FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

DEPARTMENT OF COMMUNICATION TECHNOLOGY AND NETWORK

CNS 4504 - 1:

BLOCKCHAIN

GROUP PROJECT

TITLE:

VEHICLE ACCIDENT INSURANCE CLAIM PROCESSING

LECTURER'S NAME:

DR. NORMALIA BINTI SAMIAN

NAME	**MATRIC NO**
ANIS NADIRA BINTI NOOR MAISAM	210423
MOONNA BINTI ZULKAFLY	212137
NURDIYANA ATHIRAH BINTI MOHD ASMAN	211641
HASRINAH BINTI KURONG	213110

Table of content

1.0 Executive Summary

Overview :

- Vehicle accident insurance claim processing aims to enhance efficiency and transparency using DApp technology for automated, secure, and streamlined claim verification and settlement processes.

Problem Summary :

- **Complexity and Lengthy Processes**: The traditional claim process involves intricate steps from accident reporting to settlement, posing significant challenges for smaller insurers with limited resources. This complexity can lead to inefficiencies, delays in claim resolution, and customer dissatisfaction.

- **Documentation Management**: Accident victims bear the burden of compiling extensive documentation, such as police reports and witness statements, which adds complexity and prolongs processing times. Simplifying documentation processes is crucial to improve efficiency and reduce the stress on claimants.

- **Communication and Coordination**: Coordinating between accident victims, repair shops, and insurers often leads to delays and misunderstandings, especially in smaller organizations with limited communication channels. Effective coordination is essential to ensure timely evaluations, accurate approvals, and prompt settlements.

- **Timeliness of Settlement**: Prompt settlement of claims is critical for maintaining customer satisfaction and trust. Delays in approval and payment processes can disrupt the daily lives of accident victims who rely on their vehicles, emphasizing the need for efficient claim handling.

- **Technology Integration Challenges**: Implementing advanced technologies like blockchain for smart contracts can present technical and operational hurdles, particularly for smaller insurers and startups seeking quick and efficient solutions in claim processing.

Competitor Analysis :

Our System, Takaful, Etiqa and Traditional System all share common functionalities such as claim submission, approval and rejection, and claim status retrieval. However, there are notable differences such as :

- **User Interface Simplicity**: Our System offers a high level of simplicity, ensuring ease of use for users. Takaful and Etiqa provide a moderate to high simplicity level, while the Traditional System often relies on manual processes, resulting in a lower user interface simplicity.

- **Transaction Fees**: Our System and the Traditional System offer low transaction fees, promoting cost-effectiveness. Takaful, Etiqa, and vary in their fee structures.

- **Scalability**: Our System, Takaful, Etiqa, and exhibit high scalability, indicating their ability to handle increased workload and expand operations efficiently. In contrast, the Traditional System shows lower scalability due to its reliance on manual processes.

- **Security**: Our System, Takaful, Etiqa, and prioritize high-security measures to protect user data and transactions. The Traditional System's security varies and is often less robust compared to digital platforms.

Financial Summary :

Our blockchain-based solution introduces a transformative approach to vehicle accident insurance claims processing by leveraging the power of smart contracts to automate and secure the entire workflow, from accident reporting to claim settlement. This

innovative solution offers significant cost savings for insurance companies by reducing administrative overhead associated with manual data entry, claim verification, and communication. Additionally, the immutable nature of blockchain technology helps mitigate the risk of fraudulent claims. Efficiency gains are another major benefit, as our solution can expedite claims processing from several weeks to just a few days, enhancing customer satisfaction and operational efficiency. Automation also reduces the likelihood of human errors, ensuring accurate and consistent claims processing, which in turn decreases disputes and the need for rework.

Despite the implementation costs, which include developing and deploying smart contracts, integrating with existing systems, and ongoing maintenance, the potential return on investment (ROI) is compelling. The improved efficiency and cost savings can help insurance companies achieve a faster break-even point and long-term profitability. Furthermore, the enhanced customer experience and transparency offered by our solution can attract more customers, increase market share, and improve customer loyalty, leading to steady revenue generation. Overall, our blockchain-based solution provides substantial benefits that can revolutionize vehicle accident insurance claims processing, streamline operations, and drive business growth for insurance companies.

2.0 Introduction

The traditional process of handling vehicle accident insurance claims is often plagued by inefficiencies, delays, and a lack of transparency. This can lead to a frustrating experience for both policyholders and insurers. Common issues include lengthy paperwork, slow processing times, and unclear communication regarding the status of claims. These problems not only inconvenience customers but also increase administrative costs and the potential for disputes. To address these challenges, blockchain technology offers a powerful solution through the use of smart contracts. Smart contracts are self-executing contracts where the terms of the agreement are directly written into lines of code. These contracts automatically enforce and execute the

terms when certain conditions are met, eliminating the need for intermediaries. By implementing smart contracts in the vehicle accident insurance claim process, the entire procedure can be made more efficient, accurate, and trustworthy.

Apart from that, the implementation of a smart contract for vehicle accident insurance claim processing using Ethereum's Solidity programming language. The primary goals of this smart contract include automating the claim filing process, streamlining claim review and approval, facilitating secure payouts, and enhancing transparency and trust. Policyholders can file claims directly on the blockchain, ensuring that the claim details are immutable and transparent, reducing the chances of tampering and enhancing trust. Insurers can review and approve claims efficiently within the blockchain environment, with every action taken during the review process recorded on the blockchain, providing a clear and accountable audit trail. Once a claim is approved, the payout process is handled automatically and securely by the smart contract, minimizing the risk of errors and fraud, and ensuring that policyholders receive their due compensation promptly. All stakeholders, including policyholders and insurers, can access real-time updates on the status of claims, which helps build trust and reduce the likelihood of disputes as everyone involved has a clear understanding of the process and current status.

The following sections of this report will provide a detailed explanation of each part of the smart contract. First, we will examine the contract's structure, including the necessary variables, events, and rules. Next, we will describe how to implement the functions for filing claims, reviewing and approving claims, and processing payouts. Additionally, we will explain the mechanism that allows the contract to securely receive funds. By adopting smart contracts for vehicle accident insurance claim processing, insurers can significantly enhance operational efficiency, reduce costs, and improve customer satisfaction. This comprehensive guide aims to offer a clear understanding of how to implement this system, highlighting the practical benefits of blockchain technology in the insurance industry.

