References

Aleskerov, E., Freisleben, B., Rao, B., 1997. CARDWATCH: a neural network based database mining system for credit card fraud detection, in: Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr). Presented at the Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr), pp. 220–226. doi:10.1109/CIFER.1997.618940

Bhatla, T.P., Prabhu, V., Dua, A., 2003. Understanding Credit Card Frauds. Cards Bus. Rev. 1.

Bhusari, V., Patil, S., 2011. Study of Hidden Markov Model in Credit Card Fraudulent Detection. ResearchGate 20. doi:10.5120/2428-3263

Brause, R., Langsdorf, T., Hepp, M., 1999. Neural Data Mining for Credit Card Fraud Detection, in: ResearchGate. Presented at the Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on, pp. 103–106. doi:10.1109/TAI.1999.809773

Chang, R.-I., Lai, L.-B., Su, W.-D., Wang, J.-C., Kouh, J.-S., 2007. Intrusion Detection by Backpropagation Neural Networks with Sample-Query and Attribute-Query. ResearchGate 3, 6–10. doi:10.5019/j.ijcir.2007.76

Chiu, C.-C., Tsai, C.-Y., 2004. A Web services-based collaborative scheme for credit card fraud detection, in: 2004 IEEE International Conference on E-Technology, E-Commerce and E-Service, 2004. EEE '04. Presented at the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service, 2004. EEE '04, pp. 177–181. doi:10.1109/EEE.2004.1287306

Duman, E., Ozcelik, M.H., 2011. Detecting credit card fraud by genetic algorithm and scatter search. Expert Syst. Appl. 38, 13057–13063. doi:10.1016/j.eswa.2011.04.110

Guo, T., Li, G.-Y., 2008. Neural data mining for credit card fraud detection, in: 2008 International Conference on Machine Learning and Cybernetics. Presented at the 2008 International Conference on Machine Learning and Cybernetics, pp. 3630–3634. doi:10.1109/ICMLC.2008.4621035

Holmes, T.E., 2016. Credit card rewards and cardholder satisfaction statistics [WWW Document]. URL http://www.creditcards.com/credit-card-news/cardholder-satisfaction-rewards-security-statistics-1276.php (accessed 8.14.16).

Holms, T., n.d. Credit card fraud and ID theft statistics [WWW Document]. www.CreditCards.com. URL http://www.creditcards.com/credit-card-news/credit-card-security-id-theft-fraud-statistics-1276.php (accessed 6.27.16).

ISO 8583, 2016. . Wikipedia.

Jason Steele, 2011. Why Most Americans Should Not Use Credit Cards. MoneyCrashers.com.

Kotelawala, H., 2016. Credit Card Usage in Sri Lanka: A Breakdown. Roar.lk.

Kundu, A., Panigrahi, S., Sural, S., Majumdar, A.K., 2009. BLAST-SSAHA Hybridization for Credit Card Fraud Detection. IEEE Trans. Dependable Secure Comput. 6, 309–315. doi:10.1109/TDSC.2009.11

MD5, 2017. . Wikipedia.

Meshram, P.L., Yenganti, T., 2013. Credit and ATM Card Fraud Prevention Using Multiple Cryptographic Algorithm. Int. J. Adv. Res. Comput. Sci. Softw. Eng. 3, 1300–1305.

Patidar, R., Sharma, L., 2011. Credit Card Fraud Detection Using Neural Network. Int. J. Soft Comput. Eng. IJSCE 1, 32–38.

Raj, S.B.E., Portia, A.A., 2011. Analysis on credit card fraud detection methods, in: 2011 International Conference on Computer, Communication and Electrical Technology (ICCCET). Presented at the 2011 International Conference on Computer, Communication and Electrical Technology (ICCCET), pp. 152–156. doi:10.1109/ICCCET.2011.5762457

Sahin, Y., Duman, E., 2011. Detecting Credit Card Fraud by Decision Trees and Support Vector Machines. ResearchGate 1, 442–447.

SHA-1, 2017. . Wikipedia.

Srivastava, A., Kundu, A., Sural, S., Majumdar, A., 2008. Credit Card Fraud Detection Using Hidden Markov Model. IEEE Trans. Dependable Secure Comput. 5, 37–48. doi:10.1109/TDSC.2007.70228

Stolfo, S.J., Fan, D.W., Lee, W., Prodromidis, A.L., Chan, P.K., 1998. Credit Card Fraud Detection Using Meta-Learning: Issues and Initial Results. ResearchGate.

Syeda, M., Zhang, Y.-Q., Pan, Y., 2002. Parallel granular neural networks for fast credit card fraud detection, in: Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, 2002. FUZZ-IEEE'02. Presented at the Proceedings of the 2002

IEEE International Conference on Fuzzy Systems, 2002. FUZZ-IEEE'02, pp. 572–577. doi:10.1109/FUZZ.2002.1005055

Vadoodparast, M., Hamdan, A.R., Hafiz, 2015. Fraudulent Electronic Transaction Detection Using Dynamic KDA Model. Int. J. Comput. Sci. Inf. Secur. 12.

Vatsa, V., Sural, S., Majumdar, A.K., 2005. A Game-theoretic Approach to Credit Card Fraud Detection, in: Proceedings of the First International Conference on Information Systems Security, ICISS'05. Springer-Verlag, Berlin, Heidelberg, pp. 263–276. doi:10.1007/11593980_20

Appendix A – ISO8583 Message Sample

Sample of Online Payment

Sample Request Message

0200FABC44C12880C0000000000004000000164123456789147258200000000010000000000000000121401262761000000660544012746121419096011051000406914442324691760100091433 =19092211645932100505348627 36EPAYSWITCH ONLN LOCATION COLOMBO LKA14414416412345678914725 8

Field	Value
MTI	0200
[001] Bitmap (hex 62)	FABC44C12820C0000000000004000000
[002] Primary Account Number (n..16)	16 4123456789147258
[003] Processing Code (n6)	200000
[004] Amount, Transaction (n12)	000000100000
[005] Amount, Settlement (n12)	000000000000
[007] Trans. Date/Time (n10)	1214012627
[009] Conversion rate (n8)	10000006
[011] System trace audit number (n6)	605440
[012] Time (n6)	012746 (HHmmss)
[013] Date (n4)	1214 (MMdd)
[014] Date, expiration (n4)	1909
[018] Merchant type (n4)	6011
[022] Point of service entry mode (n3)	051
[025] Point of service condition code (n2)	00
[026] Point of service capture code (n2)	04
[032] Acquiring institution identification code (n..6)	06 914442
[035] Track 2 data (n..32)	32 4691760100091433 =19092211645932
[037] Retrieval reference number (an12)	100505348627

[043] Card acceptor terminal identification (ans..32)	36 EPAYSWITCH ONLN LOCATION COLOMBO LKA
[049] Currency code, transaction (n3)	144
[050] Currency code, settlement (n3)	144
[103] Account identification 1 (n..16)	16 4123456789147258

Sample Response Message

0200FABC44C12880C0000000000004000000164123456789147258200000000010000000000000000012140126276100000066054401274612141909601105100040691441432469176010009143 3 =19092211645932100505348627**00**36EPAYSWITCH ONLN LOCATION COLOMBO LKA144144

Field	**Value**
MTI	0210
[001] Bitmap (hex 62)	FABC44C12A20C0000000000000000000
[002] Primary Account Number (n..16)	16 4123456789147258
[003] Processing Code (n6)	200000
[004] Amount, Transaction (n12)	000000100000
[005] Amount, Settlement (n12)	000000000000
[007] Trans. Date/Time (n10)	1214012627
[009] Conversion rate (n8)	10000006
[011] System trace audit number (n6)	605440
[012] Time (n6)	012746 (HHmmss)
[013] Date (n4)	1214 (MMdd)
[014] Date, expiration (n4)	1909
[018] Merchant type (n4)	6011
[022] Point of service entry mode (n3)	051
[025] Point of service condition code (n2)	00
[026] Point of service capture code (n2)	04
[032] Acquiring institution identification code (n..6)	06 914442
[035] Track 2 data (n..32)	32 4691760100091433 =19092211645932
[037] Retrieval reference number (an12)	100505348627
[039] Response Code (an2)	Success: 00
	Declined: 07
	Not enough funds: 51
	Suspicious: 59
[043] Card acceptor terminal identification	36 EPAYSWITCH ONLN LOCATION

(n..32)	COLOMBO LKA
[049] Currency code, transaction (n3)	144
[050] Currency code, settlement (n3)	144

