FRANKFURT UNIVERSITY OF APPLIED SCIENCES

VGU Vietnamese-German University

Programming Exercise Project SS18—Manuel G. Clavel

Project "My Little Shop" Delivery

Team G (My Little Shop Web Application)

Nguyen Dao Thuy Tien - 9517
Nguyen Tien Hung - 9972
Luong Tri Dung - 9963
Nguyen Thanh Duy - 1525

May 10, 2018

Contents

1 Team Work

1.1 Work of individuals:

- Tien Nguyen: Team management, QRCode scanner, testing, documentation.
- Hung Nguyen: Front-end (design all the pages for the website), integration (connect front-end and back-end).
- Dung Luong: Databases, cloud functions (back-end process functions).
- Duy Nguyen: Security (hash function for passwords). The hash function cannot be integrated.

1.2 Work Structure:

- Code storage and backup: GitHub and Google Drive.
- Web App deploy: Firebase Service.
- Version Control Management (Databases, Front-end, Server, Functions, Authentication): Firebase.
- Communication: In class and online.

1.3 Ideas:

According to the project requirements, we decided to build a Web Application based on Firebase Cloud Service which can run on multiple platforms. Firebase provides hosting, realtime databases and supports cloud-functions which can receive requests from the clients and give responses to them. The application allows the managers to add products by scanning QR Code and modify products as well as employee's accounts. The employee can use this to scan the QR code on the products and calculate the receipts. After that, the sold products (considered as transactions in database) will automatically recorded and displayed to the shop managers on Dashboard. All the processes and data are realtime synchronized.

2 Project Description

2.1 Functional Requirements:

We aim to make an application which is a multi-store POS system. The system is served to a small retailer, allows the user to scan the QR Code, calculate the receipts, manage the products in shops and see total revenue.
These are the basic functions of our application:

- Login - Logout: Every user has to log in with username and password which are automatically assigned the role and limit the functions for the user. User can change the password.
- Dashboard: All the product information, total revenue and number of products (IN/OUT/Balance) will be displayed here. The global manager can see the records of all shops, the shop manager can only see one. Some filter options are offered such as time duration, kinds of product and transaction.
- Users (add/modify): The global manager and each shop leader can add new users and modify (change or remove) the user's information.
- Products (add/modify/import/export): The manager and each shop leader can add new products (enter product code and price) and modify the product's information. The employee can scan the QR Code by Check-out function to calculate the bill for customers. After confirming the transaction, the products in receipt will be updated to Dashboard.

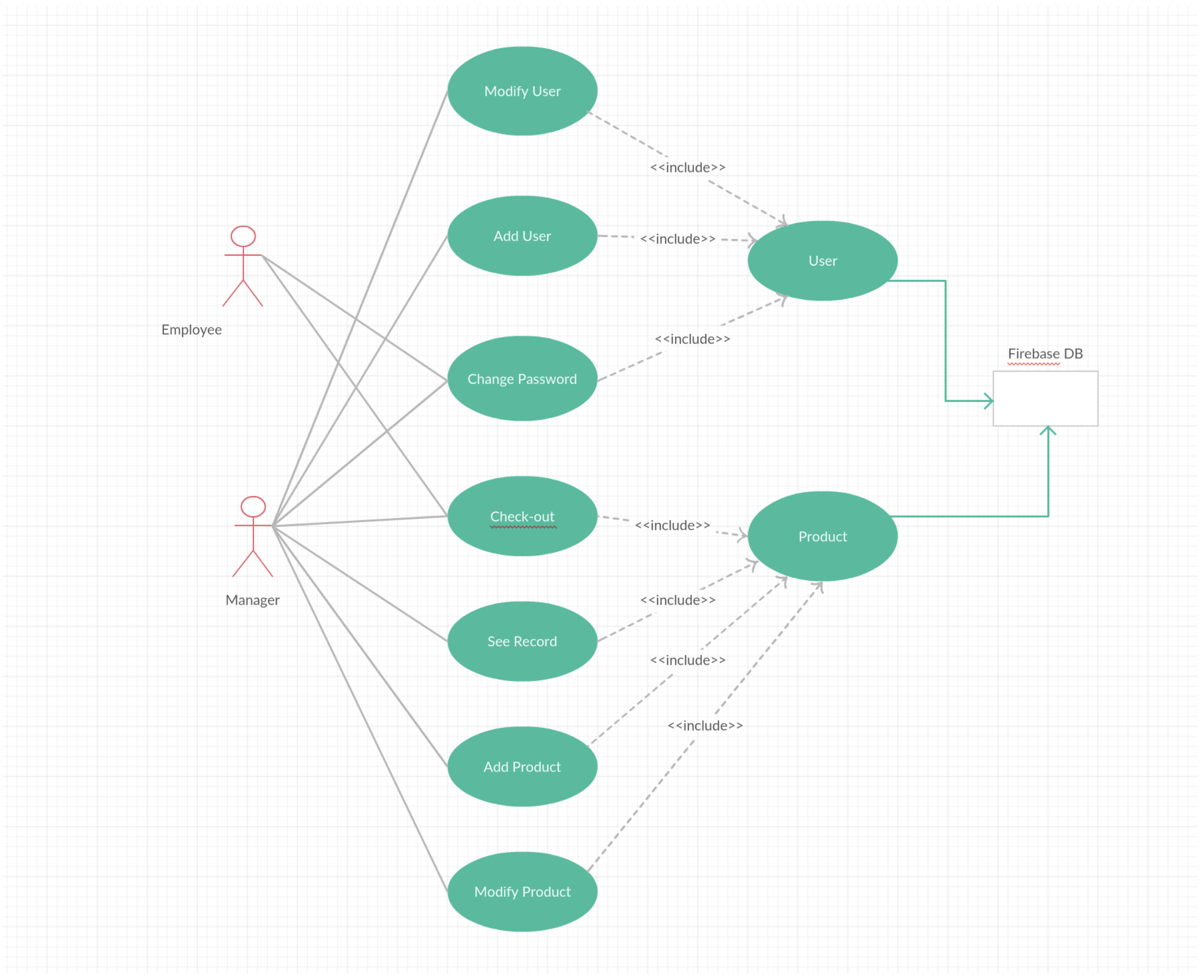
2.2 Non-functional Requirements:

