*Bachelor Thesis*  
Leuphana Universität Lüneburg  
Major Wirtschaftsinformatik

Analyse, Entwurf und prototypische Realisierung einer Flutter-Anwendung mit Web3-Authentifizierungsverfahren und Autorisierung durch ERC-721 Token am Beispiel eines Minigames

Abschlussarbeit

zur Erlangung des akademischen Grades

|  |  |
| --- | --- |
| 1. Gutachter: | Prof. Dr. Guido Barbian |
| 2. Gutachter: | Prof. Dr. Jan Wilk |
| vorgelegt von: | Anton Stamme |
| vorgelegt am: | 15. 09. 2022 |

**Bachelor of Science (B.Sc.)**

# Abstract/Zusammenfassung

**Willkommen** und vielen Dank, dass Sie sich für diese Dokumentvorlage entschieden haben.

# Inhaltsverzeichnis

[Abstract/Zusammenfassung I](#_Toc108185466)

[Inhaltsverzeichnis II](#_Toc108185467)

[Abkürzungsverzeichnis IV](#_Toc108185468)

[Tabellenverzeichnis V](#_Toc108185469)

[Abbildungsverzeichnis VI](#_Toc108185470)

[1 Einführung 1](#_Toc108185471)

[1.1 Motivation 1](#_Toc108185472)

[1.2 Zielsetzung und Aufgabenstellung 1](#_Toc108185473)

[1.3 Aufbau und Arbeitsweise 1](#_Toc108185474)

[2 Grundlagen 2](#_Toc108185475)

[2.1 Blockchain Technologie 2](#_Toc108185476)

[2.1.1 Crypto-Accounts und -Wallets 2](#_Toc108185477)

[2.1.2 Smart Contracts 2](#_Toc108185478)

[2.1.3 NFTs 2](#_Toc108185479)

[2.2 Ethereum 2](#_Toc108185480)

[2.2.1 ERC-721 2](#_Toc108185481)

[2.2.2 Solidity 2](#_Toc108185482)

[2.2.3 IPFS 2](#_Toc108185483)

[2.3 IT-Sicherheit 2](#_Toc108185484)

[2.3.1 Web3 Authentifizierung 2](#_Toc108185485)

[2.3.2 Autorisierung mittels NFTs 2](#_Toc108185486)

[2.4 Cross-Platform-Entwicklung mit Flutter 2](#_Toc108185487)

[3. Systemanalyse 3](#_Toc108185488)

[3.1 Flappy Bird 3](#_Toc108185489)

[3.1.1 Geschichte 3](#_Toc108185490)

[3.1.2 Konzept 4](#_Toc108185491)

[3.1.3 NFTs in Flappy Bird 4](#_Toc108185492)

[3.2 Anforderungsanalyse 5](#_Toc108185493)

[3.2.1 Smart Contract 6](#_Toc108185494)

[3.2.2 Flutter Anwendung 6](#_Toc108185495)

[4. Systementwurf 9](#_Toc108185496)

[4.1 Architektur 9](#_Toc108185497)

[4.2 Flutter App 9](#_Toc108185498)

[4.3 Smart Contracts 9](#_Toc108185499)

[5. Prototyische Realisierung 10](#_Toc108185500)

[5.1 Smart Contracts 10](#_Toc108185501)

[5.1.1 Ganache 10](#_Toc108185502)

[5.1.2 Skins 10](#_Toc108185503)

[5.1.3 Perks 10](#_Toc108185504)

[5.2 Flutter App 10](#_Toc108185505)

[5.2.1 Flutter Bird 10](#_Toc108185506)

[5.2.2 Web3-Authentifizierung 10](#_Toc108185507)

[5.2.3 Autorisierung mit NFTs 10](#_Toc108185508)

[6. Vorführung der Anwendung 11](#_Toc108185509)

[7. Bewertung des entwickelten Prototyps 12](#_Toc108185510)

[8. Fazit 13](#_Toc108185511)

[Literaturverzeichnis 14](#_Toc108185512)

[Literaturverzeichnis als sortierbare Tabelle 14](#_Toc108185513)

[Standard-Literaturverzeichnis 14](#_Toc108185514)

[Index 15](#_Toc108185515)

[Eidesstattliche Erklärung 16](#_Toc108185516)

# Abkürzungsverzeichnis

Ein Abkürzungsverzeichnis ist nicht für jede Arbeit notwendig. Es enthält nur fachspezifische Abkürzungen, die nicht im Duden aufgeführt bzw. allgemein gebräuchlich sind.