3.0 Problem identification

Identifying problems in the vehicle accident insurance claim process involves recognizing challenges at each stage, from reporting the accident to settling the claim. Each step has its complexities such as reporting delays can affect getting the needed police report to the insurer on time, while gathering and submitting all necessary documents accurately is crucial for claim approval. Coordination issues between the insured, repair shops, and insurers also arise during vehicle assessment, repair, and settlement stages. Addressing these challenges is key to improving efficiency, reducing delays, and ensuring customer satisfaction throughout the insurance claim process.

The problem identification can involves many recognizing specific challenges within each stage of vehicle accident insurance claim process such as :

Complexity and Lengthy Processes :

The traditional vehicle accident insurance claim process involves numerous intricate steps, starting from the initial reporting of the accident to the final settlement of the claim. This process can be particularly daunting and time-consuming for smaller insurance firms and startups that may lack the extensive resources and infrastructure of larger companies. These organizations often face challenges in efficiently managing and coordinating the various aspects of claim processing, such as gathering necessary documentation, assessing damages, and facilitating communication between all parties involved. Limited staffing and technological capabilities further compound these difficulties, potentially leading to delays in claim resolution and customer dissatisfaction. As a result, finding streamlined and effective solutions tailored to the specific needs and constraints of smaller insurers becomes crucial for improving operational efficiency and maintaining competitive viability in the insurance industry.

Documentation Management :

Accident victims face a considerable responsibility in collecting and submitting extensive documentation, such as police reports, photos of the accident scene, and statements from witnesses. This task can be overwhelming for individuals seeking simpler and more straightforward solutions to their insurance claim process. The need to compile comprehensive and accurate information adds complexity to an already stressful situation, potentially prolonging the claim processing time and requiring meticulous attention to detail. Simplifying this documentation process is essential to alleviate the burden on victims and enhance their experience during the insurance claim journey.

Communication and Coordination :

Coordinating between accident victims, repair shops, and insurance companies for assessments, repair approvals, and settlements can often result in delays and misunderstandings. This is particularly prevalent in smaller organizations that may lack robust communication channels. Effective coordination is crucial throughout these processes to ensure timely evaluations, accurate repair approvals, and prompt settlements. Miscommunications or delays can prolong the time it takes to resolve claims, causing inconvenience for the parties involved and potentially impacting customer satisfaction. Therefore, improving communication and coordination among all stakeholders is essential for streamlining the insurance claim process and delivering efficient service to accident victims.

Timeliness of Settlement :

Prompt settlement of claims is crucial for accident victims who depend on their vehicles for daily activities. Delays in the approval and payment process can significantly disrupt their routines and lead to dissatisfaction with the insurance provider. Timely settlements ensure that victims can quickly repair their vehicles and resume normal life without prolonged inconvenience. It also plays a vital role in maintaining positive relationships between insured individuals and their insurers, reinforcing trust and satisfaction in the

insurance services provided. Therefore, ensuring efficient and timely claim settlements is essential for enhancing customer experience and loyalty in the insurance industry.

Claim Settlement :

After approving the claim, it's crucial for the insurance company to quickly settle the claim by paying the repair shop for the completed repairs. Delays in this settlement process can affect the insured's ability to use their vehicle and may strain the relationship between the insured and the insurer. Prompt settlement ensures that the vehicle is repaired promptly, minimizing inconvenience for the insured and maintaining a positive relationship with the insurer.

Technology Integration:

Implementing and integrating advanced technologies, such as blockchain for smart contract solutions, may pose technical and operational challenges for small insurance companies and startups aiming for quick, efficient solutions.

4.0 Target audience

The target audience for vehicle accident insurance claim processing encompasses a broad array of stakeholders who are integral to ensuring the seamless and effective management of claims throughout their lifecycle, from initial submission to final settlement, thereby contributing to enhanced operational efficiency, customer satisfaction, and regulatory compliance within the insurance industry.

The target audience for vehicle accident insurance claim processing includes :

Insurance Companies :

Insurance companies, whether large corporations or smaller firms, are essential entities within the vehicle accident insurance claim processing framework. Their primary

objectives include optimizing claim processing workflows to improve efficiency and elevate customer satisfaction. Integral to their role is the thorough evaluation of claim validity based on policy terms and meticulous verification of supporting documentation such as accident reports and witness statements. This scrutiny enables insurers to make informed decisions regarding the approval or rejection of claims, ensuring adherence to contractual obligations while leveraging technological advancements to streamline operations and deliver timely compensation to policyholders and accident victims.

Accident Victims :

Accident victims are individuals who have experienced vehicular accidents and are navigating the insurance claim process to seek fair and timely compensation. This compensation is crucial to cover various expenses resulting from the accident, including medical bills, vehicle repairs, and other losses incurred. Throughout this process, accident victims also need to monitor the status of their claims to understand whether they will be approved or rejected by the insurance company. This monitoring is essential for managing expectations, planning finances, and addressing any discrepancies or challenges that may arise during the evaluation of their claims. Ensuring clear communication and transparency from insurers regarding the status of claims helps accident victims make informed decisions and navigate the aftermath of the accident with greater ease and confidence.

Policy Holder :

Policyholders are individuals who have vehicle insurance and may need to file a claim after being involved in a vehicle accident. They seek a straightforward and timely claims process to ensure they receive compensation for their losses without unnecessary delays. This compensation can cover various expenses, including medical bills, vehicle repairs, and other financial impacts resulting from the accident. By having a streamlined and efficient claims process, policyholders can manage their post-accident situation more effectively, reducing stress and ensuring they can quickly return to normalcy. Clear

communication and transparency throughout the claim process are crucial for policyholders to stay informed about the status of their claims, helping them make necessary plans and decisions during a challenging time.

Police :

Police play a crucial role in the vehicle accident insurance claim process. They are responsible for documenting the accident, investigating the scene, and generating an official accident report. This report is a critical piece of evidence required by insurance companies to verify the details of the incident and process the claim. The involvement of the police ensures that the facts are accurately recorded, which helps in determining fault and assessing the validity of claims. By providing a detailed and unbiased account of the accident, the police help facilitate a smoother and more efficient claims process for both the policyholder and the insurance company.