Beside the functional requirements, the project needs to satisfy some non-functional requirements:

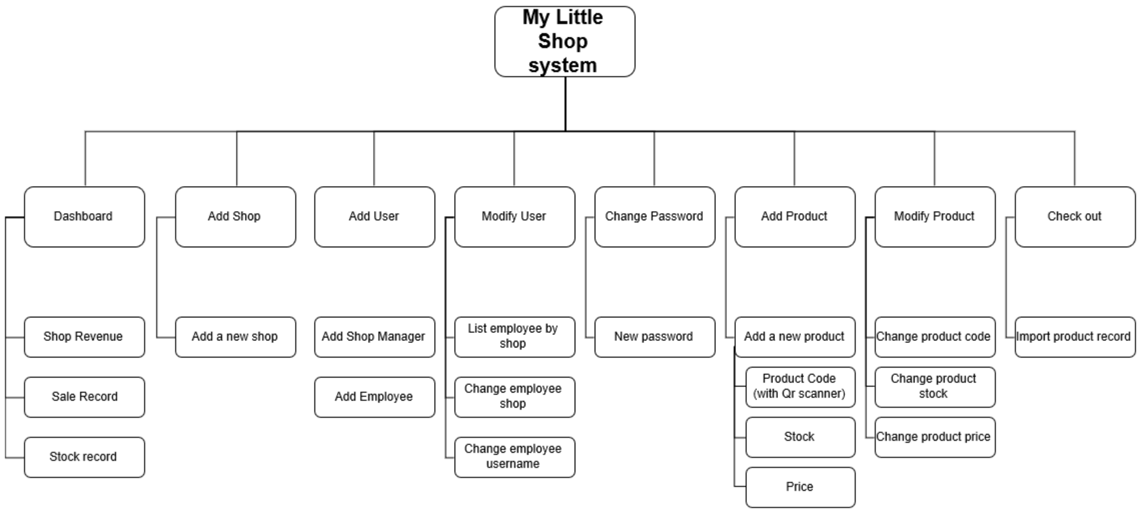
- Performance: the response time is short. Every request is responded immediately, the updated data quickly acts realtime.
- Scalability: the system still works properly though the numbers of users and requests is increasing significantly.
- Availability: the resources can be retrieved and the services are provided at any time.
- Security: the user's information (password) has to be encrypted. The internal database also must be kept securely.

3 Design Description

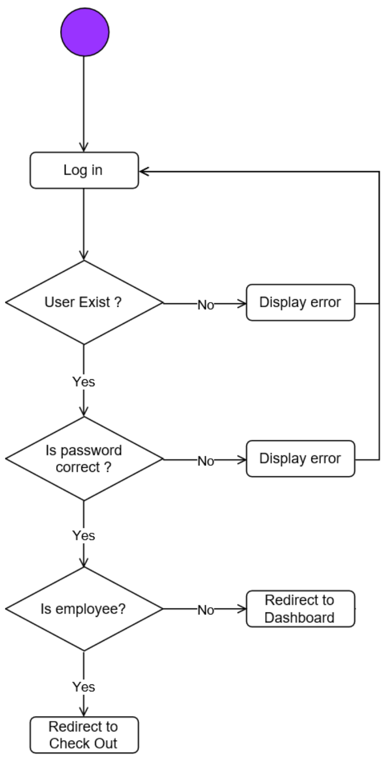
3.1 Use Case Diagram:



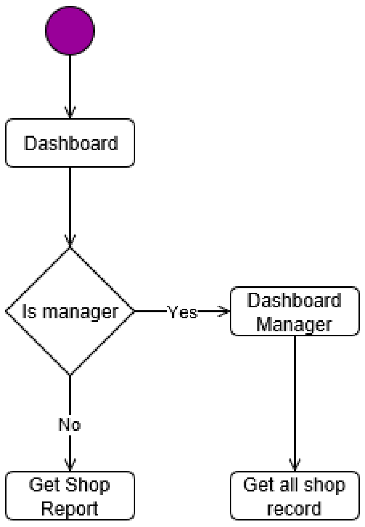
3.2 Components Diagram:



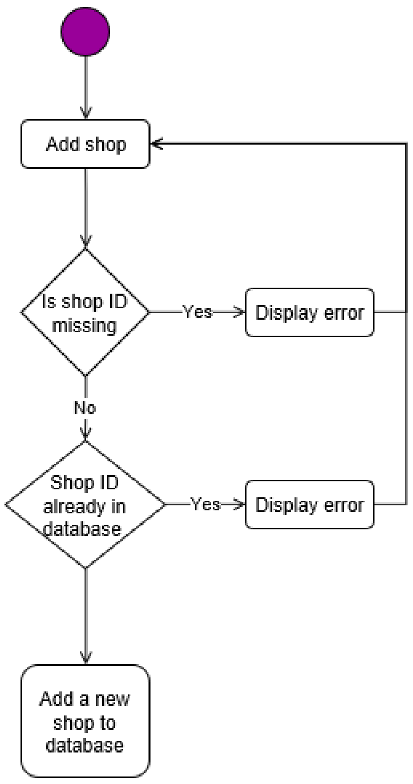
3.3 Activity Diagram:



