AIET TEAM

Parikshith Adiga	aparik20j98@gmail.com	9148931194
Vivek A Bharadwaj	bharadwaj.a.vivek@gmail.com	8105124234
Supriksha Shetty	suprikshashetty@gmail.com	9967682537
Sheeri Shetty	shettysheeri12@gmail.com	8296551312
Huda	sultana.huda.456@gmail.com	8792951532
Gobind	gobindukumar@gmail.com	9620210114

BANKING WEBSITE

18 June 2020

OVERVIEW

A website which performs operations and manages customers and their accounts.

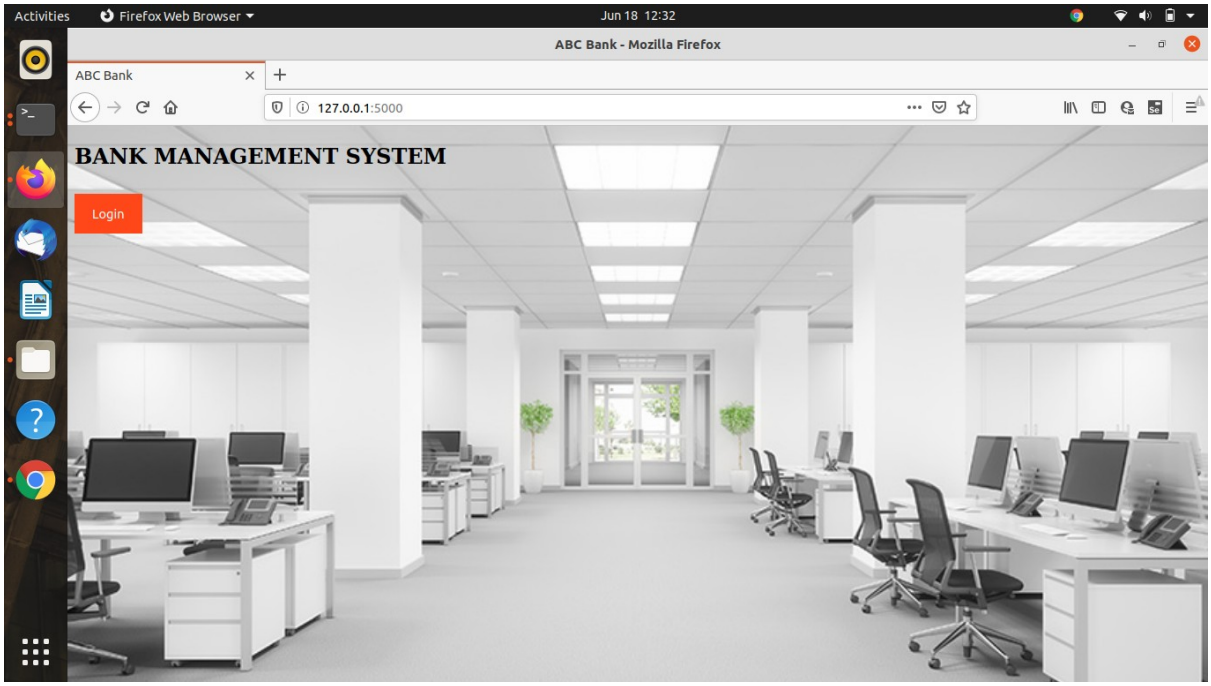
GOALS

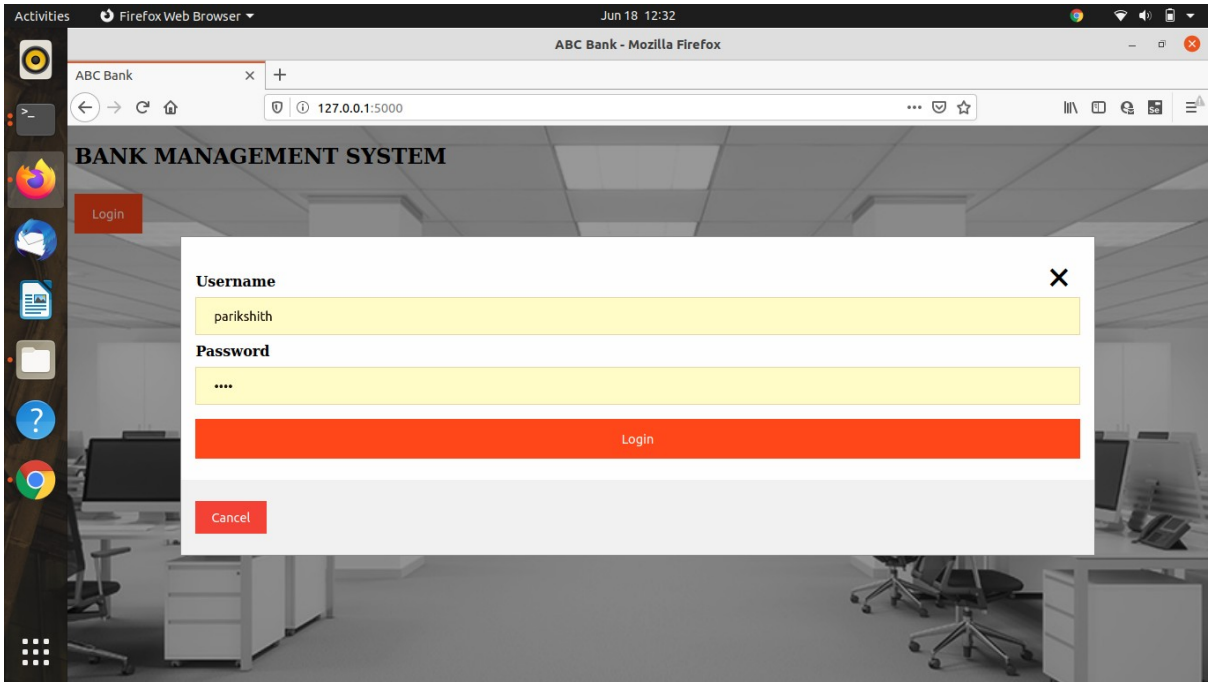
To develop a website which can perform operations like adding, deleting, searching, updating information, transfer money, status on customers and accounts.

FUNCTIONS

LOGIN

The login function is for the bank employee where he/she needs to enter the login credentials like username and password to login to the bank website. If the credentials entered are correct the user is shown a success message and is granted access to the website. If the credentials are wrong, the user is shown an error message.





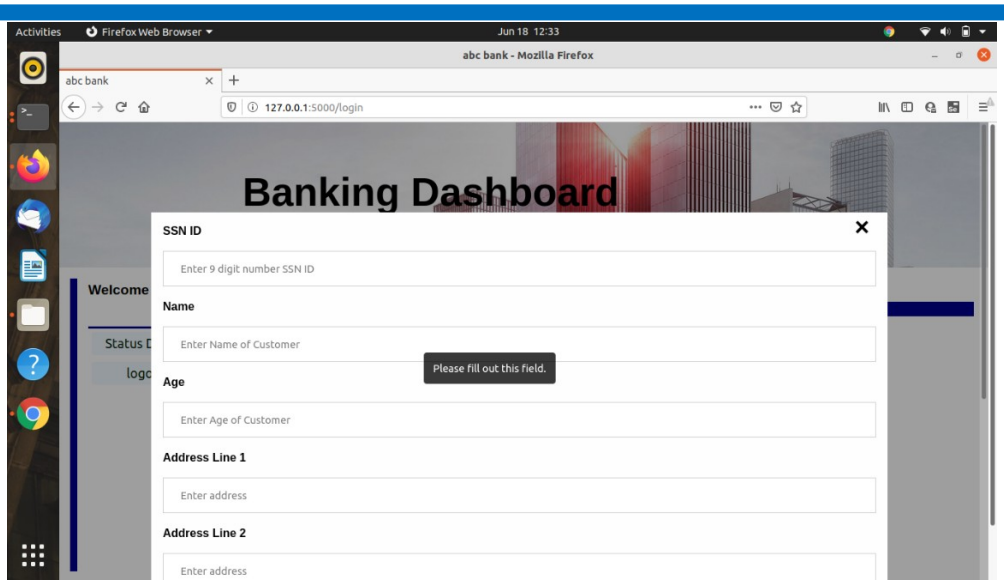
BODY:

```
@app.route('/login',methods=['POST','GET'])
def login():
    error = None
    if request.method == 'POST':
```

```
        username=request.form['uname']
        psw=request.form['psw']
        print(username)
    try:
        session['cid'],session['cssn']=None,None
        sql=("""select SSN,Username,Password from userstore where Username=%s and Password=
%s""",(username,psw))
        conn.execute(*sql)
        ssn,un,psw,=conn.fetchone()
        session['username'],session['psw'],session['SSN']=un,psw,ssn
        print(session['SSN'])
        if request.form['uname'] != un or request.form['psw'] != psw:
            error="Invalid Credentials. Please try again."
            return render_template('index.html',er=error)
        else:
            return
render_template("dash.html",welcome=session.get('username'),val=True,vala=True,vals=True)
    except:
            error="Invalid Credentials. Please try again."
            return render_template('index.html',er=error)
            print("error")
    return render_template('index.html',er=error)
```

CREATE CUSTOMER

Once the employee has logged in, he can create a new customer profile to add a new customer by clicking on ‘Create Customer’. After this is doe, a new customer profile is created and added to the bank’s website.



