# AUDIT CHAIN

## AMOS PROJECT SS2022

Supervisor Prof. Dirk Riehel

Friedrich-Alexander-University Erlangen-Nuremberg

July 27, 2022



# Outline


1. Introduction
2. Blockchain
3. RabbitMQ
4. Docker
5. Producer Dummy
6. Consumer Dummy
7. GUI
8. Testing
9. Conclusion

# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# Introduction

- Our project is a middleware based on blockchain data structures to guarantee the unchanged, compliant, in sequence, fault tolerant and buffered data flow between any kind of producers and consumers.

# Introduction

- Our project is a middleware based on blockchain data structures to guarantee the unchanged, compliant, in sequence, fault tolerant and buffered data flow between any kind of producers and consumers.
- Our partner is GRAU DATA, which specializes:

# Introduction

- Our project is a middleware based on blockchain data structures to guarantee the unchanged, compliant, in sequence, fault tolerant and buffered data flow between any kind of producers and consumers.
- Our partner is GRAU DATA, which specializes:
  - in audit-proof archiving,

# Introduction

- Our project is a middleware based on blockchain data structures to guarantee the unchanged, compliant, in sequence, fault tolerant and buffered data flow between any kind of producers and consumers.
- Our partner is GRAU DATA, which specializes:
  - in audit-proof archiving,
  - hierarchical storage management,

# Introduction

- Our project is a middleware based on blockchain data structures to guarantee the unchanged, compliant, in sequence, fault tolerant and buffered data flow between any kind of producers and consumers.
- Our partner is GRAU DATA, which specializes:
  - in audit-proof archiving,
  - hierarchical storage management,
  - ransomware protection,

# Introduction

- Our project is a middleware based on blockchain data structures to guarantee the unchanged, compliant, in sequence, fault tolerant and buffered data flow between any kind of producers and consumers.
- Our partner is GRAU DATA, which specializes:
  - in audit-proof archiving,
  - hierarchical storage management,
  - ransomware protection,
  - and scale-out storage solutions.

# Introduction

- Our mission is to deliver a MVP of a audit-proof recording of file system events to Grau Data.