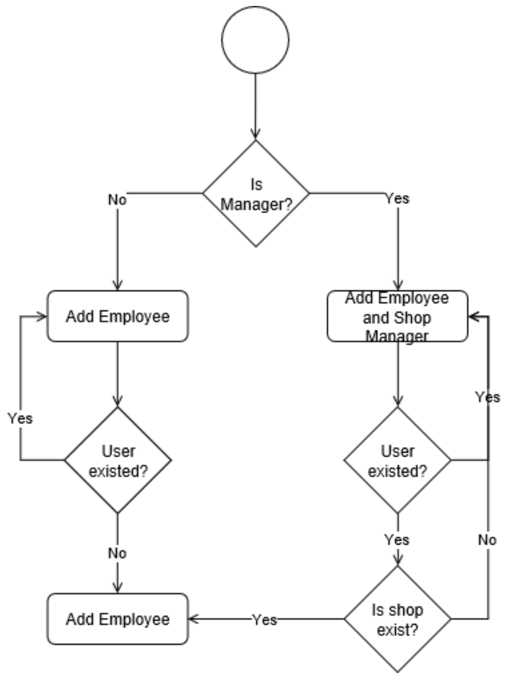
Log-in Activity Diagram



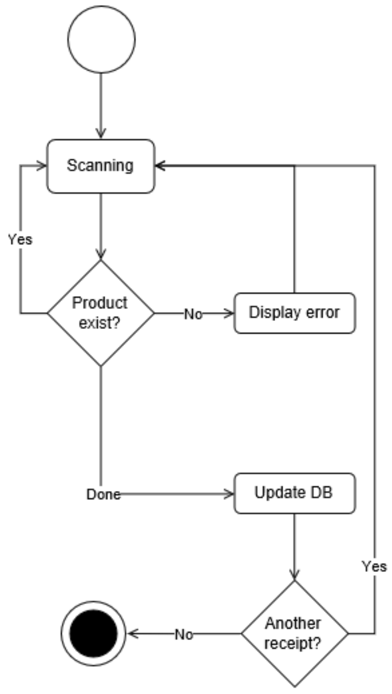
Dashboard Activity Diagram



Adding a new shop Activity Diagram

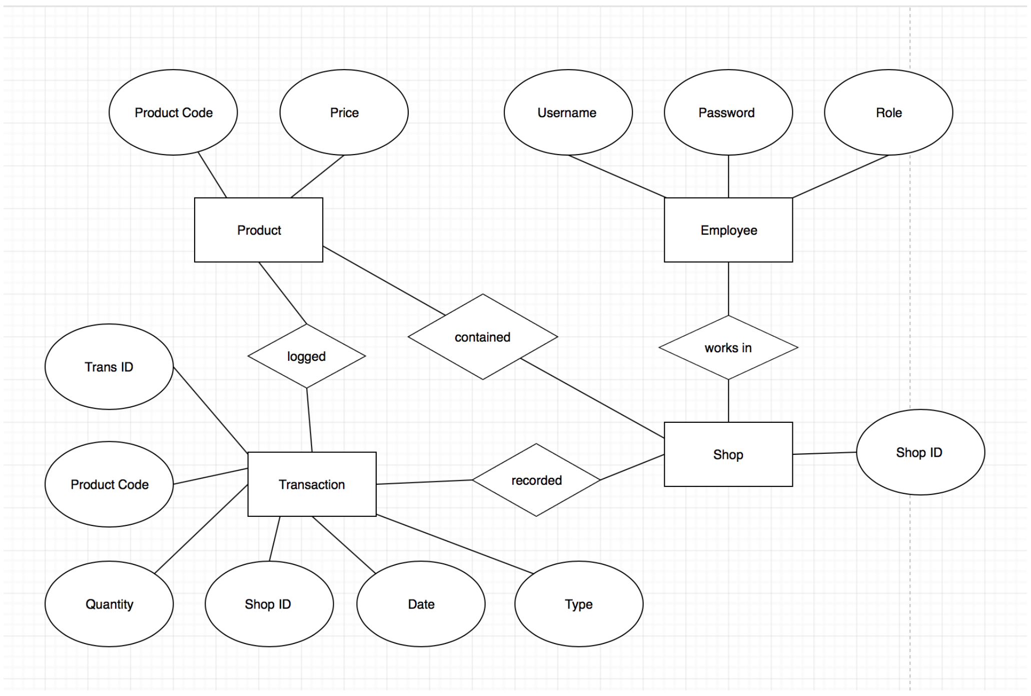


Adding a new user Activity Diagram



Check-out Activity Diagram

3.4 Entity-Relationship diagram:



4 Architecture Description

According to current context, where Internet is popular and everything can easily be connected, we come on with using Cloud Service solution which brings us many advantages. There are several Cloud Service Providers and we choose Firebase service among them.

4.1 Introduction to Firebase:

Firebase is a mobile and web application development platform acquired by Google in 2014. Firebase allows developers to build and maintain the app in a convenient way. There are many services provided in Firebase such as:

- Hosting.
- Realtime Database.
- Cloud Storage.
- Authentication.
- Cloud Functions.
- Crash Reporting.
- and many more services help enhance User Experience.

4.2 Project Structure:

Based on individual work and Firebase structure, "My Little Shop" project has the following parts:

- Hosting: "public" directory contains index.html file and app.js file controlling the main process of application.
- Database: the default realtime database provided by Firebase.
- Functions (cloud functions): all login, adding, mofifying, scanning functions are put on Cloud as back-end process which can only be accessed by Firebase server. The clients send requests to Firebase server, the server calls cloud functions and sends responses to clients.

4.3 Project Configure Instruction:

Install the Firebase CLI:

The Firebase CLI (Command Line Interface) requires Node.js and npm, which can both be installed by following the instructions on https://nodejs.org/. Installing Node.js also installs npm.
Once you have Node.js and npm installed, you can install the Firebase CLI via npm:
```
npm install -g firebase-tools
```

Initialize your app:

Once you've chosen the Firebase app you'd like to deploy, cd into your project directory and run the command:
```
firebase init
```

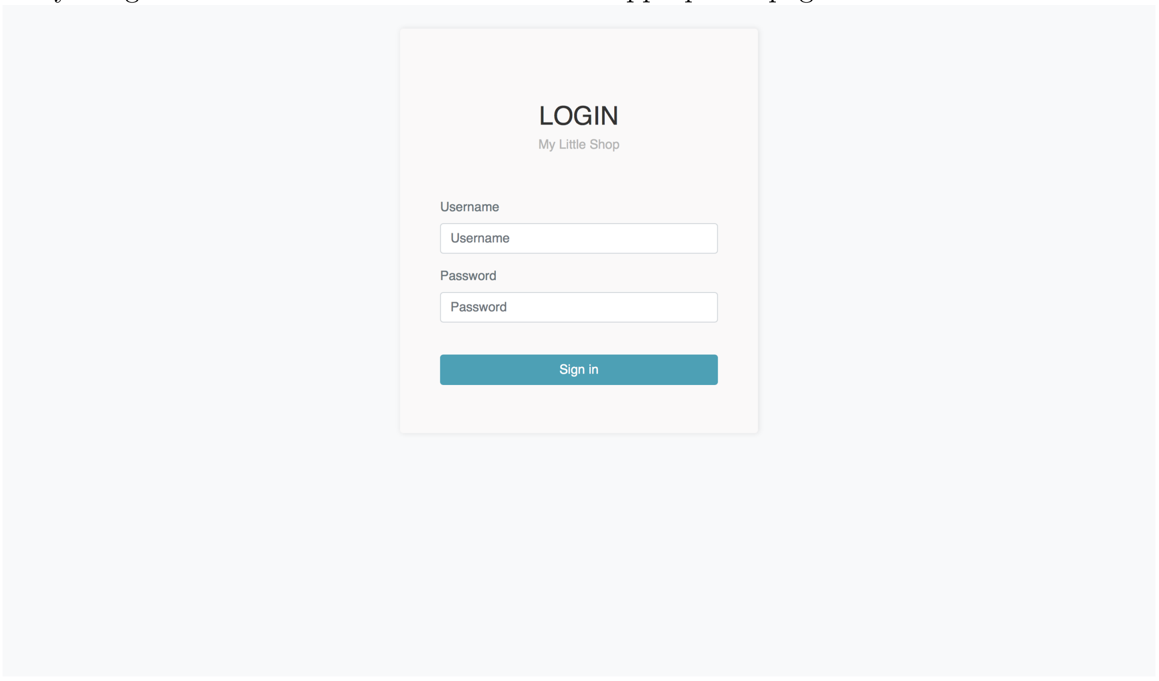
Deploy the website:

To deploy your website simply run:
```
firebase deploy
```
Your app will be deployed to the domain `<YOUR-FIREBASE-APP>.firebaseapp.com`

5 Implementation Details:

5.1 Login:

Every user must login to the website using username and password which are automatically assigned the role and redirect user to the appropriate page.

