Cyber Vulnerabilities & Mitigation Approaches for Digital Payment Applications

Smriti Vipin Madangarli

PROBLEM STATEMENT

Digital payment has become the new norm and customers are increasingly engaging in UPI transactions. The simplicity of the application does encourage widespread usage but also results in neglection of security concerns and significant vulnerability exposure in transactions. These security concerns call for better understanding of the digital transaction process and how best to address the vulnerabilities.

OBJECTIVE

The objective of this paper is to understand possible vulnerabilities in the digital transaction process (specifically using mobile payment applications) and propose additional security steps that can safeguard users.

ABSTRACT

With the emergence of digital payment technologies, UPI and its integrated applications including BHIM, GooglePay, PayTM and PhonePe have become tremendously popular in the market and their consumer - merchant satisfaction is also significantly high. This study aims at evaluating the various security risks and vulnerabilities in digital transactions, doing a case study of the possible threat domains and evaluating use-case statistics of these applications. Examining the existing threats and some of the existing mitigation strategies, the outcome of this project is to implement a model with a 3-step verification protocol and safeguard the application users against fraudulent attacks.

Keywords: digital payment, UPI, fraud detection, BHIM, mitigation strategies, application threats, device threats

INTRODUCTION

UPI (Unified Payments Interface) is a real-time payment system developed by the National Payments Corporation of India (NPCI) in 2016 that enables users to instantly transfer funds between bank accounts using their mobile devices. UPI has gained significant popularity in India as it facilitates 24*7 immediate money transfer, nationalises the concept of QR codes and does not require details such as card number, account number or IFSC for transactions [1].

NPCI has developed its own application BHIM (Bharat Interface for Money) that enables users to make simple, easy and quick payment transactions using UPI. There are a lot of third party applications that are affiliated with certain private banks which employ the functionalities of UPI ;

some of the popular ones being PhonePe, PayTM, GooglePay, Samsung Pay etc. As of April 2023, there are 414 banks live on UPI [2].

While UPI offers convenience and ease of use, like any digital payment system, it may also be vulnerable to certain risks and vulnerabilities. As of 2022-23, the Union finance ministry has recorded over 95,000 fraud cases of UPI transactions; a significant increase when compared to the statistics of 77,000 cases in 2020-21 and 84,000 cases in 2021-22 [3]. Fraudsters use various social engineering techniques to scam victims. Some of them are as listed below [4] [5]:

1. *Phishing and Social Engineering:* UPI transactions often involve sharing sensitive information such as mobile numbers, UPI IDs, or OTPs (One-Time Passwords). Attackers may use phishing techniques or social engineering tactics to trick users into revealing their credentials or OTPs, enabling them to perform unauthorised transactions.
2. *Malware and Mobile Device Exploitation:* If a user's mobile device is infected with malware or compromised, attackers can gain unauthorised access to UPI apps and manipulate transactions. This can occur through various means such as malicious apps, fake UPI apps, or operating system vulnerabilities.
3. *Man-in-the-Middle Attacks:* UPI transactions typically involve communication between the user's device, the UPI app, and the payment service provider. Attackers may attempt to intercept and manipulate the communication channel to alter transaction details, redirect funds to their accounts, or gain access to sensitive information.
4. *Fraudulent Apps and Transactions*: Attackers may develop fraudulent UPI apps that imitate legitimate ones to deceive users into performing transactions. These apps may collect sensitive information or perform unauthorised transactions without the user's knowledge.
5. *Account Takeover:* If a user's UPI account is compromised, attackers can gain control over the account and conduct fraudulent transactions. This can occur through techniques like password guessing, credential stuffing, or exploiting weak authentication mechanisms.

In this work, the principal focus is to explore additional security mechanisms such as GPS based authentication in addition to existing 2 Step verification protocols, to reduce the risk of fraudulent transactions.

LITERATURE SURVEY

1. Security Analysis of Unified Payments Interface and Payment Apps in India

Renuka Kumar, Sreesh Kishore, Hao Lau University of Michigan

Included in the Proceedings of the 29th USENIX Security Symposium August 12–14, 2020. [6]

In this research paper, Renuka Kumar et al reverse engineered the UPI protocol through the UPI apps and figured out subtle design flaws in the UPI protocol.

The research consisted of evaluating the design choices in UPI 1.0 and vulnerability risks caused. Three possible attack mechanisms were uncovered including:

A. Attack due to unauthorised registration given a user's cell number
B. Attack due to unauthorised transactions on bank accounts given a user's cell number and partial debit card number
C. Attack due to unauthorised transactions without debit card numbers

They proposed a threat model involving an application named *Mally* having the permissions android.permission.INTERNET and android.permission.RECEIVE_SMS enabled. This application *Mally* is said to have accessibility permissions including READ_PHONE_STATE. Three potential security holes were studied and it was figured out that if the attacker could find a way to bind their cellphone to the victim's cell number and obtain the OTP via sms reading function of the *Mally* app that the user harmlessly installed; it could be possible to breach the UPI security and commit theft. These security risks were studied with respect to the NPCI application BHIM and some exploitation mechanism workflows were discussed. One possible way to exploit the security hole (A) would be as follows:

- The attacker puts their cell phone in airplane mode while remaining connected to the Internet via Wi-Fi.
- The BHIM app on the attacker's phone starts a handshake with the device details and the UPI server responds with a device registration token.
- The attacker disables SMS messaging and thus the BHIM application cannot relay the token back to the UPI server.
- The BHIM app thus prompts to key-in a cell number and the attacker keys in the victim's.
- Thus the UPI server now contains the attacker's registration token linked to the victim's phone number.

In this way, the attacker is able to bind to the victim's cell phone and intercept the OTP using the *Mally* application.

Further research extended to UPI 2.0 and it was discovered that some of the attack mechanisms were prevented there. In addition to device information, UPI 2.0 also sends the device's IMEI number, SIM number, network type etc. which enables hard binding. This blocks out the potential security hole discussed above and results in a positive change.

However an important learning after studying a few other third party apps too is that the bank account number leaked from the default workflow of any of the third-party applications is enough to reset a user's passcode on another app.

2. **A Platform for Uncovering Indian Users Decision-Making Process in Unified Payment Interface (UPI) App**

Kshitiz Sharma, Nandini Bajaj*, Xinru Page, Mainack Mondal*
IIT Kharagpur, Brigham Young University [7]

The researchers off this paper aimed at and studied the decision making process of the users that could leave them vulnerable to fraud.

For this purpose, they built an application UPI-Pay which simulates the same functionality of UPI and tested out fraudulent scenarios on a group of people. Their observations concluded that application designers should make features like amount validation pages more noticeable so that customers were less likely to overlook it.

The methodology employed included developing UPI-Pay with common functionalities including home page, balance view, payee information and transaction view among others. Based on the application, an interview protocol was developed and the participants were asked to make transactions in various scenarios. The research paper hints at the limitations in its scope due to testing only in a simulated environment and not based on actual attacks with actual credentials and bank details.

The major scenarios tested out included payment and request transactions via contact number and via QR code. One of the scenarios tested upon included a minor discrepancy in the amount validation page and how users reacted to it. A large number of participants overlooked the confirmation page in haste and proceeded to pay much more than they had thought they would. Such kinds of decision-making situations call for better design construction according to the researchers.

