Interim Report I

FYP22004
Blockchain in Insurance

Student:
Tan Shan Huang 3035664582

Supervisor:
Dr. Liu, Qi

16 January 2022
Abstract

Flight delay insurance is essential to travelers as it gives the assurance of a covered accommodation and transportation in the event of a flight delay. However, the current claiming process is too slow as it is done manually. This report proposes a possible solution for this issue using the combination of blockchain and text recognition. Users will first upload supporting documents for their claim application. Then, important information will be extracted using text recognition and fed to the smart contracts. The pre-defined conditions in the smart contract will automatically approve or decline the claim application. It is expected to see this approach to outperform manual claim processing in flight delay insurance. In the long run, this will greatly reduce the operational cost of flight delay insurance providers. This project will deliver a web application to replace the insurance claim process in flight delay insurance using smart contracts and Google’s Optical Character Recognition (OCR) Engine. Currently, the development of user interfaces is completed. The development of smart contracts is currently in progress. The frontend-backend integration is expected to happen once the development of smart contracts is completed.
Acknowledgements

I would like to express my gratitude towards Dr. Liu Qi, the supervisor of this Final Year Project, for his guidance and support. I sincerely thank the Department of Computer Science, HKU for providing this opportunity for me to work on this project.
# Table of Contents

Abstract ........................................................................................................................................2

Acknowledgements .........................................................................................................................3

List of Figures .................................................................................................................................5

List of Tables .................................................................................................................................6

1 Introduction ..................................................................................................................................7
   1.1 Background Information .......................................................................................................7
   1.2 Objective ................................................................................................................................8
   1.3 Outline ..................................................................................................................................8

2 Literature Review .......................................................................................................................9
   2.1 Blockchain ............................................................................................................................9
   2.2 Optical Character Recognition ............................................................................................10

3 Project Methodology ..................................................................................................................11
   3.1 Application Workflow ..........................................................................................................11
   3.1.1 Purchase policy ...............................................................................................................11
   3.1.2 Review policy ................................................................................................................11
   3.2 Implementation ..................................................................................................................11
      3.2.1 Uploading supporting documents ...............................................................................11
      3.2.2 Off-chain data & Automation .....................................................................................12
      3.2.3 Blockchain: Polygon .....................................................................................................12
      3.2.4 Backend: Hardhat, Solidity, Express & NodeJS ...........................................................12
      3.2.5 Database: MongoDB ...................................................................................................12
      3.2.6 Frontend: ReactJS .......................................................................................................12

4 Results .........................................................................................................................................13
   4.1 Frontend Development .........................................................................................................13
   4.2 Development of Smart Contracts .......................................................................................13

5 Timeline ......................................................................................................................................14

6 Limitations ..................................................................................................................................15

7 Conclusion ...................................................................................................................................15
List of Figures

Figure 1: Anatomy of blockchain (Cherednichenko, 2020) .......................................................... 9
Figure 2: Code for funding a smart contract ................................................................................. 13
Figure 3: Insurance Plans ............................................................................................................. 18
Figure 4: Users will have to fill up personal information when purchasing a new insurance plan. ................................................................................................................................. 18
Figure 5: Landing Page .................................................................................................................. 18
Figure 6: Table showing the policies owned by users ................................................................. 19
Figure 7: Page showing the details of the current plan. Users can upload their supporting documents here. ................................................................................................................................. 19
Figure 8: Login page ..................................................................................................................... 19
List of Tables

Table 1: Implementation timeline

.......................................................... 14
1 Introduction

This section discusses the current limitation of insurance claim process, followed by the objective of this project.

1.1 Background Information

With the recent relaxation of travel restrictions around the world, travelers are finally able to travel again after 2 years of global lockdown. A study carried out by the Airports Council International found that global passenger traffic is expected to improve significantly, reaching over 70% of what it was in the pre-pandemic era (Bates, 2022). However, this surge in passenger traffic resulted in a drop in flight punctuality. A recent survey carried out by the Forbes Advisor found that 61% of the survey respondents experienced flight delays or cancellations, and 83% of those lost money as a result (Lobb & Gollub, 2022).

To better protect themselves against additional expenses caused by a flight delay, travelers will purchase a flight delay insurance.

However, the process of making a flight delay claim involves a series of time-consuming steps. First, travelers report a claim with all the supporting documents (e.g., receipts, flight ticket). Then, the insurance provider verifies the validity of claim with relevant airline company. Finally, claim will be paid out by the insurance provider. On average, insurance policyholder will receive their compensation anywhere between 1 to 2 weeks (TravelInsurance, 2022). If any issue arose during this process, it might take even longer for the policyholder to receive their payment. A long claim processing time often leads to unsatisfactory customer. Insurance Watch, a company specializing in life insurance found out that out of the all the reviews made by consumers, majority of them wanted faster payment of claims (Riskinfo, 2018). This implies that a fast and smooth claim process is critical for any insurance company to maintain customer satisfactory.