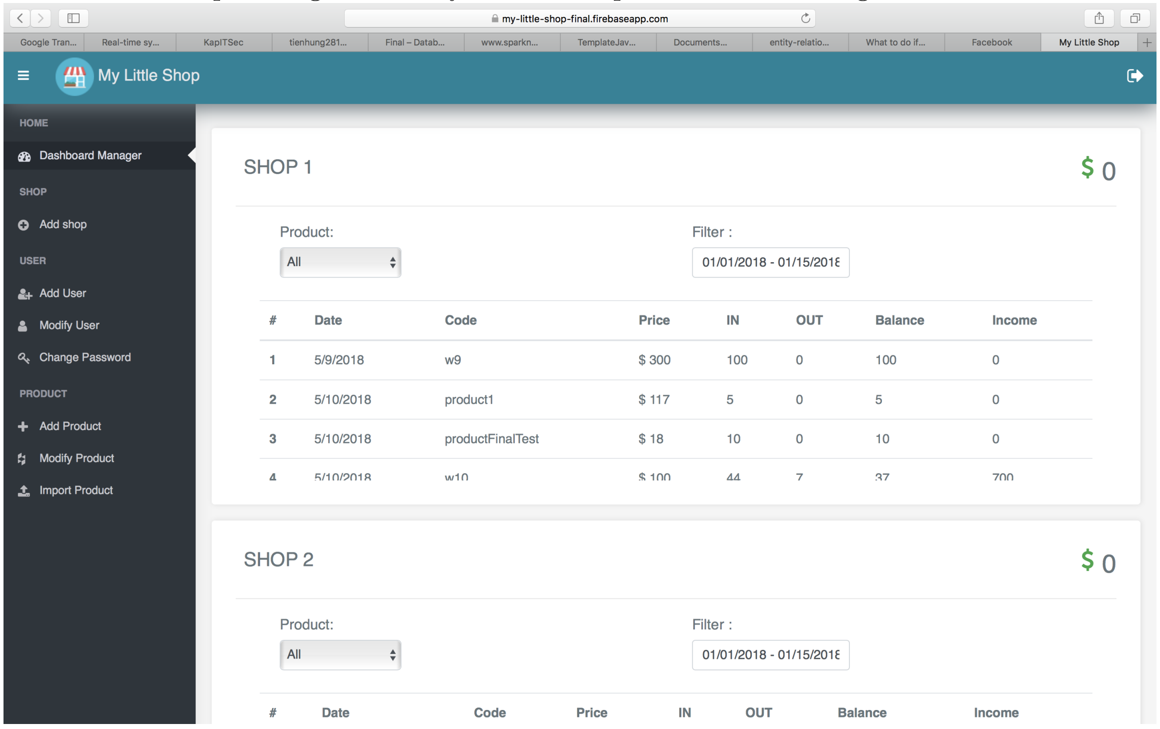


Our function can be shown in the pseudocode 5.1.

```
        exports.checkLogin = functions.https.onRequest((req,res)=>{
            cors(req, res, () => {
                const params = req.url.split("/");
                const usrId = params[1];
                const password = params[2];
                return admin.database().ref('employee/'+usrId).once('
                    value',(userSnapshot)=>{

                var usr = userSnapshot.val();
                var pass = usr.password;
                var role = -1; // default
                var shop_id = -1; // default
                var isMatched = false;

                if(password == pass){
                    isMatched = true;
                    role = usr.role;
                    shop_id = usr.shop_id;
                }

                var data = {
                    usr: usrId,
                    pass: password,
                    role : role,
                    shop_id : shop_id,
                    result: isMatched
                }

                res.send(data);
                });
            });
        });
```
Listing 5.1: Check login function

5.2 Dashboard:

The system manager can see all the sale records and stock records of all shops on dashboard. Each shop manager can only see the shop that he is working in.



