MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Project Report

on

NFT Market Place

Submitted By:

Snehil Banawal

0901CS191121

Yash Jain

0901CS191140

Faculty Mentor:

Dr. Anjula Mehto

Assistant Professor, Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR - 474005 (MP) est. 1957

MAY-JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Project Report

on

NFT Market Place

A project report submitted in partial fulfilment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Snehil Banawal

0901CS191121

Yash Jain

0901CS191140

Faculty Mentor:

Dr. Anjula Mehto

Assistant Professor, Computer Science and Engineering

Submitted to:

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE

GWALIOR - 474005 (MP) est. 1957

MAY-JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

CERTIFICATE

This is certified that **Snehil Banawal** (0901cs191121) has submitted the project report titled **NFT Market Place** under the mentorship of **Dr. Anjula Mehto** in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering from Madhav Institute of Technology and Science, Gwalior.

[illegible]

Dr. Anjula Mehto
Faculty Mentor
Assistant Professor
Computer Science and Engineering

[illegible] 09/05/2022

Dr. Manish Dixit
Professor and Head,
Computer Science and Engineering

Dr. Manish Dixit
Professor & HOD
Department of CSE
M.I.T.S. Gwalior

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

CERTIFICATE

This is certified that **Yash Jain** (0901cs191140) has submitted the project report titled **NFT Market Place** under the mentorship of **Dr. Anjula Mehto** in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering from Madhav Institute of Technology and Science, Gwalior.

[illegible]

Dr. Anjula Mehto
Faculty Mentor
Assistant Professor
Computer Science and Engineering

[illegible] 09/05/20[illegible]

Dr. Manish Dixit
Professor and Head,
Computer Science and Engineering

Dr. Manish Dixit
Professor & HOD
Department of CSE
M.I.T.S. Gwalior

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Dr. Anjula Mehto, Assistant Professor,** Computer Science and Engineering

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

[illegible]

Snehil Banawal
0901CS191121
3rd Year
Computer Science and Engineering

[illegible]

Yash Jain
0901CS191140
3rd year
Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR
(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Department of Computer Science and Engineering, for allowing** me to explore this project. I humbly thank **Dr. Manish Dixit**, Professor and Head, Department of Computer Science and Engineering, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors. I am grateful to the guidance of **Dr. Anjula Mehto, Assistant Professor,** Computer Science and Engineering for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

[illegible]

Snehil Banawal
0901CS191121
3rd Year
Computer Science and Engineering

[illegible]

Yash Jain
0901CS191140
3rd year
Computer Science and Engineering

ABSTRACT

In 2017 the world witnessed the birth of Crypto Kitties. For the first time, the world experienced a decentralized application built on blockchains but targeted towards a mainstream audience.

While Crypto Kitties felt like a toy to many, it represented a dramatic shift in how we interact with items in the digital world. While previous digital items lived on company servers, blockchain-native items lived on shared, public blockchains owned by no single party. They could be viewed anywhere, exchanged openly, and truly owned in a way that was never before possible in the digital world.

NFT Marketplace is a decentralized marketplace for NFTs of different artworks. It uses technologies such as HTML, CSS and JavaScript for Front End, Python and Django Framework for the back end server along with MySQL database to serve the images. The transactions and the actual tokens are generated on the Solidity Smart Contract that makes use of the ERC721 token standard and works in tandem with the web application by the use of Web3.

While making this project, we're excited about a brand new type of digital good called a non-fungible token, or NFT. NFTs have exciting new properties: they're unique, provably scarce, tradeable, and usable across multiple applications. Just like physical goods, you can do whatever you want with them! You could throw them in the trash, gift them to a friend across the world, or go sell them on an open marketplace. But unlike physical goods, they're armed with all the programmability of digital goods.

A core part of our vision is that open protocols like Ethereum and interoperable standards like ERC-721 will enable vibrant new economies. We're building tools that allow consumers to trade their items freely, creators to launch new digital works, and developers to build rich, integrated marketplaces for their digital items.

