*Paycheck: Online Payment Analysis and Real-Time Feedback*

Upendra Kadre   
*Computer Engineering*  
*Vishwakarma Institute of Technology*Pune, India  
upendra.kadre17@vit.edu

Saurav Kendre  
*Computer Engineering*  
*Vishwakarma Institute of Technology*Pune, India  
saurav.kendre17@vit.eduYash Sali  
*Computer Engineering*  
*Vishwakarma Institute of Technology*Pune, India  
yash.sali17@vit.edu

Prof. Virendra Pawar  
*Computer Engineering*  
*Vishwakarma Institute of Technology*Pune, India  
virendra.pawar@vit.edu Jay Maniyar  
*Computer Engineering*  
*Vishwakarma Institute of Technology*Pune, India  
jay.maniyar17@vit.edu

*Abstract*—This project is an online payment analysis and feedback system which works for reducing the thefts in online payment. A mobile application fetches the payment data from the notification of the payment app and feeds it to raspberry pi which works on the string received and displays it on the screen. The string which is displayed is appended to the ‘csv’ file which is mailed to the user whenever required.

# Introduction

Usually in any store where online payment method is available, there is always a threat of payment fraud. A customer can show the screenshot of previously paid receipt to the owner and the owner being in a hussle due to the crowd does not have much time to cross verify the payment. This can lead to a great amount of loss if such thing happens regularly. The owner or the cashier does not usually cross check in his mobile if the payment is received or not. But this project rectifies this issue by using Internet of Things (IoT) and a simple android application. The working starts with the android device, when the user receives a payment from any method, he receives a push notification or a sms. The android application fetches the notification and feeds the name of the customer, the amount paid and the method of payment to raspberry pi. Raspberry pi puts the received string into an array list. This array is then displayed on the screen with each string received as a label. Each label is assigned a number to it. We use a 4x4 matrix keypad, which deletes the entry from the screen. The entry is deleted when the user acknowledges that the payment is done and presses the label number of that particular payment on the keypad.

# MODULES

This project works in between components. Each component has its own unique function. As the project works in unison with the mobile device and raspberry pi, it is divided into different modules.

## Mobile Application:

The basic function of the mobile application (client) is to fetch the payment data and to feed it to raspberry pi (server). For this, the application scans the incoming text messages and the push notifications for the payment data and sends it to raspber ry pi.

## Display:

The String containing the name of the payer and the amount paid is now in the raspberry pi. This string is displayed on the screen in REAL TIME.

## Convert to csv:

The displayed payer details are then saved into a csv file. The file contains the all the data and can be created whenever the user wants it.

## Mailing:

The csv file which was created in the previous module is sent to the user whenever asked for using the SMTP module.

# PLATFORMS

The two platforms used in this project are android studio and raspberry pi. The first part of the project is android studio. It is the official integrated development environment (IDE) for Android application development. It is based on the IntelliJ IDEA, a Java integrated development environment for software, and incorporates its code editing and tools. The official language for Android development is Java. Large parts of Android are written in Java and its APIs are designed to be called primarily from Java. However it is possible to develop C and C++ app using the Android Native Development Kit (NDK). To support application development within the Android operating system, Android Studio uses a Gradle-based build system, emulator, code templates, and Github integration. Every project in Android Studio has one or more modalities with source code and resource files. These modalities include Android app modules, Library modules, and Google App Engine modules.

Android Studio uses an Instant Push feature to push code and resource changes to a running application. A code editor assists the developer with writing code and offering code completion, refraction, and analysis. Applications built in Android Studio are then compiled into the APK format for submission to the Google Play Store.

The software was first announced at Google I/O in May 2013, and the first stable build was released in December 2014. Android Studio is available for Mac, Windows, and Linux desktop platforms. It replaced Eclipse Android Development Tools (ADT) as the primary IDE for Android application development. Android Studio and the Software Development Kit can be downloaded directly from Google.

Raspberry Pi is the name of a series of single-board computers made by the Raspberry Pi Foundation, a UK charity that aims to educate people in computing and create easier access to computing education.

The Raspberry Pi launched in 2012, and there have been several iterations and variations released since then. The original Pi had a single-core 700MHz CPU and just 256MB RAM, and the latest model has a quad-core 1.4GHz CPU with 1GB RAM.

All over the world, people use Raspberry Pis to learn programming skills, build hardware projects, do home automation, and even use them in industrial applications.

The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins that allow you to control electronic components for physical computing and explore the Internet of Things (IoT).

There have been three generations of Raspberry Pis: Pi 1, Pi 2, and Pi 3, and there has generally been a Model A and a Model B of most generations. Model A is a cheaper variant and tends to have reduced RAM and ports like USB and Ethernet. The Pi Zero is a spinoff of the original (Pi 1) generation, made even smaller and cheaper. The model used here is Pi 3 Model B+.

The Raspberry Pi operates in the open source ecosystem: it runs Linux (a variety of distributions), and its main supported operating system, Raspbian, is open source and runs a suite of open source software. The Raspberry Pi Foundation contributes to the Linux kernel and various other open source projects as well as releasing much of its own software as open source.