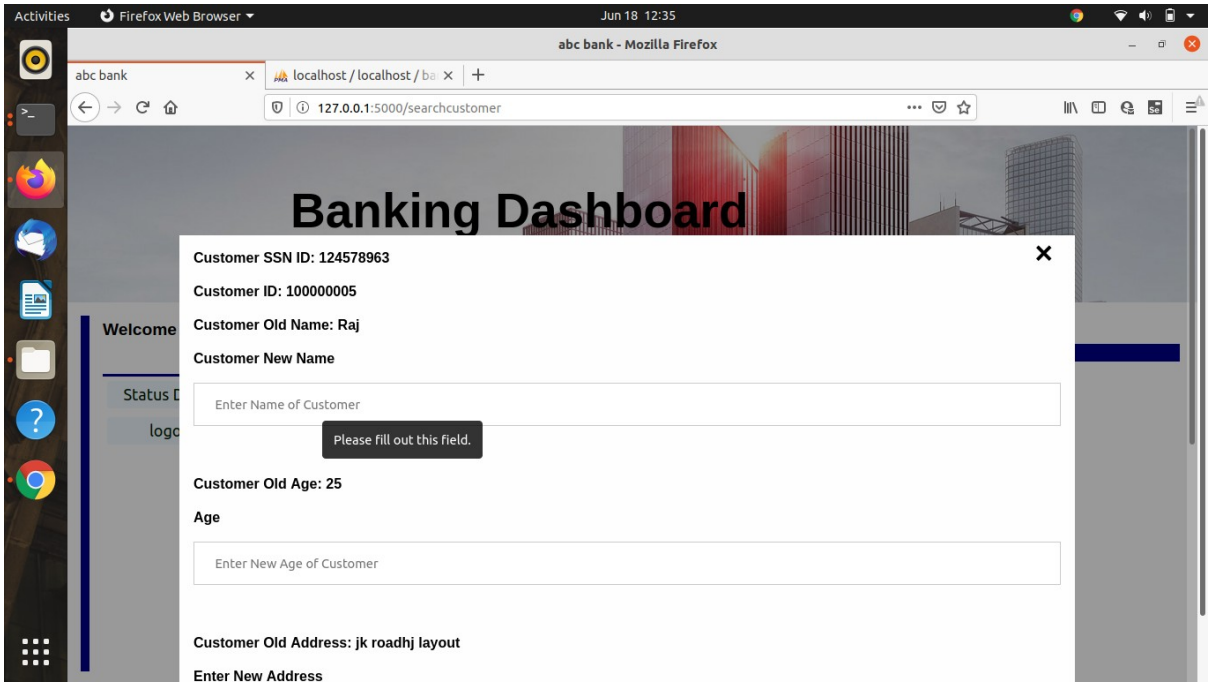
BODY:

```
@app.route('/addncus',methods=['POST','GET'])
def addncus():
    if request.method == 'POST':
        ssnid=request.form['ssn']
        if(len(ssnid)!=9):
            error="plz give 9 digit SSN"
            return
render_template('dash.html',welcome=session.get('username'),er=error,vala=True,vals=True)
        name=request.form['name']
        age=request.form['age']
        addl1=request.form['addressl1']
        addl2=request.form['addressl2']
        state=request.form['state']
        city=request.form['city']
        addl1=addl1+addl2
        global customerid
        customerid+=1
        print(customerid)
        try:
            msg="customer created successfully"
            print(msg)
            tuplev=(ssnid,customerid,name,addl1,state,city,age,msg)
            sql=("""insert into Customer(SSNID,Customerid,Name,Address,State,City,Age,message)
VALUES (%s,%s,%s,%s,%s,%s,%s,%s)""",tuplev)
            conn.execute(*sql)
```

```
        return
render_template("dash.html",welcome=session.get('username'),vala=True,vals=True)
    except:
        error="process coudnt b done"
        return
render_template('dash.html',welcome=session.get('username'),er=error,vala=True,vals=True)
        print("error")
```

UPDATE CUSTOMER

The bank employee can update any information of an already existing customer by clicking on 'Update Customer'. Here the employee can edit information about the customer from any old information to new and updated information of the customer. To update the customer information the employee has to provide the SSN number and customer ID.



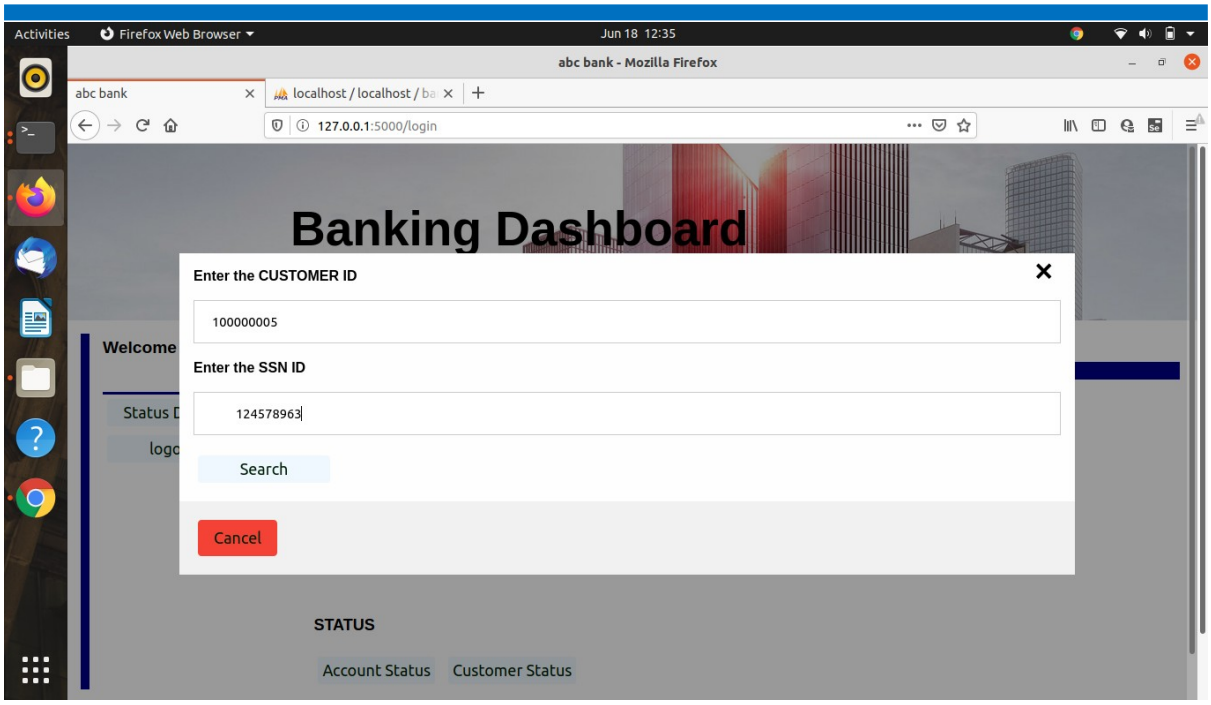
BODY:

```
@app.route('/updatecus',methods=['POST','GET'])
def updatecus():
    if(session.get('cssn')==None):
            error="plz find which customer"
            print(error)
```

```
            return
render_template('dash.html',welcome=session.get('username'),er=error,val=True,vala=True,vals=
True)

    if request.method == 'POST':

        try:

            ssn=session.get('cssn')

            print(ssn)

            sql=("""select SSNID,Customerid,Name,Address,Age from Customer where SSNID=%s""",
(ssn))

            conn.execute(*sql)

            cssid,cusid,coname,coaddress,coage=conn.fetchone()

            name=request.form['name']

            age=request.form['age']

            addl1=request.form['address']

            msg="customer updated succesfully"

            tuplev=(name,addl1,age,msg,ssn)

            sql=("""update Customer set Name=%s,Address=%s,Age=%s,message=%s where SSNID=
%s""",tuplev)

            conn.execute(*sql)

            return
render_template("dash.html",welcome=session.get('username'),vala=True,vals=True,cssid=cssid,
cusid=cusid,cname=cname,cadd=coaddress,cage=coage)

        except:

            error="process coudnt b done"

            return
render_template('dash.html',welcome=session.get('username'),vala=True,vals=True)

            print("error")
```