Appendix B – Important Code Segments

API Service Implementation

Customers.java

```
public class Customers implements Serializable {

    private static final long serialVersionUID = 1L;
    private Long customerId;
    private String fullName;
    private String nid;
    private String mobile;
    private short isActive;
    private String username;
    private String password;
    private String respo;

    public Customers() {
    }
    public Long getCustomerId() {
        return customerId;
    }
    public void setCustomerId(Long customerId) {
        this.customerId = customerId;
    }
    public String getFullName() {
        return fullName;
    }
    public void setFullName(String fullName) {
        this.fullName = fullName;
    }
    public String getNid() {
        return nid;
    }
    public void setNid(String nid) {
        this.nid = nid;
    }
    public String getMobile() {
        return mobile;
    }
    public void setMobile(String mobile) {
        this.mobile = mobile;
    }
    public short getIsActive() {
        return isActive;
    }
    public void setIsActive(short isActive) {
        this.isActive = isActive;
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
```

```
        this.password = password;
    }
    public String getRespo() {
        return respo;
    }
    public void setRespo(String respo) {
        this.respo = respo;
    }

    public Customers[] SignIn(String username, String password) {
        Customers[] ccArray = null;
        dbAccess db = new dbAccess();
        Connection con = db.GetConnection();
        try {
            Statement stmnt = con.createStatement();
            ResultSet result;
            result = stmnt.executeQuery("CALL sp_login('" + username + "',  '" +
password + "')");
            ccArray = new Customers[1];
            while (result.next()) {
                Customers cc = new Customers();
                if (result.getString("resp").equals("0x9300")) {
                    cc.setRespo(result.getString("resp"));
                } else {
                    cc.setRespo(result.getString("resp"));

cc.setCustomerId(Long.parseLong(result.getString("customer_id")));
                    cc.setUsername(result.getString("username"));
                    cc.setFullName(result.getString("full_name"));

cc.setIsActive(Short.parseShort(result.getString("is_active")));
                }
                ccArray[0] = cc;
            }
        } catch (SQLException ex) {
            Logger.getLogger(card_service_1.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return ccArray;
    }

    public String Register(Customers customers) {
        dbAccess db = new dbAccess();
        Connection con = db.GetConnection();
        String response = "";
        try {
            Statement stmnt = con.createStatement();
            ResultSet result;
            result = stmnt.executeQuery(
                    "CALL sp_register("
                            + "'" + customers.fullName + "', "
                            + "'" + customers.nid + "', "
                            + "'" + customers.mobile + "', "
                            + "'" + customers.username + "', "
                            + "'" + customers.password + "' "
                            + ")");
            while (result.next()) {
                response = result.getString("resp");
            }
        } catch (SQLException ex) {
            Logger.getLogger(card_service_1.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return response;
    }
```

```
    String ChangePass(String CustomerID, String OldPassword, String NewPassword)
{
        dbAccess db = new dbAccess();
        Connection con = db.GetConnection();
        String response = "";
        try {
            Statement stmnt = con.createStatement();
            ResultSet result;
            result = stmnt.executeQuery(
                    "CALL sp_change_password("
                            + "'" + CustomerID + "', "
                            + "'" + OldPassword + "', "
                            + "'" + NewPassword + "' "
                            + ")");
            while (result.next()) {
                response = result.getString("resp");
            }
        } catch (SQLException ex) {
            Logger.getLogger(card_service_1.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return response;
    }
}
```

CustomerChannels.java

```
public class CustomerChannels implements Serializable {

    private static final long serialVersionUID = 1L;
    private long customerId;
    private int channelId;
    private String accountName;
    private String accountSha256;
    private String accountMask;
    private String expiry;
    private String nameIn;
    private short isActive;
    private short isOn;
    private Float maxOnlineValue;
    private short isOnlineOn;
    private Float maxOfflineValue;
    private short isOfflineOn;
    private Float maxWithdrawValue;
    private short isWithdrawOn;

    public CustomerChannels() {
    }
    public long getCustomerId() {
        return customerId;
    }
    public void setCustomerId(long customerId) {
        this.customerId = customerId;
    }
    public int getChannelId() {
        return channelId;
    }
    public void setChannelId(int channelId) {
        this.channelId = channelId;
    }
    public String getAccountName() {
        return accountName;
    }
    public void setAccountName(String accountName) {
        this.accountName = accountName;
    }
    public String getAccountSha256() {
```

```
    return accountSha256;
}
public void setAccountSha256(String accountSha256) {
    this.accountSha256 = accountSha256;
}
public String getAccountMask() {
    return accountMask;
}
public void setAccountMask(String accountMask) {
    this.accountMask = accountMask;
}
public String getExpiry() {
    return expiry;
}
public void setExpiry(String expiry) {
    this.expiry = expiry;
}
public String getNameIn() {
    return nameIn;
}
public void setNameIn(String nameIn) {
    this.nameIn = nameIn;
}
public short getIsActive() {
    return isActive;
}
public void setIsActive(short isActive) {
    this.isActive = isActive;
}
public short getIsOn() {
    return isOn;
}
public void setIsOn(short isOn) {
    this.isOn = isOn;
}
public Float getMaxOnlineValue() {
    return maxOnlineValue;
}
public void setMaxOnlineValue(Float maxOnlineValue) {
    this.maxOnlineValue = maxOnlineValue;
}
public short getIsOnlineOn() {
    return isOnlineOn;
}
public void setIsOnlineOn(short isOnlineOn) {
    this.isOnlineOn = isOnlineOn;
}
public Float getMaxOfflineValue() {
    return maxOfflineValue;
}
public void setMaxOfflineValue(Float maxOfflineValue) {
    this.maxOfflineValue = maxOfflineValue;
}
public short getIsOfflineOn() {
    return isOfflineOn;
}
public void setIsOfflineOn(short isOfflineOn) {
    this.isOfflineOn = isOfflineOn;
}
public Float getMaxWithdrawValue() {
    return maxWithdrawValue;
}
public void setMaxWithdrawValue(Float maxWithdrawValue) {
    this.maxWithdrawValue = maxWithdrawValue;
}

/**
```

```
     *
     * @return
     */
    public short getIsWithdrawOn() {
        return isWithdrawOn;
    }

    /**
     *
     * @param isWithdrawOn
     */
    public void setIsWithdrawOn(short isWithdrawOn) {
        this.isWithdrawOn = isWithdrawOn;
    }

    /**
     *
     * @param userID
     * @param IsActive
     * @return
     */
    public CustomerChannels[] GetCards(Long userID, String IsActive) {
        CustomerChannels[] ccArray = null;
        dbAccess db = new dbAccess();
        Connection con = db.GetConnection();
        try {
            Statement stmnt = con.createStatement();
            ResultSet result, backResult;
            result = backResult = stmnt.executeQuery(
                    "CALL sp_get_cards("
                    + "'" + userID + "', "
                    + "'" + IsActive + "' "
                    + ")");
            int j = 0;
            while (result.next()) {
                j++;
            }
            ccArray = new CustomerChannels[j];
            int i = 0;

            while (backResult.previous()) {
                CustomerChannels cc = new CustomerChannels();
                cc.setAccountSha256(backResult.getString("account_sha256"));
                cc.setAccountMask(backResult.getString("account_mask"));
                cc.setAccountName(backResult.getString("account_name"));

cc.setIsActive(Short.parseShort(backResult.getString("is_active")));
                cc.setIsOn(Short.parseShort(backResult.getString("is_on")));
                ccArray[i] = cc;
                i++;
            }
        } catch (SQLException ex) {
            Logger.getLogger(card_service_1.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return ccArray;
    }

    /**
     *
     * @param cardData
     * @return
     */
    public String AddCard(CustomerChannels cardData) {
        dbAccess db = new dbAccess();
        Connection con = db.GetConnection();
        String response = "";
```

```
        try {
            Statement stmnt = con.createStatement();
            ResultSet result;
            result = stmnt.executeQuery(
                    "CALL sp_addcard("
                    + "'" + cardData.getCustomerId() + "', "
                    + "'" + cardData.getAccountName() + "', "
                    + "'" + cardData.getAccountSha256() + "', "
                    + "'" + cardData.getAccountMask() + "', "
                    + "'" + cardData.getExpiry() + "', "
                    + "'" + cardData.getNameIn() + "' "
                    + ")");
            while (result.next()) {
                response = result.getString("resp");
            }
        } catch (SQLException ex) {
            Logger.getLogger(card_service_1.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return response;
    }

    /**
     *
     * @param cardSha256
     * @return
     */
    public CustomerChannels[] GetCardDetails(String cardSha256) {
        CustomerChannels[] ccArray = new CustomerChannels[1];
        dbAccess db = new dbAccess();
        Connection con = db.GetConnection();
        try {
            Statement stmnt = con.createStatement();
            ResultSet result;
            result  =  stmnt.executeQuery("CALL  sp_cardview('"  +  cardSha256  +
"')");
            while (result.next()) {
                CustomerChannels cc = new CustomerChannels();
                cc.setAccountSha256(result.getString("account_sha256"));
                cc.setAccountMask(result.getString("account_mask"));
                cc.setAccountName(result.getString("account_name"));
                cc.setExpiry(result.getString("expiry"));
                cc.setNameIn(result.getString("name_in"));
                cc.setIsOn(Short.parseShort(result.getString("is_on")));

cc.setMaxOnlineValue(Float.parseFloat(result.getString("max_online_value")));

cc.setIsOnlineOn(Short.parseShort(result.getString("is_online_on")));

cc.setMaxOfflineValue(Float.parseFloat(result.getString("max_offline_value")));

cc.setIsOfflineOn(Short.parseShort(result.getString("is_offline_on")));

cc.setMaxWithdrawValue(Float.parseFloat(result.getString("max_withdraw_value")))
;

cc.setIsWithdrawOn(Short.parseShort(result.getString("is_withdraw_on")));
                ccArray[0] = cc;
            }
        } catch (SQLException ex) {
            Logger.getLogger(card_service_1.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return ccArray;
    }

    public String UpdateCard(CustomerChannels ccUpdateCard) {
```

```
        dbAccess db = new dbAccess();
        Connection con = db.GetConnection();
        String response = "";
        try {
            Statement stmnt = con.createStatement();
            ResultSet result;
            result = stmnt.executeQuery("CALL sp_updatecard("
                    + "'" + ccUpdateCard.getAccountSha256() + "', "
                    + "'" + ccUpdateCard.getCustomerId() + "', "
                    + "'" + ccUpdateCard.getAccountName() + "', "
                    + "'" + String.valueOf(ccUpdateCard.getIsOn()) + "', "
                    + "'" + ccUpdateCard.getMaxOnlineValue() + "', "
                    + "'" + String.valueOf(ccUpdateCard.getIsOnlineOn()) + "', "
                    + "'" + ccUpdateCard.getMaxOfflineValue()+ "', "
                    + "'" + String.valueOf(ccUpdateCard.getIsOfflineOn()) + "',
"
                    + "'" + ccUpdateCard.getMaxWithdrawValue()+ "', "
                    + "'" + String.valueOf(ccUpdateCard.getIsWithdrawOn()) + "'
"
                    + ")");
            while (result.next()) {
                response = result.getString("resp");
            }
        } catch (SQLException ex) {
            Logger.getLogger(card_service_1.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return response;
    }

    String DeleteCard(String CardSha256, String CustomerID) {
        dbAccess db = new dbAccess();
        Connection con = db.GetConnection();
        String response = "";
        try {
            Statement stmnt = con.createStatement();
            ResultSet result;
            result = stmnt.executeQuery(
                    "CALL sp_deletecard("
                    + "'" + CardSha256 + "', "
                    + "'" + CustomerID + "' "
                    + ")");
            while (result.next()) {
                response = result.getString("resp");
            }
        } catch (SQLException ex) {
            Logger.getLogger(card_service_1.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return response;
    }
}
```