Die Abkürzungen stehen hier in einer unsichtbaren Tabelle ohne Rahmen (Tabellenvorlage: Unsichtbare Tabelle). Mehr dazu weiter unten.

|  |  |
| --- | --- |
| Abk. 1: | Beschreibung |
| Abk. 2: | Beschreibung |
| Abk. 3: | Beschreibung |

# Tabellenverzeichnis

[Tabelle 1: Unsichtbare Steuerzeichen 3](#_Toc4947295)

[Tabelle 2: Tabellenvariante 1 (empfohlene Variante). Tabellen-Formatvorlage: Standardtabelle 14](#_Toc4947296)

[Tabelle 3: Tabellenvariante 2. Tabellen-Formatvorlage: Standardtabelle   
ohne Linien 14](#_Toc4947297)

[Tabelle 6: Formatvorlagen 19](#_Toc4947298)

[Tabelle 7: Shortcuts 20](#_Toc4947299)

# Abbildungsverzeichnis

[Abbildung 1: Seitenränder 1](#_Toc3093970)

[Abbildung 2: Steuerzeichen einblenden, Word 2007 3](#_Toc3093971)

[Abbildung 3: Steuerzeichen einblenden, Word 2000, 2002, 2003 3](#_Toc3093972)

[Abbildung 4: Shortcut ändern 4](#_Toc3093973)

[Abbildung 5: Autoformat während der Eingabe 5](#_Toc3093974)

[Abbildung 6: Beschriftung einfügen. 12](#_Toc3093975)

[Abbildung 7: Tabelle beschriften 13](#_Toc3093976)

[Abbildung 8: Aktualisierung 15](#_Toc3093977)

[Abbildung 9: Überschriften ohne Nummerierung 15](#_Toc3093978)

[Abbildung 10: Seitenzahl 16](#_Toc3093979)

[Abbildung 11: Zeile zu lang 16](#_Toc3093980)

**Hinweis:** Auf diesen Absatz folgt ein Abschnittswechsel, der nicht gelöscht werden sollte. Er ermöglicht den Wechsel von römischen zu arabischen Seitenzahlen. Mehr Infos zum Abschnittswechsel weiter unten. Drücken Sie STRG+SHIFT+PLUS um unsichtbare Steuerzeichen und damit den Abschnittswechsel ein- und auszublenden.

# 1 Einführung

## 1.1 Motivation

## 1.2 Zielsetzung und Aufgabenstellung

## 1.3 Aufbau und Arbeitsweise

# 2 Grundlagen

## 2.1 Blockchain Technologie

### 2.1.1 Crypto-Accounts und -Wallets

### 2.1.2 Smart Contracts

### 2.1.3 NFTs

## 2.2 Ethereum

### 2.2.1 ERC-721

### 2.2.2 Solidity

### 2.2.3 IPFS

## 2.3 IT-Sicherheit

### 2.3.1 Web3 Authentifizierung

### 2.3.2 Autorisierung mittels NFTs

## 2.4 Cross-Platform-Entwicklung mit Flutter

# 3. Systemanalyse

Die Verfahren der Web3-Authentifizierung und der Autorisierung durch NFTs, sollen anhand eines Minigames demonstriert werden. In diesem Abschnitt wird ein Minigame-Konzept vorgestellt, an welchem sich genannte Verfahren demonstrieren lassen. Die genauen Anforderungen an das System werden festgelegt, und bieten die Grundlage für den darauf folgenden Entwurf.

## 3.1 Flappy Bird

Als Spielkonzept wird Flappy Bird gewählt. Da die Entwicklung des Spiels nicht im Mittelpunkt dieser Arbeit steht, bietet sich ein triviales und schnell verstandenes Minigame-Konzept wie Flappy Bird an. Der Entwicklungsaufwand ist nicht sonderlich groß, und das Spiel lässt sich durch Funktionalitäten ergänzen, an welchen sich die Autorisierung mit NFTs sowie die dafür erforderliche Web3-Authentifizierung, demonstrieren lässt.