3. **An Overview of India's Unified Payments Interface (UPI): Benefits, Challenges, and Opportunities**

Dr.A.Shaji George , A.S.Hovan George, , Dr.T.Baskar , A.S.Gabrio Martin

International Research Journal (PUIRJ)Volume: 02 Issue: 01 | January-March 2023 | ISSN: 2583-5602 [8]

Dr Shaji et al examined UPI as a whole comprising the various benefits, challenges and future scope of this technology. The study methodology involved reviewing resources published in the last five years and brought to light how previous works had focused more

on security, money management and the basic UPI structure. There was a lack of analysis of the advances of UPI and thus the researchers' problem came to place.

The benefit analysis of UPI included examining how the introduction of UPI made a difference in consumer markets. Customers could avail this service even in the absence of internet by dialling the USSD code *99# provided by NPCI. The benefits of UPI extend to the elimination of the hassle involved in setting up a card machine, reduction of processing time and increase in customer-satisfaction and transparency in business transactions.

The provision of UPI gives way for numerous opportunities for banks to expand their business, enable faster transactions and provide a convenient interface for customers and merchants alike. Operational costs can be reduced and customer experience can be improved.

A few challenges pertaining to the widespread adoption of UPI were examined and the paper listed that the UPI protocol does not have the same level of encryption as other payment systems making it more susceptible to cyber attacks. The lack of awareness pertaining to digital payment working methodologies acts as one barrier towards the adoption of these services. Another challenge is the interoperability problems between mobile wallets using UPI platform and banks facilitating the service.

4. **Digital Payment Fraud Detection Methods in digital ages and Industry 4.0**

Victor Chang, Le Minh Thao Doan, Alessandro Di Stefano, Zhili Sun and Giancarlo Fortino

Computers and Electrical Engineering,, Volume 100, 2022, 107734, ISSN 0045-7906, [9]

This research paper studies various fraud detection methods relevant and applicable in Industry 4.0. Various machine learning models were proposed and evaluated to detect fraudulent transactions and their effectiveness were compared.

As quoted in the paper 'one of the popular methods to classify payment transactions is to distinguish fraud using regular labels in database training, also known as supervised learning'. The supervised learning algorithms include k-nearest neighbours (KNN), decision trees,, logistic regression and support vector machine (SVM). Supervised learning models can label historical transactions and build a predictive fraud model.

The previous modelling mechanism consists of a time consuming step: the labelling and thus unsupervised models involving anomaly identification were also tested. Feature engineering and analysis were conducted in order to estimate the impact of features selection on

classification performance. Data skewness outcomes were also addressed by examining oversampling and undersampling techniques.

The methods of logistic regression, KNN, decision tree, autoencoder and random forest were evaluated among others and finally the integration undersampling method - NearMiss was found out to be the best fit which could improve the models' performance.

5. Wallet Payments Recent Potential Threats and Vulnerabilities with its possible security Measures

Mansi Bosamia , Dharmendra Patel

International Journal of Computer Sciences and Engineering Open Access Review Paper Vol.-7, Issue-1, Jan 2019 E-ISSN: 2347-2693 [10]

Wallet payment transactions have been previously evaluated primarily for security risks. This paper aims at examining some criteria of distributed processing including performance, scalability and availability in association with digital transactions. The researchers claim the success of mobile wallets to the features of anonymity, transferability and security.

The paper focuses on a threat model that can affect the basic components of a mobile wallet. These components that can be affected are categorised as follows:

- Application User Threats
- Device Threats
- Application Threats
- Merchant Threats
- Payment Service Provider Threats
- Acquirers Threats
- Payment Network Providers Threats
- Card Issuers Threats
- Mobile Payment Applications Providers Threats

Diving deeper into each component, the paper lists out some of the possible attacks and vulnerability prevention measures associated with them.

Application users are susceptible to phishing attacks, social engineering, unintentional installation of malware and unintentional operating system permission grants. Some other possible vulnerabilities listed include the usage of public WiFi, fake access points and fake websites.

Mobile devices can be targeted and put at risk by not setting a PIN/lock or setting one that is very weak. Mobile wallet applications can be hacked and reverse engineered to reveal hardcoded passwords and encrypted data.

Merchants can accidentally corrupt the system by uploading malware on Point of Sale(POS) contactless terminals. These terminals are also vulnerable to attacks such as Man-in-the-Middle attacks and relay attacks. Payment service providers may unintentionally compromise the running softwares, payment gateways or data connectivity. Token services and payment settlement services are also at risk of compromisation on the payment network provider's side.

Related to credit cards and issuers threats, the process of enrolment and authorization can be intercepted leading to payment fraud and data risks. The cardholder's sensitive data can also get leaked and ambiguity of transactions are likely to occur.

6. Financial Vulnerability, Financial Literacy, and the Use of Digital Payment Technologies

M. M. Naeser Seldal1 · Ellen K. Nyhus

Journal of Consumer Policy (2022) 45:281–306 Received: 2 August 2021 / Accepted: 27 January 2022 /Published online: 7 March 2022 [11]

The paper focuses on studying the various aspects of financial vulnerability and its correlation with various factors such as literacy rate, generation, gender and income. This study was conducted in response to the introduction of the new Payment Service Directive(PSD2) in the EU in 2019 which led to an increase in the competition and innovation in the payment services sector and the researchers wanted to compare the usage statistics with the demographics. The study was conducted on a sample of 2209 adult Norwegians who answered a questionnaire curated by the experts from the Consumer Finance Research Center(CFRC) in Rome, Italy. The results of the questionnaire were analysed using binary logistics regressions and appropriate conclusions drawn per test were as follows:

- Use/Nonuse of Payment Technologies and Attitudes towards Third Parties: The usage probability was found to be directly related to levels of financial literacy with an odds ratio of 1.052. Gender wise distribution also indicated that women were more likely to use it when compared to men. Mobile payment and contactless payment analysis also found similar results.

- Allowing Payment Providers Access to Account Data: The willingness to allow one's bank to share information was higher among younger generations when in comparison to older generations

- Willingness to Use a Social Media Company for Money Transfers: This probability increased with higher financial literacy scores with an odds ratio of 1.031. Gender wise demographics stated that women were less willing to engage in such transactions as opposed to men and the younger generations were more receptive than the older ones.

- Difficulty with Paying Bills: This probability was found to be inversely related to literacy rates and older generations were more susceptible.

- Use of Consumer Credit to Buy Consumable Goods: Consumer credit usage was found to be less among Baby Boomers and people from the Silent Generation when compared to Millennials and Generation Z. This probability was also directly proportional to the gross household income.

- Saved Enough for at Least Three Months of Expenses: A very similar trend to the previous scenario was observed indicating that the two older generations were more likely to save when compared with their younger counterparts.

- Difficulty with Paying Bills—by Generation Group: The relationship between financial literacy and difficulty in paying bills was found to be negative considering the fact that mobile payment users were less likely to have these difficulties.

7. A Review on Electronic Payments Security

*Md Arif Hassan, Zarina Shukur, Mohammad Kamrul Hasan * and Ahmed Salih Al-Khaleefa*

Center for Cyber Security, Faculty of Information Science and Technology, National University Malaysia (UKM),

Received: 12 May 2020; Accepted: 30 May 2020; Published: 12 August 2020 [12]

The faculty from UKM conducted a comprehensive review of 131 research articles published between 2010 and 2020 and on the topic electronic payments, security issues concerning them and security properties to be implemented. The aim was to enable electronic transaction providers to strengthen their security measures by boosting their security gaps.

