

Firebase Storage Rule

Storage Rule을
적용하는 이유는?

악성 의(malicious) 유저들로 부터
파일들을 보호하기 위해서 입니다.

공식 Documentation

[https://firebase.google.com/docs/rules/get-started?
authuser=0](https://firebase.google.com/docs/rules/get-started?authuser=0)

Storage Rule
적용하는 곳으로 이동!

```
service firebase.storage {  
  match /b/{bucket}/o {  
    match /{allPaths=**} {  
      allow read, write: if request.auth != null;  
    }  
  }  
}
```

현재 상태는
authentication이 인증되면
read와 write가능

```
service <<name>> {  
  // Match the resource path.  
  match <<path>> {  
    // Allow the request if the following conditions are true.
```

```

    allow <<methods>> : if <<condition>>
  }
}


```



```

service firebase.storage {
  // The {bucket} wildcard indicates we match files in all storage buckets
  match /b/{bucket}/o {
    // Match filename
    match /filename {
      allow read: if <condition>;
      allow write: if <condition>;
    }
  }
}

```

각 경로(path)에 맞는
rule을 설정해주기

 gs://test-app-beca5.appspot.com

| <input type="checkbox"/> | Name |
|--------------------------|---|
| <input type="checkbox"/> |  message/ |
| <input type="checkbox"/> |  user_image/ |

message/

```

rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {
    ① match /message {
      ② allow read: if request.auth != null;
      ③ allow write: if request.auth != null &&
      ④ request.resource.contentType.matches('image/*') &&
      ⑤ request.resource.size < 10 * 1024 * 1024;
    }
  }
}

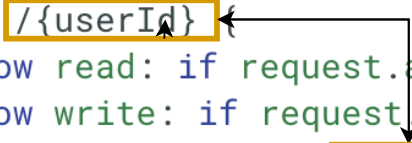
```

```
    request.resource.size < 10 * 1024 * 1024,  
  }  
}  
}
```

- ① /message 경로에 해당하는 파일들
- ② 인증이 된 유저는 read 가능
- ③ 인증이 된 유저는 write 가능
- ④ 이미지 파일만 업로드 가능
- ⑤ 파일 사이즈 10mb이하만 가능

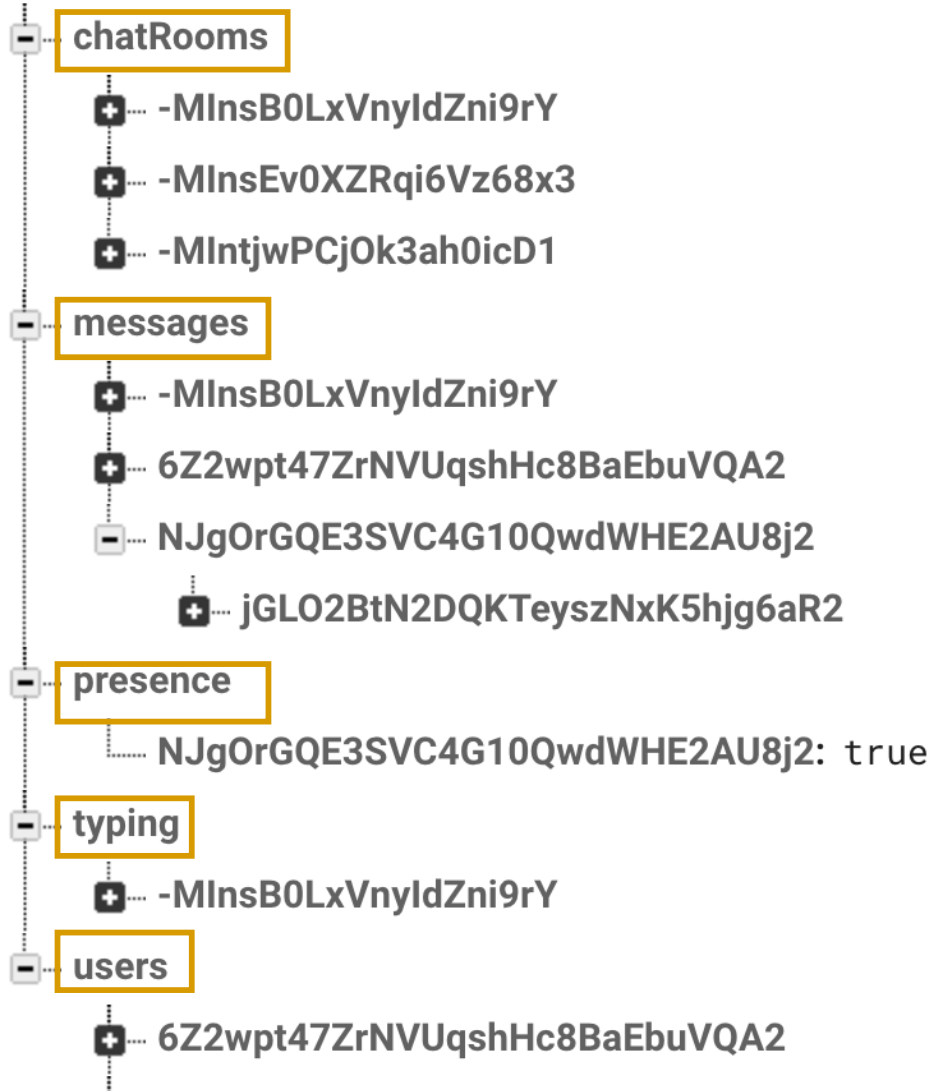
user_image/

```
match /user_image {  
  match /{userId} {  
    allow read: if request.auth != null;  
    allow write: if request.auth != null &&  
      request.auth.uid == userId &&  
      request.resource.contentType.matches('image/*') &&  
      request.resource.size < 10 * 1024 * 1024;  
  }  
}
```