SEARCH CUSTOMER

The ‘Search customer’ option is used to search the customer with known information so that operations can be performed on the profile. To search the customer profile the employee has to enter the SSN number and customer ID. After searching the employee can perform operations like update and delete.



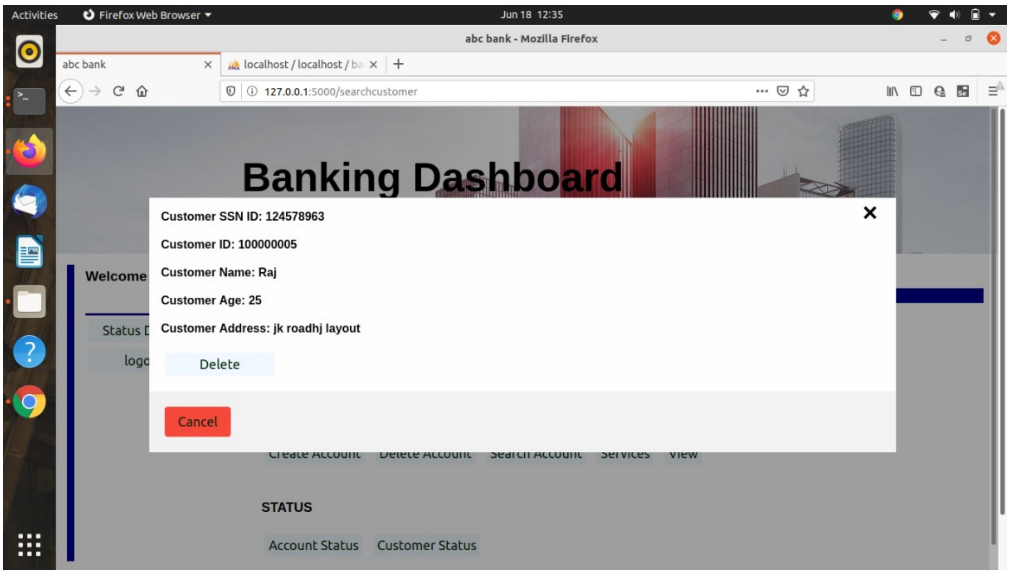
BODY:

```
@app.route('/searchcustomer',methods=['POST','GET'])
def searchcus():
    if request.method == 'POST':
        try:
            ssn=request.form['ssnid']
            cid=request.form['cid']
            sql=("""select SSNID,Customerid from Customer where Customerid=%s and SSNID=%s""",(cid,ssn))
            try:
                conn.execute(*sql)
                cssn,caid=conn.fetchone()
                if(str(caid)!=cid and str(cssn)!=ssn):
                    error="process coudnt b done due incorrect values"
                    return render_template('dash.html',welcome=session.get('username'),vala=True,vals=True,er=error)
            except:
                error="process coudnt b done due incorrect values"
                print(error)
                return render_template('dash.html',welcome=session.get('username'),vala=True,vals=True,er=error)
            session['cid'],session['cssn']=cid,ssn
            sql=("""select SSNID,Customerid,Name,Address,Age from Customer where SSNID=%s""",(ssn))
```

```
            conn.execute(*sql)

            cssid,cusid,coname,coaddress,coage=conn.fetchone()

            print(coage)

            return
render_template("dash.html",welcome=session.get('username'),vala=True,vals=True,cssid=cssid,
cusid=cusid,cname=coname,cadd=coaddress,cage=coage)

        except:

            error="process coudnt b done"

            return
render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True,er=
error)

            print("error")
```

DELETE CUSTOMER

The ‘Delete Customer’ option is used to delete a customer’s profile from the bank website if a customer wishes to close his profile.



BODY:

```
@app.route('/deletecustomer',methods=['POST','GET'])

def deletecustomer():

    if(session.get('cssn')==None):

        error="plz find which customer"

        print(error)

        return
render_template('dash.html',welcome=session.get('username'),er=error,val=True,vala=True,vals=
True)

    print((session.get('cssn')))
```

```
    sql=("""select SSNID,Customerid,Name,Address,Age from Customer where SSNID=%s""",
(session.get('cssn')))

    conn.execute(*sql)

    print((session.get('cssn')))

    cssid,cusid,cname,cadd,cage=conn.fetchone()

    sql=("""delete from Customer where SSNID=%s and Customerid=%s""",(cssid,cusid))

    conn.execute(*sql)

    session.pop('cssn', None)

    session.pop('cid', None)

    return
render_template("dash.html",vala=True,vals=True,welcome=session.get('username'),cssid=cssid,
cusid=cusid,cname=cname,cadd=cadd,cage=cage)
```

LOGOUT

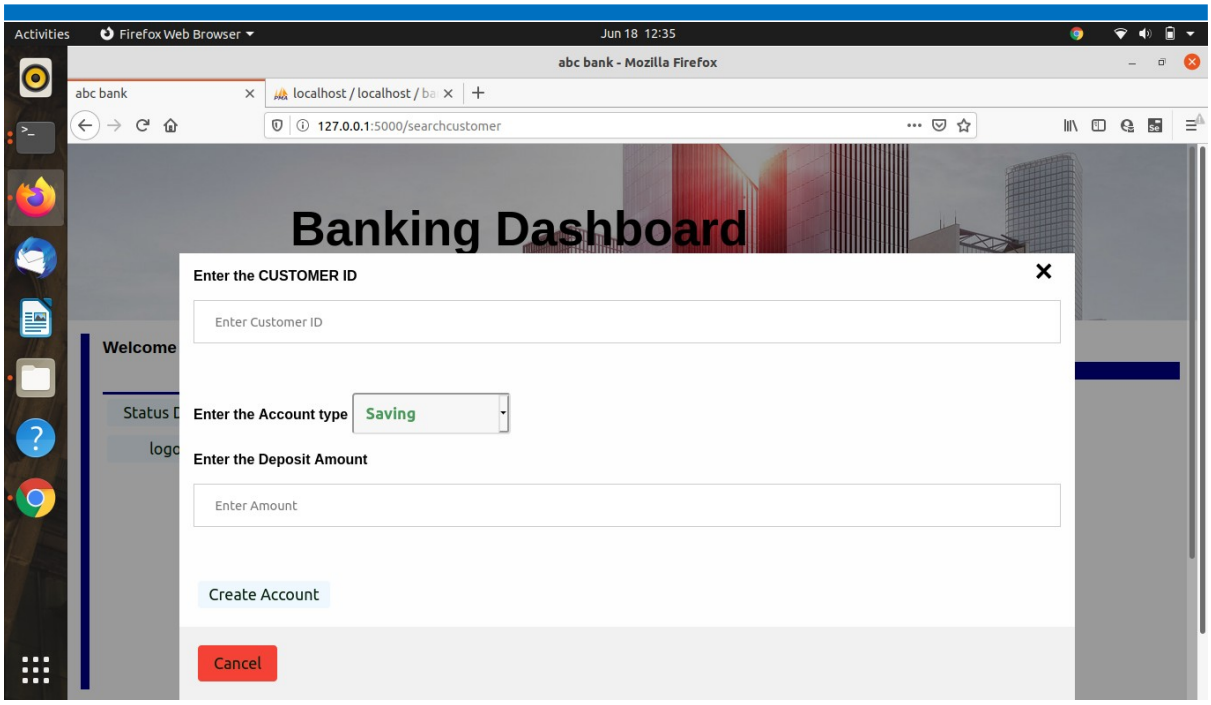
The employee can logout of the bank's website by clicking on 'Logout'.