# Introduction

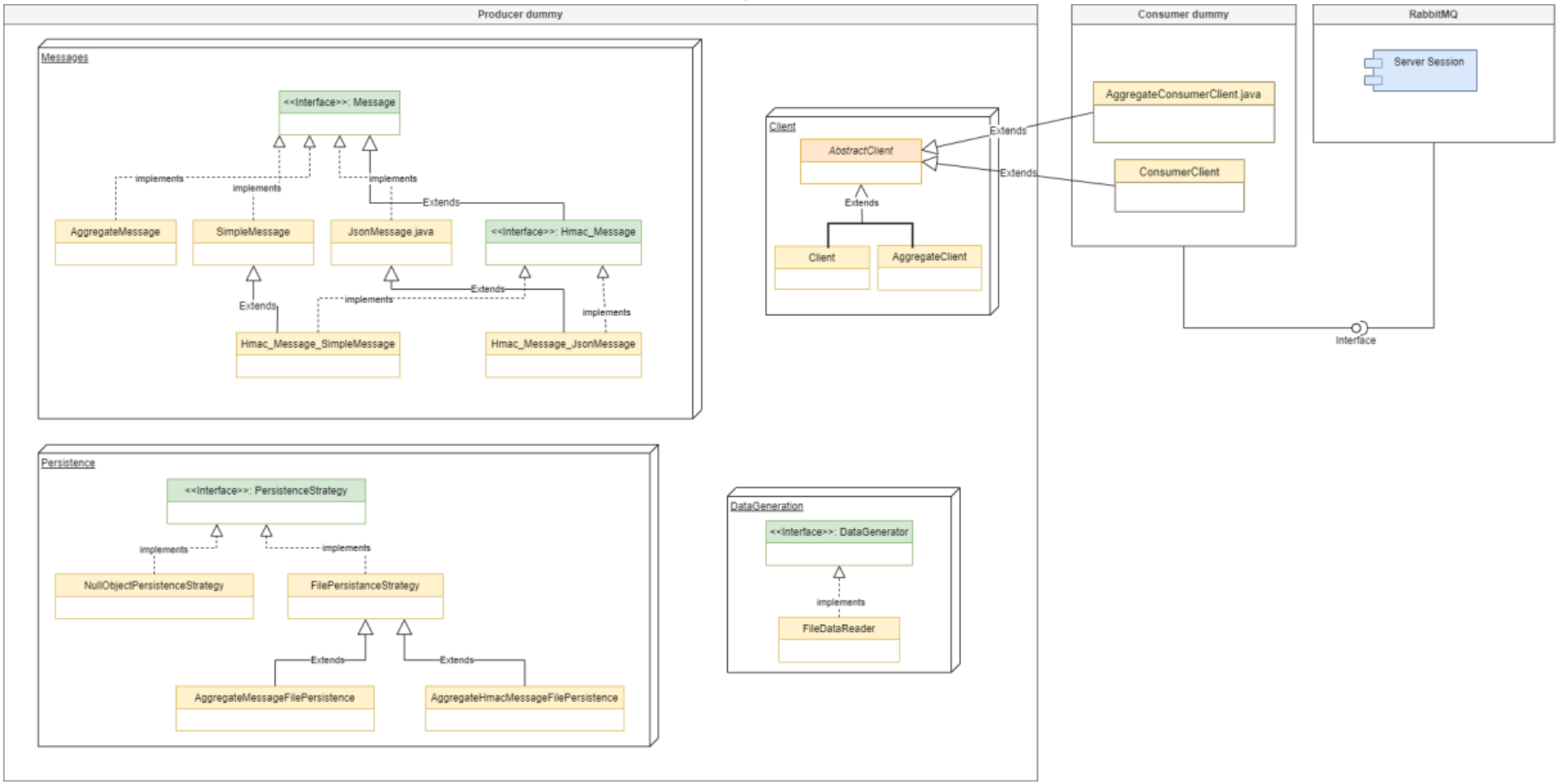
- Our mission is to deliver a MVP of a audit-proof recording of file system events to Grau Data.
- For this purpose there is a **central event queue**, which receives events from a **producer** and provides them a **consumer**, which is built on a underlying **blockchain** data structure.

# Introduction

- Our mission is to deliver a MVP of a audit-proof recording of file system events to Grau Data.
- For this purpose there is a **central event queue**, which receives events from a **producer** and provides them a **consumer**, which is built on a underlying **blockchain** data structure.
- Finally, the event will be copied in a second consumer and will be deleted from the event queue.

# Introduction

## Here is a diagram of our work



# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# Blockchain

- A Blockchain is a data structure, which is characterized by being an immutable, ordered, ”back-linked list” of blocks of transactions.

# Blockchain

- A Blockchain is a data structure, which is characterized by being an immutable, ordered, ”back-linked list” of blocks of transactions.
- A blockchain is a digital ledger that stores data in groupings called blocks.

# Blockchain

- A Blockchain is a data structure, which is characterized by being an immutable, ordered, ”back-linked list” of blocks of transactions.
- A blockchain is a digital ledger that stores data in groupings called blocks.
- When a block is full, it is closed and linked to the preceding block, producing a data chain known as the blockchain.

# Blockchain

- A Blockchain is a data structure, which is characterized by being an immutable, ordered, ”back-linked list” of blocks of transactions.
- A blockchain is a digital ledger that stores data in groupings called blocks.
- When a block is full, it is closed and linked to the preceding block, producing a data chain known as the blockchain.
- It is not only a simple back linked list, in fact each block contains a hash of the previous block.