SHOP 1 $ 0

Product: All

Filter : 01/01/2018 - 01/15/2018

#	Date	Code	Price	IN	OUT	Balance	Income
1	5/9/2018	w9	$ 300	100	0	100	0
2	5/10/2018	product1	$ 117	5	0	5	0
3	5/10/2018	productFinalTest	$ 18	10	0	10	0
4	5/10/2018	w10	$ 100	44	7	37	700

SHOP 2 $ 0

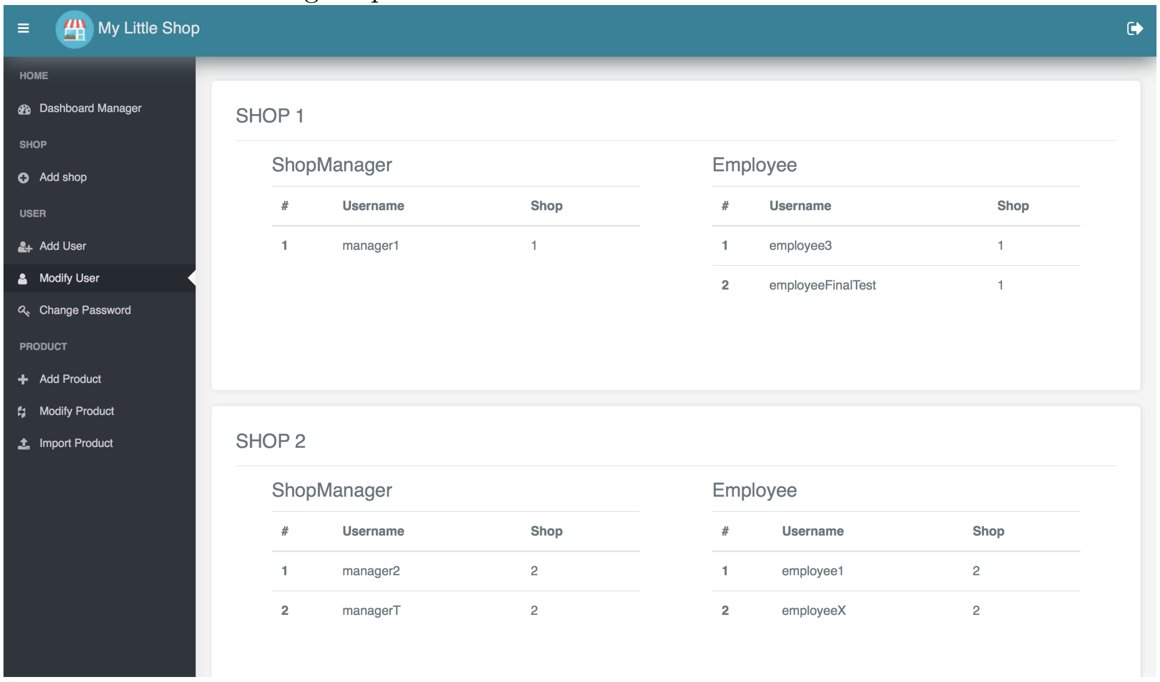
Product: All

Filter : 01/01/2018 - 01/15/2018

#	Date	Code	Price	IN	OUT	Balance	Income

5.3 User:

The system manager can add new shop managers as well as employees. Meanwhile, the shop manager can only add new employees to his shop. A manager or employee can not be added to non-existing shop.



SHOP 1

ShopManager

#	Username	Shop
1	manager1	1

Employee

#	Username	Shop
1	employee3	1
2	employeeFinalTest	1

SHOP 2

ShopManager

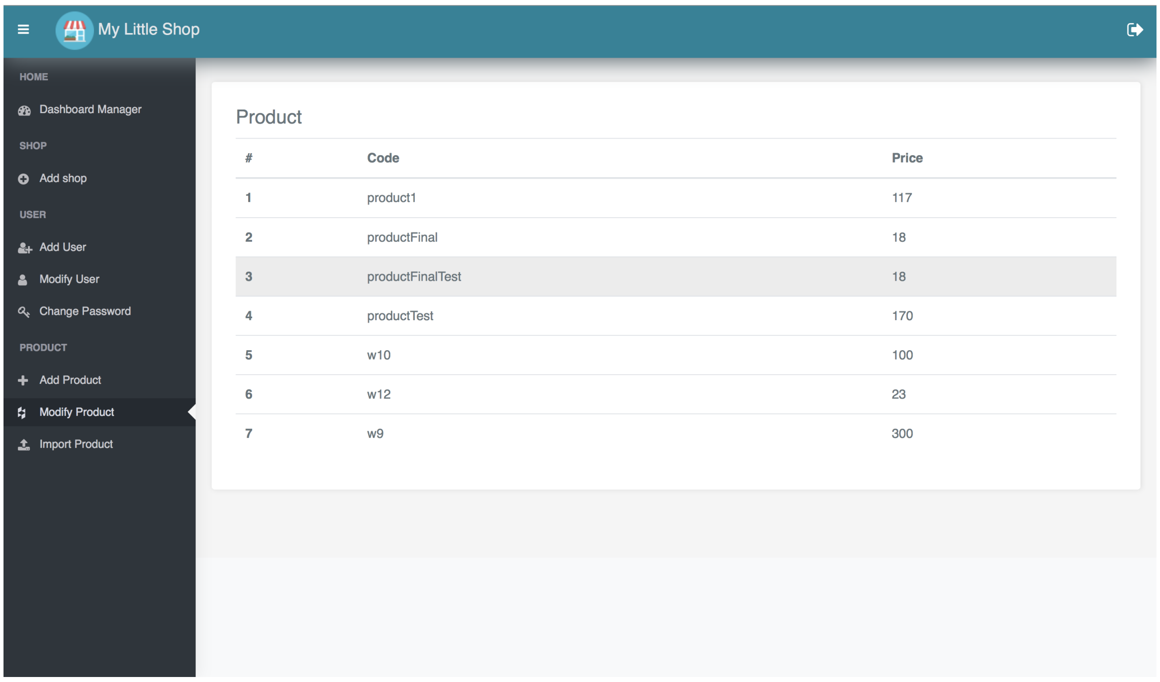
#	Username	Shop
1	manager2	2
2	managerT	2