user_service_1.java

```
public class user_service_1 {

    /**
     * This is a sample web service operation
     * @param username
     * @param password
     * @return
     */
    @WebMethod(operationName = "SignIn")
    public Customers[] SignIn(
            @WebParam(name = "username") String username,
```

```
            @WebParam(name = "password") String password) {
        Customers ccSignIn = new Customers();
        return ccSignIn.SignIn(username, password);
    }

    @WebMethod(operationName = "Register")
    public String Register(
            @WebParam(name = "fullname") String fullname,
            @WebParam(name = "nic") String nic,
            @WebParam(name = "mobile") String mobile,
            @WebParam(name = "username") String username,
            @WebParam(name = "password") String password) {
        Customers ccRegister = new Customers();
        ccRegister.setFullName(fullname);
        ccRegister.setNid(nic);
        ccRegister.setMobile(mobile);
        ccRegister.setUsername(username);
        ccRegister.setPassword(password);
        return ccRegister.Register(ccRegister);
    }

    @WebMethod(operationName = "ChangePass")
    public String ChangePass(
            @WebParam(name = "CustomerID") String CustomerID,
            @WebParam(name = "OldPassword") String OldPassword,
            @WebParam(name = "NewPassword") String NewPassword) {
        Customers ccSignIn = new Customers();
        return ccSignIn.ChangePass(CustomerID, OldPassword, NewPassword);
    }
}
```

card_service_1.java

```
public class card_service_1 {

    /**
     * This is a sample web service operation
     *
     * @param UserID
     * @param IsActive
     * @return CustomerCannels_
     */
    @WebMethod(operationName = "GetCards")
    public CustomerChannels[] GetCards(
            @WebParam(name = "userID") Long UserID,
            @WebParam(name = "IsActive") String IsActive) {
        CustomerChannels ccInactive = new CustomerChannels();
        return ccInactive.GetCards(UserID, IsActive);
    }

    /**
     *
     * @param CardSha256
     * @return
     */
    @WebMethod(operationName = "GetCardDetails")
    public  CustomerChannels[]  GetCardDetails(@WebParam(name  =  "CardSha256")
String CardSha256) {
        CustomerChannels ccInactive = new CustomerChannels();
        return ccInactive.GetCardDetails(CardSha256);
    }

    /**
     *
     * @param CustomerID
     * @param CardName
```

```
 * @param CardSha256
 * @param CardMask
 * @param Expiry
 * @param NameInCard
 * @return
 */
@WebMethod(operationName = "AddCard")
public String AddCard(
        @WebParam(name = "CustomerID") Long CustomerID,
        @WebParam(name = "CardName") String CardName,
        @WebParam(name = "CardSha256") String CardSha256,
        @WebParam(name = "CardMask") String CardMask,
        @WebParam(name = "Expiry") String Expiry,
        @WebParam(name = "NameInCard") String NameInCard) {
    CustomerChannels ccAddCard = new CustomerChannels();
    ccAddCard.setCustomerId(CustomerID);
    ccAddCard.setAccountName(CardName);
    ccAddCard.setAccountSha256(CardSha256);
    ccAddCard.setAccountMask(CardMask);
    ccAddCard.setExpiry(Expiry);
    ccAddCard.setNameIn(NameInCard);
    return ccAddCard.AddCard(ccAddCard);
}

/**
 *
 * @param CustomerID
 * @param CardName
 * @param CardSha256
 * @param IsOn
 * @param MaxOnlineValue
 * @param IsOnlineOn
 * @param MaxOfflineValue
 * @param IsOfflineOn
 * @param MaxWithdrawValue
 * @param IsWithdrawOn
 * @return
 */
@WebMethod(operationName = "UpdateCard")
public String UpdateCard(
        @WebParam(name = "CustomerID") Long CustomerID,
        @WebParam(name = "CardName") String CardName,
        @WebParam(name = "CardSha256") String CardSha256,
        @WebParam(name = "IsOn") String IsOn,
        @WebParam(name = "MaxOnlineValue") String MaxOnlineValue,
        @WebParam(name = "IsOnlineOn") String IsOnlineOn,
        @WebParam(name = "MaxOfflineValue") String MaxOfflineValue,
        @WebParam(name = "IsOfflineOn") String IsOfflineOn,
        @WebParam(name = "MaxWithdrawValue") String MaxWithdrawValue,
        @WebParam(name = "IsWithdrawOn") String IsWithdrawOn) {
    CustomerChannels ccUpdateCard = new CustomerChannels();
    ccUpdateCard.setCustomerId(CustomerID);
    ccUpdateCard.setAccountName(CardName);
    ccUpdateCard.setAccountSha256(CardSha256);
    ccUpdateCard.setIsOn(Short.parseShort(IsOn));
    ccUpdateCard.setMaxOnlineValue(Float.parseFloat(MaxOnlineValue));
    ccUpdateCard.setIsOnlineOn(Short.parseShort(IsOnlineOn));
    ccUpdateCard.setMaxOfflineValue(Float.parseFloat(MaxOfflineValue));
    ccUpdateCard.setIsOfflineOn(Short.parseShort(IsOfflineOn));
    ccUpdateCard.setMaxWithdrawValue(Float.parseFloat(MaxWithdrawValue));
    ccUpdateCard.setIsWithdrawOn(Short.parseShort(IsWithdrawOn));
    return ccUpdateCard.UpdateCard(ccUpdateCard);
}

/**
 *
 * @param CardSha256
```

```
     * @param CustomerID
     * @return
     */
    @WebMethod(operationName = "DeleteCard")
    public String DeleteCard(
            @WebParam(name = "CardSha256") String CardSha256,
            @WebParam(name = "CustomerID") String CustomerID
    ) {
        CustomerChannels ccDeleteCard = new CustomerChannels();
        return ccDeleteCard.DeleteCard(CardSha256, CustomerID);
    }
}
```

SOAP Service Requests

“Register” Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.or
   <soapenv:Header/>
   <soapenv:Body>
      <ser:Register>
         <fullname>A B C Silva</fullname>
         <nic>842458542V</nic>
         <mobile>0713096373</mobile>
         <username>silva</username>
         <password>65e84be33532fb784c48129675f9eff3a682b27
      </ser:Register>
   </soapenv:Body>
</soapenv:Envelope>
```

“Register” Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:RegisterResponse xmlns:ns2="http://services.epay.com/">
         <return>0x9000</return>
      </ns2:RegisterResponse>
   </S:Body>
</S:Envelope>
```