सार:

2017 में दुनिया ने क्रिप्टोकरंसीज का जन्म देखा। पहली बार, दुनिया ने ब्लॉकचेन पर निर्मित एक विकेंद्रीकृत अनुप्रयोग का अनुभव किया, लेकिन मुख्यधारा के दर्शकों के लिए लक्षित किया।

जबकि क्रिप्टोकरंसी कई लोगों के लिए एक खिलौने की तरह महसूस करती है, यह एक नाटकीय बदलाव का प्रतिनिधित्व करती है कि हम डिजिटल दुनिया में वस्तुओं के साथ कैसे बातचीत करते हैं। जबकि पिछले डिजिटल आइटम कंपनी के सर्वर पर रहते थे, ब्लॉकचेन-देशी आइटम साझा किए गए, सार्वजनिक ब्लॉकचेन पर किसी एक पार्टी के स्वामित्व में नहीं थे। उन्हें कहीं भी देखा जा सकता है, खुले तौर पर आदान-प्रदान किया जा सकता है, और वास्तव में इस तरह से स्वामित्व किया जा सकता है जो डिजिटल दुनिया में पहले कभी संभव नहीं था।

एनएफटी मार्केटप्लेस विभिन्न कलाकृतियों के एनएफटी के लिए एक विकेन्द्रीकृत बाजार है। यह छवियों की सेवा के लिए माई एसक्यूएल डेटाबेस के साथ बैक एंड सर्वर के लिए फ्रंट एंड, पायथन और जैंगो फ्रेमवर्क के लिए एचटीएमएल, सीएसएस और जावास्क्रिप्ट जैसी तकनीकों का उपयोग करता है। लेन-देन और वास्तविक टोकन सॉलिडिटी स्मार्ट कॉन्ट्रैक्ट पर उत्पन्न होते हैं जो ईआरसी721 टोकन मानक का उपयोग करता है और वेब 3 के उपयोग से वेब एप्लिकेशन के साथ मिलकर काम करता है।

इस प्रोजेक्ट को बनाते समय, हम एक बिलकुल नए प्रकार के डिजिटल गुड के बारे में उत्साहित हैं जिसे अपूरणीय टोकन या एनएफटी कहा जाता है। एनएफटी में रोमांचक नए गुण हैं: वे अद्वितीय, सिद्ध रूप से दुर्लभ, व्यापार योग्य और कई अनुप्रयोगों में प्रयोग करने योग्य हैं। भौतिक वस्तुओं की तरह, आप उनके साथ जो चाहें कर सकते हैं! आप उन्हें कूड़ेदान में फेंक सकते हैं, उन्हें दुनिया भर के किसी मित्र को उपहार में दे सकते हैं, या उन्हें किसी खुले बाज़ार में बेच सकते हैं। लेकिन भौतिक वस्तुओं के विपरीत, वे डिजिटल सामान की सभी प्रोग्राम योग्यता से लैस हैं।

हमारी दृष्टि का एक मुख्य हिस्सा यह है कि एथेरियम जैसे खुले प्रोटोकॉल और ईआरसी -721 जैसे इंटरऑपरेबल मानक जीवंत नई अर्थव्यवस्थाओं को सक्षम करेंगे। हम ऐसे टूल बना रहे हैं जो उपभोक्ताओं को अपने आइटम का स्वतंत्र रूप से व्यापार करने की अनुमति देते हैं, निर्माता नए डिजिटल कार्यों को लॉन्च करने के लिए, और डेवलपर्स को अपने डिजिटल आइटम के लिए समृद्ध, एकीकृत मार्केटप्लेस बनाने की अनुमति देते हैं।

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1: Introduction

1.1 Project Overview

NFT Marketplace is a decentralized marketplace for NFTs of different artworks. It uses technologies such as HTML, CSS and JavaScript for Front End, Python and Django Framework for the back end server along with MySQL database to serve the images. The transactions and the actual tokens are generated on the Solidity Smart Contract that makes use of the ERC721 token standard and works in tandem with the web application by the use of Web3.

While making this project, we're excited about a brand new type of digital good called a non-fungible token, or NFT. NFTs have exciting new properties: they're unique, provably scarce, tradeable, and usable across multiple applications. Just like physical goods, you can do whatever you want with them! You could throw them in the trash, gift them to a friend across the world, or go sell them on an open marketplace. But unlike physical goods, they're armed with all the programmability of digital goods.

1.2 Usage

NFT marketplaces are gaining traction in industries such as gaming, art, social networking, and music, nearly capturing every market that deals with digital assets. With the influx of Metaverse projects, NFT marketplaces have grown into prominence and relevance even further.

With time, the features of NFT marketplaces are upgrading, and limitations like lack of NFT interoperability are getting addressed. Cross-chain bridging, niche-specific NFTs, NFT swapping, and compatibility with multiple metaverse projects are some of the advanced features of contemporary NFT marketplaces.

One important factor to consider before developing your NFT Marketplace project is the niche. However, above and beyond niche, there are numerous layers to consider like the blockchain protocol, the NFT storge, architecture designing, and others. Here, this insight covers all about NFT marketplace development.