### 3.1.1 Geschichte

Flappy Bird ist ein Minigame, welches im Jahr 2013 von dem vietnamesischen Entwickler Dong Nguyen für iOS und Android Geräte veröffentlicht wurde (source). Nachdem das Spiel für über ein halbes Jahr kaum Aufmerksamkeit erlangte, erlebte es Anfang 2014 eine enorme Hypewelle, welche das Spiel im Apple AppStore zwischenzeitlich auf den 1. Platz der am häufigsten heruntergeladenen Spiele brachte (source). Die ikonische Geschichte von Flappy Bird endete schon kurze Zeit später am 10. Februar 2014, als der Entwickler, sich trotz enormen – auch wirtschaftlichen – Erfolges, dazu entschied das Spiel aufgrund von Kritik aus dem AppStore und Google PlayStore zu entfernen (source). Seinen plötzlichen Erfolg hat das Spiel wohl seiner absoluten Simplizität zu verdanken, welche gleichzeitig einen gewissen Suchtfaktor mit sich brachte. Das Spiel, dessen Entwicklung laut Angaben von Entwickler Dong Nguyen nur zwei bis drei Tage dauerte (source), funktioniert wie folgt:

### 3.1.2 Konzept

Die Spielfigur ist ein kleiner gelber Vogel, welcher sich konstant durch eine zweidimensionale Spielwelt nach rechts bewegt. Auf dem Weg begegnet der Vogel in regelmäßigen Abständen Hindernissen, den es auszuweichen gilt. Das einzige was der Spieler hierfür tun muss, ist es, auf den Bildschirm zu tippen, um den Vogel „springen“ zu lassen. Dabei ist es völlig egal wo auf dem Bildschirm getippt wird. Die Hindernisse werden durch vertikale grüne Röhren dargestellt, welche auf einer zufälligen Höhe eine Öffnung zum Durchfliegen haben (siehe Abbildung 1). Sobald der Vogel eine Röhre berührt, stürzt er ab, und die Runde endet. Ziel des Spiel ist es, in einem Durchlauf durch möglichst viele Röhren zu fliegen, ohne dabei mit einer zu kollidieren. Die Anzahl der durchflogenen Röhren ergibt die Punktzahl, an der sich gemessen werden kann.

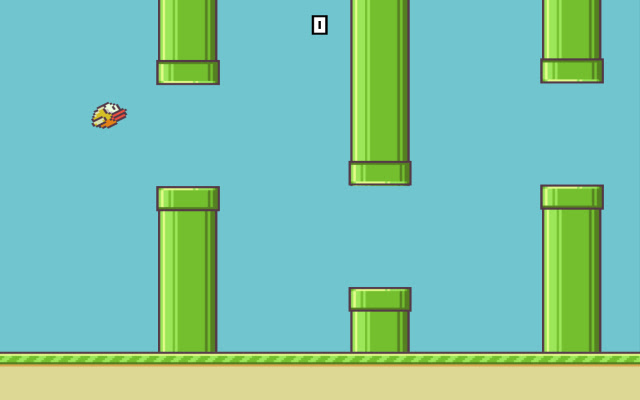


Abbildung 1: FlappyBird Screenshot

### NFTs in Flappy Bird

Damit an diesem Spielkonzept ein Autorisierungsverfahren mit NFTs veranschaulicht werden kann, wird es durch Funktionalitäten ergänzt, die erst nach erfolgreicher Autorisierung eines bestimmten NFTs freigeschaltet werden. Auf die folgenden Zwei Weisen, soll von NFTs Gebrauch gemacht werden können:

Skins

Bestimmte NFTs sollen dazu genutzt werden können im Spiel alternatives Aussehen für den Vogel („Skins“) freizuschalten. Wird der Besitz eines Skin-NFTs nachgewiesen, hat der Spieler vor dem Start der Runde die Möglichkeit dieses alternative Aussehen zu wählen. Diese „Skins“ stellen eine rein optische Veränderung des Spiels dar, und bieten keine tatsächlichen Vorteile.

Perks

Andere NFTs sollen dazu genutzt werden können bestimmte Vorteile im Spiel („Perks“) zu gewähren. Dazu könnten zum Beispiel Power-Ups wie ein Schild gehören, welches im Spiel eingesammelt werden kann, um die nächste Kollision mit einer Röhre zu vermeiden. Ein weiteres Power-Up könnte nach dem Einsammeln das Spiel für eine gewisse Zeit verlangsamen, was das Spiel vereinfacht.