Firestore Database Rule

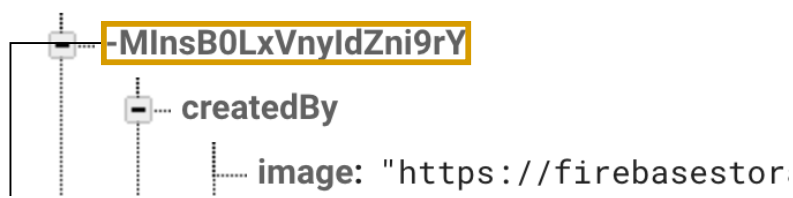
test-app-beca5



각각의 Collection을 위한 Rule 설정해주기

chatRooms

... chatRooms



name: "John Ahn"
description: "first chat room"
id: "-MInsB0LxVnyIdZni9rY"
name: "first room"

```
{
  "rules": {
    "chatRooms": {
      ".read": "auth != null",
      "$chatRoomId": {
        ".write": "auth != null",
        ".validate": "newData.hasChildren(['id', 'name', 'createdBy', 'description'])",
        "id": {
          ".validate": "newData.val() === $chatRoomId"
        },
        "name": {
          ".validate": "newData.val().length > 0"
        },
        "description": {
          ".validate": "newData.val().length > 0"
        }
      }
    }
  }
},
}
```

messages

messages

```
-MInsB0LxVnyIdZni9rY
  -MJNakZgBsJSGex0kr-E
    content: "111"
    timestamp: 1602435615445
    user
      id: "NJgOrGQE3SVC4G10QwdWHE2AU8j2"
      image: "https://firebasestorage.goo
      name: "John Ahn"
  -MJmD3wV0GS_jlmGHT0I
    image: "https://firebasestorage.google
    timestamp: 1602865353348
    user
      id: "y1AGwpaqQddpXIQDNVnzL3aqU1S2"
      image: "http://gravatar.com/avatar
      name: "iaewon1"
```

name: 'jaewon'

```
"messages": {
  ".read": "auth != null",
  ".write": "auth != null",
  "content": {
    ".validate": "newData.val().length > 0"
  },
  "image": {
    ".validate": "newData.val().length > 0"
  },
  "user": {
    ".validate": "newData.hasChildren(['id', 'name', 'image'])"
  }
},
```

presence & typing

· presence

```
└─ NJgOrGQE3SVC4G10QwdWHE2AU8j2: true
└─ y1AGwpaqQddpXIQDNVnzL3aqUIS2: true
```

... typing

```
└─ -MlnsB0LxVnyldZni9rY
  └─ y1AGwpaqQddpXIQDNVnzL3aqUIS2: "jaewon1"
```

```
"presence": {
  ".read": "auth != null",
  ".write": "auth != null"
},
"typing": {
  ".read": "auth != null",
  ".write": "auth != null"
},
```

