретный ключ**".

Project settings

General Cloud Messaging Integrations **Service accounts** Data privacy Users and permissions Alerts

[Manage service account permissions](#)

 **Firebase Admin SDK**

Legacy credentials

 **Database secrets**

All service accounts

 [2 service accounts](#)

Database secrets

 Database secrets are currently deprecated and use a legacy Firebase token generator. Update your source code with the Firebase Admin SDK.

[Learn more](#)

Create custom database authentication tokens using a legacy Firebase token generator. At least one secret must exist at all times. [Learn more](#)

[Add secret](#)

Database

Secrets

me-shop-bf0a7-default... X2UXG57nYomwVMzV4v4qKvkEsGQDGDlsqNh2alxu

Copy Secret

Часть 2: Запуск Веб-панели (GitHub Pages)

- Создайте репозиторий на [GitHub](#).
- В корне репозитория создайте файл `index.html` и вставьте в него код панели.

Код для index.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>МЭ Магазин | Админ-Панель</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>
  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>
  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-
database.js"></script>
</head>
```

```
<body class="bg-gray-900 text-gray-200 font-sans antialiased min-h-screen flex flex-col">

  <div id="login-screen" class="flex flex-1 items-center justify-center p-4">
    <div class="bg-gray-800 p-8 rounded-lg shadow-lg w-full max-w-md border border-gray-700">
      <h2 class="text-3xl font-bold text-center text-blue-400 mb-6">Вход в систему</h2>
      <form id="login-form" class="space-y-4">
        <div>
          <label class="block text-sm font-medium text-gray-400">Никнейм</label>
          <input type="text" id="email" required class="mt-1 block w-full bg-gray-700 border border-gray-600 rounded-md shadow-sm py-3 px-4 text-white focus:outline-none focus:ring-blue-500 focus:border-blue-500">
        </div>
        <div>
          <label class="block text-sm font-medium text-gray-400">Пароль</label>
          <input type="password" id="password" required class="mt-1 block w-full bg-gray-700 border border-gray-600 rounded-md shadow-sm py-3 px-4 text-white focus:outline-none focus:ring-blue-500 focus:border-blue-500">
        </div>
        <div id="login-error" class="text-red-500 text-sm hidden text-center">Неверный логин или пароль.</div>
        <button type="submit" class="w-full flex justify-center py-3 px-4 border border-transparent rounded-md shadow-sm text-base font-medium text-white bg-blue-600 hover:bg-blue-700 focus:outline-none transition">
          Войти
        </button>
      </form>
    </div>
  </div>

  <div id="dashboard" class="hidden flex-1 flex flex-col">
    <nav class="bg-gray-800 border-b border-gray-700 sticky top-0 z-10">
      <div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">
        <div class="flex items-center justify-between h-16">
```

```

        <div class="flex items-center">
            <span class="text-xl font-bold text-blue-400 mr-4">МЭ
Маразин</span>

            <div class="hidden sm:flex space-x-2">
                <button onclick="switchTab('users')" class="tab-btn px-4 py-2 rounded-md text-sm font-medium bg-gray-900 text-white" id="tab-users">Игроки</button>

                <button onclick="switchTab('shop')" class="tab-btn px-4 py-2 rounded-md text-sm font-medium text-gray-300 hover:bg-gray-700 hover:text-white transition" id="tab-shop">Товары</button>

                <button onclick="switchTab('logs')" class="tab-btn px-4 py-2 rounded-md text-sm font-medium text-gray-300 hover:bg-gray-700 hover:text-white transition" id="tab-logs">Логи</button>
            </div>
        </div>

        <div class="flex items-center space-x-4">
            <span id="user-email" class="text-sm text-gray-400 hidden md:block"></span>

            <button onclick="logout()" class="px-4 py-2 rounded-md text-sm font-medium text-white bg-red-600 hover:bg-red-700 transition">Выход</button>
        </div>
    </div>

    <div class="sm:hidden pb-3 pt-1 flex space-x-2 overflow-x-auto no-scrollbar">

        <button onclick="switchTab('users')" class="tab-btn flex-none px-4 py-2 rounded-md text-sm font-medium bg-gray-900 text-white" id="tab-users-mob">Игроки</button>

        <button onclick="switchTab('shop')" class="tab-btn flex-none px-4 py-2 rounded-md text-sm font-medium text-gray-300 hover:bg-gray-700 hover:text-white transition" id="tab-shop-mob">Товары</button>