The kind of research conducted varied over published journals, conferences/proceedings and industrial level research which lead to the conclusion that countries with the most discussion about digital payment in the banking sector included India, Malaysia, Nigeria, Bangladesh and so on with around 19 articles published pertaining to the said domain: 'E-wallet Design and requirement'.

Electronic payments have become vulnerable to security strikes due to alterations in web-based transactions and lack of protection measures. The insecure transmission of client sensitive information such as cards and payment account details demand improvement in information technology systems.

This research lead to the identification of six major security properties that are required to be implemented in safe digital payment systems:

- Confidentiality: the property that assures that the information is shared solely among authorised individuals or companies
- Integrity: the property that guarantees that information will not be replaced, harmed or tampered with in the course of a transaction
- Authentication: the property that requires a confirmation of identity before enabling services. This can be extended to Multi Factor Authentication involv9ing mechanisms such as biometry, token devices and smart cards
- Non Repudiation
- Robustness: the property that ensures greater productivity and efficiency
- Efficiency: the property that provides ensures the functionalities if transactions do not conflict with other mobile systems and non-payment functions
- Availability: then property which demands the service be available whenever the user/customer requires it.

8. Security of Mobile Payments and Digital Wallets

European Union Agency for Network and Information Security (ENISA). December 2016.[13]

This article examines the workflows of some major third-party mobile banking applications, discusses associated threats and also suggests a few recommendations to mitigate the identified threats.

Mobile payment applications that have been under this study include Apple Pay, Android Pay and Samsung Pay. The generalised workflow model can be said to comprise the following steps:

- Card enrolment
- User Authentication
- Device Authentication
- Data protection: which can include the following mechanisms
    - Tokenization
    - Secure Element generation
    - Encryption of credit/debit card information
    - NFC(Near Field Communication) control
    - Payment authorization localisation within NFC
    - Device fingerprinting

Keeping the risks and consequences at hand, the article suggests and lists a few security measures as the minimal requirements.

- Customers should actively check their Operating System and update frequently and should avoid engagement via untrusted networks
- The authentication mechanisms can be strengthened by enforcing stronger PINs/pattern locks
- Device configuration should be effective enough to wipe out remote data in case of loss or compromisation of said device
- Merchants should readily update their Point of Sale(POS) software and should be made aware of the risks associated with hardware tampering of POS.
- Sensitive information should not be hard coded as much as possible and anti-reversing techniques should be applied on payment protocols

In addition to the above measures, the article also stresses on how secure payment is the responsibility of service providers too including TSP(token service providers), Mobile OS providers who must continuously evaluate and update based on security issues identified. The applications should provide visibility to the customers regarding safety measures applied which encourages more widespread usage of these applications. Effective risk management programs should also be implemented to focus on mitigation of application risks and information threats.

9. **Assessing factors influencing consumers' non-adoption intention: Exploring the dark sides of mobile payment**

Rajat Kumar Beheraa, Pradip Kumar Balab, Nripendra P. Ranac

Information Technology & People, DOI: 10.1108/ITP-03-2022-0223. [14]

The popularity of mobile payment and adoption of this mechanism has led to the researchers conducting a study on the dark sides of mobile payment and some of the causes for less adoption. They evaluated factors such as usage, risk, tradition, innovation resistance and self-compassion which could moderate the relationship between innovation resistance and non-adoption intention.

The dark side of mobile payment was evaluated based on the following metrics:

- Usage threats: The presence of individuals who are accustomed to traditional payment methods and refuse to switch over could impact the usage of digital payment applications. In addition to this the lack of speedy Internet connections in developing countries could also act as a risk.
- Risk threats: Users are susceptible to phishing and social engineering attacks while devices are susceptible towards unauthorised access related fraud. Mobile applications are also at the risk of being reverse engineered. These threats can extend to merchants and other payment service providers.
- Value threats: The introduction of digital payment and acceptance could be hampered if the new technology does not substantially add to a value that existed.
- Image threats: Biases, prejudices, rumours among customers can result in a non adoption strategy. This challenge is especially prevalent in the older generation owing to the generation gap which could eliminate the usage of mobile payment strategies.
- Traditional threats: Similar to the previous threat, this arises as a result of norms, customs and family cultures which do not advocate the switch of traditional payment mechanisms
- Complexity threats: The complexity of usage an incorporation into daily routines are negatively impacting the adoption of this kind of payment
- Price threats: The equipment upgrade required in case the customer doesn't have one that currently supports is also considered as a threat.

The research landed at the conclusion that some of the dark sides of mobile payments do result in resistance towards adoption. Leaving out the price concerns, the other concerns based on the metrics were deemed relevant with 'risk threats' and 'image threats' being the leading factors.

10. Unified Payment Interface (UPI): A Digital Innovation and Its Impact on Financial Inclusion and Economic Development

Shailesh Rastogi, Chetan Panse, Arpita Sharma, Venkata Mrudula Bhimavarapu*

Symbiosis Institute of Business Management, Symbiosis International (Deemed University), Pune, India. *Received January 30, 2021; Revised April 13, 2021; Accepted June 4, 2021. Universal Journal of Accounting and Finance, Vol. 9, No. 3, pp. 518-530, 2021. DOI: 10.13189/ujaf.2021.090326.* [15]

This paper aims at studying the impact of UPI on economic development and its relation to financial literacy. The 3 hypotheses being tested in this paper are as follows:

- UPI impacts the Financial Literacy
- Financial Stability mediates the relationship between Literacy and financial Inclusion
- Trust mediates the relationship between Financial Inclusion and Economic Development

To administer this research a survey was conducted on 500 Maharashtrian participants who held accounts in the PMJDY (Pradhan Mantri Jan Dhan Yojana) since they would be familiar with and have access to some of the basic features of digital transactions and the SEM(Structural Equation Modelling) methodology was applied.

The results of the tests concluded that all the three hypotheses were accepted. This directly implies the contribution of UPI towards the growth of financial literacy, establishing UPI as one of the biggest innovations in digital banking. The major driving factors that can contribute towards the sustenance of UPI include the accessibility to the technology, the convenience, cost effectiveness and citizens voluntary exclusion in the financial inclusion. The second hypothesis concluded that financial inclusion and literacy are directly impacted and mediated by financial stability and the third hypothesis confirms the role trust plays in financial inclusion and economic development.

The study resulted in the following implications. Firstly, the government should provide platforms such as UPI and extend the accessibility towards a wider crowd. This accessibility should come with cost effectiveness and convenience.

The second implication states how building a supportive infrastructure to promote financial literacy is essential and managers are responsible for educating clients and maintaining transparency.

The third implication hints at how UPI and financial inclusion together will multiply the developmental effect and for this to take place stability and trust must be present which promote financial inclusion and further economic development.

PROPOSED SYSTEM

Existing authentication protocols in general utilise a 2-factor verification approach by employing otp based verification sent to the registered mobile number of a user. However, in case a user's phone has been stolen and it is unlocked, anyone can misutilise the phone to perform unauthorised financial transactions causing significant damage. In order to mitigate this glaring security breach it is imperative that we understand the current authentication approach in detail. Based on a detailed analysis of the existing authentication process, this paper explores the possibility of employing a 3-factor based authentication by adding additional Geo Tagging and GPS tracking on top of standard OTP authentication

Payment Authentication can happen through knowledge-based factor authentication i.e using information in possession of the user alone. The OTP service comes under this category. The knowledge-based factor authentication follows the Challenge-Handshake Authentication Protocol as shown in Fig. 1 [16].