Employee

#	Username	Shop
1	employee1	2
2	employeeX	2

5.4 Product:

The shop manager can add new products to his shop without assigning shop ID. The system manager needs to point out the shop ID when adding a new product. It can be seen that the client cannot type the code for a product, therefore the website is designed to scan the QR Code for adding a new product, the user needs to enter only valid values for price and stock.



Product

#	Code	Price
1	product1	117
2	productFinal	18
3	productFinalTest	18
4	productTest	170
5	w10	100
6	w12	23
7	w9	300

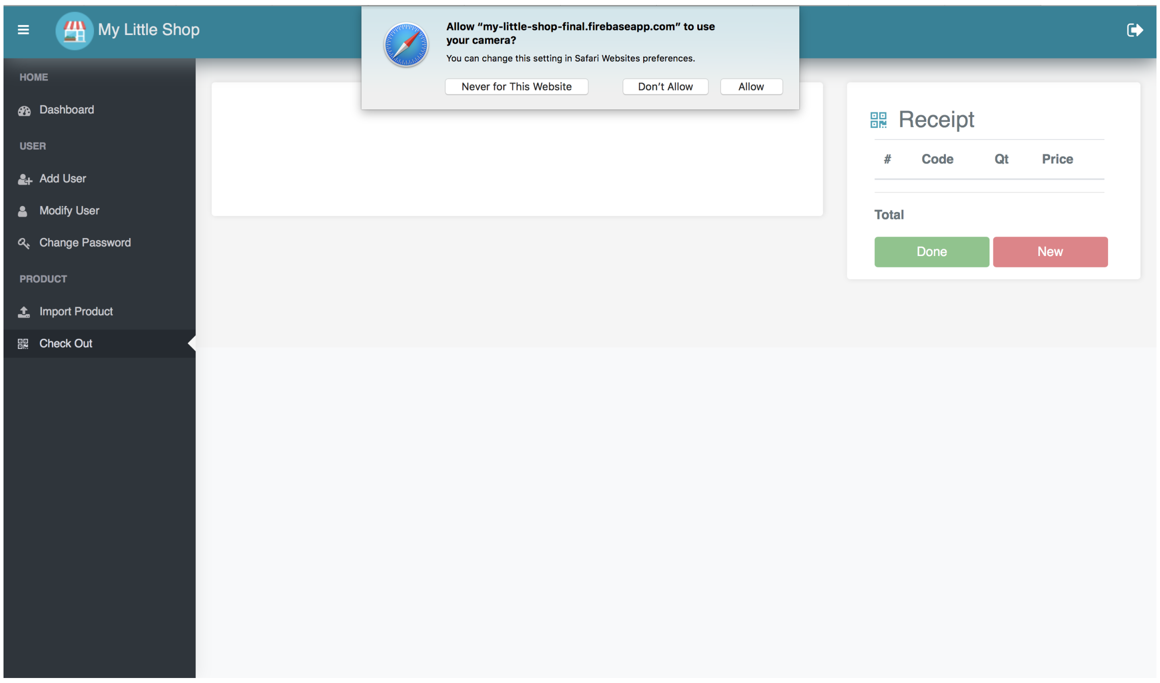
Our function can be shown in the pseudocode 5.2.

```
        exports.modifyProduct = functions.https.onRequest((req,res)=>{
            cors(req,res,() =>{
                const params = req.url.split("/");
                const product = JSON.parse(decodeURI(params[1]));

                var product_code = product.code;
                var old_product_code = product.oldCode;
                var price = product.price;
                var data;
                var result;
                var updates = {};
                var productExist = false;

                admin.database().ref('/products/'+product_code).once('
                    value',(snapshot)=>{
                    if(snapshot.exists()){
                        productExist = true;
                        data = {
                            price: price

                        }
                        updates['/products/'+product_code] = data;
                        admin.database().ref().update(updates);


                    } else{
                        admin.database().ref('/products/'+
                            old_product_code).once('value',(
                            productSnapshot)=>{
```

```

                            data = {
                                price: price

                            }

                            updates['/products/'+product_code] = data;
                            admin.database().ref().update(updates);
                            admin.database().ref().child('/products/'+
                                old_product_code).remove();
                        });
                    }
                    result = {
                        productExist: productExist,
                        shopExist: shopExist
                    }
                    res.send(result);
                });
            });
        });

        exports.removeProduct = functions.https.onRequest((req,res)=>{
            cors(req,res,() =>{
                const params = req.url.split("/");
                const product_code = params[1];
                const shop_id = params[2];
                admin.database().ref().child('shop/'+shop_id+'/products/'
                    +product_code).remove();
                return res.send(true);
            });
        });
```
Listing 5.2: Modify and Remove Product function