        <button onclick="switchTab('logs')" class="tab-btn flex-none px-4 py-2 rounded-md text-sm font-medium text-gray-300 hover:bg-gray-700 hover:text-white transition" id="tab-logs-mob">Логи</button>
    </div>
</div>
</nav>

<main class="max-w-7xl w-full mx-auto px-4 sm:px-6 lg:px-8 py-6">

```



```
<div id="content-users" class="tab-content block">
  <h2 class="text-2xl font-bold mb-4">Управление балансом</h2>
  <div class="bg-gray-800 shadow rounded-lg overflow-hidden border
border-gray-700">
    <div class="hidden sm:flex bg-gray-700 p-4 text-xs font-medium
text-gray-300 uppercase tracking-wider">
      <div class="flex-1">Никнейм</div>
      <div class="w-48 text-center">Баланс</div>
    </div>
    <div id="users-list" class="divide-y divide-gray-700"></div>
  </div>
</div>

<div id="content-shop" class="tab-content hidden">
  <h2 class="text-2xl font-bold mb-4">Товары на витрине</h2>
  <div class="bg-gray-800 shadow rounded-lg overflow-hidden border
border-gray-700">
    <div class="hidden md:flex bg-gray-700 p-4 text-xs font-medium
text-gray-300 uppercase tracking-wider gap-4">
      <div class="flex-1">Название</div>
      <div class="w-32">Категория</div>
      <div class="w-24">Цена</div>
      <div class="w-[180px] text-center">Действия</div>
    </div>
    <div id="shop-list" class="divide-y divide-gray-700"></div>
  </div>
</div>

<div id="content-logs" class="tab-content hidden">
  <h2 class="text-2xl font-bold mb-4">История операций</h2>
  <div class="bg-gray-800 shadow rounded-lg overflow-hidden border
border-gray-700">
    <div id="logs-list" class="divide-y divide-gray-700"></div>
  </div>
</div>
</main>
</div>
```

```
<script src="./firebase-config.js"></script>

<script>
  firebase.initializeApp(firebaseConfig);
  const auth = firebase.auth();
  const db = firebase.database();

  auth.onAuthStateChanged((user) => {
    if (user) {
      document.getElementById('login-screen').classList.add('hidden');
      document.getElementById('dashboard').classList.remove('hidden');
      document.getElementById('user-email').innerText =
user.email.replace('@shop.local', '');
      loadData();
    } else {
      document.getElementById('login-screen').classList.remove('hidden');
      document.getElementById('dashboard').classList.add('hidden');
    }
  });

  document.getElementById('login-form').addEventListener('submit', (e) => {
    e.preventDefault();
    let email = document.getElementById('email').value.trim();
    const password = document.getElementById('password').value;

    if (!email.includes('@')) {
      email = email + '@shop.local';
    }

    auth.signInWithEmailAndPassword(email, password).catch(() => {
      document.getElementById('login-error').classList.remove('hidden');
    });
  });

  function logout() { auth.signOut(); }

  function switchTab(tabId) {
```

```

        document.querySelectorAll('.tab-content').forEach(el =>
el.classList.add('hidden'));
        document.querySelectorAll('.tab-btn').forEach(el => {
            el.classList.remove('bg-gray-900', 'text-white');
            el.classList.add('text-gray-300');
        });

        document.getElementById('content-' + tabId).classList.remove('hidden');
        document.getElementById('tab-' + tabId).classList.add('bg-gray-900', 'text-
white');
        if(document.getElementById('tab-' + tabId + '-mob')) {
            document.getElementById('tab-' + tabId + '-mob').classList.add('bg-
gray-900', 'text-white');
        }
    }

function loadData() {
    // ПОЛЬЗОВАТЕЛИ
    db.ref('users').on('value', (snapshot) => {
        const users = snapshot.val() || {};
        const list = document.getElementById('users-list');
        list.innerHTML = '';
        for (const [name, rawData] of Object.entries(users)) {
            let data = rawData;
            if (typeof rawData === 'string') { try { data =
JSON.parse(rawData); } catch(e) { data = { balance: 0 }; } }
            const balance = data.balance !== undefined ? data.balance : 0;
            list.innerHTML += `
                <div class="p-4 flex flex-col sm:flex-row sm:items-center
justify-between gap-4 hover:bg-gray-750 transition">
                    <div class="font-bold text-lg sm:text-base text-
white">${name}</div>
                    <div class="flex items-center gap-3">
                        <div class="flex-1 sm:flex-none">
                            <label class="text-xs text-gray-400 block sm:hidden
mb-1">Баланс</label>
                            <input type="number" id="bal-${name}"
value="${balance}" class="w-full sm:w-28 bg-gray-700 text-white px-3 py-2 rounded

```