BODY:

```
@app.route('/logout',methods=['POST','GET'])

def logout():

    if request.method == 'POST':

        if 'lout' in request.form:

                session.pop('username', None)


                session.pop('passwrd', None)

                session.pop('SSN', None)

                return render_template("index.html")
```

ADD ACCOUNT

The bank employee can create an account by entering credentials of the customer if the customer wishes to add a new account in order to preform monetary transactions from the bank's website.



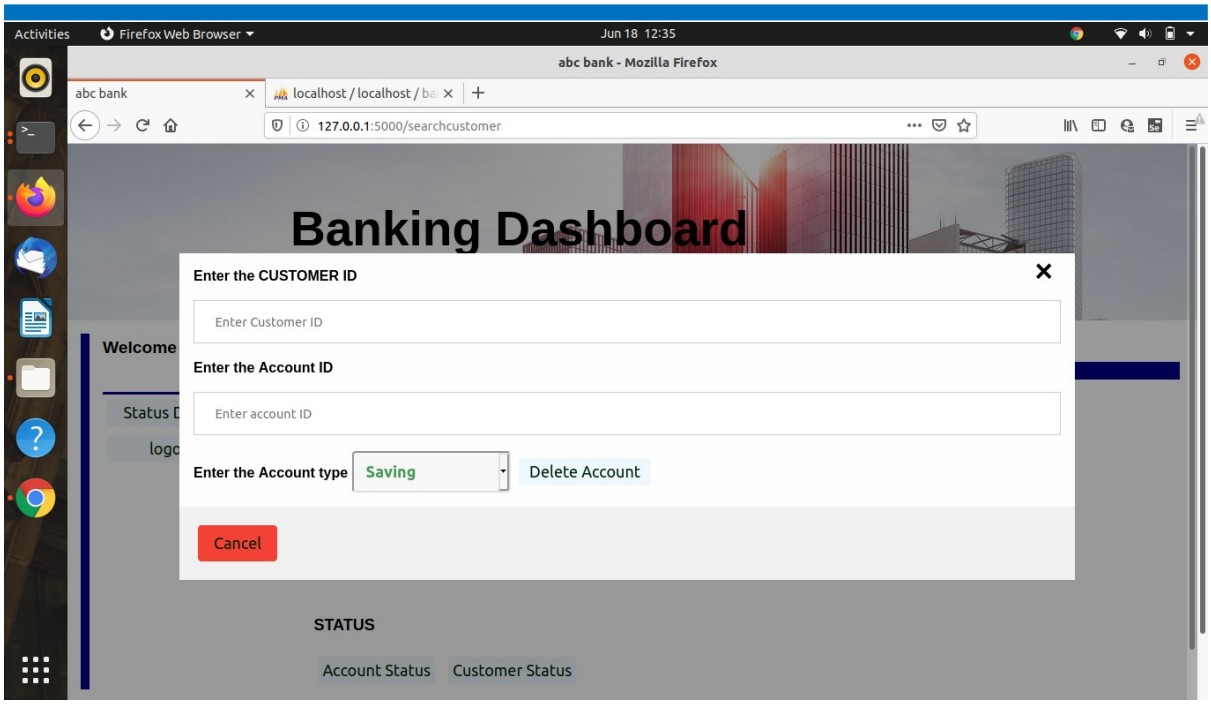
BODY:

```
@app.route('/addaccount',methods=['POST','GET'])
def addaccount():
    if request.method == 'POST':
        cid=request.form['cid']
        acctype=request.form['acctype']
        amt=request.form['amount']
        try:
            sql=("""select Customerid from Customer where Customerid=%s""",(cid))
            conn.execute(*sql)
            caid=conn.fetchone()
            if(str(caid[0])!=cid):
                error="process coudnt b done due incorrect values"
                print(error)
                return
render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True,er=
error)
        except:
            error="process coudnt b done due incorrect values"
            print(error)
            return
render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True,er=
error)
        try:
```

```
            global accid
            accid+=1
            print(accid)
            msg="account created succesfully"
            tuplev=(accid,cid,amt,acctype,msg)
            print(tuplev)
            sql=("""insert into Account(Accountid,cusid,Balance,accounttype,msg) VALUES (%s,%s,%s,
%s,%s)""",tuplev)
            print(conn.execute(*sql))
            tuplev=(accid,cid,amt,msg)
            sql=("""insert into personalaccount(aid,cid,abalance,msg)VALUES(%s,%s,%s,%s)""",tuplev)
            conn.execute(*sql)
            return render_template('dash.html',welcome=session.get('username'))
        except:
            error="process coudnt b done"
            return
render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True)
            print("error")
```

DELETE ACCOUNT

The bank employee can delete a customer’s account from the bank website if the customer wishes to end his monetary transaction on the platform by clicking on ‘Delete Account’.



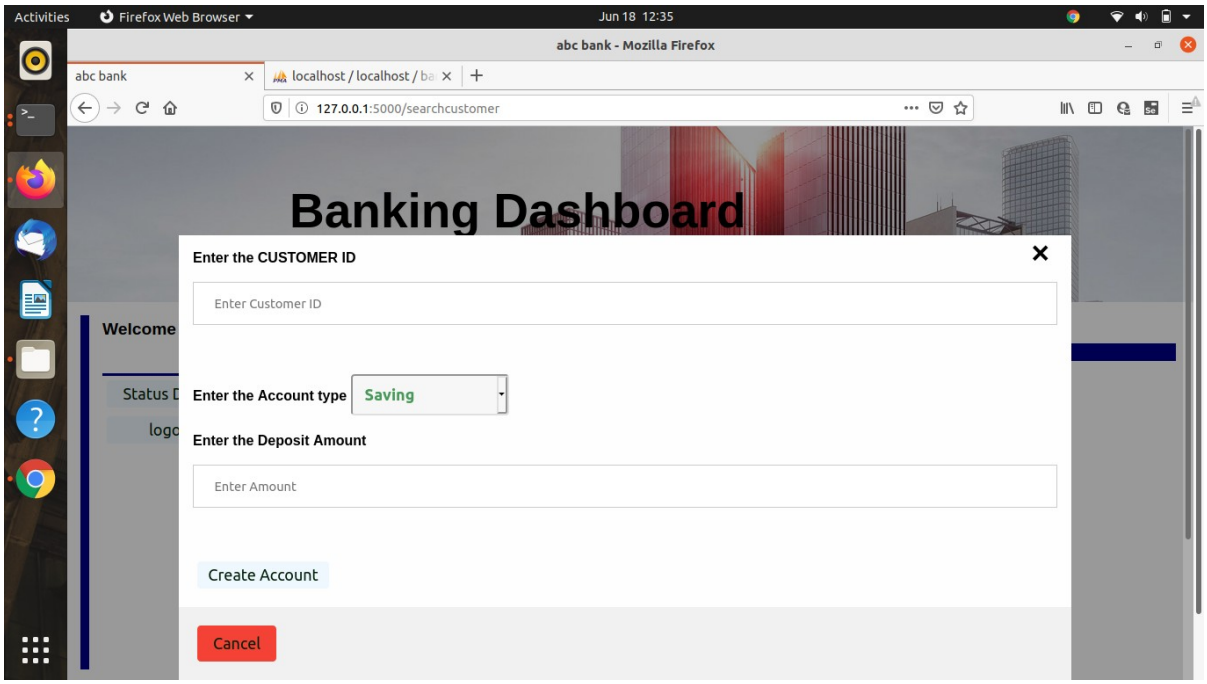
BODY:

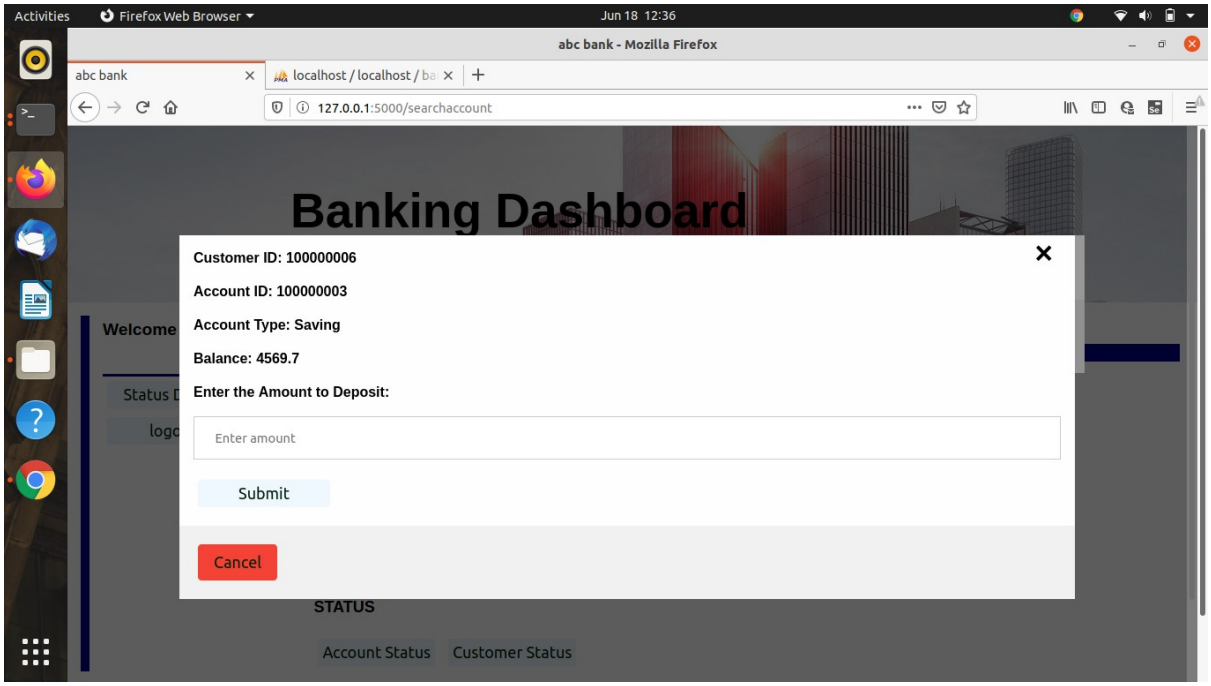
```
@app.route('/deleteaccount',methods=['POST','GET'])
def deleteaccount():
    if request.method == 'POST':
        cid=request.form['cid']
        acctype=request.form['acctype']
        aid=request.form['aid']
        try:
            sql=("""select Accountid,cusid from Account where cusid=%s and Accountid=%s""",(cid,aid))
            conn.execute(*sql)
            caid,fcid=conn.fetchone()
            print(caid)
        except:
            error="process coudnt b done due incorrect values"
            print(error)
            return render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True,er=error)
        try:
            tuplev=(aid,cid)
            sql=("""delete from Account where Accountid=%s and cusid=%s""",tuplev)
            conn.execute(*sql)
            amt="0"
```

```
            msg="account deleted"
            tuplev=(aid,cid,amt,msg)
            print(tuplev)
            sql=("""insert into personalaccount(aid,cid,abalance,msg)VALUES(%s,%s,%s,%s)""",tuplev)
            conn.execute(*sql)
            return render_template('dash.html',welcome=session.get('username'))
        except:
            error="process coudnt b done"
            return
render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True)
            print("error")
```

DEPOSIT MONEY

The bank employee can deposit money to a customer's account from the bank website. In order to do this, the employee has to provide the account ID provided to the customer.





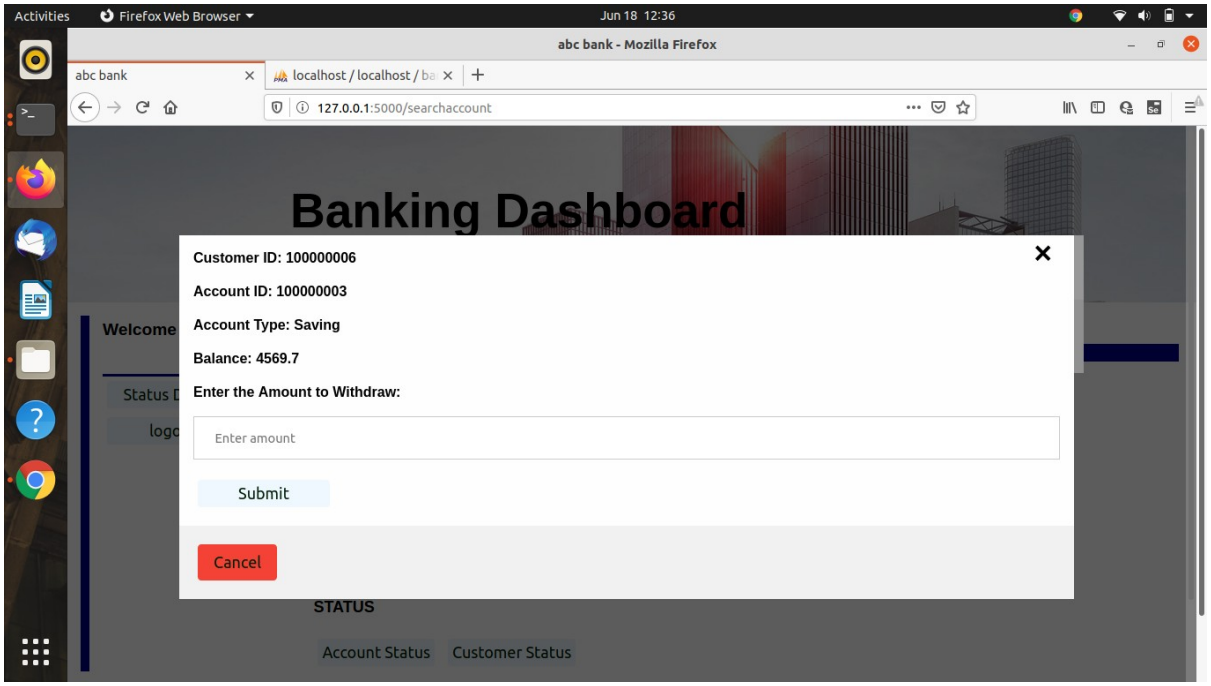
BODY:

```
@app.route('/depositmoney',methods=['POST','GET'])
def depositmoney():
    if request.method == 'POST':
        amt=request.form['amt']
        try:
            sql=("""select Accountid,cusid,balance,accounttype from Account where Accountid=%s""",(session.get('aid')))
            conn.execute(*sql)
            aid,cusid,bal,at=conn.fetchone()
            bal=float(amt)+float(bal)
            msg="amount is debited"
            tuplev=(bal,msg,(session.get('aid')))
            sql=("""update Account set balance=%s,msg=%s where Accountid=%s""",tuplev)
            conn.execute(*sql)
            tuplev=((session.get('aid')),cusid,bal,msg)
            sql=("""insert into personalaccount(aid,cid,abalance,msg)VALUES(%s,%s,%s,%s)""",tuplev)
            conn.execute(*sql)
            sql=("""select Accountid,cusid,balance,accounttype from Account where Accountid=%s""",(session.get('aid')))
            conn.execute(*sql)
```

```
        aid,cusid,bal,at=conn.fetchone()

        print(aid)

        return
render_template('dash.html',welcome=session.get('username'),val=True,vals=True,aid=aid,cusid
=cusid,bal=bal,at=at)

    except:

        error="process coudnt b done"

        return
render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True,er=
error)

        print("error")
```

WITHDRAW MONEY

The bank employee can withdraw money for the customer by providing the account ID provided for the customer.