# Blockchain

- A Blockchain is a data structure, which is characterized by being an immutable, ordered, ”back-linked list” of blocks of transactions.
- A blockchain is a digital ledger that stores data in groupings called blocks.
- When a block is full, it is closed and linked to the preceding block, producing a data chain known as the blockchain.
- It is not only a simple back linked list, in fact each block contains a hash of the previous block.
- It is not possible to edit the blockchain by adding or removing blocks in the middle or changing them

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:
  - Greater assurance

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:
  - Greater assurance
  - Greater safety

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:
  - Greater assurance
  - Greater safety
  - Enhanced efficiencies

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:
  - Greater assurance
  - Greater safety
  - Enhanced efficiencies
- The idea of a blockchain was first conceived as the mechanism supporting Bitcoin, but there are some more applications as well like:

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:
  - Greater assurance
  - Greater safety
  - Enhanced efficiencies
- The idea of a blockchain was first conceived as the mechanism supporting Bitcoin, but there are some more applications as well like:
  - Money transfers

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:
  - Greater assurance
  - Greater safety
  - Enhanced efficiencies
- The idea of a blockchain was first conceived as the mechanism supporting Bitcoin, but there are some more applications as well like:
  - Money transfers
  - Secure Internet of Things networks (IoT)

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:
  - Greater assurance
  - Greater safety
  - Enhanced efficiencies
- The idea of a blockchain was first conceived as the mechanism supporting Bitcoin, but there are some more applications as well like:
  - Money transfers
  - Secure Internet of Things networks (IoT)
  - Data storage

# Benefits and Application of Blockchain

- The way data of Blockchain is structured differs significantly between a traditional database and it has plenty of benefits benefits such as:
  - Greater assurance
  - Greater safety
  - Enhanced efficiencies
- The idea of a blockchain was first conceived as the mechanism supporting Bitcoin, but there are some more applications as well like:
  - Money transfers
  - Secure Internet of Things networks (IoT)
  - Data storage
  - Real Estate etc.

# Implementation of Blockchain

- For the implementation of Blockchain, Java was used as the programming language and also a seperated branch on IntelIJ was created.

# Implementation of Blockchain

- For the implementation of Blockchain, Java was used as the programming language and also a seperated branch on IntelIJ was created.
- Each block contains a hashmap of messages.

# Implementation of Blockchain

- For the implementation of Blockchain, Java was used as the programming language and also a seperated branch on IntelIJ was created.
- Each block contains a hashmap of messages.
- At the same time, we ensured the right sequence of incoming messages and the consistent message flow.

# Implementation of Blockchain

- For the implementation of Blockchain, Java was used as the programming language and also a seperated branch on IntelIJ was created.
- Each block contains a hashmap of messages.
- At the same time, we ensured the right sequence of incoming messages and the consistent message flow.
- Some of the features that we implemented are the following:

# Implementation of Blockchain

- For the implementation of Blockchain, Java was used as the programming language and also a seperated branch on IntelIJ was created.
- Each block contains a hashmap of messages.
- At the same time, we ensured the right sequence of incoming messages and the consistent message flow.
- Some of the features that we implemented are the following:
  - Added the feature that checks automatically itself to see if some messages stored have been tempered,

# Implementation of Blockchain

- For the implementation of Blockchain, Java was used as the programming language and also a seperated branch on IntelIJ was created.
- Each block contains a hashmap of messages.
- At the same time, we ensured the right sequence of incoming messages and the consistent message flow.
- Some of the features that we implemented are the following:
  - Added the feature that checks automatically itself to see if some messages stored have been tempered,
  - Added the feature that checks if there is a specific block in the blockchain and if it contains tempered messages,

# Implementation of Blockchain

- For the implementation of Blockchain, Java was used as the programming language and also a seperated branch on IntelIJ was created.
- Each block contains a hashmap of messages.
- At the same time, we ensured the right sequence of incoming messages and the consistent message flow.
- Some of the features that we implemented are the following:
  - Added the feature that checks automatically itself to see if some messages stored have been tempered,
  - Added the feature that checks if there is a specific block in the blockchain and if it contains tempered messages,
  - Added the feature that checks automatically itself to see if some messages stored in a range have been tempered.