```

border border-gray-600 focus:outline-none focus:border-blue-500">
    </div>
    <div class="pt-5 sm:pt-0">
        <button onclick="saveBalance('${name}')" class="bg-
green-600 hover:bg-green-700 text-white font-medium py-2 px-4 rounded transition w-full
sm:w-auto">Сохранить</button>
    </div>
</div>
</div>
</div>
    `;
}
});

// ТОВАРЫ
db.ref('shop').on('value', (snapshot) => {
    let shopData = snapshot.val() || {};
    if (typeof shopData === 'string') { try{ shopData =
JSON.parse(shopData); } catch(e) { shopData = {}; } }
    const itemsRaw = shopData.items || [];
    const list = document.getElementById('shop-list');
    list.innerHTML = '';
    Object.keys(itemsRaw).forEach(key => {
        const item = itemsRaw[key];
        if(!item) return;
        list.innerHTML += `
            <div class="p-4 flex flex-col md:flex-row md:items-center gap-4
hover:bg-gray-750 transition">
                <div class="flex-1">
                    <label class="text-xs text-gray-400 block md:hidden mb-
1">Название товара</label>
                    <input type="text" id="name-${key}"
value="${item.name}" class="w-full bg-gray-700 text-white px-3 py-2 rounded border
border-gray-600 focus:outline-none focus:border-blue-500">
                </div>
                <div class="w-full md:w-32">
                    <label class="text-xs text-gray-400 block md:hidden mb-
1">Категория</label>
                    <input type="text" id="cat-${key}"

```

```

value="${item.category}" class="w-full bg-gray-700 text-white px-3 py-2 rounded border
border-gray-600 focus:outline-none focus:border-blue-500">
    </div>
    <div class="w-full md:w-24">
        <label class="text-xs text-gray-400 block md:hidden mb-
1">Цена</label>
        <input type="number" id="price-${key}"
value="${item.price}" class="w-full bg-gray-700 text-white px-3 py-2 rounded border
border-gray-600 focus:outline-none focus:border-blue-500">
    </div>
    <div class="flex gap-2 w-full md:w-[180px] pt-2 md:pt-0
border-t border-gray-700 md:border-t-0 mt-2 md:mt-0">
        <button onclick="saveItem('${key}')" class="flex-1 bg-
blue-600 hover:bg-blue-700 text-white font-medium py-2 px-3 rounded
transition">Сохранить</button>
        <button onclick="deleteItem('${key}')" class="flex-1
bg-red-600 hover:bg-red-700 text-white font-medium py-2px-3 rounded
transition">Удалить</button>
    </div>
</div>
`
});
});
});

// ЛОГИ
db.ref('logs').orderByKey().limitToLast(100).on('value', (snapshot) => {
    const logsRaw = snapshot.val() || {};
    const list = document.getElementById('logs-list');
    list.innerHTML = '';
    const logsArray = [];
    for (const key in logsRaw) {
        let log = logsRaw[key];
        if (typeof log === 'string') { try { log = JSON.parse(log); }
catch(e) { continue; } }
        logsArray.push(log);
    }
    logsArray.reverse();
    logsArray.forEach(log => {

```

```

        let actionColor = "text-gray-300";
        if(log.action.includes("ПОКУПКА")) actionColor = "text-green-400";
        if(log.action.includes("ПРОДАЖА") || log.action.includes("СКУПКА"))
actionColor = "text-yellow-400";
        if(log.action.includes("УДАЛЕН") || log.action.includes("ОШИБКА"))
actionColor = "text-red-400";

        list.innerHTML += `
            <div class="p-4 hover:bg-gray-750 transition flex flex-col gap-
1">

                <div class="flex flex-wrap justify-between items-center
gap-x-4 gap-y-1 border-b border-gray-700 pb-2 sm:border-0 sm:pb-0">
                    <div class="flex items-center gap-2">
                        <span class="font-bold text-base text-
white">${log.user}</span>