1.3 Objective and Scope

Blockchain technology and NFTs can offer the perfect opportunity for artists and content creators to obtain financial remuneration for their works. Now, artists don't have to depend anymore on auction houses or galleries for selling their artwork. On the contrary, an artist could just sell their work to a buyer in the form of NFT. This also helps the artists in getting a better share of the profits.

Interestingly, NFTs also involve the scope for royalties that entitle the original creator to a certain percentage of subsequent sales of the artwork. Many people interested in finding out the top NFT marketplace would become eager to find out how they can start an NFT collection. Interested buyers might be looking for ways to buy NFTs. So, let us take a look into the things that go into making NFTs available for selling and buying

1. You need a digital wallet for storing NFTs and cryptocurrency to pay for transactions on your selected blockchain platform.

2. Purchase cryptocurrency, possibly Ether, or the currency supported on your selected NFT provider.

3. Users could move cryptocurrency from exchanges to their preferred wallets.

1.4 Project Features

Following are some of the features that are added to the NFT marketplace:

- **Storefront**
  NFT marketplace should have a storefront that offers users all the information required for an item: bids, owners, preview or price history.

- **Searching for items**
  An NFT marketplace platform should support management to allow users to search collectibles. Use a search bar on the site and add categories.

- **Create listings**
  A user should be able to create and submit collectibles. Using this feature, a user should upload files and fill in the token information such as name, tags, description.

- **Buy and Sell**
  The NFT marketplace platform should have a feature that allows users to buy and sell for NFTs listed on the platform. The selling feature makes the nft available for sale.

- **Wallet**
  The NFT Marketplace Platform should have a wallet that allows users to store, send and receive non-fungible tokens. The easiest way to integrate this feature is to provide users with a connected wallet that they already use. For example, you can integrate the most popular wallets like Metamask, Formatic or MyEtherWallet.

1.5 Feasibility

1.5.1 Operational Feasibility

The operational cost of this project will be lower than even the most common web applications. The primary problem associated with Operational Costs are things like Database storage, Backups, Scaling, Compute Resources and more. Since the web application is not actually handling the transactions and the associated encryption, we do not need to worry about the problems mentioned above.

As far as the operational costs of the Smart contract goes, it will virtually be zero. Once deployed, the contract will take the necessary operation fee or **gas** money from the user itself while handling a transaction. This is due to the server less and decentralized nature of block chain which eliminates all the concerns with scaling and upgrading. The only cost incurred will be deployment of the contract.

1.5.2 Economic Feasibility

The project is aimed at providing the freedom to artists and the competing platforms that do so charge a hefty amount of money for their troubles but since we aim at making this project solely for the betterment of the artists' community, we would not charge service charges associated with the platform. Everything is transparent and so the trust factor and approval rates rise.

The web application, while being versatile and secure would not require a whole lot of compute or data storage since it is only to assist the user in accessing and operating the smart contract.

1.6 System Requirements

- Linux Operating System
- 2GB RAM
- 30GB of Secondary Storage
- Primary and Auxiliary power for zero downtime
- Python 3.8
- Internet Connectivity

Chapter 2: Literature Review

NFT or Non-Fungible token, in general terms, is the ownership token of a digital asset stored in an immutable record on a blockchain.

To break the definition down, a digital asset can be anything from a digital photograph of an old mansion to a rare audio clip of baby whale whisper, the age-old internet meme of Nyan Cat, the screenshot of Jack Dorsey's first-ever tweet, or a random collection of vector art.

Since all these arts could not have any specific value, their price is determined by the demand and supply paradigm. This is why most of the NFTs are auctioned rather than sold at a fixed price. And this is also why they are called to be non-fungible.

So contrary to NFT, a fungible item is one that can be broken down into smaller units and can still retain some of the value. Fungible items, like currency, can also be exchanged for another currency of the same value. However, since NFTs are unique artforms, *unique* being the keyword here, they cannot be exchanged for something of the same value.

Now, moving on to the next part of the definition, **NFTs are ownership tokens**, and precisely so. When someone buys or sells an NFT, they do not give away a hard copy of the art. Rather the seller transfers the ownership of the digital art to the buyer. This means any number of people can still download the art and use it for their respective purposes, but the owner of the art will remain with the buyer until they sell it to someone else.

In this way, NFTs are quite similar to the artwork in a museum. Many people can come and appreciate it, they may even hang a copy of it in their homes, yet the owner of the original artwork remains with the museum.