Claim assessors :

Claim assessors known as insurance adjusters, are crucial to the vehicle accident insurance claim process. They are responsible for evaluating the damage to vehicles, reviewing submitted documentation, and determining the extent of the insurer's liability. This involves inspecting the accident scene, examining the vehicle's condition, and sometimes interviewing involved parties and witnesses to gather comprehensive information. The assessors' evaluations help estimate repair costs and ensure that the claim is valid and accurately reflects the damages incurred. By providing an objective and thorough assessment, claim assessors play a vital role in ensuring fair and timely compensation for policyholders, thereby facilitating a smoother and more efficient claim resolution process.

The Road Transport Department Malaysia (JPJ) :

The Road Transport Department Malaysia, commonly known as JPJ (Jabatan Pengangkutan Jalan Malaysia), plays a pivotal role in the vehicle accident insurance

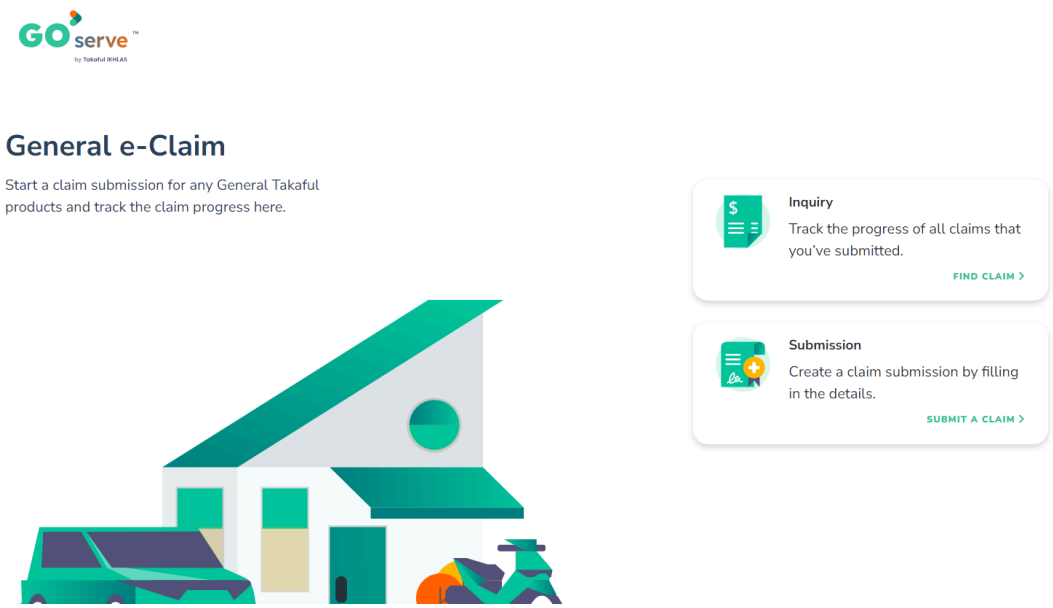
claim process. JPJ is tasked with the registration and licensing of vehicles and drivers, in addition to the enforcement of road transport laws within Malaysia. In the event of a vehicle accident, JPJ's involvement is multifaceted. Firstly, JPJ receives accident reports, which are often a prerequisite for insurance claims and legal documentation. The department maintains comprehensive records of vehicle ownership and driver licensing, which are essential for verifying the authenticity of the involved parties and ensuring the accuracy of the submitted documentation. In instances of serious accidents, JPJ collaborates with police authorities to conduct thorough investigations, thereby providing a detailed understanding of the accident circumstances, which is crucial for fair and accurate claim assessments. Moreover, JPJ ensures that all vehicles and drivers adhere to Malaysia's road transport regulations, which is vital for the legitimacy and processing of insurance claims. Through these responsibilities, JPJ significantly contributes to the efficiency, transparency, and accuracy of the vehicle accident insurance claim process, thereby supporting both policyholders and insurance companies.

5.0 Solution overview

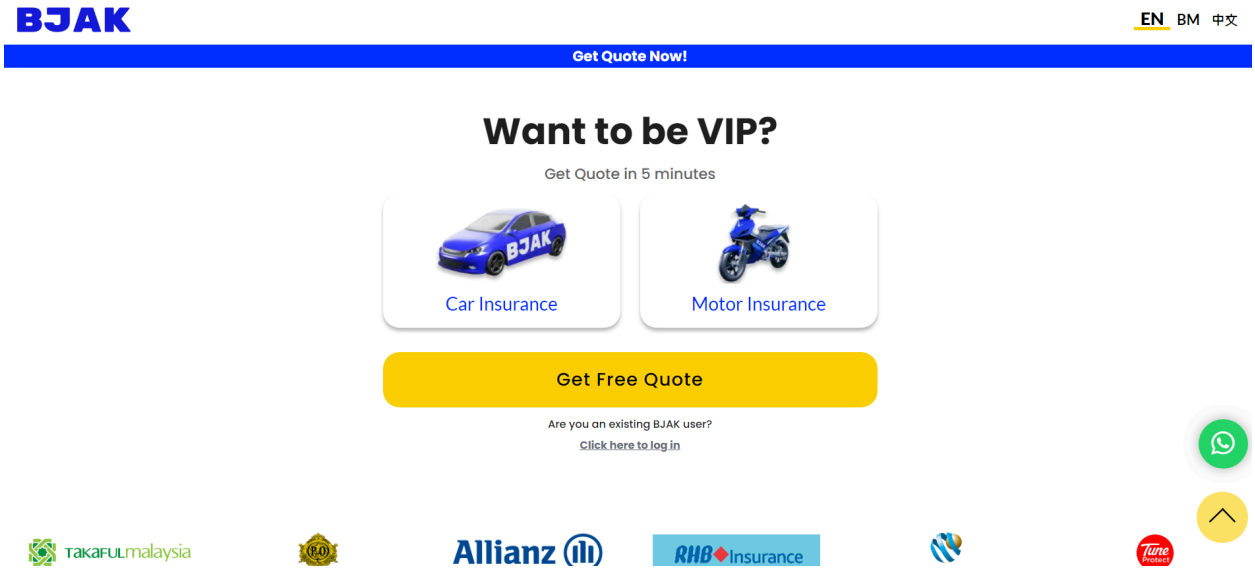
The insurance claim processing system is architectured as a decentralized application (DApp) on the Ethereum blockchain, offering a decentralized infrastructure for secure data storage and smart contract execution using Solidity. This DApp introduces automation, transparency, and security into traditional insurance claim processing methods. Its user-friendly interface, developed with ReactJS, enables insurance companies to submit claims and users to track claim statuses. The DApp manages the entire claim process from submission to verification, enhancing efficiency and reducing fraud risks. Insurance companies can analyze damages and update information, while users can easily track claim statuses. Ethereum's smart contract functionality enables process automation, reducing manual intervention and enhancing efficiency. Overall, Ethereum provides a reliable platform for building secure, transparent, and efficient DApps. The system utilizes ReactJS for the front end, Node.js

for the backend, and Solidity for smart contracts, ensuring a secure and efficient process for managing insurance claims.