                        <span class="text-sm font-bold bg-gray-900 px-2 py-
0.5 rounded ${actionColor}">${log.action}</span>
                    </div>
                    <span class="text-xs font-mono text-gray-
400">${log.time}</span>
                </div>
                <div class="text-sm text-gray-300 break-words mt-
1">${log.details}</div>
            </div>
        `;
    });
});
}

window.saveBalance = function(name) {
    const newBal = document.getElementById(`bal-${name}`).value;
    db.ref(`users/${name}/balance`).set(Number(newBal))
        .then(() => showToast('Баланс успешно обновлен', 'success'))
        .catch(e => showToast('Ошибка сохранения', 'error'));
}

window.saveItem = function(key) {
    const name = document.getElementById(`name-${key}`).value;

```

```

const cat = document.getElementById(`cat-${key}`).value;
const price = document.getElementById(`price-${key}`).value;

db.ref('shop').once('value').then((snapshot) => {
  let shopData = snapshot.val() || {};
  if (typeof shopData === 'string') { try { shopData =
JSON.parse(shopData); } catch(e) { shopData = { items: [] }; } }
  if (shopData.items && shopData.items[key]) {
    shopData.items[key].name = name;
    shopData.items[key].category = cat;
    shopData.items[key].price = Number(price);
    db.ref('shop').set(shopData)
      .then(() => showToast('Товар успешно обновлен', 'success'))
      .catch(e => showToast('Ошибка сохранения', 'error'));
  }
});
}

window.deleteItem = function(key) {
  if (!confirm('Удалить товар навсегда?')) return;
  db.ref('shop').once('value').then((snapshot) => {
    let shopData = snapshot.val() || {};
    if (typeof shopData === 'string') { try { shopData =
JSON.parse(shopData); } catch(e) { shopData = { items: [] }; } }
    if (shopData.items) {
      let itemsArray = Array.isArray(shopData.items) ? shopData.items :
Object.values(shopData.items);
      itemsArray.splice(Number(key), 1);
      shopData.items = itemsArray.filter(Boolean);
      db.ref('shop').set(shopData)
        .then(() => showToast('Товар удален', 'success'))
        .catch(e => showToast('Ошибка удаления', 'error'));
    }
  });
}

function showToast(message, type) {
  const toast = document.createElement('div');

```

```

        toast.className = `fixed bottom-4 right-4 sm:bottom-8 sm:right-8 px-6 py-4
rounded-lg shadow-2xl text-white font-medium transition-all duration-300 transform
translate-y-10 opacity-0 ${type === 'success' ? 'bg-green-600' : 'bg-red-600'} z-50`;
        toast.innerText = message;
        document.body.appendChild(toast);
        requestAnimationFrame(() => { toast.classList.remove('translate-y-10',
'opacity-0'); });
        setTimeout(() => { toast.classList.add('translate-y-10', 'opacity-0');
setTimeout(() => toast.remove(), 300); }, 3000);
    }
</script>
</body>
</html>