You may now ask that when a museum has the ownership of certain artwork, they have a certificate as well as the art itself with themselves to prove their ownership; how do NFTs find a workaround that. And this is where the last part of the definition enters the image. All NFT transactions are recorded in a blockchain. Since blockchains are immutable, any particular NFT entity's record will always show its latest owner, with documented records of all the owners before that.

2.1 Why Blockchain?

Here are some oversimplified reasons on why we are even using blockchain instead of traditional ways:

Decentralized. No single entity controls the operation of the contract and thus it is fair in every sense of word.

Can't be hacked. Because to hack it you would need to change the data on all network nodes simultaneously or fool the proof-of-concept check for them. Practically impossible.

Anonymity and open for all. There are addresses, not DOBs and Aadhaar numbers.

2.2 Why Ethereum?

Simply put we chose Ethereum Network because of the extensive support for Smart Contracts.

Solidity allows different types of use cases for Ethereum. It enables people to use tokens and non-fungible tokens on Ethereum.

Decentralized Autonomous Organizations (DAOs) is a new type of online organizational structure. We use Solidity to write DAOs.

You can look at its extensive usage in Kickstarter projects and see why it is desirable.

Chapter 3: Preliminary design

3.1 Software Development Life Cycle Model

3.1.1 Rapid Application Development

Reason: since the software size was not much large and there was a time-bound and the project was made in modules therefore in this project, I used Rapid Application Development. A software project can be implemented using this model if the project can be broken down into small modules wherein each module can be assigned independently to separate teams. These modules can finally be combined to form the final product.