## Anforderungsanalyse

Die Anforderungsanalyse ist ein wichtiger Bestandteil des Softwareentwicklungsprozess. Anhand von möglichen Anwendungsszenarien werden die Kernfunktionen und damit die funktionalen Anforderungen an das System festgelegt.

Da die Rahmenbedingungen dieses Projektes vorgeben, dass es sich um eine Flutter Anwendung sowie um ERC-721 Token handeln wird, lässt sich bereits in der Analysephase zwischen den beiden entscheidenden Komponenten differenzieren: Zum einen die Flutter Anwendung, in welcher das Spielkonzept sowie die Verfahren der Web3 Authentifizierung und der Autorisierung durch NFTs realisiert werden. Und zum anderen der ERC-721 Smart Contract, welcher auf einer Blockchain hinterlegt wird und für die Verwaltung der NFTs zuständig ist, von denen in der Flutter Anwendung Gebrauch gemacht werden kann.

Ziel dieser Analyse sind konkrete Listen der Anforderungen an den Smart Contract sowie an die Flutter Anwendung, sodass im Anschluss ein Entwurf des Gesamtsystems angefertigt werden kann. Das zu entwickelnde Gesamtsystem wird von nun an als „**Flutter Bird**“ bezeichnet, da es das Spielkonzept von Flappy Bird nachahmen soll, und mit Flutter entwickelt wird.

### 3.2.1 Smart Contract

Die grundlegenden Anforderungen an einen ERC-721 Smart Contract sind fest vorgeschrieben und gelten universell für jeden ERC-721 Smart Contract (siehe 2.2.1 ERC-721). Darüber hinaus sollen die von diesem Smart Contract verwalteten NFTs folgende Eigenschaften besitzen:

|  |  |
| --- | --- |
| Zufällig generierte Skins | Der Smart Contract soll jedem Token eine zufällig generierte Bilddatei zuordnen können, welche sich als Skin im Flutter Bird Spiel eignet. |
| Hohe Varianz an Skins | Der Smart Contract soll mindestens 500 Token verwalten, um eine hinreichende Varianz an Skins zu ermöglichen. |
| Metadaten | Der Smart Contract soll jedem Token individuelle Metadaten zuweisen können, welche die Eigenschaften und Merkmale des dazugehörigen Skins festhalten. |
| Bilddatein im dezentralen Datenspeicher | Die Metadaten eines jeden Tokens sollen eine Referenz zu der entsprechenden Bilddatei des Skins enthalten, welche im IPFS gespeichert ist. |
|  |  |
|  |  |
|  |  |

### 3.2.2 Flutter Anwendung

Um einen Überblick über mögliche Anwendungsszenarien zu schaffen, wird die Flutter Anwendung in einem UML Use Case Diagrammen dargestellt. Abbildung 2 zeigt einen imaginären Nutzer der App und die potentiellen Anwendungsszenarien, durch welche dieser Nutzer mit der App interagieren kann. Zentrales Anwendungsszenario wird es sein, Flutter Bird zu spielen. Das in Kapitel 3.1 Flappy Bird gezeigte Spielkonzept soll nachgeahmt werden, sodass es dem Ursprungsspiel im Aussehen und in der Funktionsweise ähnelt. Dieses Szenario wird durch die Möglichkeit erweitert, das Spiel mit einem Skin-NFT anstelle der Standard-Spielfigur zu spielen. Damit der Nutzer dazu autorisiert werden kann, mit einem solchen Skin-NFT zu spielen, ist es erforderlich, dass die Anwendung mit dem auf der Blockchain hinterlegtem ERC-721 Smart Contract interagieren kann, um den Besitz eines solchen Skin-NFTs nachzuweisen (siehe 2.3.2 Autorisierung mittels NFTs). Dieser Nachweis wiederum erfordert, dass der Spieler die Möglichkeit hat sich innerhalb der Anwendung mit einem Crypto Wallet zu authentifizieren (siehe 2.3.1 Web3 Authentifizierung).

