CubeIt 3. výstup – Dokumentace

Fassinger Maxmilian

Table of Contents

[Serverové Notifikace 3](#_Toc33910817)

[Upozornění na novou zprávu 3](#_Toc33910818)

[Vysvětlení kódu: 4](#_Toc33910819)

[Upozornění prodání předmětu v herním obchodu (market) 5](#_Toc33910820)

[Vysvětlení kódu 5](#_Toc33910821)

[Reklamy v aplikaci 6](#_Toc33910822)

[Vytváření Privacy Policy a copyright 6](#_Toc33910823)

[Živý souboj 6](#_Toc33910824)

[Vyhodnocení splnění mise 6](#_Toc33910825)

[Obnova a aktualizace serverových pravidel 7](#_Toc33910826)

# Serverové Notifikace

Pro tento úkol jsem pracoval v prostředí Firebase Cloud Functions, který je psaný v jazyce Node.JS. Níže je uvedena ukázka kódu.

### Upozornění na novou zprávu

const functions = require('firebase-functions');

const admin = require('firebase-admin');

admin.initializeApp();

const change = admin.firestore();

exports.notifyOnNewMessage = functions.firestore.document('users/{username}/Inbox/{messageId}')

.onCreate(docSnapshot, context => {

const message = docSnapshot.data();

const recipentUsername = message['receiver'];

const senderUsername = message['sender'];

return admin.firestore().doc('/users/' + recipentUsername).get().then(userDoc => {

const registrationTokens = userDoc.get('registrationTokens');

const notificationBody = message['content'];

const notificationTitle = message['subject']

const payload = {

notification: {

title: notificationTitle,

body: notificationBody,

) }

return admin.messaging().sendToDevice(registrationTokens, payload);

})});

### Vysvětlení kódu:

Server sleduje všechny dokumenty v kolekci Users/(Uživatelské jméno)/Inbox. Pokud je vytvořen nový dokument, zapíše se informace o upozornění (příjemce, odesilatel a obsah upozornění). Dále se načtou tokeny příjemce (jedná se o jedinečný identifikátor uživatele) a obsah se zašle na všechny zařízení uživatele.

### Upozornění prodání předmětu v herním obchodu (market)

const functions = require('firebase-functions');

const admin = require('firebase-admin');

admin.initializeApp();

const change = admin.firestore();

exports.notifyOnItemSell = functions.firestore.document('market/{item}/').onChange(docSnapshot, context).then(docSnap =>{

const itemDoc = docSnap.data();

const currentTime = admin.firestore.FieldValue.serverTimestamp();

if (itemDoc['buyer'] != null) {

return functions.firestore.document('/users/' + itemDoc["seller"]).get().then(userDoc => {

const registrationTokens = userDoc.data('registrationTokens');

const payload = {

notification:

{

title: itemDoc["seller"] + ", one of your item was just sold!",

body: itemDoc["buyer"] + " has bought your " + itemDoc['itemName'] + " for market price!",

}

}

return admin.messaging().sendToDevice(registrationTokens, payload)

})

}

else {

}})

### Vysvětlení kódu

Server sleduje všechny změny v kolekci „market“. Pokud se změní pole „buyer“ (kupující) – který je při zápisu předmětu prázdný, považuje se předmět jako prodaný. Poté se načtou tokeny uživatele a odešle se upozornění.

# Reklamy v aplikaci

#### Spolupráce s Jakubem Kostkou

V současné době není možné do aplikace implementovat ani zkušební reklamy – z důvodu zablokování našeho AdMob účtu. Důvod není v momentálně jasný, čekáme na odpověď od Google Support. Zablokování AdMob účtu totiž zabrání přístup do portálu, přes který je možné získat identifikátory pro testovací zařízení.

# Vytváření Privacy Policy a copyright

#### Spolupráce s Davidem Boříkem

Na tomto úkolu jsem spolupracoval s Davidem Boříkem, jelikož jsem rodilý mluvčí angličtiny (hra je v angličtině). Pomáhal jsem mu z hlediska gramatiky a korektury.

# Živý souboj

#### Spolupráce s Jakubem Kostkou

Vývoj na této časti převzal Jakub Kostka – po konzultaci se věnuji serveru/databázi.