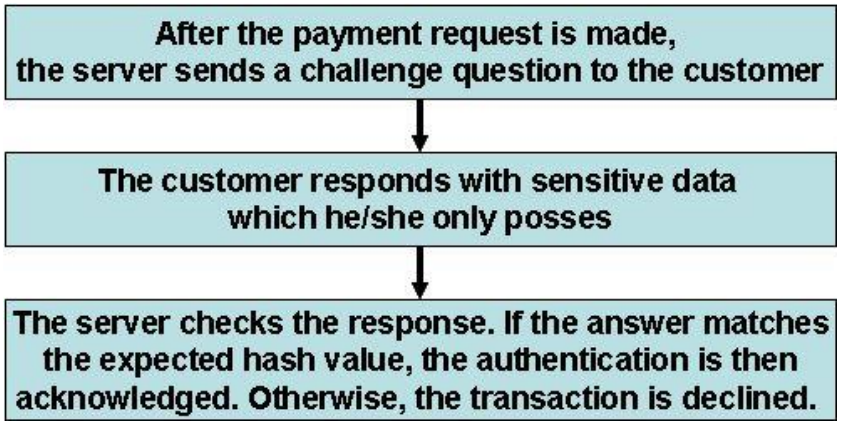


Fig.1: Challenge-Handshake Authentication Protocol

Another method of authentication is based on user location. Devices connected over the internet can pick up their geographic coordinates and can thus act as reliable location confirmation. Geolocation, though not directly used to verify the actual identity, can potentially detect a transaction in a completely different city or country which has a higher possibility of being a fraudulent one. This system is already employed in credit card authentications and this paper proposes to extend this towards digital payments via mobile applications.

Geotagging services are already used by banks to get insights into regional penetrations of digital payments and identify the scope for deploying additional payment touchpoints [17]. Also, GPS data can act as an additional layer of security by allowing banks to determine whether a transaction aligns with a particular user's location and if not allows them to respond effectively by shutting down fraudulent transactions. [18]

The proposed model employs GPS verification followed by OTP authentication which is implemented as follows. Firstly the user will initiate their transaction which prompts the application to verify the location before proceeding. This triggers the GPS tracking mechanism which tries to match the location coordinates of the device from where the transaction is taking place with the known/ previously stored location coordinates of the user. These locations are measured on the basis of latitudes and longitudes. If the coordinates match or are within a certain limit, say within 1 degree (or more based on the application and level of security), then the OTP is sent to the registered number and the transaction can proceed as per its original flow. If there is a discrepancy, the application will suspect that the device has been compromised and instead of sending the OTP to the registered number, it will be sent to an alternate number (which is taken during the time of registration and is encrypted). If this transaction is initiated by the user themselves, then they would have access to the alternate number and can complete it successfully.

The key factor to secure this flow of transactions is to securely encrypt the alternate number and utilise a highly trusted number for the same. OTP authorization can be further enhanced by incorporating a human-robot captcha verification step before sending the OTP.

MODULE DESCRIPTION

The proposed model focuses on implementing a 3-factor based authentication system with both GPS tracking and OTP verification which has the additional step of human/robot captcha based verification. The model can be described as containing the following modules

1. Bank: The interface on the bank's side through which the actual transactions take place. This module can include the functionalities of checking the balance and transaction initiation(sending money). The operations performed have an actual impact on the user's bank account and balance. Once the corresponding verifications are successful, the bank successfully terminates the transaction.

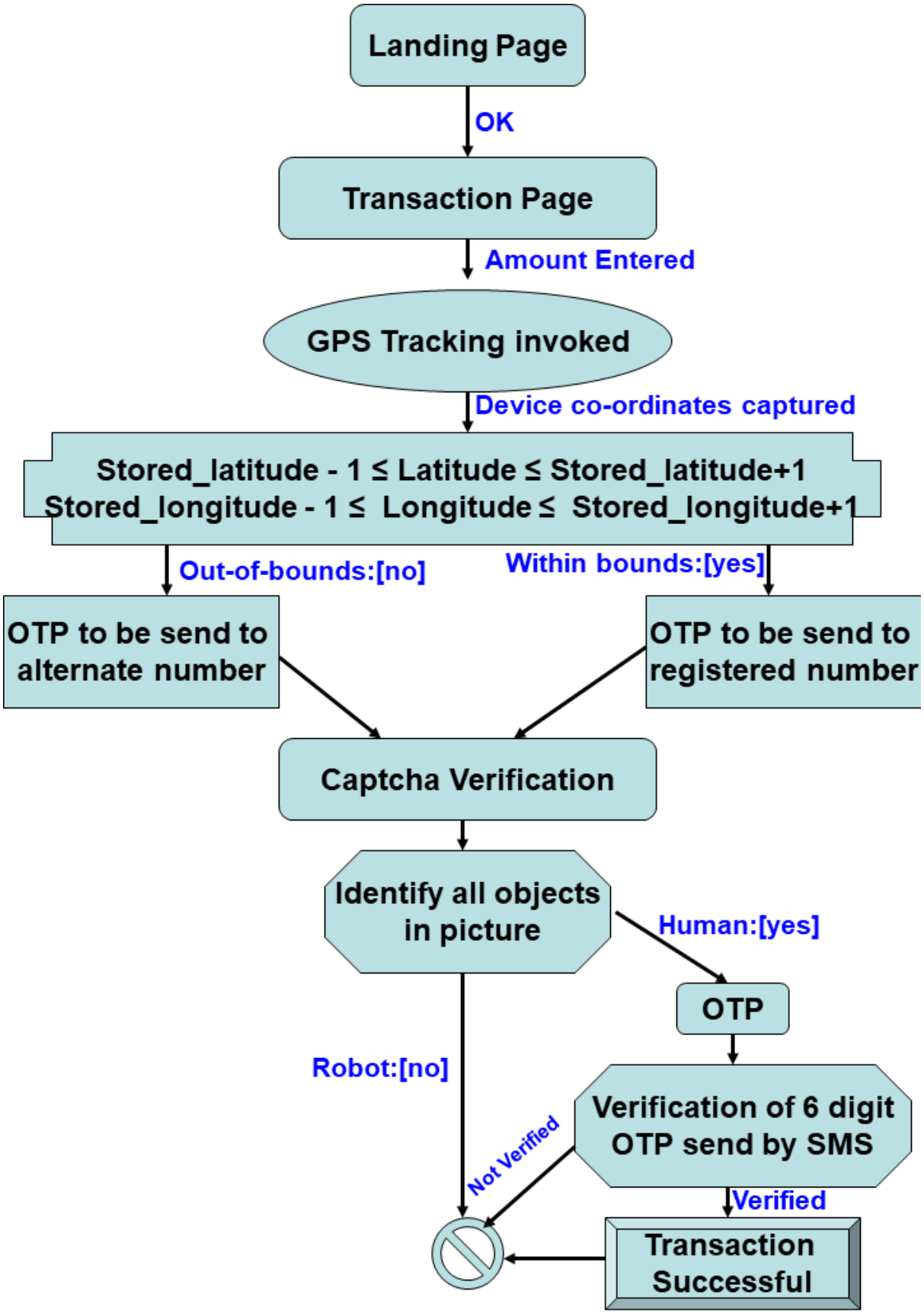
2. GPS: This module is responsible for tracking the GPS coordinates of the device from which the transaction is initiated. The attributes it stores include the previously known/registered location of the user and the current location of the transaction. Both these locations are stored in the form of 2 parameters: its latitude and longitude. This module employs the functionality of checking if the latitude/longitude is within bounds(within an assumed limit of 1 degree) and triggering the appropriate OTP functionality.