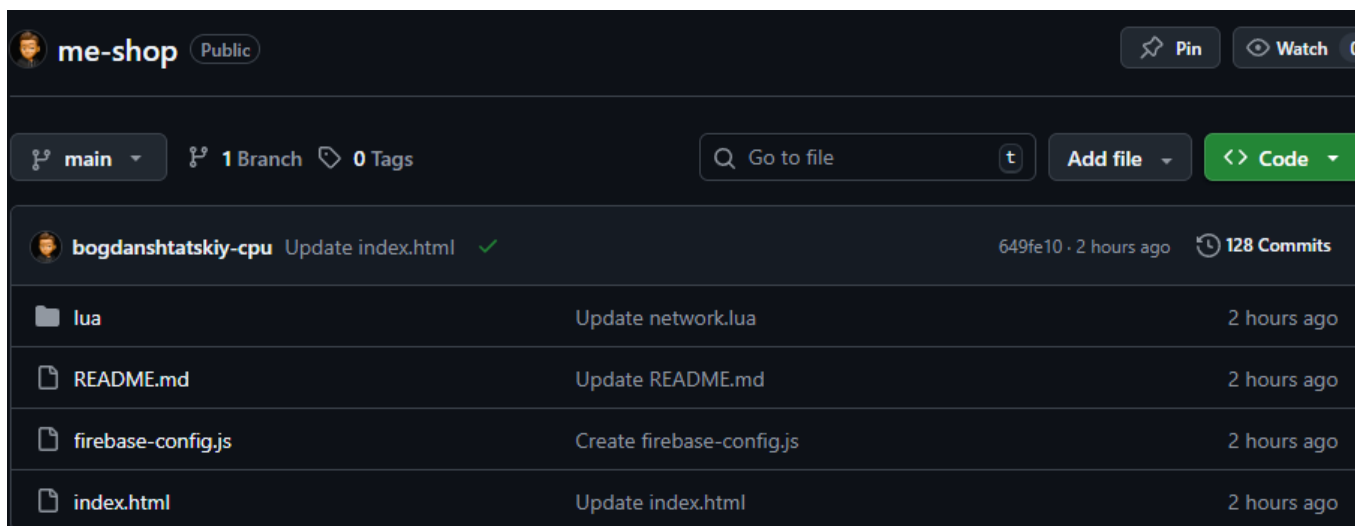
```

- Рядом создайте файл `firebase-config.js` и вставьте туда данные из блокнота в формате:

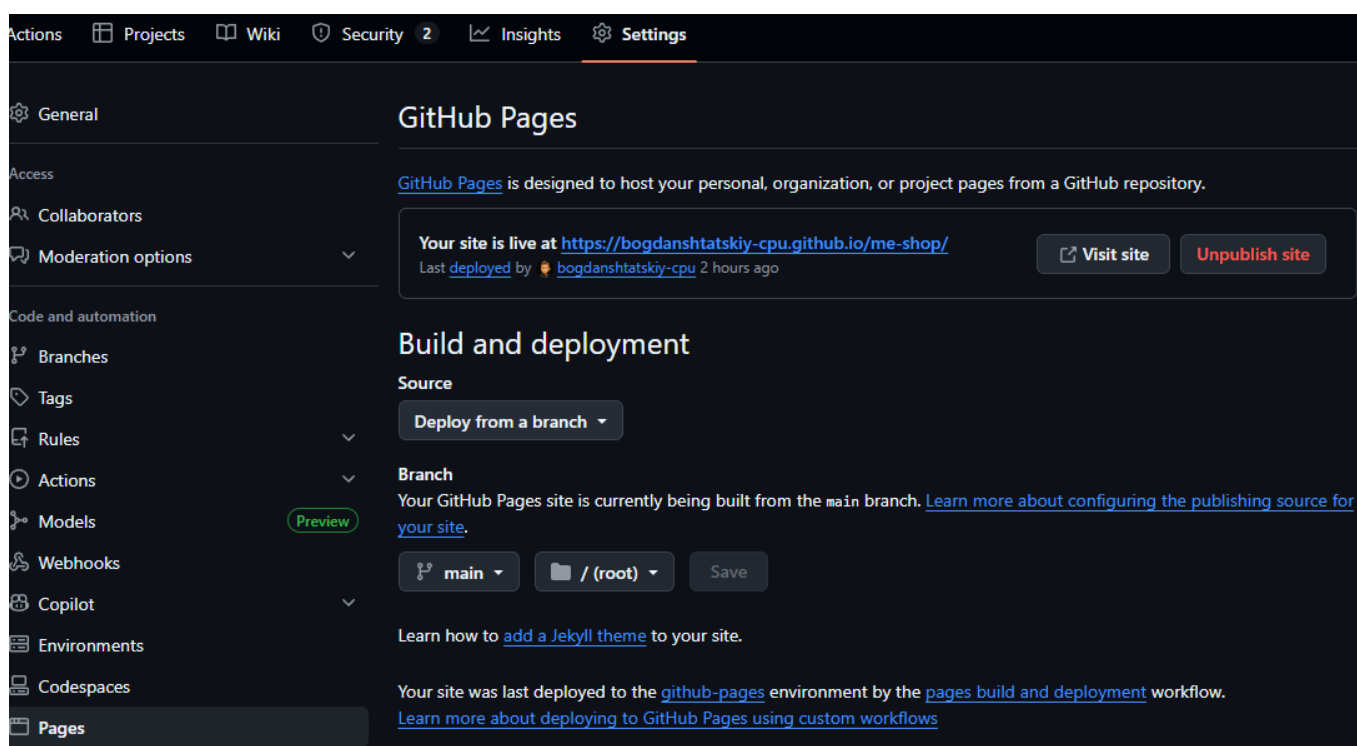
```