“SignIn” Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.or
   <soapenv:Header/>
   <soapenv:Body>
      <ser:SignIn>
         <username>silva</username>
         <password>65e84be33532fb784c48129675f9eff3a682b27
      </ser:SignIn>
   </soapenv:Body>
</soapenv:Envelope>
```

“ SignIn” Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envel
   <S:Body>
      <ns2:SignInResponse xmlns:ns2="http://services.epay.
         <return>
            <item>
               <customerId>11</customerId>
               <fullName>A B C Silva</fullName>
               <isActive>0</isActive>
               <respo>0x9000</respo>
               <username>silva</username>
            </item>
         </return>
      </ns2:SignInResponse>
   </S:Body>
</S:Envelope>
```

“AddCard” Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap
   <soapenv:Header/>
   <soapenv:Body>
      <ser:AddCard>
         <CustomerID>11</CustomerID>
         <CardName>A B C Bank</CardName>
         <CardSha256>18df5e12539da8a7432f6b93687c6b14273151dbc9c
         <CardMask>41472XXXXXXX9456</CardMask>
         <Expiry>03/20</Expiry>
         <NameInCard>A B C Silva</NameInCard>
      </ser:AddCard>
   </soapenv:Body>
</soapenv:Envelope>
```

“AddCard” Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:AddCardResponse xmlns:ns2="http://services.epay.com/"
         <return>0x9000</return>
      </ns2:AddCardResponse>
   </S:Body>
</S:Envelope>
```

“UpdateCard” Request

“UpdateCard” Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:UpdateCardResponse xmlns:ns2="http://services.epay.co
         <return>0x9000</return>
      </ns2:UpdateCardResponse>
   </S:Body>
</S:Envelope>
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap
   <soapenv:Header/>
   <soapenv:Body>
      <ser:UpdateCard>
         <CustomerID>11</CustomerID>
         <CardName>Primary</CardName>
         <CardSha256>18df5e12539da8a7432f6b93687c6b14273151dbc9c
         <IsOn>1</IsOn>
         <MaxOnlineValue>50000</MaxOnlineValue>
         <IsOnlineOn>1</IsOnlineOn>
         <MaxOfflineValue>20000</MaxOfflineValue>
         <IsOfflineOn>0</IsOfflineOn>
         <MaxWithdrawValue>30000</MaxWithdrawValue>
         <IsWithdrawOn>1</IsWithdrawOn>
      </ser:UpdateCard>
   </soapenv:Body>
</soapenv:Envelope>
```

“GetCards” List Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap
   <soapenv:Header/>
   <soapenv:Body>
      <ser:GetCards>
         <userID>11</userID>
         <IsActive>1</IsActive>
      </ser:GetCards>
   </soapenv:Body>
</soapenv:Envelope>
```

“GetCards” List Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:GetCardsResponse xmlns:ns2="http://services.epay.com/
         <return>
            <item>
               <accountMask>41472XXXXXXX9456</accountMask>
               <accountName>Primary</accountName>
               <accountSha256>18df5e12539da8a7432f6b93687c6b1427
               <channelId>0</channelId>
               <customerId>0</customerId>
               <isActive>1</isActive>
               <isOfflineOn>0</isOfflineOn>
               <isOn>1</isOn>
               <isOnlineOn>1</isOnlineOn>
               <isWithdrawOn>1</isWithdrawOn>
            </item>
         </return>
      </ns2:GetCardsResponse>
   </S:Body>
</S:Envelope>
```

“GetCardDetails” Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap
   <soapenv:Header/>
   <soapenv:Body>
      <ser:GetCardDetails>
         <CardSha256>18df5e12539da8a7432f6b93687c6b14273151dbc9c
      </ser:GetCardDetails>
   </soapenv:Body>
</soapenv:Envelope>
```

‘GetCardDetails” Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:GetCardDetailsResponse xmlns:ns2="http://services.epa
         <return>
            <item>
               <accountMask>41472XXXXXXX9456</accountMask>
               <accountName>Primary</accountName>
               <accountSha256>18df5e12539da8a7432f6b93687c6b1427
               <channelId>0</channelId>
               <customerId>0</customerId>
               <expiry>03/20</expiry>
               <isActive>0</isActive>
               <isOfflineOn>0</isOfflineOn>
               <isOn>1</isOn>
               <isOnlineOn>1</isOnlineOn>
               <isWithdrawOn>1</isWithdrawOn>
               <maxOfflineValue>20000.0</maxOfflineValue>
               <maxOnlineValue>50000.0</maxOnlineValue>
               <maxWithdrawValue>30000.0</maxWithdrawValue>
               <nameIn>A B C Silva</nameIn>
            </item>
         </return>
      </ns2:GetCardDetailsResponse>
   </S:Body>
</S:Envelope>
```

“DeleteCard” Request

“DeleteCard” Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelop
   <S:Body>
      <ns2:DeleteCardResponse xmlns:ns2="http://services.epa
         <return>0x9000</return>
      </ns2:DeleteCardResponse>
   </S:Body>
</S:Envelope>
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
   <soapenv:Header/>
   <soapenv:Body>
      <ser:DeleteCard>
         <CardSha256>18df5e12539da8a7432f6b93687c6b14273151d
         <CustomerID>11</CustomerID>
      </ser:DeleteCard>
   </soapenv:Body>
</soapenv:Envelope>
```

“GetTransactions” Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap
   <soapenv:Header/>
   <soapenv:Body>
      <ser:GetTransactions>
         <CustomerID>11</CustomerID>
         <Range>100</Range>
      </ser:GetTransactions>
   </soapenv:Body>
</soapenv:Envelope>
```

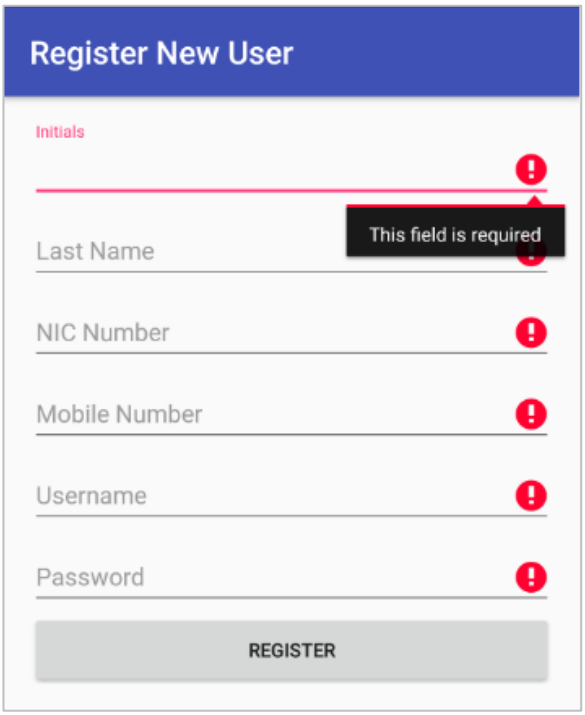
“GetTransactions” Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:GetTransactionsResponse xmlns:ns2="http://services.ep
         <return>
            <item>
               <accountMask>41472XXXXXXX9456</accountMask>
               <amount>1000.0</amount>
               <merchant>EPAYSWITCH TEST LOCATION COLOMBO LKA</m
               <status>SUCCESS</status>
               <trnTime>2017-01-28 10:27:25.0</trnTime>
            </item>
            <item>
               <accountMask>41472XXXXXXX9456</accountMask>
               <amount>1000.0</amount>
               <merchant>EPAYSWITCH TEST LOCATION COLOMBO LKA</m
               <status>ABOVE LIMIT</status>
               <trnTime>2017-01-28 10:27:01.0</trnTime>
            </item>
```