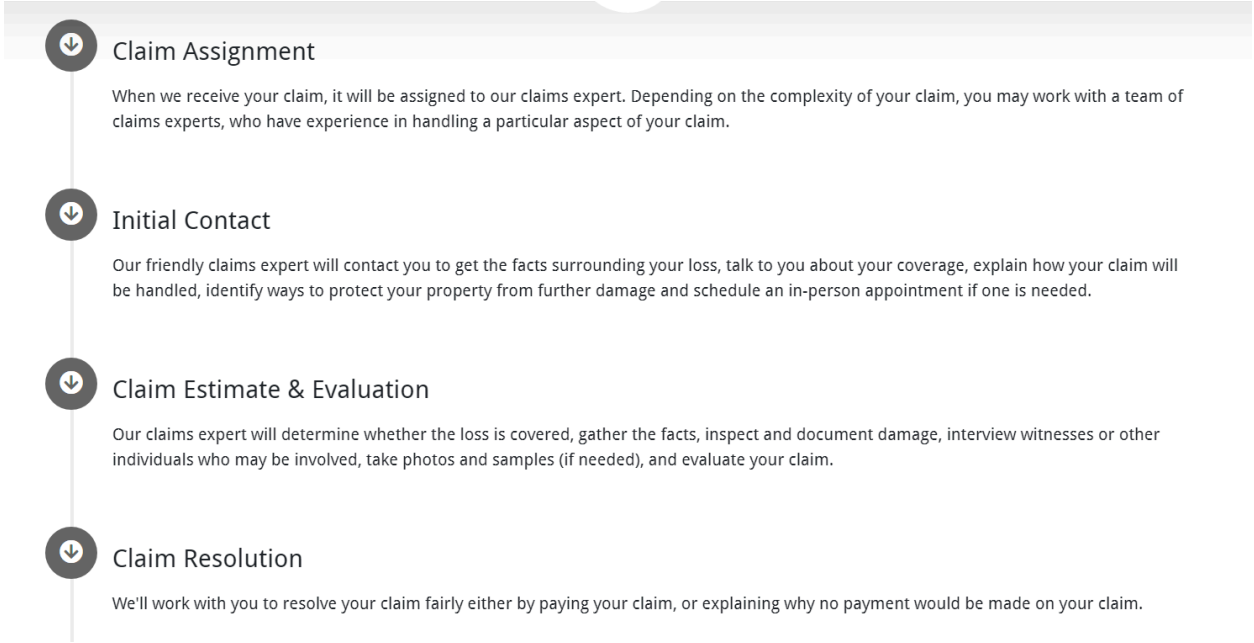
6.0 Competitive analysis



https://go.takaful-ikhlas.com.my/eclaim/general/



https://bjak.my/en?utm_source=google&utm_medium=cpc&utm_campaign=gg_bjak-insurance_sem_brd_phrase&utm_content=bijak%20quotation&gclid=Cj0KCQjwsuSzBhCLARIsAIcdLm4y_vN0daoHUo4e0PKt8I-9AAaxxTpVabGjqEV2aKUITUEYPSB0rXsaApCXEALw_wcB



https://www.etiqa.com.my/v2/claims

Features	Our System	Takaful	etiqa	BJAK	Traditional System
Submit Claim	Yes	Yes	Yes	Yes	Yes
Approve/Reject Claim	Yes	Yes	Yes	Yes	Yes
Retrieve Claim Status	Yes	Yes	Yes	Yes	Often Manual
User Interface Simplicity	High	Moderate	High	High	Low
Transaction Fees	Low	Varies	Varies	Varies	Low
Scalability	High	High	High	High	Low
Security	High	High	High	High	Varies

- Takaful uses various custom platforms tailored to their digital insurance needs but lacks specific mentions of technologies like blockchain or specific programming languages
- Etiqa and BJAK also employ custom digital solutions but do not specify the technology stack in the available data.

7.0 Implementation plan

Phase 1: Planning and Design

Week 1-2: Requirements Gathering

- Identify stakeholders (accident victims, insurance companies)
- Define functional requirements (claim submission, status retrieval)
- Define non-functional requirements (security, performance)

Week 3-4: System Design

- Design system architecture (frontend, backend, blockchain integration)
- Create wireframes and UI designs
- Establish a project management plan with timelines and responsibilities
- Prepare technical documentation for development and testing phases

Phase 2: Development

Week 5-6: Smart Contract Development

- Develop smart contracts using Solidity for claim submission, processing, and status retrieval
- Test contracts on Ethereum test network (e.g., Ropsten, Rinkeby)

Week 7-8: Frontend Development

- Build user interfaces using React for accident victims and insurance companies
- Integrate smart contracts with frontend using Web3.js
- Implement user authentication and authorization

Week 9-10: Backend Development

- Set up Node.js backend to manage user data and interactions
- Develop APIs to interact with smart contracts and frontend
- Implement database (e.g., MongoDB) for storing non-blockchain data

Phase 3: Testing

Week 11-12: Unit and Integration Testing

- Test smart contracts using Truffle and Ganache
- Perform frontend and backend integration tests
- Conduct performance testing to ensure system scalability

Week 13-14: User Acceptance Testing

- Conduct testing sessions with stakeholders
- Gather feedback and make necessary adjustments
- Conduct security testing to ensure data protection and privacy

Phase 4: Deployment

Week 15: Deployment

- Deploy smart contracts on Ethereum mainnet
- Deploy frontend and backend to cloud platforms (e.g., AWS, Heroku)
- Set up continuous integration and continuous deployment (CI/CD) pipelines

Phase 5: Maintenance and Monitoring

Ongoing: Post-Deployment Support

- Monitor system performance and security
- Provide user support and perform regular updates
- Implement analytics to track usage and gather insights for future improvements
- Conduct regular audits to ensure compliance with industry standards and regulations

Milestones:

1. Requirements and Design Completion (End of Week 4)
2. Smart Contracts Deployed on Test Network (End of Week 6)
3. Frontend and Backend Integration Complete (End of Week 10)
4. User Acceptance Testing Complete (End of Week 14)
5. System Deployed and Live (End of Week 15)
6. Continuous Monitoring and Support (Ongoing)

8.0 Financial projection

Our blockchain-based solution represents a fundamental shift in the vehicle accident insurance claims processing paradigm. By leveraging the power of smart contracts, we aim to automate and secure the entire workflow, from accident reporting to claim settlement, thereby revolutionizing the way insurance companies handle claims. At this stage, we will provide a comprehensive overview of the potential financial impact of our solution, focusing on qualitative projections that highlight the key benefits for insurance companies.