const firebaseConfig = {
  apiKey: "AIzaSyBsc5Mz9eooWlwg2JJXrNvIfaHFewXNAzA",
  authDomain: "me-shop-bf0a7.firebaseio.com",
  databaseURL: "https://me-shop-bf0a7-default-rtdb.europe-west1.firebaseio.com",
  projectId: "me-shop-bf0a7",
  storageBucket: "me-shop-bf0a7.firebaseio.com",
  messagingSenderId: "816771532922",
  appId: "1:816771532922:web:401b081c4623f0f3b153c3"
};

```

- Зайдите в **Settings** репозитория -> вкладка **Pages**. В разделе Branch выберите `main`, папку `/ (root)` и нажмите Save. Через 2 минуты ваш личный сайт-админка заработает! Ссылка появится сверху вместе с кнопкой Visit site.



Часть 3: Установка в Игре

- На компьютере в Minecraft скачайте установщик:

```
wget -f https://raw.githubusercontent.com/bogdanshtatskiy-cpu/me-shop/main/luainstaller.lua installer.lua
```

- Напишите `installer` и нажмите Enter. Скрипт сам скачает все нужные файлы.
- Откройте настройки командой `edit config.lua`.

- `config.use_database = true` (оставляем так).
- Впишите ваш `firebase_url` (ссылка на БД без `/` на конце) и `db_secret` (секретный ключ).
- Впишите свой ник в `config.admins`.
- Настройте часовой пояс (например, `config.timezone = 2` для Киева/UTC+2).
- Сохраните (Ctrl+S) и закройте (Ctrl+W).
- Напишите `reboot`. После перезагрузки введите `main`. **Готово!** Ваш магазин подключен к облаку. После добавления товаров откройте веб-панель на телефоне или ПК, залогиньтесь (просто введя ник и пароль) и управляйте витриной!

Путь Б: Локальная установка (Только внутри игры, без сайта и БД).

ПУТЬ Б: ЛОКАЛЬНАЯ УСТАНОВКА (БЕЗ САЙТА И БД)

Если вы не хотите заморачиваться с базами данных, регистрациями и сайтами, магазин будет отлично работать прямо на жестком диске компьютера в Майнкрафте.

- На компьютере в Minecraft скачайте установщик:

```
wget -f https://raw.githubusercontent.com/bogdanshtatskiy-cpu/me-shop/main/luainstaller.lua installer.lua
```

- Напишите `installer` и нажмите Enter.
- Откройте настройки командой:

```
edit config.lua
```

- Внесите **две главные правки**:
 - Измените `config.use_database = true` на `config.use_database = false`. (Это отключит все сетевые запросы и фоновые синхронизации, магазин будет работать молниеносно).
 - Впишите свой никнейм в блоке `config.admins`.
 - Поля `firebase_url` и `db_secret` просто оставьте пустыми — они вам не понадобятся.
 - Сохраните файл (Ctrl+S) и закройте (Ctrl+W).
 - Перезагрузите ПК (`reboot`) и напишите `main`. **Готово!** Автономный, независимый магазин работает прямо на вашем сервере.

Как пользоваться магазином?

Добавление товара (Для Админа):

- В игре нажмите «АВТОРИЗАЦИЯ» -> «АДМИН ПАНЕЛЬ».
- Нажмите «ДОБАВИТЬ ЗАПИСЬ».
- Положите нужный предмет (например, Алмаз) в левый сундук (сундук сканера) и нажмите ОК на экране. Скрипт сам прочитает скрытые ID и NBT-теги.
- Введите цену и выберите категорию. Нажмите Сохранить.

Покупки и Корзина (Для Игроков):

- Нажмите кнопку «**В КОРЗИНУ**» или «**КУПИТЬ**» под любыми товарами.
- Наберите нужное количество предметов (хоть по 1000 штук за раз).
- Перейдите в корзину и нажмите «ОПЛАТИТЬ». Система сама проверит, хватает ли у вас ЭМов и есть ли свободное место в сундуке выдачи.

Скупка: Хотите, чтобы игроки продавали вам ресурсы? Админ добавляет предметы в раздел "Скупка" с указанием цены. Игрок скидывает полную сумку булыжника в левый сундук, нажимает кнопку «**ПРОДАТЬ ВСЁ**», и система мгновенно конвертирует ресурсы в валюту на баланс игрока!