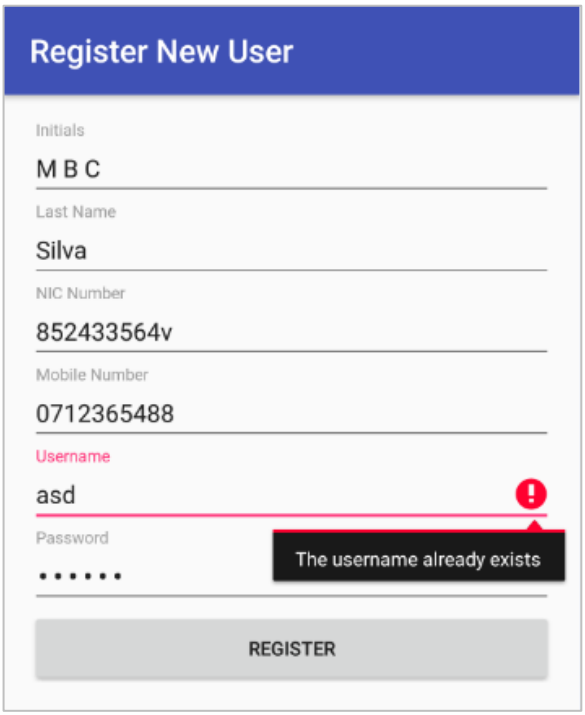
Appendix C – User Interfaces

Mobile Application

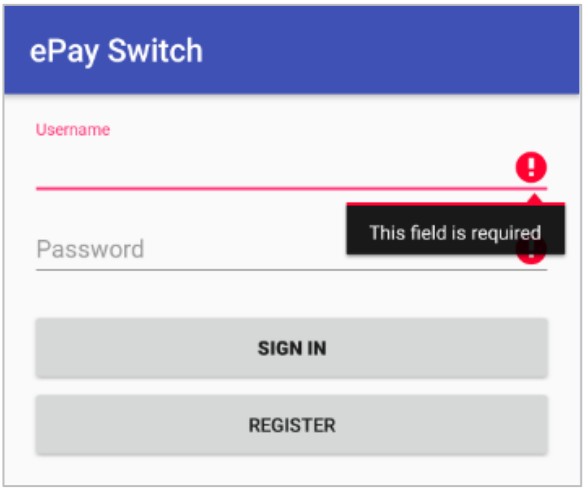
Register All Fields Required



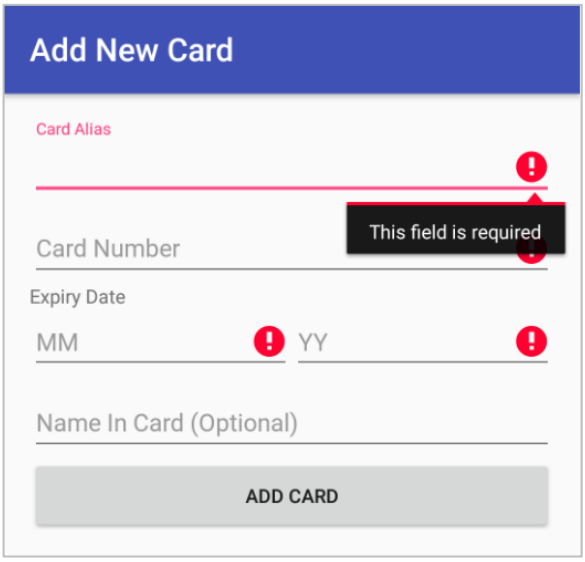
Register Username Exists



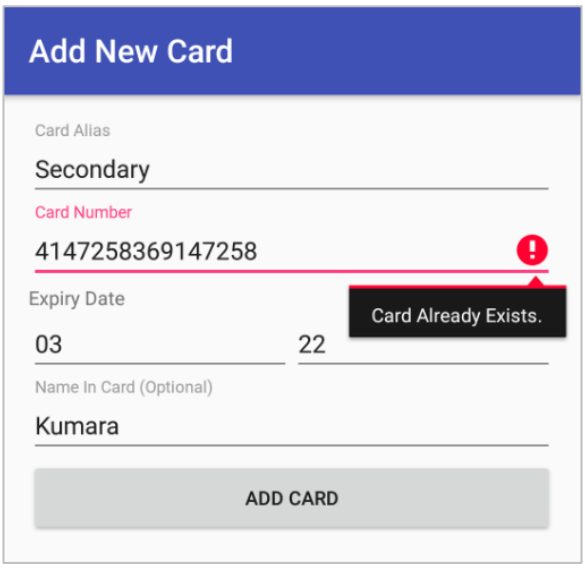
Sign In All Fields Required



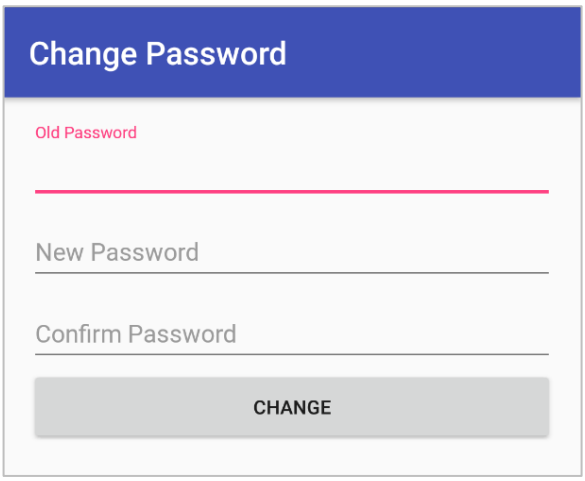
Add New Card All Fields Required



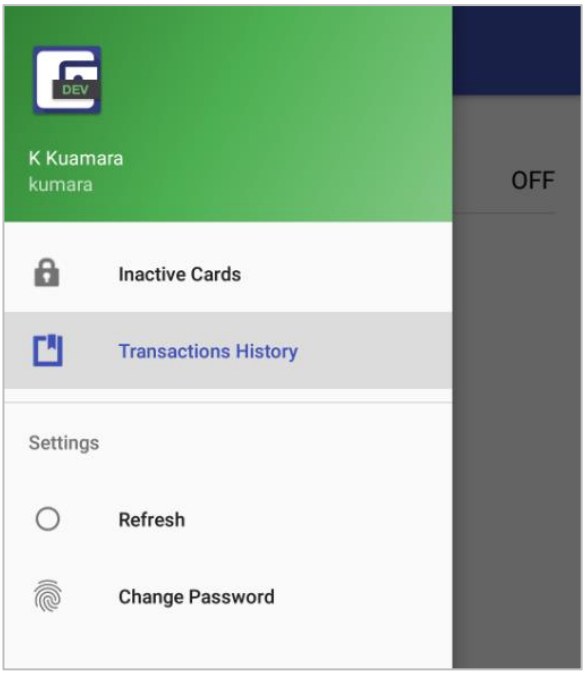
Add New Card Already Exists



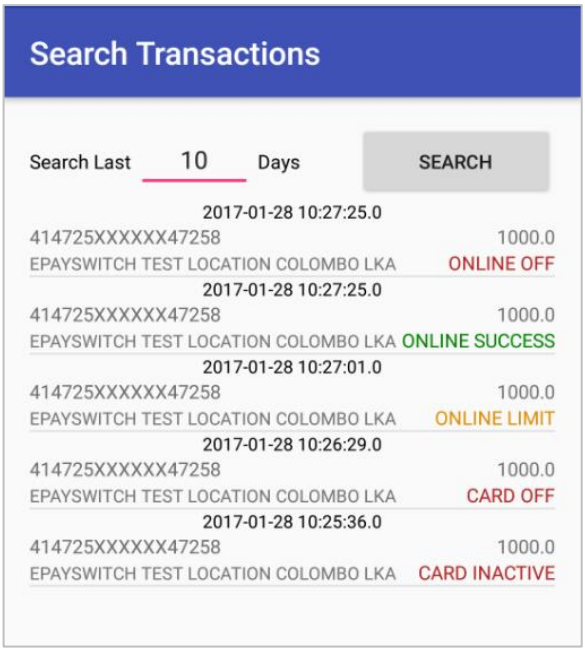
Password Change



Menu Drawer

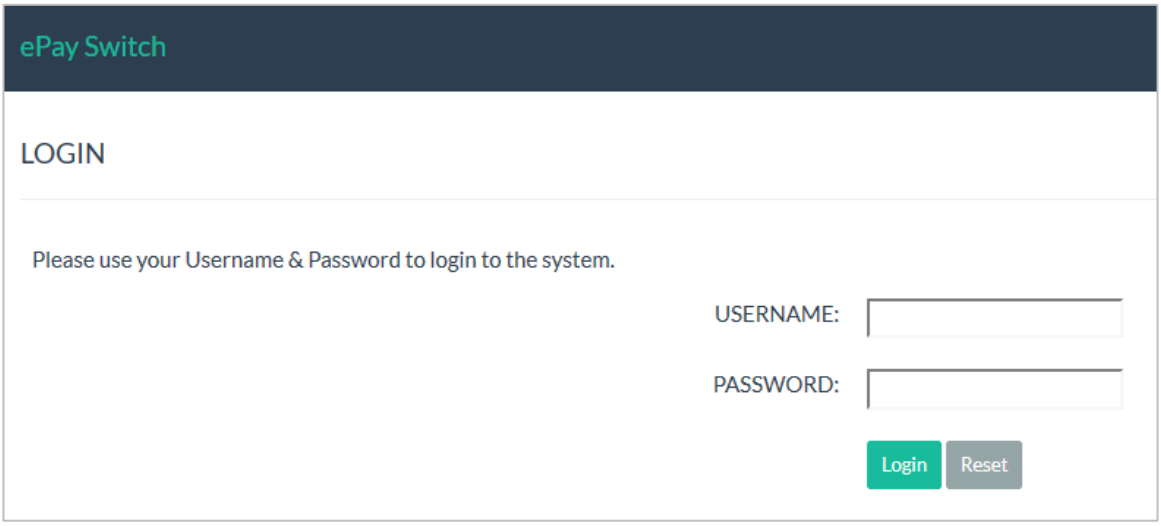


Transaction Summary

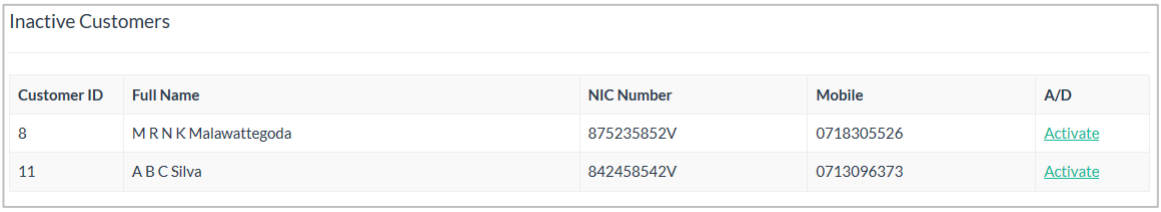


Customer and Cards Manager

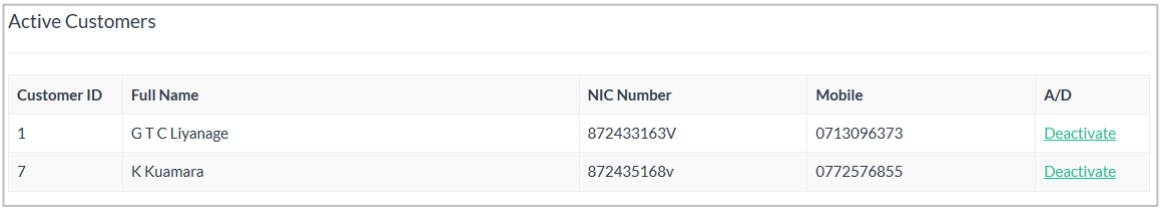
User Login



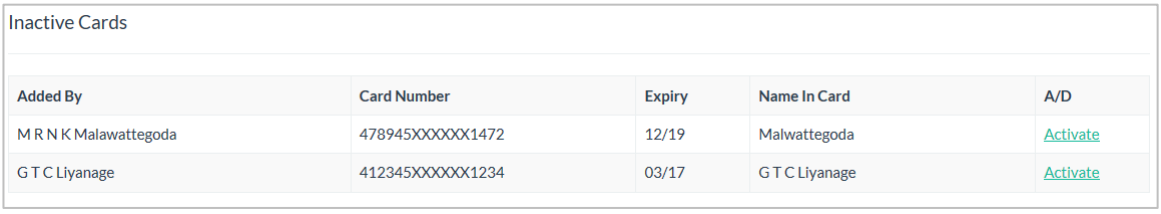
Inactive Customers (Activate Customers)



Active Customers (Deactivate Customers)



Inactive Cards (Activate Cards)



Active Cards (Deactivate Cards)

Active Cards

Added By	Card Number	Expiry	Name In Card	A/D
K Kuamara	414725XXXXXX4725	12/20	A B C Kumara	Deactivate
K Kuamara	412345XXXXXX1230	12/18	Kumara	Deactivate

Appendix D – Questionnaire and Responses

ePaySwitch

This questionnaire was prepare as evaluation technique to the research "Implementing a Software Switch and a Mobile Application to Prevent Fraud and Control the Usage of Electronic Transactions" done by G. T. C. Liyanage as a partial fulfillment of the M. Sc. in IT at University of Moratuwa, Faculty of Information Technology.

Please be kind enough to spend a little from your valuable time to answer this as it will help me in completing the evaluation process of the above mentioned task.