Even so, many insurance providers are still slow in realizing the importance of a smooth claim process. A study found that out of all the resources spent by insurance provider on improving their existing business model, innovation in claim processing only accounted for a mere 5 percent (McKinsey, 2017). There are a few reasons leading to a slow claim process. First, a high volume of insurance claim. As all the supporting documents must be verified manually, it takes a lot of manpower to process the insurance claim. Second, the potential
of insurance fraud. Insurance companies spend extra time and effort in reviewing every claim request before compensating the policyholders. The reason is because insurance frauds are causing insurance companies a huge cost. A study carried out by the Coalition Against Insurance Fraud found that it cost the United States $310 billion dollars annually on insurance fraud (Coalition Against Insurance Fraud, 2022).

1.2 Objective
Given the existing slow process in insurance claim, there is a need for a system with faster document reviewing process, as well as a way in preventing insurance fraud. This project therefore intends to build a web application using blockchain and OCR to replace the existing slow insurance claim process in flight delay insurance.

Smart contract, a self-executing contract stored on a blockchain can be used to automate insurance claim. Essentially, all the details of an insurance policy will be written in code into a smart contract. When all the predetermined conditions in the smart contract are fulfilled, payment will be made to the policyholder. As to counter insurance fraud, blockchain’s property of immutable transaction is perfect for this issue. Since all insurance claims are recorded to the blockchain, no repeated transactions will be allowed. Finally, a slow document reviewing process can be solved by utilizing OCR. Important information can be extracted and fed to smart contracts for verification.

1.3 Outline
Section 1 introduces the background information of this project, followed by the objectives of this implementation. Then, Section 2 provides more information on blockchain and OCR. Project methodology of this project, both frontend and backend will be shown in Section 3. Section 4 and 5 presents the current progress and a tentative project schedule of this project. The limitation of this project is discussed in Section 6. Finally, Section 7 provides a conclusion of this project.
2 Literature Review

This section provides more information on blockchain and OCR and their relation to this project.

2.1 Blockchain

A blockchain is a type of Distributed Ledger Technology. Different from a typical database which store data in only one location, blockchain stores information in a decentralized way. On top of that, blockchain transactions are immutable because of its data structure. It consists of a growing list of nodes, securely linked using cryptography. Each node contains their own unique hash value, and a cryptographic hash to the previous node (see Figure 1). Whenever someone attempts to edit any node in a blockchain, all the subsequent nodes must be updated as well or else it would be noticed by other participants of the network and rejecting this update. For an example, in Figure 1, if the attacker attempts to modify block no. 2, the hash value will no longer be “6BQ1”. It will be immediately noticed by block no. 3 and this modification will be rejected. Therefore, it is computationally impossible to tamper a blockchain, thus effectively prevent the issue of double claiming.

This project will utilize blockchain to deliver a system free of fraud. The immutable blockchain transaction property eliminates the possibility of double claiming in all insurance claim application.

![Figure 1: Anatomy of blockchain (Cherednichenko, 2020)](image-url)
2.2 Optical Character Recognition

OCR is a method used in text recognition. An OCR program can extract data from scanned documents, camera images and image-only pdfs. Upon inputting a scanned-in image, the image is analyzed for light and dark areas. Only dark areas are processed as it represents the characters. Then, it is further processed to find alphabetic letters or numeric digits. Finally, pattern recognition or feature recognition algorithm will be used to obtain the desired result (IBM Cloud Education, 2022).

Using OCR, the supporting documents from an insurance claim can be processed much faster. This aligns with the project’s objective of delivering a faster insurance claim process.
3 Project Methodology
This section presents both application workflow and implementation of this project in detail.

3.1 Application Workflow
Users will login with their username and password. Depending on whether the policyholder wants to purchase a new policy or review their existing policies, different window is shown.

3.1.1 Purchase policy
After selecting a new insurance policy and filing in required information, user will be directed to a payment page. The cost of premium includes the cost of writing smart contracts to the blockchain, the cost of using a third-party blockchain oracle and the cost of premium of the insurance plan.

3.1.2 Review policy
All the policies owned by the user will be shown in a table. The address of the smart contracts associated with the insurance policy, and the status of the policies are included in the table as well. When the policyholder wants to submit a claim request, they will first be directed to upload all the supporting documents. Upon successful verification, compensation will be made.

3.2 Implementation
The following section provides an overview of each technology and the justification behind the choice.