Cost Savings:

One of the primary benefits of our solution is the potential for significant cost savings for insurance companies. By automating the claim submission and approval process, we can drastically reduce administrative overhead. This includes labor costs related to manual data entry, claim verification, and communication. Furthermore, the use of blockchain technology ensures the immutability of claims data, reducing the risk of fraudulent claims and resulting in additional cost savings.

Efficiency Gains:

Our smart contract-enabled solution has the potential to greatly improve the efficiency of the claims processing workflow. Traditional claims processing can be a time-consuming process, taking several weeks and involving multiple parties and numerous document verifications. By leveraging blockchain technology, we can expedite this process to just a few days, greatly improving customer satisfaction and operational efficiency. Additionally, automation reduces the likelihood of human errors, ensuring that all claims are processed accurately and consistently, thereby reducing disputes and rework.

Revenue Generation:

In addition to cost savings, our solution offers significant revenue generation opportunities for insurance companies. By offering a streamlined, secure, and efficient claims processing solution, insurance companies can attract more customers. This can lead to an increase in market share and revenue. Furthermore, the improved customer

experience due to faster and more transparent claims processing can enhance customer loyalty, reducing churn rates and maintaining a steady revenue stream.

Implementation Costs:

While there are significant benefits to be gained from our solution, there are also costs associated with its implementation. Initial costs involve developing and deploying the smart contracts and user interface. This includes hiring blockchain developers, UI/UX designers, and quality assurance testers. Additionally, there are costs associated with integrating the blockchain solution with existing insurance company systems and ensuring seamless data flow and compatibility. Ongoing maintenance and updates to the system are also necessary to address any issues, implement new features, and keep up with technological advancements.

Potential Return on Investment (ROI):

Despite the implementation costs, our solution offers a compelling potential ROI for insurance companies. The projected ROI will be influenced by factors such as the extent of adoption and implementation of our solution by insurance companies. By reducing operational costs and enhancing customer satisfaction, insurance companies can achieve a faster break-even point and long-term profitability.

Overall, our blockchain-based solution offers a transformative approach to vehicle accident insurance claims processing, providing substantial benefits in terms of cost savings, efficiency gains, and revenue generation. By embracing this technology, insurance companies can not only streamline their operations but also improve customer experience and drive business growth.

9.0 Team expertise

Overview

Our team comprises four dedicated and skilled members, each bringing unique expertise essential for the successful development and implementation of our blockchain-based solution.

Smart Contract Development:

2 Students for Smart Contract Development:

Anis Nadira:

- **Role:** Smart Contract Developer
- **Expertise:**
    - Basic experience in Solidity programming for smart contract development.
    - Experience in designing and implementing secure and efficient smart contracts.
    - Knowledge of best practices in smart contract development, including access control and error handling.
    - Ability to write comprehensive tests to validate smart contract behavior.
    - Basic understanding of cryptographic principles and consensus algorithms.

Moonna Zulkafly:

- **Role:** Smart Contract Developer
- **Expertise:**
    - Similar to Anis Nadira, with a focus on developing robust and secure smart contracts.
    - Collaborates closely with Student 1 to ensure the smart contract meets all requirements.
    - Contributes to testing and validation efforts to ensure the smart contract's functionality and security.
    - Basic understanding of blockchain technology and its applications.

2 Students for User Interface Development:

Diyana Athirah:

- **Role:** UI/UX Designer
- **Expertise:**
    - Skilled in developing Web3 user interfaces to interact with smart contracts.
    - Experience in ensuring the security and reliability of dApps through best practices.
    - Knowledgeable in incorporating features such as transaction history, user profiles, and data visualization.
    - Implements error handling and user feedback mechanisms to enhance the user experience.

Hasrinah Kurong:

- **Role:** UI/UX Developer
- **Expertise:**
    - Similar to Diyana Athirah, with a focus on developing intuitive and user-friendly interfaces.
    - Works closely with Student 3 to design and implement the UI features of the dApp.
    - Collaborates with the smart contract developers to ensure seamless integration between the UI and the smart contracts.
    - Basic experience in writing, testing, and deploying smart contracts on the Ethereum blockchain.

Together, our team is well-equipped to develop and implement a successful blockchain-based solution for vehicle accident insurance claims processing. Each member brings valuable skills and expertise to the table, ensuring that our solution meets all requirements and provides a seamless user experience.

