Mobile Computing

Report about the development of our application: 'ShoppingApp'

Hong Kong Baptist University

Engelina Saarloos - 15501701

Sun Xiaomeng - 12251216

INTRODUCTION

E-commerce is unthinkable to life without nowadays. It is defined as the exchange in products or services using information technology such as the World Wide Web. The increase in mobile phone use has played a dominant role in the popularity gain of E-commerce which we call it Mobile Commerce. Nowadays almost every company has a mobile application. Since the creation of a mobile application is cheap, compared to owning an actual physical store, companies can sell to a broad consumer range without making many costs. Such application has very huge potential.

For this reason we make an application in the Mobile Commerce area. It is an online shop in which customers can view items including their price through their mobile phones rather than going out far to the retail store.

COMPILING ENVIRONMENT

Platform: iOS version 8.4

Software: Appcelerator Studio version 5.1.1.GA

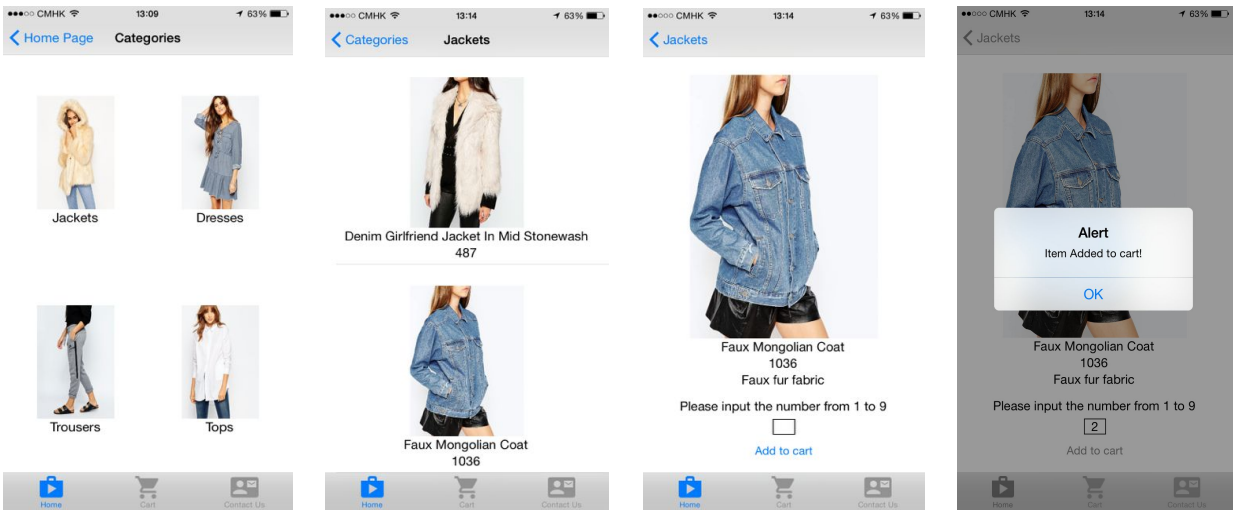
Database: SQLite Manager Firefox version 0.8.3.1

Testing Device: iPhone 6 (v8.3)

INTERFACES

Our app is a 3-tab application. The first tab is the home page. There are two rows.

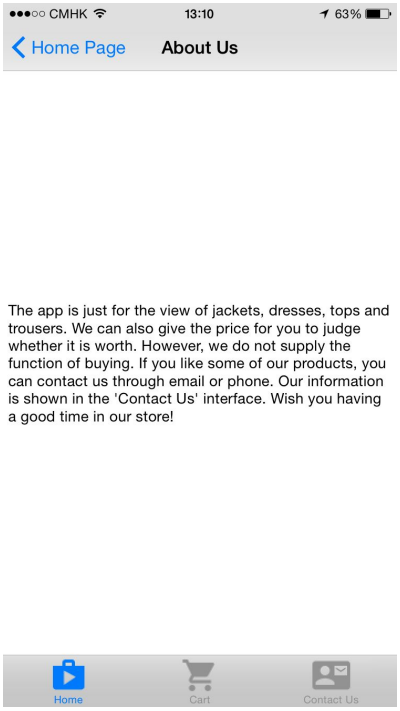
Through the first row we can view all the items in our store and choose some of them to add into the cart.