# Vyhodnocení splnění mise

#### Spolupráce s Oldřichem Čihákem

Vývoj na této část převzal Oldřich Čihák po konzultaci s Jakubem Kostkou.

# Obnova a aktualizace serverových pravidel

Serverové pravidla jsem aktualizoval podle osnovy Jakuba Kostky.

Aktuální podoba serverových pravidel:

rules\_version = '2';

service cloud.firestore {

match /databases/{database}/documents {

match /admin/v0.1-cihak/UserPreferences/{preferance} {

allow read, write: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList

&& request.auth.uid == resource.id;

match /History/{document=\*\*} {

allow read;

}}

match /admin/v0.1-cihak/PublicTemplates/{document=\*\*} {

allow read: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /users/{username} {

allow read;

allow create: if request.auth.uid != null;

allow write: if (request.auth.uid == resource.data.userId &&

request.resource.data.username == resource.data.username &&

request.resource.data.userId == resource.data.userId);

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;

match /Timestamp/{document=\*\*} {

allow read;

allow create: if request.auth.uid != null;

allow write: if request.auth.uid == get(/databases/$(database)/documents/users/$(request.auth.token.name)).data.userId;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /ActiveQuest/{document=\*\*}{

allow read;

allow create: if request.auth.uid != null;

allow write: if request.auth.uid == get(/databases/$(database)/documents/users/$(request.auth.token.name)).data.userId;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /Allies/{document=\*\*}{

allow read;

allow create: if request.auth.uid != null;

allow write: if request.auth.uid == get(/databases/$(database)/documents/users/$(request.auth.token.name)).data.userId;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /Inbox/{document=\*\*}{

allow read;

allow create: if request.auth.uid != null;

allow write: if request.auth.uid == get(/databases/$(database)/documents/users/$(request.auth.token.name)).data.userId;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /Socials/{document=\*\*}{

allow read;

allow create: if request.auth.uid != null;

allow write: if request.auth.uid == get(/databases/$(database)/documents/users/$(request.auth.token.name)).data.userId;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}}

match /RocketGame/{document=\*\*}{

allow read: if request.auth.uid != null;

allow create: if request.auth.uid != null;}

match /factions/{document=\*\*} {

allow read;

allow create: if get(/databases/$(database)/documents/users/$(request.auth.token.name)).data.factionID == null;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;

//Member role

allow update: if (resource.data.members[request.auth.token.name].role == "MEMBER" &&

request.resource.data.allyFactions == resource.data.allyFactions &&

request.resource.data.captureDate == resource.data.captureDate &&

request.resource.data.democracy == resource.data.democracy &&

request.resource.data.description == resource.data.description &&

request.resource.data.enemyFactions == resource.data.enemyFactions &&

request.resource.data.experience == resource.data.experience &&

request.resource.data.externalDescription == resource.data.externalDescription &&

request.resource.data.id == resource.data.id &&

request.resource.data.invitationMessage == resource.data.invitationMessage &&

request.resource.data.leader == resource.data.leader &&

request.resource.data.members == resource.data.members &&

request.resource.data.name == resource.data.name &&

request.resource.data.openToAllies == resource.data.openToAllies &&

request.resource.data.pendingInvitationsFaction == resource.data.pendingInvitationsFaction &&

request.resource.data.pendingInvitationPlayer == resource.data.pendingInvitationPlayer &&

request.resource.data.recruiter == resource.data.recruiter &&

request.resource.data.taxPerDay == resource.data.taxPerDay &&

request.resource.data.warnMessage == resource.data.warnMessage);

//Moderator role

allow update: if (resource.data.member[request.auth.token.name].role == "MODERATOR" &&

request.resource.data.allyFactions == resource.data.allyFactions &&

request.resource.data.captureDate == resource.data.captureDate &&

request.resource.data.democracy == resource.data.democracy &&

request.resource.data.enemyFactions == resource.data.enemyFactions &&

request.resource.data.experience == resource.data.experience &&

request.resource.data.externalDescription == resource.data.externalDescription &&

request.resource.data.id == resource.data.id &&

request.resource.data.leader == resource.data.leader &&

request.resource.data.name == resource.data.name &&

request.resource.data.pendingInvitationsFaction == resource.data.pendingInvitationsFaction &&

request.resource.data.recruiter == resource.data.recruiter);