10.0 Code and output

Insurance.sol

```
// SPDX-License-Identifier: MIT
```

```
pragma solidity 0.8.19;

contract Insurance {
    enum ClaimStatus {
        Pending,
        Approved,
        Rejected
    }

    struct Claim {
        address victim;
        address insuranceCompany;
        string policeReport;
        string repairEstimate;
        uint256 claimAmount;
        ClaimStatus status;
    }

    mapping(uint256 => Claim) public claims;
    uint256 public claimCount;
    address public insuranceCompany;  // Address of the insurance company

        event ClaimSubmitted(uint256 indexed claimId, address indexed victim, string
policeReport, string repairEstimate, uint256 claimAmount);
    event ClaimApproved(uint256 indexed claimId, uint256 claimAmount);
    event ClaimRejected(uint256 indexed claimId);

    // Constructor to set the insurance company address
    constructor(address _insuranceCompany) {
        insuranceCompany = _insuranceCompany;
    }
```

```
    // Function to submit a claim
    function submitClaim(
        string memory _policeReport,
        string memory _repairEstimate,
        uint256 _claimAmount
    ) public {
        claims[claimCount] = Claim(
            msg.sender,
            insuranceCompany,
            _policeReport,
            _repairEstimate,
            _claimAmount,
            ClaimStatus.Pending
        );
            emit ClaimSubmitted(claimCount, msg.sender, _policeReport, _repairEstimate,
_claimAmount);
        claimCount++; // Increase the claimCount after emitting the event
    }

    function approveClaim(uint256 _claimId) public {
        require(claims[_claimId].victim != address(0), "Claim does not exist");
        require(msg.sender == insuranceCompany, "Only insurance company can approve
claim");
        claims[_claimId].status = ClaimStatus.Approved;
        emit ClaimApproved(_claimId, claims[_claimId].claimAmount);
    }

    function rejectClaim(uint256 _claimId) public {
        require(claims[_claimId].victim != address(0), "Claim does not exist");
```

```
        require(msg.sender == insuranceCompany, "Only insurance company can reject claim");
        claims[_claimId].status = ClaimStatus.Rejected;
        emit ClaimRejected(_claimId);
    }

    function getClaimStatus(uint256 _claimId) public view returns (string memory) {
        Claim memory claim = claims[_claimId];
        if (claim.status == ClaimStatus.Pending) {
            return "Pending";
        } else if (claim.status == ClaimStatus.Approved) {
                                return string(abi.encodePacked("Approved with amount: ", uint2str(claim.claimAmount)));
        } else {
            return "Rejected";
        }
    }

    function uint2str(uint _i) internal pure returns (string memory) {
        if (_i == 0) {
            return "0";
        }
        uint j = _i;
        uint len;
        while (j != 0) {
            len++;
            j /= 10;
        }
        bytes memory bstr = new bytes(len);
        uint k = len;
        while (_i != 0) {
```

```
            k = k-1;
            uint8 temp = (48 + uint8(_i - _i / 10 * 10));
            bytes1 b1 = bytes1(temp);
            bstr[k] = b1;
            _i /= 10;
        }
        return string(bstr);
    }
}
```

2_claim.js

```
const Insurance = artifacts.require("Insurance");

module.exports = async function(deployer, network, accounts) {
    // You guys tak bagitau account[0] tu insuranceCompany.

    // So yang kat approveClaim tu, Benda ni tak meet
      //   require(msg.sender == insuranceCompany, "Only insurance company
can reject claim");

   const insuranceCompanyAddress = accounts[0]; // or set this to the
specific address you want to use

  await deployer.deploy(Insurance, insuranceCompanyAddress);
};
```

App.jsx

```
import React, { useState, useEffect } from 'react';
import './App.css';
import Web3 from 'web3';
import InsuranceClaim from '/src/artifacts/Insurance.json';
```

```
const App = () => {
  const [web3, setWeb3] = useState(null);
  const [accounts, setAccounts] = useState([]);
  const [insuranceContract, setInsuranceContract] = useState(null);
  const [claimId, setClaimId] = useState('');
  const [policeReport, setPoliceReport] = useState('');
  const [repairEstimate, setRepairEstimate] = useState('');
  const [claimAmount, setClaimAmount] = useState('');
  const [claimStatus, setClaimStatus] = useState('');
  const [message, setMessage] = useState('');

  useEffect(() => {
    const loadBlockchainData = async () => {
      try {
        if (window.ethereum) {
          window.web3 = new Web3(window.ethereum);
          await window.ethereum.enable();
        } else if (window.web3) {
          window.web3 = new Web3(window.web3.currentProvider);
        } else {
                                        window.web3    =    new    Web3(new
Web3.providers.HttpProvider('http://localhost:7545'));
        }

        const web3Instance = window.web3;
        setWeb3(web3Instance);

        const accounts = await web3Instance.eth.getAccounts();
        setAccounts(accounts);
        console.log("Accounts:", accounts);

        const networkId = await web3Instance.eth.net.getId();
        const deployedNetwork = InsuranceClaim.networks[networkId];
        if (deployedNetwork) {
          const insurance = new web3Instance.eth.Contract(
            InsuranceClaim.abi,
            deployedNetwork.address,
          );
          setInsuranceContract(insurance);
          console.log("Insurance Contract:", insurance);
```

```
        } else {
          setMessage('Smart contract not deployed to detected network.');
        }
      } catch (error) {
        console.error('Error loading blockchain data:', error);
        setMessage('Error loading blockchain data.');
      }
    };

    loadBlockchainData();
  }, []);

  const submitClaim = async () => {
    try {
      await insuranceContract.methods
                              .submitClaim(policeReport,    repairEstimate,
web3.utils.toWei(claimAmount, 'ether'))
        .send({ from: accounts[0] });

      // Retrieve submitted claim information
                                 const     submittedClaimId     =     await
insuranceContract.methods.getSubmittedClaimId().call();
                              const     submittedClaimAmount     =     await
insuranceContract.methods.getSubmittedClaimAmount().call();
                              const     submittedClaimStatus     =     await
insuranceContract.methods.getClaimStatus(submittedClaimId).call();

            setMessage('Claim  submitted  successfully.  Claim  ID:  '  +
submittedClaimId + ', Amount: ' + web3.utils.fromWei(submittedClaimAmount,
'ether') + ' ETH, Status: ' + submittedClaimStatus);
    } catch (error) {
      console.error('Error submitting claim:', error);
      setMessage('Error submitting claim.');
    }
  };

  const approveClaim = async () => {
    try {
       await insuranceContract.methods.approveClaim(claimId).send({ from:
accounts[0] });
```

```
      setMessage('Claim approved successfully.');
    } catch (error) {
      console.error('Error approving claim:', error);
      setMessage('Error approving claim.');
    }
  };

  const rejectClaim = async () => {
    try {
        await insuranceContract.methods.rejectClaim(claimId).send({ from:
accounts[0] });
      setMessage('Claim rejected successfully.');
    } catch (error) {
      console.error('Error rejecting claim:', error);
      setMessage('Error rejecting claim.');
    }
  };

  const getClaimStatus = async () => {
    try {
                                    const      status      =      await
insuranceContract.methods.getClaimStatus(claimId).call();
      setClaimStatus(status);
      setMessage('Claim status retrieved successfully.');
    } catch (error) {
      console.error('Error getting claim status:', error);
      setMessage('Error getting claim status.');
    }
  };

  return (
    <div className="main-container">
      <header>
        <h1 className="title">Insurance DApp</h1>
      </header>

      <div className="main-content">
        <div className="account">
          <b>Current Account:</b> {accounts[0]}
        </div>
```

```
      </div>

      <div className="main-process">
        <div className="claim-form">
          <h3>Submit Claim</h3>
          <input
            type="text"
            placeholder="Claim ID"
            onChange={(e) => setClaimId(e.target.value)}
          />
          <input
            type="text"
            placeholder="Police Report"
            value={policeReport}
            onChange={(e) => setPoliceReport(e.target.value)}
          />
          <input
            type="text"
            placeholder="Repair Estimate"
            value={repairEstimate}
            onChange={(e) => setRepairEstimate(e.target.value)}
          />
          <input
            type="text"
            placeholder="Claim Amount (ETH)"
            value={claimAmount}
            onChange={(e) => setClaimAmount(e.target.value)}
          />
                  <button  className="button"  onClick={submitClaim}>Submit
Claim</button>
        </div>

        <div className="claim-approve">
          <h3>Approve Claim</h3>
          <input
            type="text"
            placeholder="Claim ID"
            onChange={(e) => setClaimId(e.target.value)}
          />
```