*Java Swing* is a part of Java Foundation Classes (JFC) that is *used to create window-based applications*. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java. Unlike AWT, Java Swing provides platform-independent and lightweight components. The Java Foundation Classes (JFC) are a set of GUI components which simplify the development of desktop applications. Unlike AWT components, Swing components are not implemented by platform-specific code. Instead, they are written entirely in Java and therefore are platform-independent. Additionally, this framework provides a layer of abstraction between the code structure and graphic presentation of a Swing-based GUI. Since early versions of Java, a portion of the Abstract Window Toolkit (AWT) has provided platform-independent APIs for user interface components. In AWT, each component is rendered and controlled by a native peer component specific to the underlying windowing system. Much of the Swing API is generally a complementary extension of the AWT rather than a direct replacement. In fact, every Swing lightweight interface ultimately exists within an AWT heavyweight component because all of the top-level components in Swing extend an AWT top-level container.

By contrast, Swing components are often described as *lightweight* because they do not require allocation of native resources in the operating system's windowing toolkit. The AWT components are referred to as *heavyweight components*.[[13]](https://en.wikipedia.org/wiki/Swing_(Java)#cite_note-13)

*CSV* stands for ‘*comma-separated values*‘. CSV files are mostly used for creating data files either for exporting the data OR importing the data. Java language does not provide any native support for effectively handling CSV files. Without using 3rd party libraries, we may end up creating your own *CSV parser*. There is usually no advantage in re-inventing the wheel, So it is advisable to use such 3rd party tools for parsing the CSV files.

*OpenCSV* is such a tool which can be used to read a csv file in java or write data to CSV file. Here we use CSVWriter class of OpenCSV module. CSVWriter a customizable module where we can use custom separator, custom quote character or custom line terminator while writing a CSV file using java application code. The CSVWriter module separates the received string into words by detecting a blank space. These separated words are appended into a .csv file. The .csv file is cleared and reset when the java server restarts.

The basic file-handling for creating .csv file and mailing service is done on raspberry pi.

An important module that is used is *smtplib* in python. Python provides *smtplib* module, which defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon. An SMTP object has an instance method called sendmail, which is typically used to do the work of mailing a message. It takes three parameters −

* The *sender* − A string with the address of the sender.
* The *receivers* − A list of strings, one for each recipient.
* The *message* − A message as a string formatted as specified in the various RFCs.

An e-mail requires a *From*, *To*, and *Subject* header, separated from the body of the e-mail with a blank line. To send the mail we use *smtpObj* to connect to the SMTP server on the local machine and then use the *sendmail* method along with the message, the from address, and the destination address as parameters. If we are not running an SMTP server on a local machine, we use *smtplib* client to communicate with a remote SMTP server.

To send an e-mail with mixed content requires to set *Content-type* header to *multipart/mixed*. Then, text and attachment sections can be specified within *boundaries.*

A boundary is started with two hyphens followed by a unique number, which cannot appear in the message part of the e-mail. A final boundary denoting the e-mail's final section must also end with two hyphens.

Attached files should be encoded with the *pack("m")* function to have base64 encoding before transmission.

# WORKING

When an amount is paid to the user, the user gets either a push notification or a text message. The android application then scans the input message for payment related words like credited, debited, paid, paytm, etc. If any of such word is found in the message, the message is passed to the application in form of list by the *onNotificationPosted*function which brings out the incoming notifications into the home screen of the notification reader application.*CustomList adapter Function* performs up the storing of the notification from the notifications tab in form of a combined list and then passes it on to the next function button present up on the R-pi listener which lets it function out to make the output string present on the r -pi module and lets it get processed. A connection between the android device and raspberry pi is established using socket programming. For sending the data to the raspberry pi, the IP address and the java server port are given to the application. If the connection is established, the string that is passed into the application is sent to the raspberry pi. The received string is further parsed for specific details as the name of the payer and the amount. When these details are found, the display module starts its function. Using java swing classes and awt APIs the string is displayed on the screen of the monitor. As new inputs are sent to the r pi, these are appended onto the screen. Java swing classes are used for formatting the strings in ways like font, size, color, etc. CSVWriter is a module which is used in this project to create a csv version of the data. It divides the string into parts separated by space. It adds the separated word in a collection called array list. These separated words are then appended into the csv file. Using the SMTP and MIME module of python, the csv file is fetched using file handling, it is then scanned for the type of data in it. Once it has confirmed that it is the right file, it creates a timestamp using time module. The initial time is set before hand as well as the interval. The email is sent at the initial time that is set and keeps on sending at the specified interval.

##### Acknowledgment

We would like to thank our college Vishwakarma Institute of Technology for providing us the opportunity to work on such projects. We would also like to thank our guide Prof. Virendra Pawar for guiding and helping us during the course of this project.

##### References

1. https://www.raspberrypi.org
2. https://developer.android.com/studi