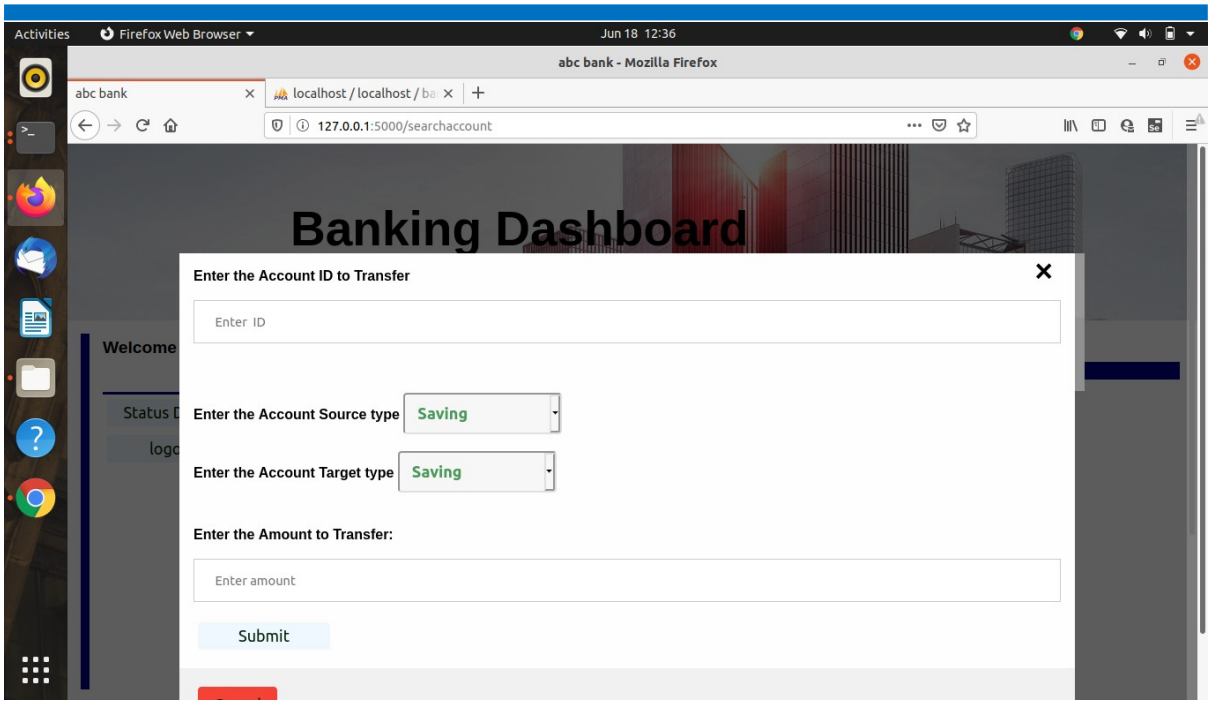
BODY:

```
@app.route('/withdrawmoney',methods=['POST','GET'])

def withdrawmoney():

    if request.method == 'POST':

        amt=request.form['amt']

        try:

            sql=("""select Accountid,cusid,balance,accounttype from Account where Accountid=%s""",
(session.get('aid')))
```

```
            conn.execute(*sql)

            aid,cusid,bal,at=conn.fetchone()

            bal=float(bal)-float(amt)

            msg="amount is credited"

            tuplev=(bal,msg,(session.get('aid')))

            sql=("""update Account set balance=%s,msg=%s where Accountid=%s""",tuplev)

            conn.execute(*sql)

            tuplev=((session.get('aid')),cusid,bal,msg)

            sql=("""insert into personalaccount(aid,cid,abalance,msg)VALUES(%s,%s,%s,%s)""",tuplev)

            conn.execute(*sql)

            sql=("""select Accountid,cusid,balance,accounttype from Account where Accountid=%s""",
(session.get('aid')))

            conn.execute(*sql)

            aid,cusid,bal,at=conn.fetchone()

            print(aid)

            return
render_template('dash.html',welcome=session.get('username'),val=True,vals=True,aid=aid,cusid
=cusid,bal=bal,at=at)

        except:

            error="process coudnt b done"

            return
render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True)

            print("error")
```

TRANSFER MONEY

The bank employee can transfer money from one account to another by entering both the sender account ID and the receiver’s account ID.



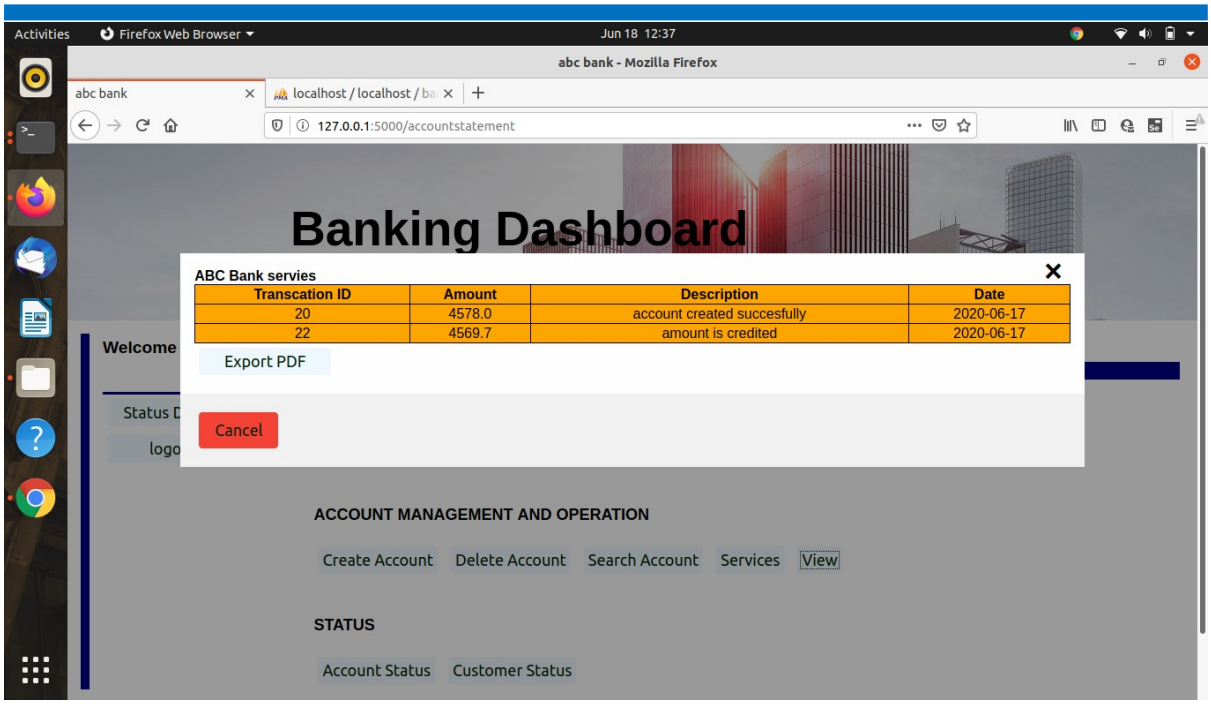
BODY:

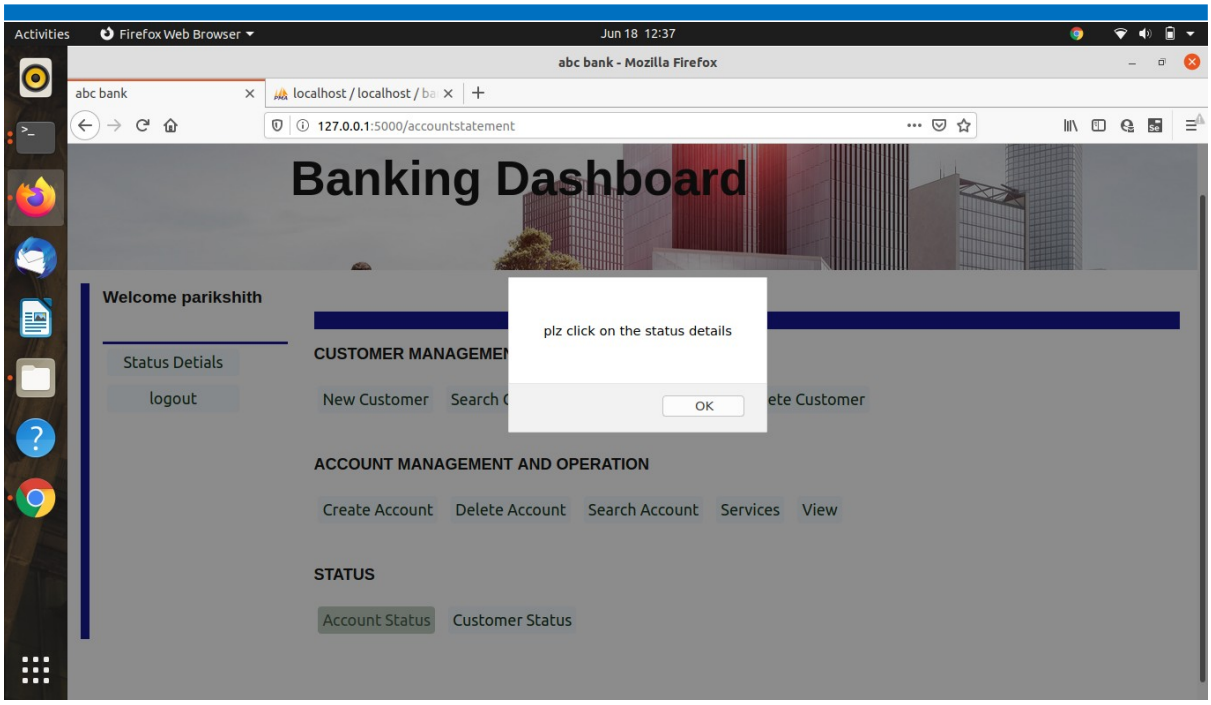
```
@app.route('/transfermoney',methods=['POST','GET'])
def transfermoney():
    if request.method == 'POST':
        aidt=request.form['cid']
        amt=request.form['amt']
        try:
            sql=("""select Accountid,cusid,balance,accounttype from Account where Accountid=%s""",(session.get('aid')))
            conn.execute(*sql)
            aid,cusid,bal,at=conn.fetchone()
            print(bal)
            bal=float(bal)-float(amt)
            print(bal)
            msg="amount is credited"
            tuplev=(bal,msg,(session.get('aid')))
            sql=("""update Account set balance=%s,msg=%s where Accountid=%s""",tuplev)
            conn.execute(*sql)
            tuplev=((session.get('aid')),cusid,bal,msg)
            sql=("""insert into personalaccount(aid,cid,abalance,msg)VALUES(%s,%s,%s,%s)""",tuplev)
            conn.execute(*sql)
```

```
            sql=("""select Accountid,cusid,balance,accounttype from Account where Accountid=
%s""",aidt)

            conn.execute(*sql)

            aid,cusid,bal,at=conn.fetchone()

            print(bal)

            bal=float(bal)+float(amt)

            msg="amount is debited"

            print(aidt)

            tuplev=(bal,msg,aidt)

            sql=("""update Account set balance=%s,msg=%s where Accountid=%s""",tuplev)

            conn.execute(*sql)

            print(msg,aidt)

            tuplev=((session.get('aid')),cusid,bal,msg)

            sql=("""insert into personalaccount(aid,cid,abalance,msg)VALUES(%s,%s,%s,%s)""",tuplev)

            conn.execute(*sql)

            sql=("""select Accountid,cusid,balance,accounttype from Account where Accountid=%s""",
(session.get('aid')))

            conn.execute(*sql)

            aid,cusid,bal,at=conn.fetchone()

            print(aid)

            return
render_template('dash.html',welcome=session.get('username'),val=True,vals=True,aid=aid,cusid
=cusid,bal=bal,at=at)

        except:

            error="process coudnt b done"

            return
render_template('dash.html',welcome=session.get('username'),val=True,vala=True,vals=True)

            print("error")
```

ACCOUNT STATEMENT

The bank employee can provide the account statement of the customer’s account activity by clicking on ‘view’. All the transactions of the customer’s account can be viewed here.


Activities
Firefox Web Browser
Jun 18 12:37
abc bank - Mozilla Firefox
abc bank
localhost / localhost / ba
127.0.0.1:5000/accountstatement
Banking Dashboard
ABC Bank servies
Transcation ID
Amount
Description
Date
20
4578.0
account created succesfully
2020-06-17
22
4569.7
amount is credited
2020-06-17
Export PDF
Cancel
Welcome
ACCOUNT MANAGEMENT AND OPERATION
Create Account
Delete Account
Search Account
Services
View
STATUS
Account Status
Customer Status

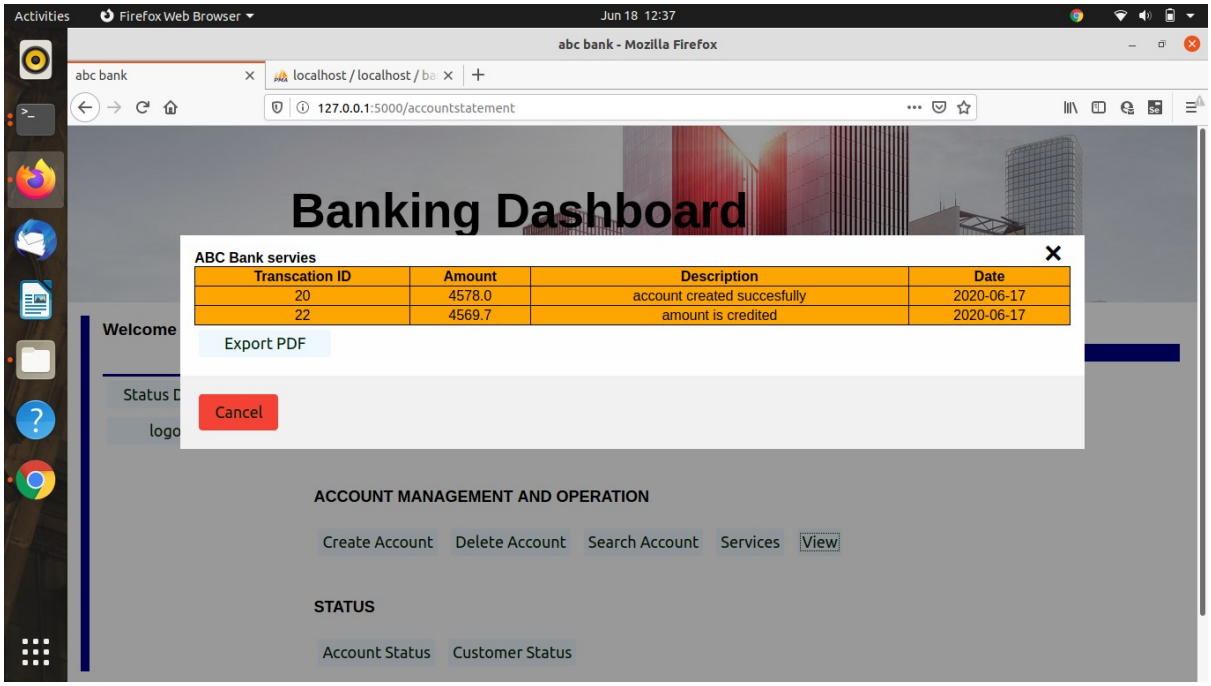


BODY:

```
@app.route('/accountstatement',methods=['POST','GET'])
def accountstatement():
    if request.method == 'POST':
        sdate=request.form['startdate']
        edate=request.form['lastdate']
        print(sdate)
        print(edate)
        try:
            tuplel=((session.get('aid')),sdate,edate)
            sql=("""select transid,abalance,msg,datea from personalaccount where aid=%s and datea
BETWEEN %s and %s""",tuplel)
            conn.execute(*sql)
            data=conn.fetchall()
            return
render_template('dash.html',welcome=session.get('username'),val=True,data=data,vals=True)
        except:
            error="process coudnt b done"
            return
render_template('dash.html',welcome=session.get('username'),val=True,data="none",er=error,val
a=True,vals=True)
            print("error")
```

ACCOUNT STATEMENT WITH PDF DOWNLOAD

The bank employee can also provide the customer with a PDF document of the account statement. This document includes all the transactions done to the customer's account and can be downloaded with the 'view' option.