# Implementation of Blockchain

- For the implementation of Blockchain, Java was used as the programming language and also a seperated branch on IntelIJ was created.
- Each block contains a hashmap of messages.
- At the same time, we ensured the right sequence of incoming messages and the consistent message flow.
- Some of the features that we implemented are the following:
  - Added the feature that checks automatically itself to see if some messages stored have been tempered,
  - Added the feature that checks if there is a specific block in the blockchain and if it contains tempered messages,
  - Added the feature that checks automatically itself to see if some messages stored in a range have been tempered.
- Finally, we tested our program by creating the JUnittest with a direct channel between producer dummy and consumer without the queue.

# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:
  - Reliability

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:
  - Reliability
  - Many Clients

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:
  - Reliability
  - Many Clients
  - Multi-protocol

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:
  - Reliability
  - Many Clients
  - Multi-protocol
  - High Available Queues

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:
  - Reliability
  - Many Clients
  - Multi-protocol
  - High Available Queues
- The data needs to be transmitted via RabbitMQ.

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:
  - Reliability
  - Many Clients
  - Multi-protocol
  - High Available Queues
- The data needs to be transmitted via RabbitMQ.
- In our project, after the producer dummy successfully generates an event, it goes to RabbitMQ (event queue).

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:
  - Reliability
  - Many Clients
  - Multi-protocol
  - High Available Queues
- The data needs to be transmitted via RabbitMQ.
- In our project, after the producer dummy successfully generates an event, it goes to RabbitMQ (event queue).
- In order to ensure the proper functioning of RabbitMQ, we proceeded to install it inside the Virtual Machine.

# RabbitMQ

- RabbitMQ is a messaging broker, which offers applications a common platform for sending and receiving messages, ensuring that messages are protected until they are received.
- The benefits of RabbitMQ are the following:
  - Reliability
  - Many Clients
  - Multi-protocol
  - High Available Queues
- The data needs to be transmitted via RabbitMQ.
- In our project, after the producer dummy successfully generates an event, it goes to RabbitMQ (event queue).
- In order to ensure the proper functioning of RabbitMQ, we proceeded to install it inside the Virtual Machine.
- RabbitMQ is a 3rd party program that we are using as a component for queuing events.

# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# Docker

- Docker is a containerization platform available as an open source project.

# Docker

- Docker is a containerization platform available as an open source project.
- A container combines application source code with the OS libraries and dependencies required for running the application in any environment.

# Docker

- Docker is a containerization platform available as an open source project.
- A container combines application source code with the OS libraries and dependencies required for running the application in any environment.
- Developers can package applications into containers.

# Docker

- Docker is a containerization platform available as an open source project.
- A container combines application source code with the OS libraries and dependencies required for running the application in any environment.
- Developers can package applications into containers.
- We set Docker instead of Virtual Machine because we want:

# Docker

- Docker is a containerization platform available as an open source project.
- A container combines application source code with the OS libraries and dependencies required for running the application in any environment.
- Developers can package applications into containers.
- We set Docker instead of Virtual Machine because we want:
  - our system resources to be used efficiently (consume less main memory, easy to port etc),

# Docker

- Docker is a containerization platform available as an open source project.
- A container combines application source code with the OS libraries and dependencies required for running the application in any environment.
- Developers can package applications into containers.
- We set Docker instead of Virtual Machine because we want:
  - our system resources to be used efficiently (consume less main memory, easy to port etc),
  - the software works better.

# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# Producer Dummy

- A producer dummy consists of three sub-components:
  - a database,
  - a data generator,
  - a persistence mechanism (storage buffer).