Thank You.

Sincerely,
#Thilanga (G T C Liyanage)

*Required

ePaySwitch

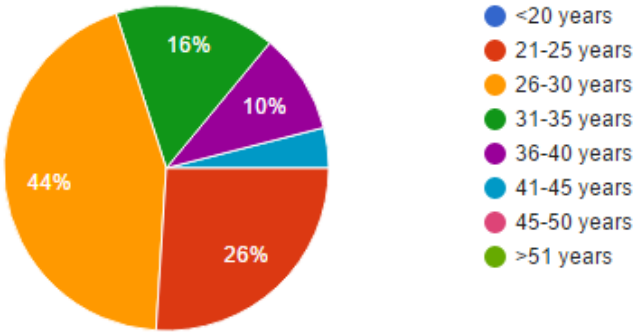
50 responses

1. **Please Specify Your Age Group** *

Mark only one oval.

- <20 years
- 21-25 years
- 26-30 years
- 31-35 years
- 36-40 years
- 41-45 years
- 45-50 years
- >51 years

Please Specify Your Age Group (50 responses)

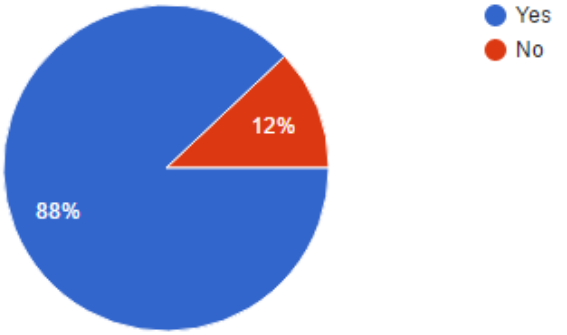


2. **Do you have a Credit or Debit Card?** *

Mark only one oval.

- ◯ Yes *Skip to question 3.*
- ◯ No *Skip to question 12.*

Do you have a Credit or Debit Card? (50 responses)

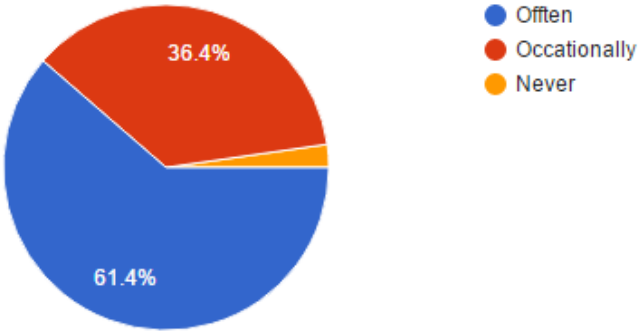


3. **How frequently you use Credit/Debit card for transactions?** *

Mark only one oval.

- ◯ Offten *Skip to question 4.*
- ◯ Occationally *Skip to question 12.*
- ◯ Never *Skip to question 12.*

How frequently you use Credit/Debit card for transactions?
(44 responses)

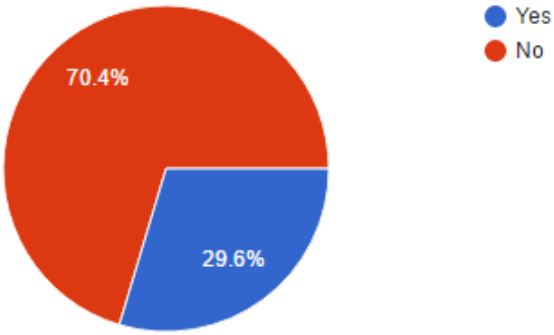


4. **Have you ever lost your card?** *

Mark only one oval.

- Yes
- No

Have you ever lost your card? (27 responses)



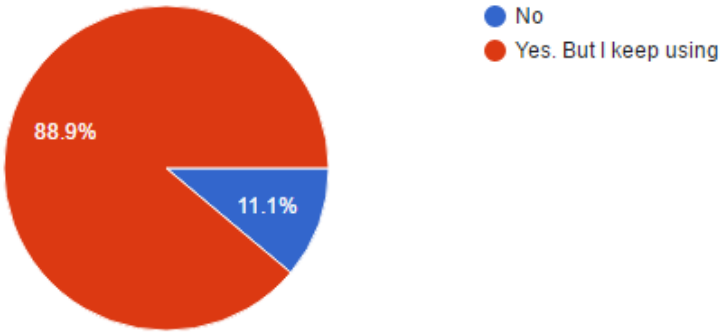
5. **Have you ever thought someone will steel your card & perform transaction before you know?** *

Mark only one oval.

- No
- Yes. But I keep using

Have you ever thought someone will steel your card & perform transaction before you know?

(27 responses)



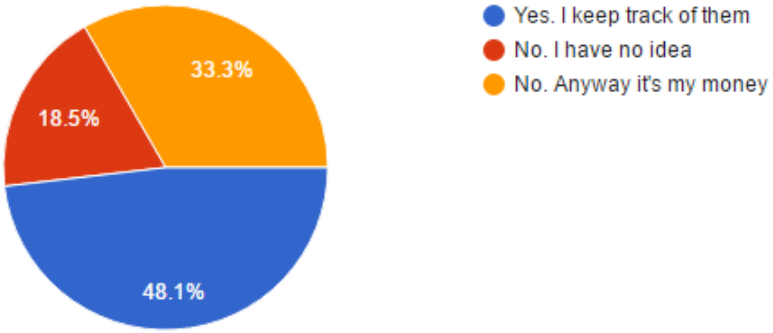
6. **Do you aware of transaction amounts you perform via cards?** *

Mark only one oval.

- ◯ Yes. I keep track of them
- ◯ No. I have no idea
- ◯ No. Anyway it's my money

Do you aware of transaction amounts you perform via cards?