3. OTP: OTP stands for One-Time-Password. This module has 2 submodules - one to verify if the user is a human and the second to verify the unique 6 digit code sent to the user via sms

    3.1. Captcha: This module has a function that verifies if the user is a human or a robot. This is done by displaying a grid of random pictures and asking the user to humanely identify the embedded objects asked(eg: chimney, stairs etc..)Once the user correctly identifies the images containing the object, the system classifies the user as a human and lets them proceed with the transaction. This is a functionality that is highly deployed by many applications and acts as a layer of security against certain types of transactions.

    3.2. SMS OTP: This module has a function that generates a random string of six integers and sends it to the mobile number passed over from the GPS stage(users registered number if matched, alternate number otherwise). If the user enters the correct sequence of numbers, the transaction is allowed to proceed and blocked otherwise.

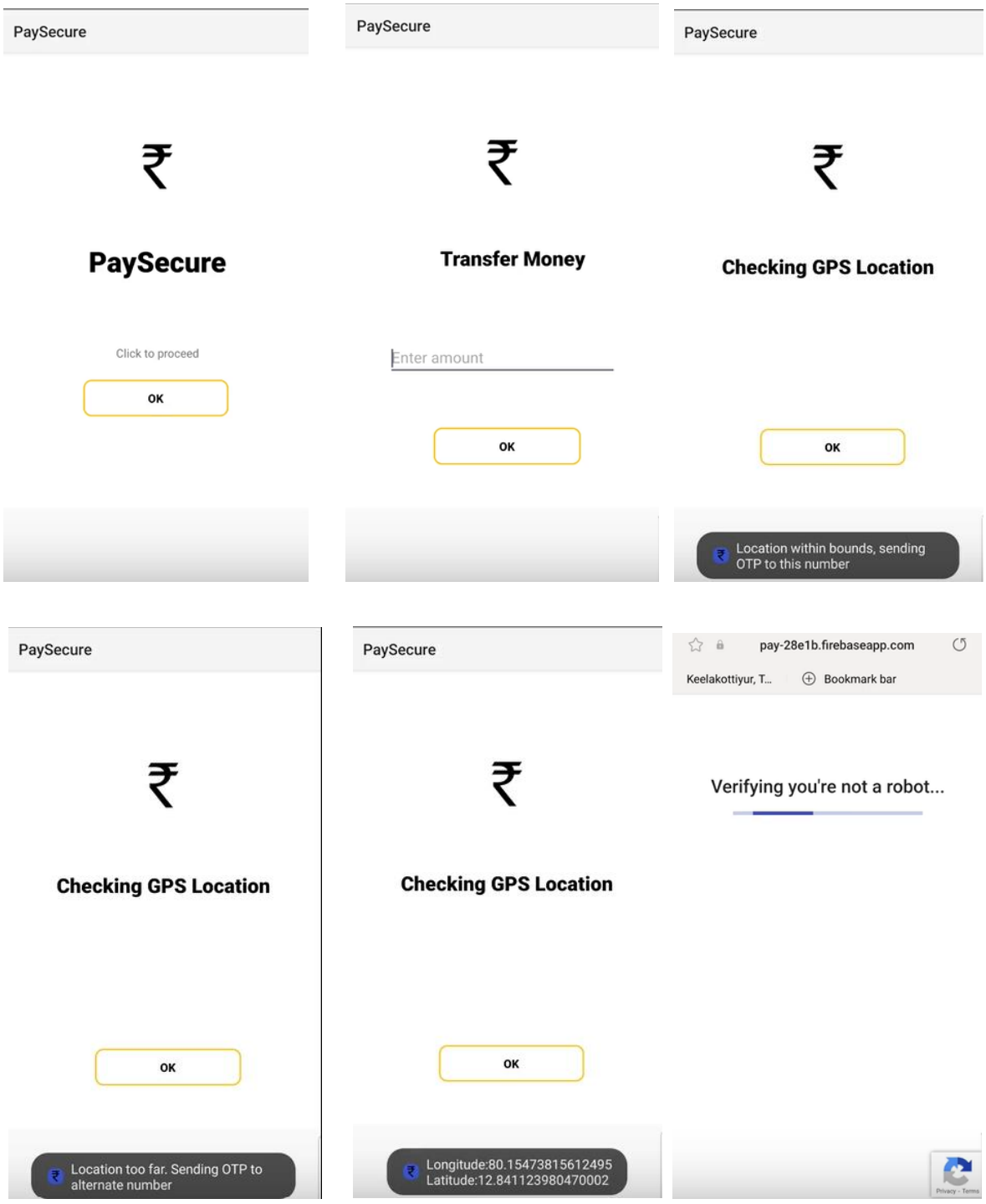
ARCHITECTURE DIAGRAM

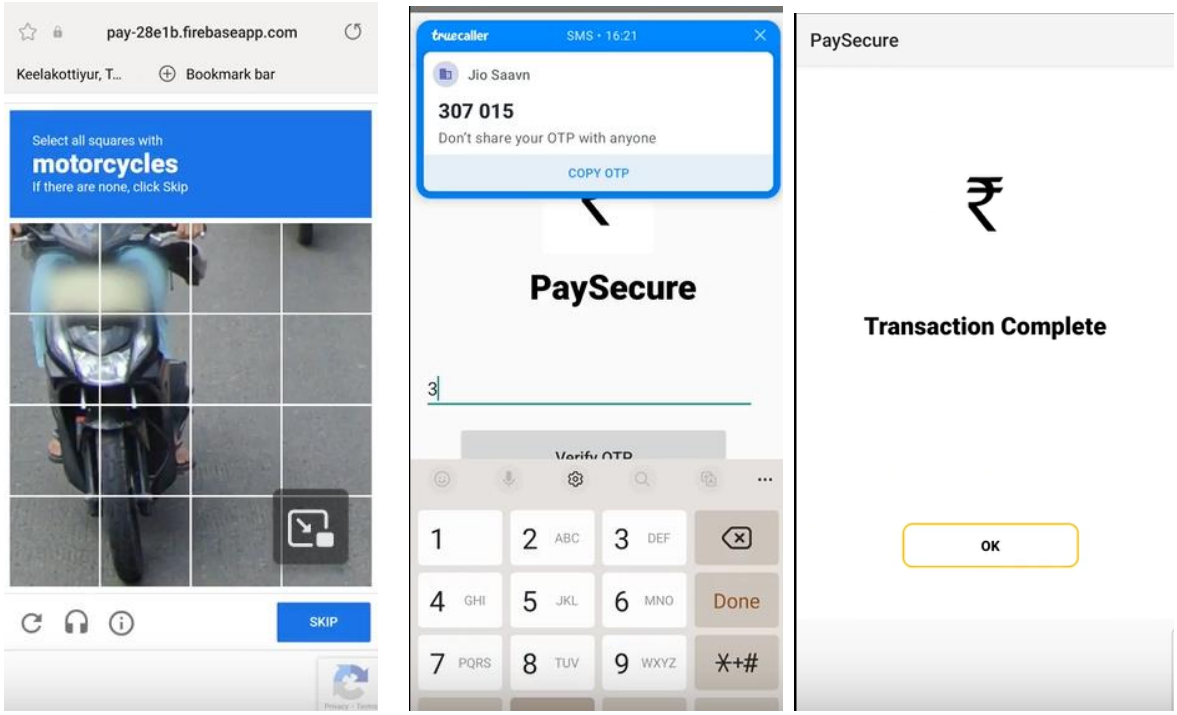


IMPLEMENTATION

To illustrate this workflow, a dummy application *PaySecure* has been created.
This application has been developed using android studio and the OTP sending and verification have been deployed using Firebase.[19] The location tracking has been implemented using the Android Location API and Location Manager [20]