//Leader role

allow update: if (resource.data.members[request.auth.token.name].role == "LEADER" &&

request.resource.data.captureDate == resource.data.captureDate &&

request.resource.data.experience == resource.data.experience &&

request.resource.data.id == resource.data.id &&

request.resource.data.name == resource.data.name &&

request.resource.data.pendingInvitationsFaction == resource.data.pendingInvitationsFaction);

//Invited user

allow update: if (request.auth.token.name in resource.data.pendingInvitationsFaction &&

request.resource.data.actionLog == resource.data.actionLog &&

request.resource.data.allyFactions == resource.data.allyFactions &&

request.resource.data.captureDate == resource.data.captureDate &&

request.resource.data.democracy == resource.data.democracy &&

request.resource.data.description == resource.data.description &&

request.resource.data.enemyFactions == resource.data.enemyFactions &&

request.resource.data.experience == resource.data.experience &&

request.resource.data.externalDescription == resource.data.externalDescription &&

request.resource.data.fame == resource.data.fame &&

request.resource.data.id == resource.data.id &&

request.resource.data.invitationMessage == resource.data.invitationMessage &&

request.resource.data.leader == resource.data.leader &&

request.resource.data.level == resource.data.level &&

request.resource.data.name == resource.data.name &&

request.resource.data.openToAllies == resource.data.openToAllies &&

request.resource.data.pendingInvitationPlayer == resource.data.pendingInvitationPlayer &&

request.resource.data.recruiter == resource.data.recruiter &&

request.resource.data.taxPerDay == resource.data.taxPerDay &&

request.resource.data.warnMessage == resource.data.warnMessage);

}

match /surfaces/{document=\*\*}{

allow read;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /npcs/{document=\*\*}{

allow read;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /GenericDB/{document=\*\*} {

allow read;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /test/{document=\*\*} {

allow read, write: if true;}

match /FightLog/{document=\*\*} {

allow read, create: if request.auth.uid != null;

allow update: if request.auth.token.name == resource.data.winnerName; }

match /CommunityStories/{document=\*\*} {

allow read, delete, create: if request.auth.uid != null;}

match /charclasses/{document=\*\*} {

allow read; }

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /items/{document=\*\*} {

allow read: if request.auth.uid != null;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /spells/{document=\*\*} {

allow read;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /story/{document=\*\*} {

allow read;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList; }

match /globalDataChecksum/{document=\*\*}{

allow read;

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /market/{document=\*\*}{

allow read;

allow create: if request.auth.uid != null;

allow delete: if request.auth.token.name == resource.data.seller;

allow update: if (request.auth.token.name == resource.data.seller &&

request.resource.data.afterExpiryCubeCoins == resource.data.afterExpiryCubeCoins &&

request.resource.data.afterExpiryCubix == resource.data.afterExpiryCubix &&

request.resource.data.afterExpiryDate == resource.data.afterExpiryDate &&

request.resource.data.closeAfterExpiry == resource.data.closeAfterExpiry &&

request.resource.data.creationTime == resource.data.creationTime &&

request.resource.data.id == resource.data.id &&

request.resource.data.item == resource.data.item &&

request.resource.data.itemClass == resource.data.itemClass &&

request.resource.data.itemLevel == resource.data.itemLevel &&

request.resource.data.itemName == resource.data.itemName &&

request.resource.data.itemQuality == resource.data.itemQuality &&

request.resource.data.itemType == resource.data.itemType &&

request.resource.data.seller == resource.data.seller);}

match /Server/AuthorizedUsers {

allow read, write, create, delete, update: if request.auth.uid in

get(/databases/$(database)/documents/Server/AuthorizedUsers).data.usersList;}

match /Server/Generic {

allow read; }

match /Server/GlobalLiveMessage {

allow read; }

match /CommunityStories/{document=\*\*} {

allow create: if request.auth.uid != null;

allow update, update:

if request.auth.name == resource.data.author &&

request.resource.data.id == resource.data.id &&

request.resource.data.reward == resource.data.reward &&

request.resource.data.uploadData == resource.data.uploadData; } } }