(27 responses)



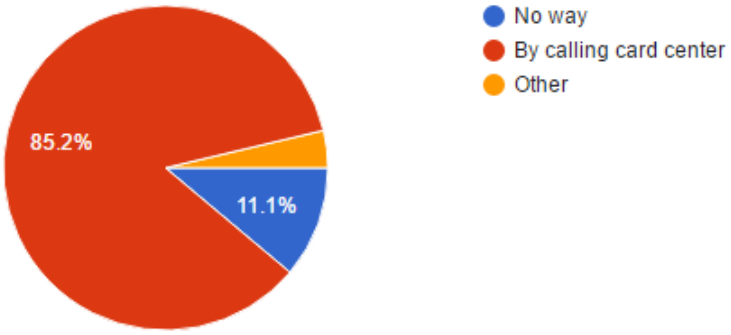
7. **If some one steel your card and try to perform transactions, how will YOU be able to block it?** *

Mark only one oval.

- ◯ No way
- ◯ By calling card center
- ◯ Other: _____

If some one steel your card and try to perform transactions, how will YOU be able to block it?

(27 responses)



8. **What if you have an application to switch on/off your card in order to perform a transaction? Will you try it?** *

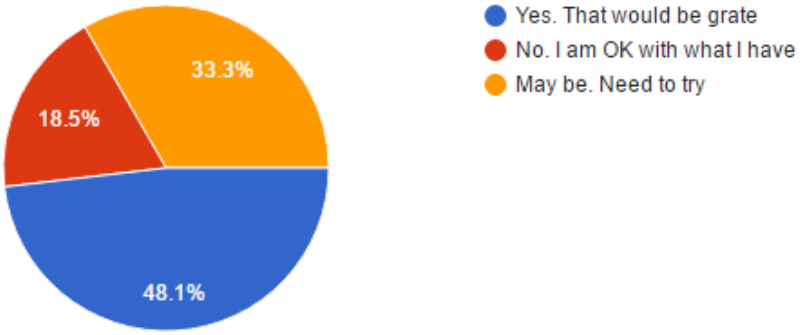
Mark only one oval.

- Yes. That would be grate
- No. I am OK with what I have
- May be. Need to try

Skip to question 10.

What if you have an application to switch on/off your card in order to perform a transaction? Will you try it?

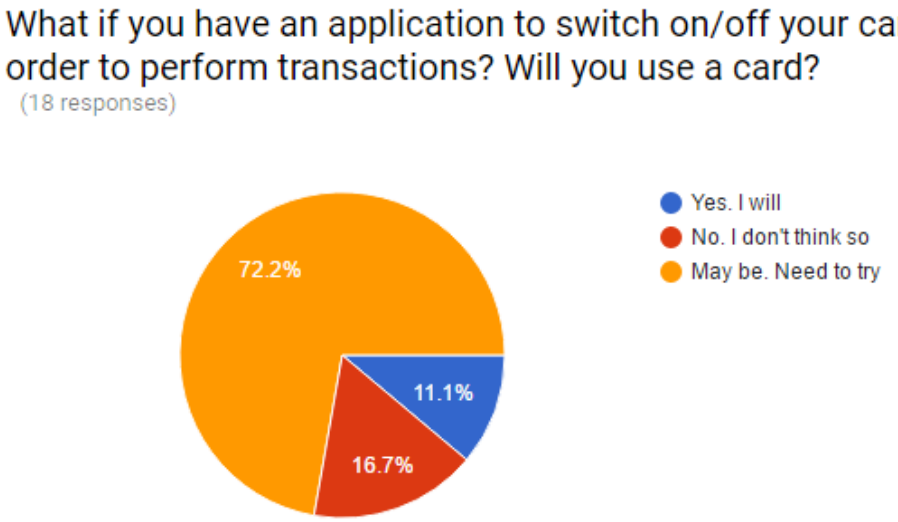
(27 responses)



9. **What if you have an application to switch on/off your card in order to perform transactions? Will you use a card?** *

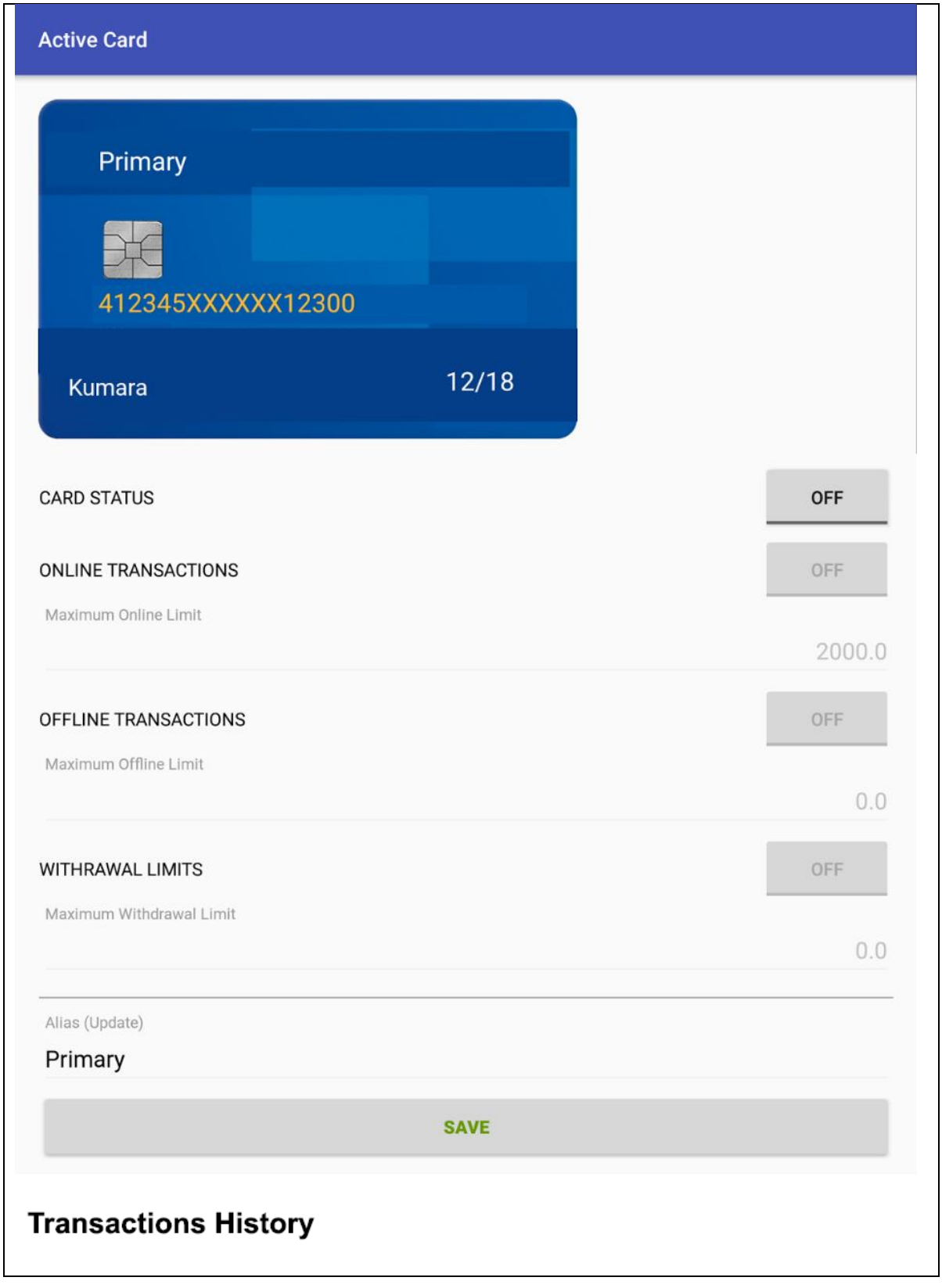
Mark only one oval.