5.5 Check-out:

The employees need this function in order to calculate the receipts for customers. Whenever a receipt is confirmed ”done”, all the products will be updated on sale and stock records on dashboard.



Allow "my-little-shop-final.firebaseapp.com" to use your camera?

You can change this setting in Safari Websites preferences.

Our function can be shown in the pseudocode 5.3.

```
        exports.saveRecord = functions.https.onRequest((req,res)=>{
            cors(req,res,() =>{
                const params = req.url.split("/");
                const product = JSON.parse(decodeURI(params[1]));


                var updates = {};

                var shop_id = product.shopID;
                var time = product.time;
                var code = product.product_code;
                var qty = product.qty;

                var type = 'export';
                var data;

                data = {
                    shopID : shop_id,
                    code: code,
                    qty : qty,
                    time: time,
                    type: type
                }


                var databaseRef = admin.database().ref('transaction');
                var newTransactionKey = databaseRef.push().key;
                console.log('Add transaction: '+newTransactionKey);
                updates['/transaction/'+newTransactionKey] = data;
                admin.database().ref().update(updates);
                res.send(true);
            });
        });

        exports.checkProduct = functions.https.onRequest((req,res)=>{
            cors(req,res,() =>{
                const params = req.url.split("/");
                const product_code = params[1];

                var isFound = false;

                var data = {
                    result: isFound
                }

                admin.database().ref('products').once('value',(snapshot)
                    =>{
                    snapshot.forEach(function(productID){
                        if(productID.key==product_code){
                            console.log('Existed');
                            isFound = true;
                            data = {
                                product_code: product_code,
                                price: productID.val().price,
                                result: isFound
                            }
                            res.send(data);
                            return;
                        }
                    })
                    res.send(data);
                });
            });
        });
```

Listing 5.3: Modify and Remove Product function

6 Conclusions and Future Work

6.1 Firebase Review:

After completing the ”My Little Shop” project, we have some certain experiences with Firebase platform. Firebase brings the web developers many benefits. However, it also has several drawbacks which might be improved in the near future. Firebase is quite new and still in developing stage, it is getting more and more attention from the community. We can expect that this potential platform will be more complete and stable soon.

6.1.1 Advantages:

- Simple: Firebase has a simple interface, very useful and short documentation which are good for the developers to use.
- Realtime Database: It does not require much effort to gain realtime data system. How fast everything runs is very impressive.
- Static Hosting: Zero-configuration hosting runs on Google Cloud platform.
- Authentication. Firebase auth includes a built-in email/password authentication system. It also supports OAuth2 for Facebook, Google, Twitter and GitHub.

6.1.2 Disadvantages:

- Bad data structure: Complex queries are impossible. The whole data tree is downloaded so it is impossible to control the resources. It is providing a data service with no searching or filtering capabilities.
- Relations: Dealing relations with NoSQL is hard, dealing relations with Firebase is a nightmare. Developers have to handle links between different data branches themselves.
- Inconsistencies: Firebase supports offline operations. It works like GitHub allowing users to deploy the project, but the main issue is that it does not have ”commit” and merge the changes.

6.2 Risk Managements:

Firebase has several drawbacks which make developer team face with certain risks.
The ability of customizing the authentication is limited except for using the supported identity providers available on Firebase. We decided ourselves to save the username and password of users on realtime database and try to encrypt them for security. However, at the moment, the hash function is not integrated to the project.
Firebase aims to help developers create an application simpler and quicker. However, we are forced to use the features offered by Firebase. In the future, in case Firebase does not work anymore, we have very little to recover data and reuse source code. One possible solution we can come up with is to use another Cloud Service Provider such as Microsoft Azure or Amazon Web Services and link to another Database such as MongoDB. The price for those services are certainly higher and the work requires more efforts but the quality of product is obviously much better.

6.3 Further Work:

Thanks to this project, we learned to build a real web application for running on multiple platform and solving certain tasks. We also learned to work as a team, how to plan the whole project such as decide how the website and database look like, how to split the work to individuals and integrate different parts, how to communicate and how to overcome our problems together.
In the moment, we have not satisfied certain situations and planned to improve in the near future:

- Solve product price changes problem (how to manage the prices over the time and do not lose any data).
- Support discount function.
- Print the receipt function.
- Credit card payment function.
- Improve security (2-step authentication and use hash function to encrypt the data).