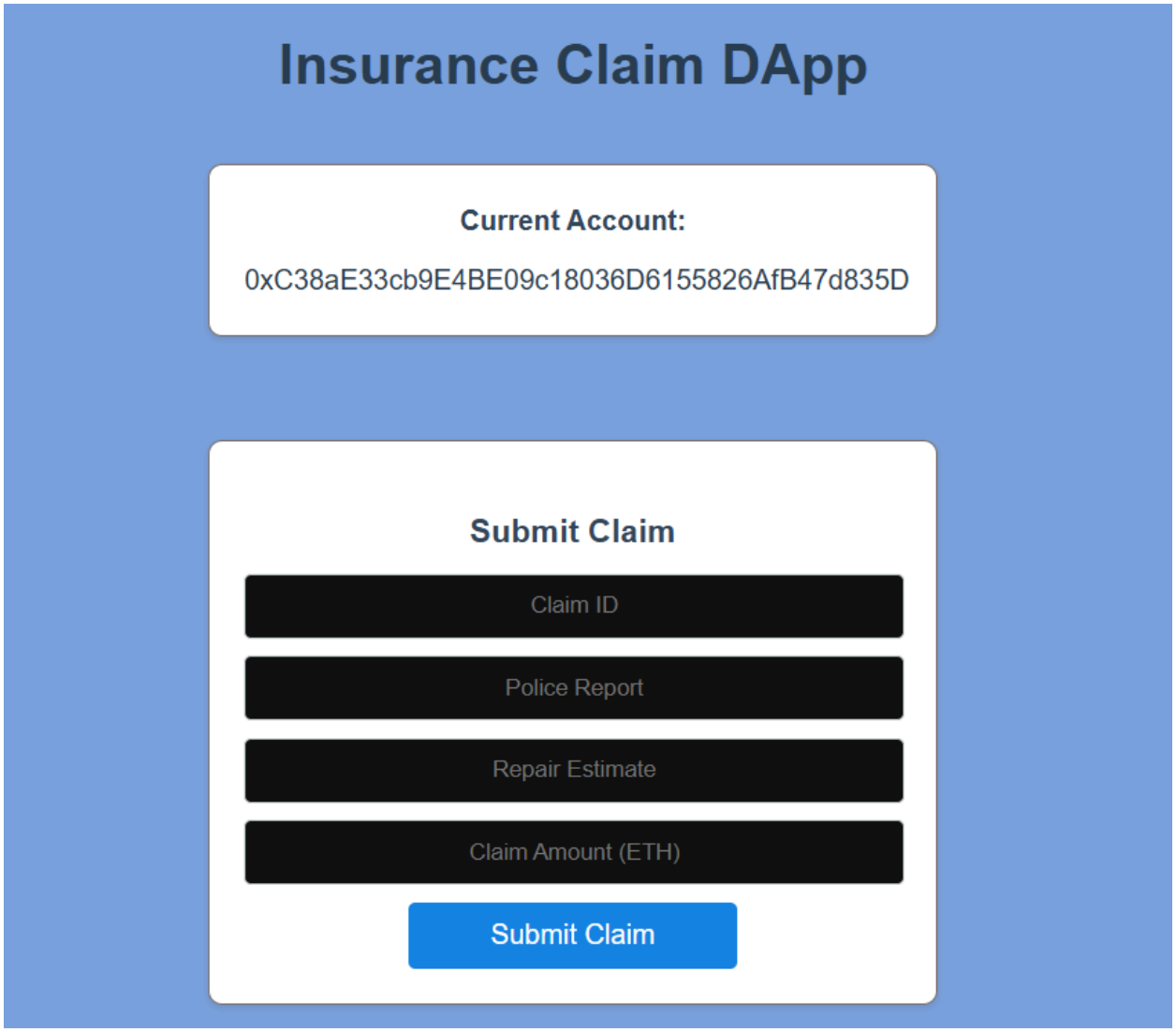
```
                <button  className="button"  onClick={approveClaim}>Approve
Claim</button>
        </div>

        <div className="claim-reject">
          <h3>Reject Claim</h3>
          <input
            type="text"
            placeholder="Claim ID"
            onChange={(e) => setClaimId(e.target.value)}
          />
                  <button  className="button"  onClick={rejectClaim}>Reject
Claim</button>
        </div>

        <div className="claim-status">
          <h3>Get Claim Status</h3>
          <input
            type="text"
            placeholder="Claim ID"
            onChange={(e) => setClaimId(e.target.value)}
          />
                  <button  className="button"  onClick={getClaimStatus}>Get
Status</button>
          <p>{claimStatus}</p>
        </div>
      </div>

      {message && <p className="message">{message}</p>}
    </div>
  );
};

export default App;
```