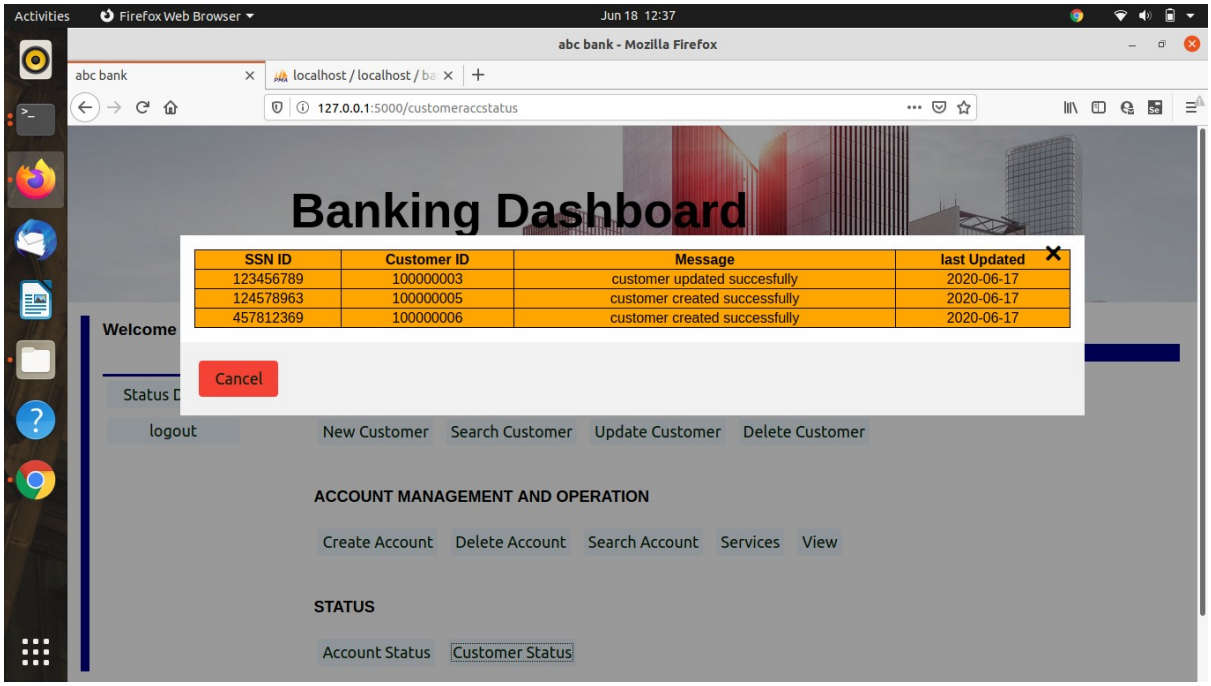
BODY:

```
$('#cmd').click(function () {
      var table = tableToJson($('#htmlexportPDF').get(0))
      var doc = new jsPDF('p','pt', 'a3', true);
      doc.cellInitialize();
      $.each(table, function (i, row){

          console.debug(row);
         $.each(row, function (j, cell){
            doc.cell(10, 50,200, 50, cell, i);  // 2nd parameter=top margin,1st=left margin 3rd=row
cell width 4th=Row height
         })
      })
      doc.save('sample-file.pdf');
});
   function tableToJson(table) {
   var data = [];
   var headers = [];
   for (var i=0; i<table.rows[0].cells.length; i++) {
```

```
        headers[i] = table.rows[0].cells[i].innerHTML.toLowerCase().replace(/ /gi,'');
    }
    for (var i=0; i<table.rows.length; i++) {
        var tableRow = table.rows[i];
        var rowData = {};
        for (var j=0; j<tableRow.cells.length; j++) {
            rowData[ headers[j] ] = tableRow.cells[j].innerHTML;
        }
        data.push(rowData);
    }
    return data;
}
});
```

CUSTOMER ACCOUNT STATUS

The bank employee can provide the details about the status of a customer account to the customer if asked for. The status of the account can be checked by clicking on the 'Status' button.



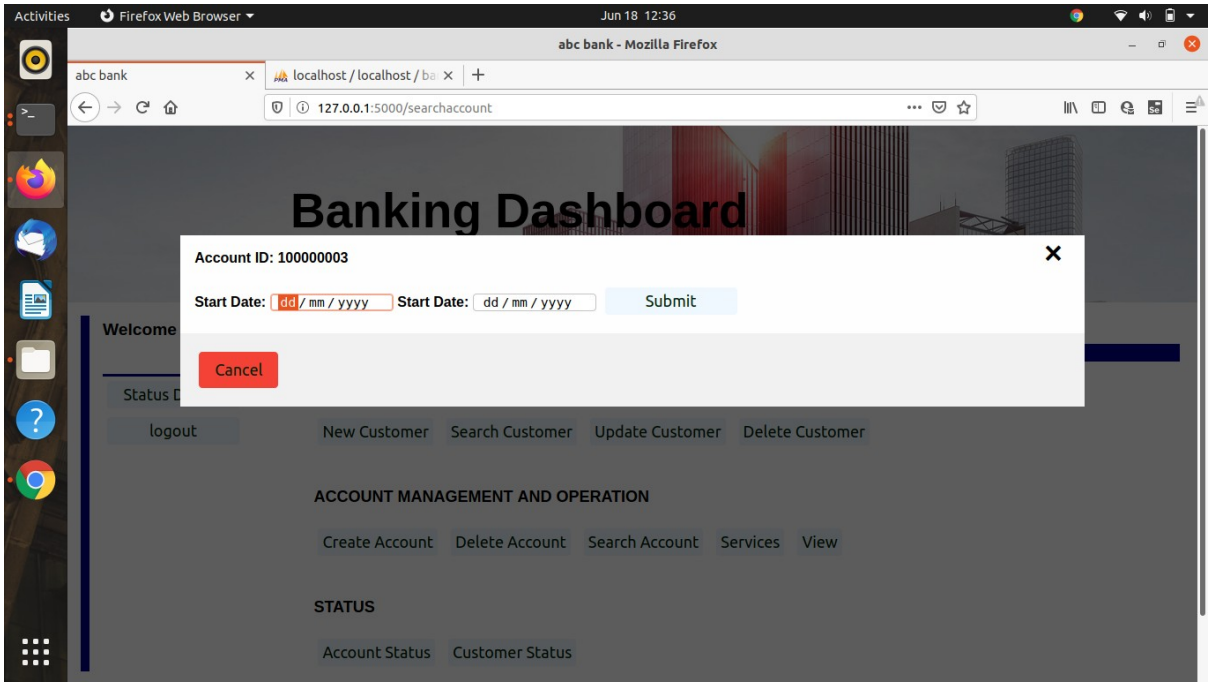
BODY:

```
@app.route('/customeraccstatus',methods=['POST','GET'])
def customerstatus():
    try:
        sql=("""select SSNID,Customerid,message,datec from Customer""")
```

```
        conn.execute(sql)

        cdata=conn.fetchall()

        sql=("""select Accountid,cusid,balance,Accounttype,msg,cdate from Account""")

        conn.execute(sql)

        adata=conn.fetchall()

        return
render_template('dash.html',welcome=session.get('username'),val=True,cdata=cdata,adata=adat
a,vala=True)

    except:

        error="process coudnt b done"

        return
render_template('dash.html',welcome=session.get('username'),val=True,data="none",er=error,val
a=True,vals=True)

        print("error")
```

SEARCH ACCOUNT

The bank employee can also search for a customer's account that is present in the website by clicking on the 'Search Account' option and providing the account ID provided to the customer.



BODY:

```
@app.route('/searchaccount',methods=['POST','GET'])

def searchaccount():

    if request.method == 'POST':
```

```
        aid=request.form['aid']
        try:
            sql=("""select Accountid,cusid from Account where Accountid=%s """,(aid))
            try:
                conn.execute(*sql)
                caid,cssn=conn.fetchone()
                if str(caid)!=aid:
                    error="process coudnt b done due incorrect values"
                    print(error)
                    return
render_template('dash.html',welcome=session.get('username'),val=True,er=error,vala=True,vals=
True)
            except:
                error="process coudnt b done due incorrect value"
                print(error)
                return
render_template('dash.html',welcome=session.get('username'),val=True,er=error,vala=True,vals=
True)
            session['aid']=aid
            print(session.get('aid'))
            sql=("""select Accountid,cusid,balance,accounttype from Account where Accountid=%s""",
(aid))
            conn.execute(*sql)
            aid,cusid,bal,at=conn.fetchone()
            return
render_template('dash.html',vals=True,welcome=session.get('username'),val=True,aid=aid,cid=c
usid,bal=bal,at=at)
        except:
            error="process coudnt b done"
            return
render_template('dash.html',welcome=session.get('username'),val=True,er=error,vala=True,vals=
True)
            print("error")
```