# Producer Dummy

- A producer dummy consists of three sub-components:
  - a database,
  - a data generator,
  - a persistence mechanism (storage buffer).
- When the producer dummy is triggered, the data generator catches data events (message) from the database and then forwards it to the client session and persistence mechanism.

# Producer Dummy

- A producer dummy consists of three sub-components:
  - a database,
  - a data generator,
  - a persistence mechanism (storage buffer).
- When the producer dummy is triggered, the data generator catches data events (message) from the database and then forwards it to the client session and persistence mechanism.
- The persistence mechanism is a contingent component that stores the last data event and gets triggered in case of any failure.

# Producer Dummy

- A producer dummy consists of three sub-components:
  - a database,
  - a data generator,
  - a persistence mechanism (storage buffer).
- When the producer dummy is triggered, the data generator catches data events (message) from the database and then forwards it to the client session and persistence mechanism.
- The persistence mechanism is a contingent component that stores the last data event and gets triggered in case of any failure.
- In our case, producer dummy creates the events that are then stored in the central queue which will be used later from consumer dummy.

# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# Consumer Dummy

- The consumer takes events from the queue in order to perform further actions with them (that are not of interest in our application).

# Consumer Dummy

- The consumer takes events from the queue in order to perform further actions with them (that are not of interest in our application).
- Consumers can approach the queue, in order to get the events and work with them accordingly. Events are recorded serially and the central event queue is transmitted via the network queue.

# Consumer Dummy

- The consumer takes events from the queue in order to perform further actions with them (that are not of interest in our application).
- Consumers can approach the queue, in order to get the events and work with them accordingly. Events are recorded serially and the central event queue is transmitted via the network queue.
- In our case, after the implementation of producer dummy, our consumer dummy would use the events that are created and stored in the central queue for further processing, like storing.

# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# GUI

- A Graphical User Interface (GUI) allows users to interact with computing devices without entering text commands into a console.

# GUI

- A Graphical User Interface (GUI) allows users to interact with computing devices without entering text commands into a console.
- Users can perform any action without having to know any programming language in this friendly visual environment.

# GUI

- A Graphical User Interface (GUI) allows users to interact with computing devices without entering text commands into a console.
- Users can perform any action without having to know any programming language in this friendly visual environment.
- In our project, we implemented a nice GUI and at the same time we improved the functionality of Blockchain with the following features:
  - from sequence number return the message if tempered,
  - from two sequence number return the tempered messages in that interval,
  - show how many events does the Blockchain hold,
  - show how many files does the Blockchain hold,
  - show much space the Blockchain occupies.

# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# Testing

- JUnit is a unit testing open-source framework for the Java programming language.

# Testing

- JUnit is a unit testing open-source framework for the Java programming language.
- We use this framework to write and execute automated tests.

# Testing

- JUnit is a unit testing open-source framework for the Java programming language.
- We use this framework to write and execute automated tests.
- However, in Java, there are test cases that must be re-executed every time a new code is added.

# Testing

- JUnit is a unit testing open-source framework for the Java programming language.
- We use this framework to write and execute automated tests.
- However, in Java, there are test cases that must be re-executed every time a new code is added.
- This is done to make sure that nothing in the code is broken.

# Testing

- JUnit is a unit testing open-source framework for the Java programming language.
- We use this framework to write and execute automated tests.
- However, in Java, there are test cases that must be re-executed every time a new code is added.
- This is done to make sure that nothing in the code is broken.
- We have JUnit Tests for the following cases:
  - JUnit Tests Producer Dummy
  - JUnit Tests Consumer Dummy
  - JUnit Tests Blockchain

# Table of Contents

1 Introduction
2 Blockchain
3 RabbitMQ
4 Docker
5 Producer Dummy
6 Consumer Dummy
7 GUI
8 Testing
9 Conclusion

# Conclusion

We had a great time together and we really enjoyed our time!