The UI is minimalistic with more focus on functionality:





The implementation so far can be visualised through the following videos demonstrating the 2 cases:

1. When the user's device is identified to be within bounds, the OTP is sent to the same number
   [21]

2. When the user's device is identified to be out of bounds, the OTP is sent to the alternative number
   [22]

In case the OTP entered is incorrect, the transaction does not proceed ahead [23]

SAMPLE CODE

The sample application is coded in Java and contains the following classes:

- MainActivity: which is the starting page of the application from where the flow is initiated
- Transfer: where the amount entered by the user is captured
- GPSTrack: where the location constraints of the user are examined and appropriate phone number is chosen for OTP
- OTP: where the captcha and OTP are sent to the user for verification
- HomePage: the page displayed after successful transaction

MainActivity.java

```
package com.example.paysecure;

import android.content.Intent;
import android.view.View;
import android.widget.Button;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    Button b1;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        b1 = findViewById(R.id.page1);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(MainActivity.this, Transfer.class);
                startActivity(i);
            }
        });
    }
}
```

Transfer.java

```
package com.example.paysecure;

import android.content.Intent;
import android.view.View;

import android.widget.Button;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
```

```
public class Transfer extends AppCompatActivity {

    Button submit;
    EditText amount;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_transfer);
        submit = findViewById((R.id.notif));
        amount = findViewById(R.id.amount);
        submit.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View view) {
                Intent i = new Intent(Transfer.this, GPSTrack.class);
                startActivity(i);
            }
        });

    }
}
```

GPSTrack.java

```
package com.example.paysecure;

import static android.Manifest.permission.ACCESS_COARSE_LOCATION;
import static android.Manifest.permission.ACCESS_FINE_LOCATION;

import android.annotation.TargetApi;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.IBinder;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;
import java.util.List;

public class GPSTrack extends AppCompatActivity {


    private ArrayList permissionsToRequest;
    private ArrayList permissionsRejected = new ArrayList();
    private ArrayList permissions = new ArrayList();

    String phone_number;
    private final static int ALL_PERMISSIONS_RESULT = 101;
    LocationTrack locationTrack;

    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_gpstrack);

        permissions.add(ACCESS_FINE_LOCATION);
        permissions.add(ACCESS_COARSE_LOCATION);

        permissionsToRequest = findUnAskedPermissions(permissions);
        //get the permissions we have asked for before but are not
granted..
        //we will store this in a global list to access later.


        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {


            if (permissionsToRequest.size() > 0)
                requestPermissions((String[])
permissionsToRequest.toArray(new String[permissionsToRequest.size()]),
ALL_PERMISSIONS_RESULT);
        }


                locationTrack = new LocationTrack(GPSTrack.this) {
                    @Override
                    public void onLocationChanged(@NonNull List<Location>
locations) {
                        super.onLocationChanged(locations);
                    }

                    @Override
                    public void onFlushComplete(int requestCode) {
                        super.onFlushComplete(requestCode);
                    }

                    @Override
                    public void onStatusChanged(String s, int i, Bundle
bundle) {

                    }

                    @Nullable
                    @Override
                    public IBinder onBind(Intent intent) {
                        return null;
                    }
                };


                if (locationTrack.canGetLocation()) {


                    double longitude = locationTrack.getLongitude();
                    double latitude = locationTrack.getLatitude();

                    validate_location(longitude, latitude);
                    Toast.makeText(getApplicationContext(), "Longitude:" +
Double.toString(longitude) + "\nLatitude:" + Double.toString(latitude),
Toast.LENGTH_SHORT).show();
                } else {
```

```
                    locationTrack.showSettingsAlert();
                }

            }

    private void validate_location(double longitude, double latitude) {
        double control_longitute = 80.15741256996989;
        double control_latitude = 12.842172933742404;
        if (longitude >= control_longitute + 1 || longitude <=
control_longitute - 1)
        {
            //phone_number = "7972610132";
            phone_number = "9663399805";
            Toast.makeText(getApplicationContext(), "Location too far.
Sending OTP to alternate number", Toast.LENGTH_SHORT).show();
        }
        else if (longitude <= control_longitute - 1 || latitude <=
control_latitude - 1) {
            //phone_number = "7972610132";
            phone_number = "9663399805";
            Toast.makeText(getApplicationContext(), "Location too far.
Sending OTP to alternate number", Toast.LENGTH_SHORT).show();
        }
        else {
            phone_number = "9353358462";
            Toast.makeText(getApplicationContext(), "Location within
bounds, sending OTP to this number", Toast.LENGTH_SHORT).show();

        }
        new Handler().postDelayed(() -> {
            Intent i = new Intent(this, OTP.class);
            i.putExtra("message_key", phone_number);
            startActivity(i);
            finish();
        }, 3000);
    }


    private ArrayList findUnAskedPermissions(ArrayList wanted) {
        ArrayList result = new ArrayList();

        for (Object perm : wanted) {
            if (!hasPermission(perm)) {
                result.add(perm);
            }
        }

        return result;
    }

    private boolean hasPermission(Object permission) {
        if (canMakeSmores()) {
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                return (checkSelfPermission(permission.toString()) ==
PackageManager.PERMISSION_GRANTED);
            }
        }
        return true;
    }

    private boolean canMakeSmores() {
```

```
        return (Build.VERSION.SDK_INT > Build.VERSION_CODES.LOLLIPOP_MR1);
    }


    @TargetApi(Build.VERSION_CODES.M)
    @Override
    public void onRequestPermissionsResult(int requestCode, String[]
permissions, int[] grantResults) {

        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);
        switch (requestCode) {

            case ALL_PERMISSIONS_RESULT:
                for (Object perms : permissionsToRequest) {
                    if (!hasPermission(perms)) {
                        permissionsRejected.add(perms);
                    }
                }

                if (permissionsRejected.size() > 0) {


                    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                        if
(shouldShowRequestPermissionRationale(permissionsRejected.get(0).toString()
)) {
                            showMessageOKCancel("These permissions are
mandatory for the application. Please allow access.",
                                    new DialogInterface.OnClickListener() {
                                        @Override
                                        public void onClick(DialogInterface
dialog, int which) {
                                            if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.M) {

requestPermissions((String[]) permissionsRejected.toArray(new
String[permissionsRejected.size()]), ALL_PERMISSIONS_RESULT);
                                            }
                                        }
                                    });
                            return;
                        }
                    }

                }

                break;
        }

    }

    private void showMessageOKCancel(String message,
DialogInterface.OnClickListener okListener) {
        new AlertDialog.Builder(GPSTrack.this)
                .setMessage(message)
                .setPositiveButton("OK", okListener)
                .setNegativeButton("Cancel", null)
                .create()
                .show();
    }
```

```
    @Override
    protected void onDestroy() {
        super.onDestroy();
        locationTrack.stopListener();
    }
}
```