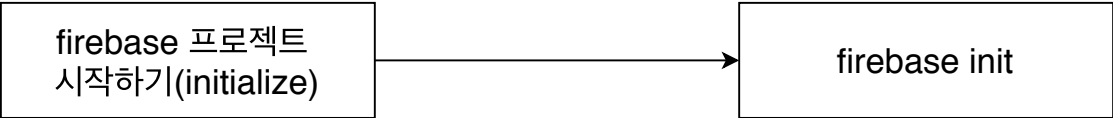
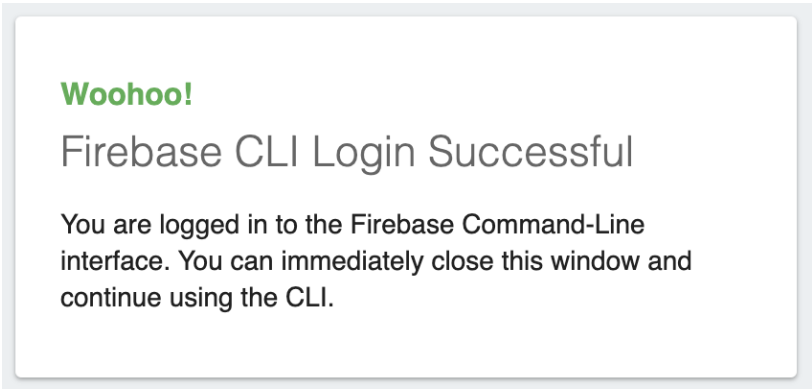
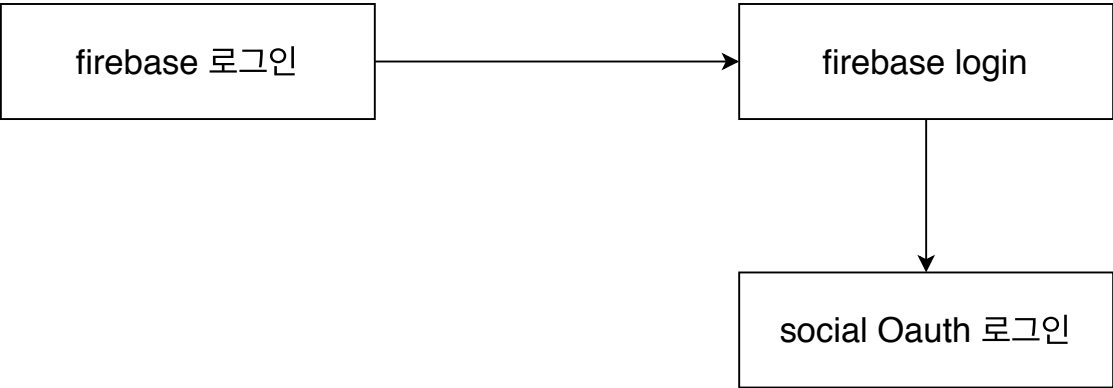
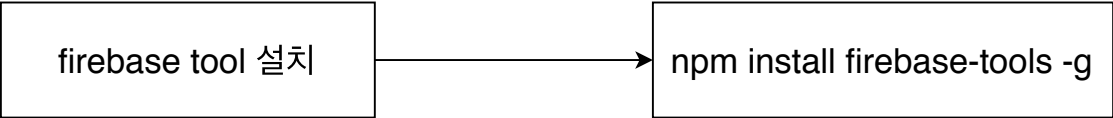
users

... SJ7Sha7kLCUVbLhwYftCct897UR2

```
└─ image: "http://gravatar.com/avatar/"
└─ name: "jaewon"
```

```
"users": {
  ".read": "auth != null",
  "$uid": {
    ".write": "auth != null && auth.uid === $uid",
    ".validate": "newData.hasChildren(['name', 'image'])",
    "name": {
      ".validate": "newData.val().length > 0"
    },
    "image": {
      ".validate": "newData.val().length > 0"
    }
  }
}
```

애플리케이션 배포하기



Database,
Storage 선택

```
? Which Firebase CLI features do you want to set up for this folder? Press Space to select features, then Enter to confirm your choices. (Press <space> to select, <a> to toggle all, <i> to invert selection)
```

```
Selection/
> Database: Deploy Firebase Realtime Database Rules
  ○ Firestore: Deploy rules and create indexes for Firestore
  ○ Functions: Configure and deploy Cloud Functions
  ○ Hosting: Configure and deploy Firebase Hosting sites
  ○ Storage: Deploy Cloud Storage security rules
  ○ Emulators: Set up local emulators for Firebase features
  ○ Remote Config: Get, deploy, and rollback configurations for Remote Config
```

Project Setup

=== Project Setup

First, let's associate this project directory with a Firebase project. You can create multiple project aliases by running **firebase use --add**, but for now we'll just set up a default project.

```
? Please select an option:
> Use an existing project
  Create a new project
  Add Firebase to an existing Google Cloud Platform project
  Don't set up a default project
```

```
? Please select an option: Use an existing project
? Select a default Firebase project for this directory:
  chat-bot-app-test (chat-bot-app-test)
> jaewon-react-slack (jaewon-react-slack)
```

Database & Storage Setup

=== Database Setup

Firebase Realtime Database Rules allow you to define how your data should be structured and when your data can be read from and written to.


```
? What file should be used for Database Rules? database.rules.json
✓ Database Rules for test-app-beca5 have been downloaded to database.rules.json.
Future modifications to database.rules.json will update Database Rules when you run
firebase deploy.
```

=== Storage Setup

Firebase Storage Security Rules allow you to define how and when to allow uploads and downloads. You can keep these rules in your project directory and publish them with **firebase deploy**.

```
? What file should be used for Storage Rules? (storage.rules)
```


`{}` database.rules.json

 firebase.json

배포(deploy)를 위한 빌드

npm run build

```
{
  "hosting": {
    "public": "./build"
  },
  "database": {
    "rules": "database.rules.json"
  },
  "storage": {
    "rules": "storage.rules"
  }
}
```

빌드하기

firebase deploy