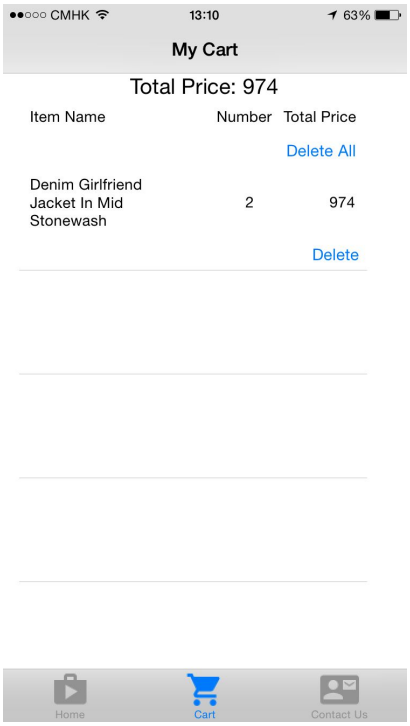


By clicking the second row, the user can get a brief introduction of our app.

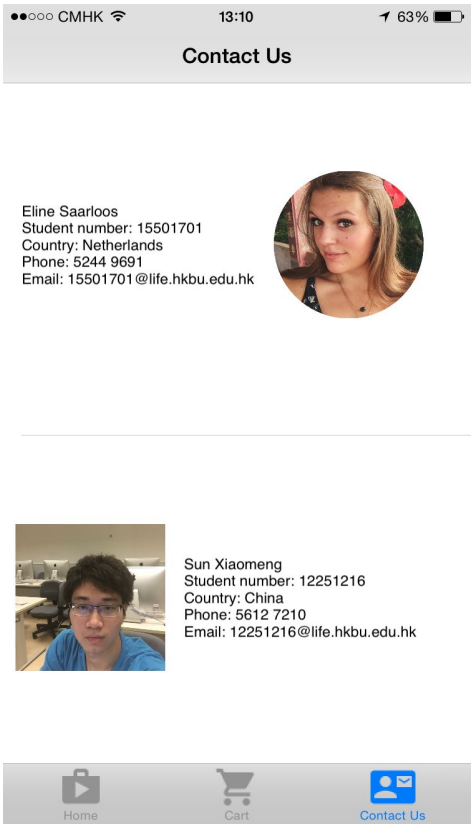
After the user add some items in the cart, she can go to the second tab “Cart” to see what items she has chosen. The total price will also be shown in this tab. If

the user wants to remove some items, just click the “Delete” button in the item row. Of course, the user can empty the cart by clicking the “Delete All” button.

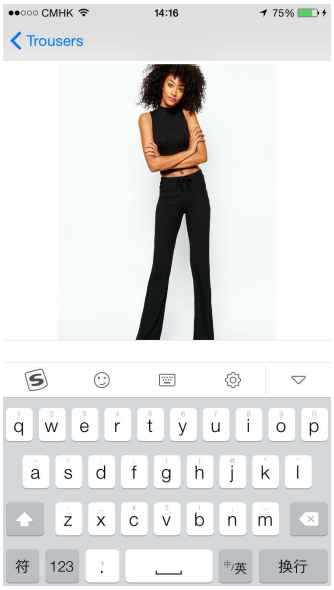




The third tab is the information about the app developers that are us. There the user can find our name, country, picture, phone number and email address. If the user wants to buy some item in our store, she can contact us and we will talk to them.

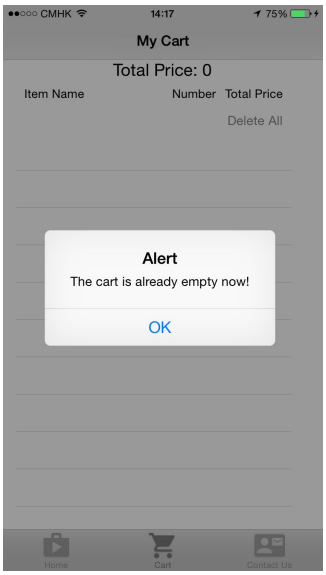


FUNCTIONS ACHIEVED IN APP



The first is the “jump into another page” function. By clicking a certain text or picture in the app, we will go to another page to do further operation, which is the basic function of an app.

The second is the input function. We can input some information by keyboard to do some operations. This is supported in our app.



The third one is the add and delete function. For a shopping app, the user should be able to add things they want and delete things they dislike. To achieve this, we establish a database and make our app connect with it. So whatever the user does, our app will access to database and do operations.