OTP.java

```
package com.example.paysecure;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.android.gms.tasks.TaskExecutors;
import com.google.firebase.FirebaseException;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.PhoneAuthCredential;
import com.google.firebase.auth.PhoneAuthOptions;
import com.google.firebase.auth.PhoneAuthProvider;

import java.util.concurrent.TimeUnit;

public class OTP extends AppCompatActivity {

    // variable for FirebaseAuth class
    private FirebaseAuth mAuth;

    // variable for our text input
    // field for phone and OTP.
    private EditText  edtOTP;

    // buttons for generating OTP and verifying OTP
    private Button verifyOTPBtn, generateOTPBtn;

    // string for storing our verification ID
    private String verificationId, phone_no;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_otp);

        // below line is for getting instance
        // of our FirebaseAuth.
        mAuth = FirebaseAuth.getInstance();

        // initializing variables for button and Edittext.
        Intent i = getIntent();
```

```
        phone_no = i.getStringExtra("message_key");
        edtOTP = findViewById(R.id.idEdtOtp);
        verifyOTPBtn = findViewById(R.id.idBtnVerify);
        //generateOTPBtn = findViewById(R.id.idBtnGetOtp);


        // send OTP method for getting OTP from Firebase.
        String phone = "+91" + phone_no;
        sendVerificationCode(phone);




        // initializing on click listener
        // for verify otp button
        verifyOTPBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // validating if the OTP text field is empty or not.
                if (TextUtils.isEmpty(edtOTP.getText().toString())) {
                    // if the OTP text field is empty display
                    // a message to user to enter OTP
                    Toast.makeText(OTP.this, "Please enter OTP",
Toast.LENGTH_SHORT).show();
                } else {
                    // if OTP field is not empty calling
                    // method to verify the OTP.
                    verifyCode(edtOTP.getText().toString());
                }
            }
        });
    }

    private void signInWithCredential(PhoneAuthCredential credential) {
        // inside this method we are checking if
        // the code entered is correct or not.
        mAuth.signInWithCredential(credential)
                .addOnCompleteListener(new OnCompleteListener<AuthResult>()
{
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task)
{
                        if (task.isSuccessful()) {
                            // if the code is correct and the task is
successful
                            // we are sending our user to new activity.
                            Intent i = new Intent(OTP.this,
HomePage.class);
                            startActivity(i);
                            finish();
                        } else {
                            // if the code is not correct then we are
                            // displaying an error message to the user.
                            Toast.makeText(OTP.this,
task.getException().getMessage(), Toast.LENGTH_LONG).show();
                        }
                    }
                });
    }
```

```
    private void sendVerificationCode(String number) {
        // this method is used for getting
        // OTP on user phone number.
        PhoneAuthOptions options =
                PhoneAuthOptions.newBuilder(mAuth)
                        .setPhoneNumber(number)            // Phone number
to verify
                        .setTimeout(60L, TimeUnit.SECONDS) // Timeout and
unit
                        .setActivity(this)                 // Activity (for
callback binding)
                        .setCallbacks(mCallBack)           //
OnVerificationStateChangedCallbacks
                        .build();
        PhoneAuthProvider.verifyPhoneNumber(options);
    }

    // callback method is called on Phone auth provider.
    private PhoneAuthProvider.OnVerificationStateChangedCallbacks

            // initializing our callbacks for on
            // verification callback method.
            mCallBack = new
PhoneAuthProvider.OnVerificationStateChangedCallbacks() {

        // below method is used when
        // OTP is sent from Firebase
        @Override
        public void onCodeSent(String s,
PhoneAuthProvider.ForceResendingToken forceResendingToken) {
            super.onCodeSent(s, forceResendingToken);
            // when we receive the OTP it
            // contains a unique id which
            // we are storing in our string
            // which we have already created.
            verificationId = s;
        }

        // this method is called when user
        // receive OTP from Firebase.
        @Override
        public void onVerificationCompleted(PhoneAuthCredential
phoneAuthCredential) {
            // below line is used for getting OTP code
            // which is sent in phone auth credentials.
            final String code = phoneAuthCredential.getSmsCode();

            // checking if the code
            // is null or not.
            if (code != null) {
                // if the code is not null then
                // we are setting that code to
                // our OTP edittext field.
                edtOTP.setText(code);

                // after setting this code
                // to OTP edittext field we
                // are calling our verifycode method.
                verifyCode(code);
            }
        }
```

```
        // this method is called when firebase doesn't
        // sends our OTP code due to any error or issue.
        @Override
        public void onVerificationFailed(FirebaseException e) {
            // displaying error message with firebase exception.
            Toast.makeText(OTP.this, e.getMessage(),
Toast.LENGTH_LONG).show();
        }
    };

    // below method is use to verify code from Firebase.
    private void verifyCode(String code) {
        // below line is used for getting
        // credentials from our verification id and code.
        PhoneAuthCredential credential =
PhoneAuthProvider.getCredential(verificationId, code);

        // after getting credential we are
        // calling sign in method.
        signInWithCredential(credential);
    }
}
```

HomePage.java

```
package com.example.paysecure;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class HomePage extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home_page);
    }
}
```

The design for each page is an xml file containing similar contents to the one pasted below:

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@color/white"
        android:backgroundTintMode="add"
        android:forceHasOverlappingRendering="false"
        android:gravity="center_horizontal"
```

```
        android:orientation="vertical"
        android:theme="@style/Base.Theme.PaySecure">

        <ImageView
            android:id="@+id/img_send_otp"
            android:layout_width="120dp"
            android:layout_height="120dp"
            android:layout_marginTop="90dp"
            android:src="@drawable/rupee" />


        <androidx.constraintlayout.widget.ConstraintLayout
            android:id="@+id/frameLayout2"
            android:layout_width="match_parent"
            android:layout_height="388dp"
            tools:layout_editor_absoluteY="210dp">

            <Button
                android:id="@+id/page1"
                android:layout_width="191dp"
                android:layout_height="wrap_content"
                android:layout_marginStart="108dp"
                android:layout_marginBottom="120dp"
                android:background="@drawable/backgroundbox"
                android:text="OK"
                android:textColor="@color/colortextfirst"
                android:textStyle="bold"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintStart_toStartOf="parent" />

            <ProgressBar
                android:id="@+id/probar1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:visibility="gone"
                app:layout_constraintBottom_toTopOf="parent"
                app:layout_constraintEnd_toStartOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="175dp"
                android:layout_marginBottom="194dp"
                android:gravity="center"
                android:text="Click to proceed"
                android:textColor="@color/colortextsecond"
                android:textSize="14sp"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintTop_toTopOf="parent"
                tools:ignore="MissingConstraints"
                tools:layout_editor_absoluteX="-16dp" />

            <TextView
                android:layout_width="251dp"
                android:layout_height="83dp"
                android:layout_marginStart="118dp"
                android:layout_marginTop="53dp"
                android:layout_marginEnd="119dp"
                android:layout_marginBottom="280dp"
```

```
                android:fontFamily="sans-serif-black"
                android:text="PaySecure"
                android:textAlignment="center"
                android:textColor="@color/colortextfirst"
                android:textSize="34sp"
                android:textStyle="bold"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent"
                tools:ignore="MissingConstraints" />