3.2.1 Uploading supporting documents
To extract data from user uploaded documents, an OCR engine will be used. Tesseract OCR Engine, an OCR maintained by Google is chosen as it is widely used in the industry and has a lot of resources provided by the community. Since the frontend of this project will be built using ReactJS, Tesseract.js, a wrap around library on Google’s Tesseract OCR Engine is selected. Tesseract.js works in the browser using plain script tags with a CDN. This will save a lot of time as compared to manually connecting this project and Google’s Tesseract OCR.
3.2.2 Off-chain data & Automation
To verify a flight delay claim, a real time data of flight information is compulsory. A custom flight data feed will be built using Chainlink library. On top of that, Chainlink Automation will be used to enable conditional execution of smart contracts through their decentralized platform.

3.2.3 Blockchain: Polygon
Polygon is chosen as the blockchain choice for this project due to a few reasons. First, Polygon shares the decentralized security of Ethereum by check-pointing all the transactions on its chain (MATIC chain) to the Ethereum. Second, transactions in Polygon are much faster at 7,000 per second as compared to Ethereum’s 15 transactions per second (Polygon, 2022). These properties of the Polygon blockchain ensures all the blockchain transactions are safe and fast, which is essential to this project.

3.2.4 Backend: Hardhat, Solidity, Express & NodeJS
The backend server will be created using Express, a minimalist web framework for Node.js. Node.js is selected as the server-side programming language as it has high built-in functionality. As for the development of smart contracts will be done using a combination of Hardhat and Solidity. Hardhat is an Ethereum development environment whereas Solidity is a programming language designed to target the Ethereum Virtual Machine. This combination will speed up the entire development for the smart contracts.

3.2.5 Database: MongoDB
MongoDB is a non-relational database for this application. It is chosen as it has smooth integration with both NodeJS and Express. It will be used for verified access to the web application and to store insurance policies owned by the users in the form of smart contract address.

3.2.6 Frontend: ReactJS
As a web application will be built for this project, the frontend library developed by Facebook, React.js will be used. Depending on the user group, different interfaces will be shown. The web application will use the backend API to perform different operations such as Create, Read, Update and Delete.
4 Results

This section discusses the progress made during the first semester. Feasibilities of different technologies was explored to determine the optimal combination required for this project. The development of user interfaces and smart contracts will be discussed.

4.1 Frontend Development

The development of core components needed for this project has been completed. This includes a login page, a landing page, a page showing details of different insurance plans, a payment page, a page showing policies currently owned and finally a page for users to upload their supporting documents (see Appendix A).

4.2 Development of Smart Contracts

The implementation of funding a smart contract has been completed (see Figure 2). This process is essential as blockchain oracles require funding on the smart contracts for them to make an API call. The completion of this step gives good progress as it allows the development of integrating insurance policies into smart contract to start.

Figure 2: Code for funding a smart contract
## 5 Timeline

This section presents the overall timeline for the implementation of this project. In the second semester, the development will mainly focus on backend and frontend-backend integration.

<table>
<thead>
<tr>
<th>Time Period</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>2022 October</td>
<td>• Research on feasibility of different blockchains and technologies required for this project</td>
</tr>
<tr>
<td>2022 November - December</td>
<td>• Frontend development</td>
</tr>
<tr>
<td>2023 January</td>
<td>• Custom flight data feed</td>
</tr>
<tr>
<td></td>
<td>• Smart contracts development</td>
</tr>
<tr>
<td></td>
<td>• Information extraction using Tesseract</td>
</tr>
<tr>
<td>2023 February – March</td>
<td>• Database development</td>
</tr>
<tr>
<td></td>
<td>• User verification system</td>
</tr>
<tr>
<td></td>
<td>• Frontend and backend integration</td>
</tr>
<tr>
<td></td>
<td>• Smart contract testing</td>
</tr>
<tr>
<td>2023 April</td>
<td>• Product testing</td>
</tr>
</tbody>
</table>

*Table 1: Implementation timeline*
6 Limitations
Currently, there is no way to verify the ownership of the supporting documents uploaded by the user. One possible solution is to request customer data from airline companies to verify the ownership of the flight ticket. Several emails have been sent to different airline companies to see if they are willing to provide such data. Another solution is to request identification document from the user when they are making the claim. User would have to upload two documents, namely, identity card and a selfie of them holding the identity card.

7 Conclusion
This report has proposed a possible solution for slow insurance claim processing. A web application will be built by utilizing smart contracts and OCR to deliver a faster and smoother claim process. Several core components have been built using React Js and were split into different pages. The development of funding a smart contract has been completed and this allows smart contracts to make API call through a blockchain oracle. The process of integrating an insurance policy into smart contract will be implemented next. Once the development of smart contracts is finished, the frontend-backend integration will be carried out.
References


Appendix A

Figure 3: Insurance Plans

Figure 4: Users will have to fill up personal information when purchasing a new insurance plan.

Figure 5: Landing Page
Figure 6: Table showing the policies owned by users

Figure 7: Page showing the details of the current plan. Users can upload their supporting documents here.

Figure 8: Login page