The fourth one is the app icon. We make a different icon of our app to make it look more beautiful.

FUNCTIONS USED OF MOBILE PHONE

The first one is the touch screen. For operations, users should touch the screen to send information to the app.

The second one is the keyboard. For data input, the user should use the keyboard to input the number of items she want.

The third is the database. We use database to store useful information and make changes to it if the user want to do some operations.

HOW DATABASE WORKS

First, we establish a database called “shop.sqlite” using SQLite Manager Firefox. In the database, we create two tables. One is called “items” and the other is called “cart”. We add columns to them and add some records in “items”.

Second, we connect the database with our app by javascript in the “models” folder to create it as a model.

```
config: {
    adapter: {
        type: "sql",
        collection_name: "items",
        db_file:"/shop.sqlite",
        idAttribute: "id"
    }
},
```

After that, in “index.js”, we get the data from database and send it to the collections which we will use in the view.

```
$.index.open();

Alloy.Collections.items.fetch();
Alloy.Collections.cart.fetch();
```

Then in the main view which is “index.xml”, we import them as a collection of data.

```
<Alloy>
    <Collection src="items"/>
    <Collection src="cart" />
```

After importing, we can use the data in the database in the main page.

```
<TextArea editable="false" font="" textAlign="left"   width="150dp" left="20dp" value="{item_name}"   />
<TextArea editable="false" font="" textAlign="center" width="12dp"  left="50dp" value="{item_number}" />
<TextArea editable="false" font="" textAlign="right"  width="50dp"  left="50dp" value="{total_price}" />
<Button title="Delete" textAlign="center" left="85%" onClick="deleteOne" cid="{id}"/>
```

However, to use data in other page is a little bit different.

The first is to import the collection.

```
<Alloy>
    <Collection src="items"/>
    <Window backgroundColor="white">
```

Then we use a data filter to store the collection of data that we want to get. We also need to define the function in the javascript file.

```
 <TableView dataCollection="items" dataFilter="showDetails">

function showDetails (collection) {
    return collection.where({id:args.pid});
}
```

After that, we can use the data in the database in other viewers.

```
<TableViewRow layout="vertical" height="100%">
    <ImageView  image="{images}" height="350dp" width="250dp" top="25dp" />
    <Label  text="{name}"  />
    <Label  text="{price}" />
    <Label  text="{info}"  />
</TableViewRow>
```

DATABASE OPERATIONS

Add one record:

```
var model = Alloy.createModel("cart", {item_id:item.id, item_name: item.name,
    item_price: item.price, item_number: numberToBuy, total_price: item.price * numberToBuy});
Alloy.Collections.cart.add(model);
model.save();
```

Delete one record:

```
var model = cartObject.get(e.source.cid);
model.destroy();
```

Delete all records(Not drop table):

```
var numberOfItem = cartObject.length;
if (numberOfItem > 0) {
    for (var i = 0; i < numberOfItem; i++) {
        var model = cartObject.at(0);
        model.destroy();
    }
} else {
    alert("The cart is already empty now!");
}
```

Get certain value from a certain record:

```
for (var i = 0; i < numberOfItem; i++) {
    var model = cartObject.at(i);
    totalPrice += model.get("total_price");
}
```

CHALLENGES

All challenges in our app are about the database.

The first is how to get connection with the database. As we have mentioned above, there are several steps to make connections. However, we spent many days to work with that and finally solved it .

The second one is how to modify the database in the app. We have connect with the database but it is also difficult to add, delete a record or get some value in the database. The process is very complex and it is not easy to understand very fast. So it also took us some days to make it.

LIMITATIONS

As we have mentioned before, our app is just for a view of items. So users cannot make purchases in our app. This is the first limitation and we think it is a pity not achieving this function.

The second limitation is that we do not make a starting animation because of the time limit. The starting animation now is the default one in the appcelerator studio which is not very beautiful we think.

CONTRIBUTORS

Eline: Propose the idea of app

Create the main interfaces

Create database

Add table “items” and add columns, contents to it

Make connections to the “items” table

Supply all the pictures

Xiaomeng: Make adjustments to the interfaces and add some interfaces

Add table “cart” and add columns to it

Make connections to the “cart” table

Add the “add”,”delete” and “show total price” functions

Add a new icon for the app

Adjust the code style

Testing and debugging