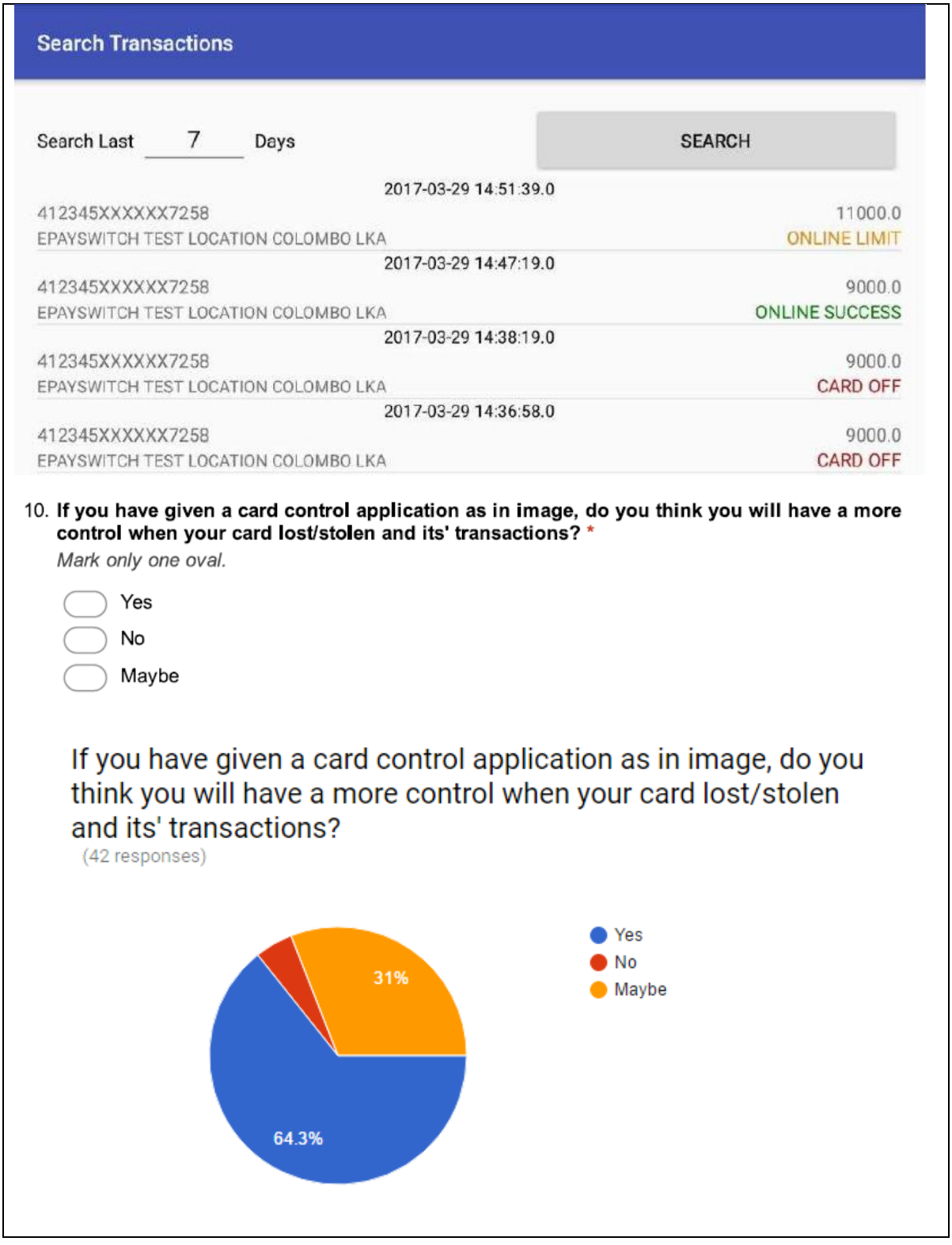
- ◯ Yes. I will *Skip to question 10.*
- ◯ No. I don't think so *Stop filling out this form.*
- ◯ May be. Need to try *Skip to question 10.*



In the below shown figure, we suggest you an application which can control your card transactions. You can switch on/off your card completely or for specific types on transactions.

Card Control App


Active Card
Primary
412345XXXXXX12300
Kumara
12/18
CARD STATUS
OFF
ONLINE TRANSACTIONS
OFF
Maximum Online Limit
2000.0
OFFLINE TRANSACTIONS
OFF
Maximum Offline Limit
0.0
WITHRAWAL LIMITS
OFF
Maximum Withdrawal Limit
0.0
Alias (Update)
Primary
SAVE
Transactions History



10. **If you have given a card control application as in image, do you think you will have a more control when your card lost/stolen and its' transactions?** *

Mark only one oval.

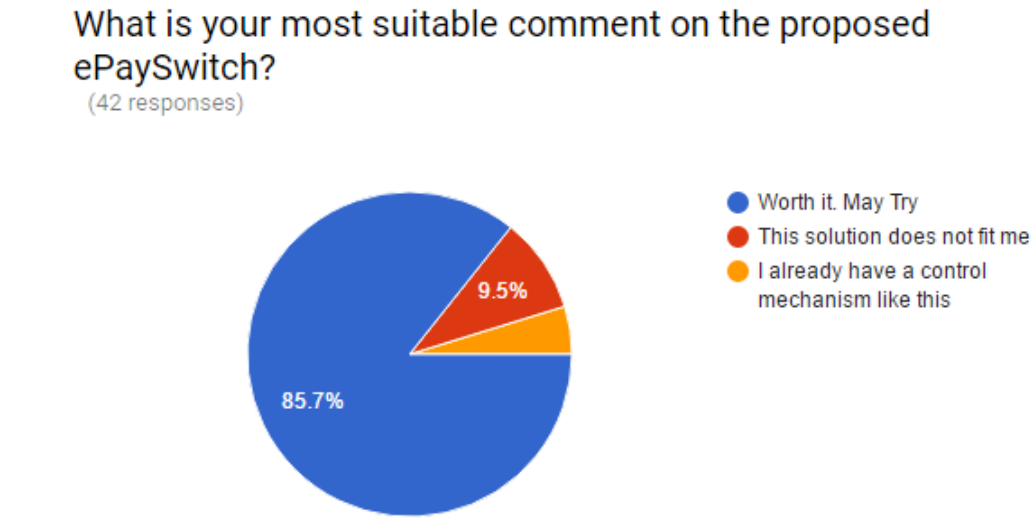
- Yes
- No
- Maybe

11. **What is your most suitable comment on the proposed ePaySwitch?** *

Mark only one oval.

- ◯ Worth it. May Try
- ◯ I already have a control mechanism like this
- ◯ This solution does not fit me

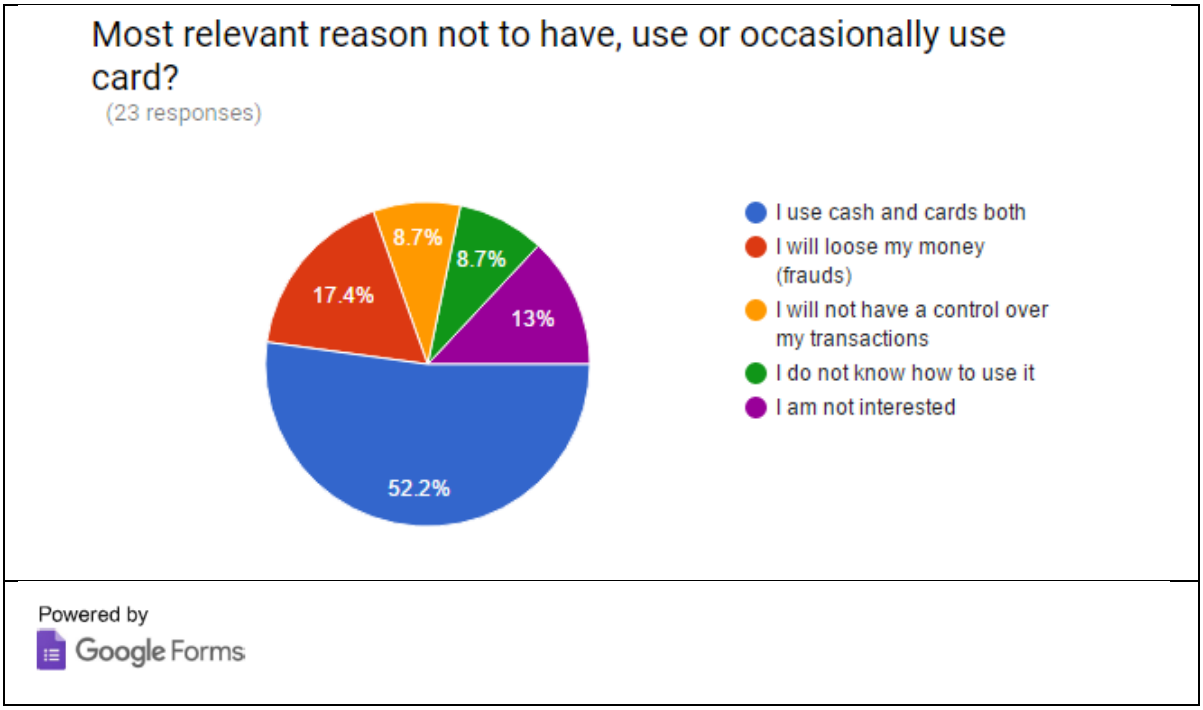
Stop filling out this form.



12. **Most relevant reason not to have, use or occasionally use card?** *

Mark only one oval.

- ◯ I use cash and cards both *Skip to question 9.*
- ◯ I will loose my money (frauds) *Skip to question 9.*
- ◯ I will not have a control over my transactions *Skip to question 9.*
- ◯ I do not know how to use it *Stop filling out this form.*
- ◯ I am not interested *Stop filling out this form.*


Most relevant reason not to have, use or occasionally use card?
(23 responses)
8.7%
8.7%
17.4%
13%
52.2%
I use cash and cards both
I will loose my money (frauds)
I will not have a control over my transactions
I do not know how to use it
I am not interested
Powered by
Google Forms