        </androidx.constraintlayout.widget.ConstraintLayout>
    </LinearLayout>
</ScrollView>
```

RESULTS AND DISCUSSION

The proposed workflow has been implemented through the dummy application *PaySecure*. The application functions by taking in the user's input of a certain transaction amount. This triggers the GPS tracking mechanism which verifies the location of the transaction. In case the location is too far off, the transaction is suspected to be a fraudulent one and the subsequent step of sending the OTP is affected, i.e the number the code is sent to is the alternate number. Successful verification of the OTP leads to a successful and safe transaction.

The usage of GPS tracking in combination to the existing 2-factor authentication definitely acts as an additional layer of security. This combats the situations in which a person's device is lost when their account details are compromised. Addressing the query posed in the paper[6] , we can say that the attacker wouldn't be able to commit a fraud even with the access to the victim's messages (through the READ_SMS permission) given that the attack is from a significantly farther location.

This methodology has its limitations some of them being the complicated workflow which may frustrate the user, the compromise that can happen if the attacker gets a hold of the alternative phone number and the scope of attack within the geographic limits set up.

CONCLUSION

In this paper, digital payment security and the associated risks have been studied and explored. Implementing a GPS based 3-factor authentication system, we have proposed an alternative workflow for payment applications that can be employed. The inclusion of GPS based tracking can definitely elevate the security of a digital payment application. The associated benefits and limitations have been discussed.

The future scope of this research can focus on integrating / implementing a solution based on the in – built 'Find My Phone' application in both Android and iOS devices. Additionally, biometric based security measures can also be studied and employed.

REFERENCES

[1] https://www.npci.org.in/what-we-do/upi/product-overview

[2]NPCI UPI Product Statistics: https://www.npci.org.in/what-we-do/upi/product-statistics

[3]https://www.hindustantimes.com/india-news/over-95-000-upi-fraud-cases-reported-in-2022-23-centre-in-parliament-101679541121388.html

[4]Common Types of UPI Frauds and how to avoid them: https://www.bqprime.com/personal-finance/pfx-common-types-of-upi-frauds-and-how-to-stay-safe

[5]UPI Payments: Top Security Issues and ways around: https://community.nasscom.in/communities/digital-transformation/fintech/upi-payments-top-security-issues-possible-way-arounds-an-opinion.html

[6]Renuka Kumar, Sreesh Kishore, Hao Lau University of Michigan. Security Analysis of Unified Payments Interface and Payment Apps in India. Included in the Proceedings of the 29th USENIX Security Symposium August 12–14, 2020. https://www.usenix.org/conference/usenixsecurity20/presentation/kumar

[7]Kshitiz Sharma (IIT Kharagpur), Nandini Bajaj (IIT Kharagpur), Xinru Page (Brigham Young University), Mainak Mondal(IIT Kharagpur). A Platform for Uncovering Indian Users' Decision-Making Process in Unified Payment Interface (UPI) Apps https://www.usenix.org/system/files/soups2022-poster62_sharma_abstract_final.pdf

[8]Dr.A.Shaji George1, A.S.Hovan George2, Dr.T.Baskar3, A.S.Gabrio Martin4. 1,4Masters IT Solutions, Chennai, Tamil Nadu, India. 2Student, Tbilisi State Medical University, Tbilisi, Georgia. 3Professor, Department of Physics, Shree Sathyam College of Engineering and Technology, Sankari Taluk, Tamil Nadu, India. An Overview of India's Unified Payments Interface (UPI): Benefits, Challenges, and Opportunities. Partners Universal International Research Journal (PUIRJ)Volume: 02 Issue: 01 | January-March 2023 | ISSN: 2583-5602 https://zenodo.org/record/7723154

[9]Victor Chang1*, Le Minh Thao Doan2 Alessandro Di Stefano2 Zhili Sun3and Giancarlo Fortino4 1. Department of Operations and Information Management, Aston Business School, Aston University, Birmingham, UK 2. Cybersecurity, Information Systems and AI Research Group, School of Computing, Engineering and Digital Technologies, Teesside University, Middlesbrough, UK 3. Institute for Communication Systems (ICS), 5G&6G Innovation Centre, University of Surrey, Guildford, Surrey, UK 4. Department of Informatics, Modeling, Electronics and Systems (DIMES) of the University of Calabria (Unical), Rende (CS), Italy.Digital Payment Fraud Detection Methods in digital ages and Industry 4.0. Computers and Electrical Engineering,, Volume 100, 2022, 107734, ISSN 0045-7906, https://www.sciencedirect.com/science/article/abs/pii/S0045790622000465?via%3Dihub

[10]Mansi Bosamia & Dharmendra Patel, 1,2 Smt. Chandaben Mohanbhai Patel Institute of Computer Applications, CHARUSAT, Changa, Gujarat, India Wallet Payments Recent Potential Threats and Vulnerabilities with its possible security Measures. https://www.researchgate.net/publication/332113623_Wallet_Payments_Recent_Potential_Threats_and_Vulnerabilities_with_its_possible_security_Measures

[11]M. M. Naeser Seldal & Ellen K. Nyhus Financial Vulnerability, Financial Literacy, and the Use of Digital Payment Technologies https://link.springer.com/article/10.1007/s10603-022-09512-9

[12]Md Arif Hassan A Review on Electronic Payments Security https://www.researchgate.net/publication/343598898_A_Review_on_Electronic_Payments_Security
[13]European Union Agency for Network and Information Security (ENISA). December 2016. Security of Mobile Payments and Digital Wallets - ENISA https://www.enisa.europa.eu/publications/mobile-payments-security/at_download/fullReport
[14]Rajat Kumar Beheraa,*, Pradip Kumar Balab, Nripendra P. Ranac. Information Technology & People, DOI: 10.1108/ITP-03-2022-0223. Assessing factors influencing consumers' non-adoption intention: Exploring the dark sides of mobile payment. https://www.emerald.com/insight/content/doi/10.1108/ITP-03-2022-0223/full/html
[15]Shailesh Rastogi*, Chetan Panse, Arpita Sharma, Venkata Mrudula Bhimavarapu. Symbiosis Institute of Business Management, Symbiosis International (Deemed University), Pune, India. Received January 30, 2021; Revised April 13, 2021; Accepted June 4, 2021. Universal Journal of Accounting and Finance, Vol. 9, No. 3, pp. 518-530, 2021. DOI: 10.13189/ujaf.2021.090326. Unified Payment Interface (UPI): A Digital Innovation and Its Impact on Financial Inclusion and Economic Development. https://www.hrpub.org/journals/article_info.php?aid=11076
[16]Payment Authentication, Ebanx https://business.ebanx.com/en/resources/payments-explained/payment-authentication
[17]Geotagging of payment system touchpoints and its benefits - The Hindu BusinessLine https://www.thehindubusinessline.com/blexplainer/bl-explainer-geotagging-of-payment-system-touchpoints-and-its-benefits/article65302108.ece
[18]Securing FinTech Apps With GPS Data – Velmie. https://www.velmie.com/post/securing-fintech-apps-with-gps-data
[19] https://www.geeksforgeeks.org/firebase-authentication-with-phone-number-otp-in-android/
[20] https://www.digitalocean.com/community/tutorials/android-location-api-tracking-gps
[21]Transaction with registered number (Demo) https://drive.google.com/file/d/1iBYMO5AhPV6od3VykEZdoQD8MnALMdzz/view?usp=sharing
[22]Transaction with alternate number (Demo) https://drive.google.com/file/d/1ANw8TLHnKrNFjs6GFxNWLFC5jiA4HiFP/view?usp=sharing
[23]OTP wrong (Demo) https://drive.google.com/file/d/1dq-MlIXQbwee0zf8-WbuZrfnX_t9jIN6/view?usp=sharing