App.css

```
.main-container {
  display: flex;
```

```
  flex-direction: column;
  align-items: center;
  justify-content: center; /* Center vertically */
  font-family: Arial, sans-serif;
  min-height: 100vh; /* Ensure full height for vertical centering */
  text-align: center; /* Center text inside elements */
  background-color: #7ba2dd; /* Light background color */
  margin: 0 auto; /* Center horizontally */
  max-width: 800px; /* Limit maximum width */
}

.title {
  margin: 20px;
  font-size: 2em;
  color: #2c3e50; /* Dark blue color for title */
}

.main-content, .main-process {
  width: 60%;
  max-width: 800px;
  margin: 20px auto; /* Center horizontally */
}

.account, .claim-form, .claim-approve, .claim-reject, .claim-status {
  background: #ffffff; /* White background for sections */
  padding: 20px;
  border: 1px solid #7b7070; /* Light grey border */
  border-radius: 8px;
  margin-bottom: 20px;
  width: 90%; /* Ensure elements take full width for alignment */
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1); /* Slight shadow for depth */
  color: #2c3e50; /* Dark blue color for text */
}

.account b, .claim-form h3, .claim-approve h3, .claim-reject h3,
.claim-status h3 {
  display: block;
  margin-bottom: 10px;
  color: #34495e; /* Slightly darker blue for headings */
}
```

```
input[type="text"] {
  width: calc(100% - 20px);
  padding: 10px;
  margin-bottom: 10px;
  border: 1px solid #bdc3c7; /* Light grey border for inputs */
  border-radius: 4px;
  text-align: center; /* Center the text inside the input fields */
}

.button {
  width: 50%;
  padding: 10px;
  background-color: #1784e4; /* Blue color for buttons */
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
   transition: background-color 0.3s; /* Smooth transition for hover effect
*/
}

.button:hover {
  background-color: #3498db; /* Lighter blue on hover */
}

.message {
  color: #e74c3c; /* Red color for error messages */
  font-weight: bold;
}
```


Insurance Claim DApp
Current Account:
0xC38aE33cb9E4BE09c18036D6155826AfB47d835D
Submit Claim
Claim ID
Police Report
Repair Estimate
Claim Amount (ETH)
Submit Claim


Approve Claim
Claim ID
Approve Claim
Reject Claim
Claim ID
Reject Claim
Get Claim Status
Claim ID
Get Status

11.0 Conclusion

This project has demonstrated the transformative potential of smart contracts within the blockchain ecosystem, specifically leveraging Solidity on the Ethereum platform. By implementing fundamental features of an insurance claim processing system, the project has showcased the efficiency, transparency, and security that blockchain technology can offer. Through the use of smart contracts, the system automates the verification and execution of insurance claims, reducing the need for manual intervention and minimizing the risk of fraud. The decentralized nature of the Ethereum blockchain ensures that all transactions are immutable and transparent, enhancing trust among stakeholders.

Despite encountering several constraints, the project has provided valuable insights and recommendations for overcoming these challenges, paving the way for future enhancements. The integration of AI and machine learning for predictive analytics, as well as the development of comprehensive dashboards, offers promising avenues for further innovation. In conclusion, this project illustrates the immense benefits of integrating smart contracts and blockchain technology into traditional insurance processes. By harnessing the power of Ethereum and Solidity, the system achieves unparalleled levels of efficiency, security, and transparency. This project not only addresses current industry needs but also sets a strong foundation for future advancements in the realm of decentralized finance and automated contract execution.

12.0 Recommendations

Since the proposed features are fundamental and basic components of an insurance claim processing system, several constraints have arisen. To address these, some recommendations are suggested for improvement:

1. Integrating Artificial Intelligence (AI) for automated claim verification and Machine Learning (ML) for predictive analytics. AI can automate the verification of claims by analyzing data quickly and accurately. This reduces the time taken to process claims, which traditionally requires manual review by human agents. Machine Learning models can analyze historical claim data to identify patterns and predict the likelihood of fraud or error. This leads to more accurate assessments, reducing the chances of incorrect claim approvals or denials.

2. Third-Party Integrations by connecting with external services such as repair shops and financial institutions. Integrating with repair shops allows for direct communication and updates regarding the repair status of insured items (e.g., vehicles). This eliminates the need for customers to manually provide repair details. Linking with financial institutions can automate the payment process, ensuring that claims are settled quickly and efficiently without manual handling.

3. Real-Time Notifications by implementing push notifications and SMS alerts for claim updates. Customers receive immediate updates about their claim status, such as approval, processing, or required additional information. By keeping customers informed throughout the claim process, it reduces uncertainty and enhances their overall experience, leading to higher satisfaction rates.

4. Develop advanced analytics dashboards for better insights. Insurance companies can use these dashboards to gain insights into claim trends, customer behavior, and operational performance. This data-driven approach helps in making informed decisions about policy adjustments, risk management, and resource allocation. Also, customers can access detailed reports about their claims and policies, helping them understand their coverage, claim history, and potential future risks.

5. Customer Support with chatbots and live agents. Customers can get help anytime they need it, without waiting for business hours. Chatbots can handle common queries and guide users through processes, while live agents can handle more complex issues. Providing round-the-clock support ensures that customers feel valued and supported, increasing their overall satisfaction and loyalty to the insurance provider.

Integrating AI and automation, third-party services, real-time notifications, advanced analytics, and 24/7 customer support can significantly enhance the efficiency, accuracy, and user experience of the claim processing system. These upgrades not only streamline operations but also provide customers with timely, relevant information and support, leading to better service and higher satisfaction.

13.0 References :

Ante, Lennart. “Smart Contracts on the Blockchain – a Bibliometric Analysis and Review.” *Telematics and Informatics*, Oct. 2020, p. 101519, https://www.sciencedirect.com/science/article/pii/S0736585320301787?casa_token=ctfYmc5_tbcAAAAA:YLEpqkUsG50jHgM-TzFJ7ycJ2dczmD5v_8MQ11X0R9iiP1O46qDR6KESlNumHc6-1bGvLTS53BjK#bi005. Accessed 25 Oct. 2020.

Khan, Shafaq Naheed, et al. “Blockchain Smart Contracts: Applications, Challenges, and Future Trends.” *Peer-To-Peer Networking and Applications*, vol. 14, no. 1, 18 Apr. 2021, pp. 2901–2925, link.springer.com/article/10.1007/s12083-021-01127-0, https://link.springer.com/article/10.1007/s12083-021-01127-0.

Gatteschi, Valentina, et al. “Blockchain and Smart Contracts for Insurance: Is the Technology Mature Enough?” *Future Internet*, vol. 10, no. 2, 20 Feb. 2018, p. 20, www.mdpi.com/1999-5903/10/2/20, https://www.mdpi.com/1999-5903/10/2/20