Diagram

Description automatically generated

Abbildung : Darstellung der Kernfunktionen von Flutter Bird durch ein Use Case Diagramm

Es lassen sich die folgenden Kernanforderungen an die Flutter Anwendung ableiten:

|  |  |
| --- | --- |
| Flutter Bird spielen | Die Anwendung soll das Spielkonzept von Flappy Bird nachahmen. Auch ohne den Besitz eines Skin-NFTs soll ein Nutzer das Spiel spielen können. |
| Spielen mit Skin-NFT | Die Anwendung soll es autorisierten Nutzern erlauben mit einem alternativen Skin zu spielen. Hierzu kann der Nutzer vor dem Start des Spiel zwischen allen besessenen Skins wählen. |
| Autorisierung durch NFT | Das System soll Nutzer dazu autorisieren können mit einem Skin-NFT zu spielen, indem der Besitz eines solchen nachgewiesen wird. |
| Authentifizierung mit Crypto Wallet | Das System soll ein Verfahren implementieren, welches die Authentifizierung mit einem Crypto-Wallet ermöglicht. |
|  |  |
|  |  |
|  |  |
|  |  |

# 4. Systementwurf

## 4.1 Architektur

## 4.2 Flutter App

## 4.3 Smart Contracts

# 5. Prototyische Realisierung

## 5.1 Smart Contracts

### 5.1.1 Ganache

### 5.1.2 Skins

### 5.1.3 Perks

## 5.2 Flutter App

### 5.2.1 Flutter Bird

### 5.2.2 Web3-Authentifizierung

### 5.2.3 Autorisierung mit NFTs

# 6. Vorführung der Anwendung

# 7. Bewertung des entwickelten Prototyps

# 8. Fazit

# Literaturverzeichnis

Entscheiden Sie sich für eine Variante des Verzeichnisses. Sie sind beide ähnlich formatiert, das eine ist eine sortierbare Tabelle, die manuell befüllt wird, das andere ist ein Standard-Verzeichnis. Formatiert wird das Verzeichnis mit der Absatzformatvorlage „Literaturverzeichnis“.

## Literaturverzeichnis als sortierbare Tabelle

|  |
| --- |
| Eco, Umberto: Wie man eine wissenschaftliche Abschlussarbeit schreibt. UTB, 2007. |
| Fahnenstich, Klaus/Haselier, G. Rainer: Microsoft Home and Student 2007: Microsoft Press Deutschland, 2007. |
| Karmasin, Matthias/Ribing, Rainer: Die Gestaltung wissenschaftlicher Arbeiten. UTB, 2008. |
| Nicol, Natascha/Albrecht, Ralf: Wissenschaftliche Arbeiten schreiben mit Word 2007. Addison-Wesley, 2007. |
| Ravens, Tobias: Wissenschaftlich mit Word arbeiten. Pearson Studium, 2004. |

## Standard-Literaturverzeichnis

Eco, Umberto: Wie man eine wissenschaftliche Abschlussarbeit schreibt. UTB, 2007.

Fahnenstich, Klaus/Haselier, G. Rainer: Microsoft Home and Student 2007: Microsoft Press Deutschland, 2007.Karmasin, Matthias, und Rainer Ribing. Die Gestaltung wissenschaftlicher Arbeiten. UTB, 2008.

Karmasin, Matthias/Ribing, Rainer: Die Gestaltung wissenschaftlicher Arbeiten. UTB, 2008.

Nicol, Natascha/Albrecht, Ralf: Wissenschaftliche Arbeiten schreiben mit Word 2007. Addison-Wesley, 2007.

Ravens, Tobias: Wissenschaftlich mit Word arbeiten. Pearson Studium, 2004.

# Index

Abbildungen 18

Abkürzungsverzeichnis V

Abstände 14

Arial 9

Automatische Nummerierung 12

Beschriftung 18

Bindestriche 16

Dokumentstruktur 9

Einzüge 15

Formatvorlagenleiste 8

Fußnoten 17

Gedankenstriche 16

Geschützte Leerzeichen 16

Inhaltsverzeichnis 22

Kopfzeilen 15

Nummerierungen 16

Satzspiegel 8

Shortcuts 10, 27

Steuerzeichen 9

Tabellen, unsichtbar 12

Überschriften 14

Überschriften, mehrere 14

Zitate 17

# Eidesstattliche Erklärung

Ich erkläre hiermit, dass ich diese Bachelorarbeit selbstständig ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel verfasst habe. Alle den benutzten Quellen wörtlich oder sinngemäß entnommenen Stellen sind als solche einzeln kenntlich gemacht.

Diese Arbeit ist bislang keiner anderen Prüfungsbehörde vorgelegt worden und auch nicht veröffentlicht worden.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ort, Datum, Unterschrift