3.2 ER Diagram

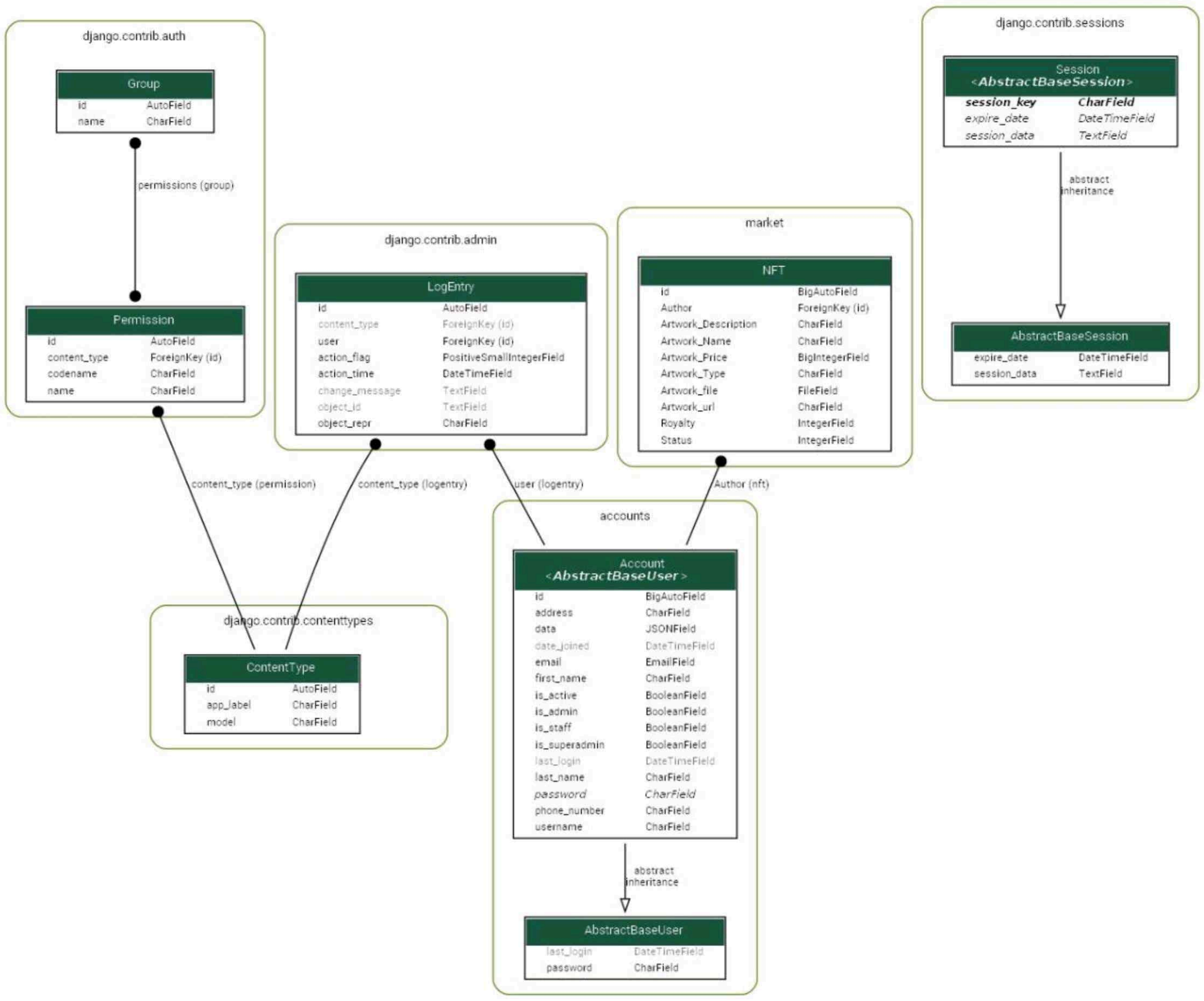


Fig. 3.2.1 ER Diagram

3.3 Tools & Technologies

3.3.1 Python

Python is a high-level, interpreted, interactive, and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently whereas other languages use punctuation, and it has fewer syntactical constructions than other languages.

3.3.2 Visual Studio Code

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

3.3.3 Remix IDE

Remix IDE allows developing, deploying and administering smart contracts for Ethereum like blockchains. It can also be used as a learning platform.

3.3.4 Solidity

Solidity is a high-level object-oriented language for creating smart contracts. Smart contracts are programmes that control how accounts behave in the Ethereum state. Solidity is a curly-bracket programming language aimed towards the Ethereum Virtual Machine (EVM). C++, Python, and JavaScript all have an impact. Solidity is statically typed and, among other things, enables inheritance, libraries, and sophisticated user-defined types.

3.3.5 Ethereum

Ethereum is a platform for digital money, international payments, and applications. The community has created a thriving digital economy, as well as innovative new ways for creators to make money online. It's accessible to anybody, wherever in the globe — all you need is an internet connection.

3.3.6 Smart Contract

Smart contracts are essentially programmes that run when certain criteria are satisfied and are recorded on a blockchain. They're usually used to automate the execution of an agreement so that all parties may be confident of the conclusion right away, without the need for any intermediaries or time waste. They can also automate a workflow, starting the following step when certain circumstances are satisfied.

3.3.6 Front End

We have used Html, CSS & JavaScript to develop our NFT Market Place website, to integrate it with blockchain we have use Web3 JS.

3.3.7 Back End

Solidity Smart Contract on Ethereum Network, Python and Django Framework in harmony and MySQL Database to hold metadata

3.3.8 Libraries Used

3.3.7.1 Pillow

The Python Imaging Library improves your Python interpreter's image processing capabilities. This library supports a wide range of file formats, has a fast internal representation, and can handle pretty complex picture processing. The image library's core is built to provide quick access to data saved in a few fundamental pixel formats. It should serve as a robust basis for any image processing application.

3.3.7.2 Open Zeppelin

OpenZeppelin Contracts reduces risk by utilising battle-tested smart contract libraries for Ethereum and other blockchains. It comprises the most widely used ERC standard implementations.

3.3.7.3 Django Extensions

Django Extensions is a set of Django Framework extensions. Management commands, extra database fields, admin extensions, and many more features are available.

3.3.7.4 Asgiref

ASGI is a standard for Python asynchronous web apps and servers to communicate with each other, and positioned as an asynchronous successor to WSGI

3.3.7.5 PyYAML

YAML is a data serialisation format for humans and scripting languages. PyYAML is a Python implementation of a YAML parser and emitter. PyYAML comes with a full YAML 1.1 parser, Unicode support, pickle support, a robust extension API, and logical error messages. PyYAML includes both conventional YAML tags and Python-specific tags for representing any Python object.

3.3.7.6 Py CryptoDome

PyCryptodome is a Python module that contains low-level cryptographic primitives. Python 2.7, Python 3.5 and later, and PyPy are all supported.

Chapter 4: Final Design and Analysis

4.1 Final URL Mappings and Design

4.1.1 Admin

The admin panel of the web application allows you to edit the database and manage user access controls. It is the last line of defence against malfunctions in the web application itself.

4.1.2 Accounts

Accounts section URLs allow the user to review their profile, edit it, change passwords, check their NFTs and much more behind the scenes functionality that make sure that the web application is secure and accessed through proper channels.

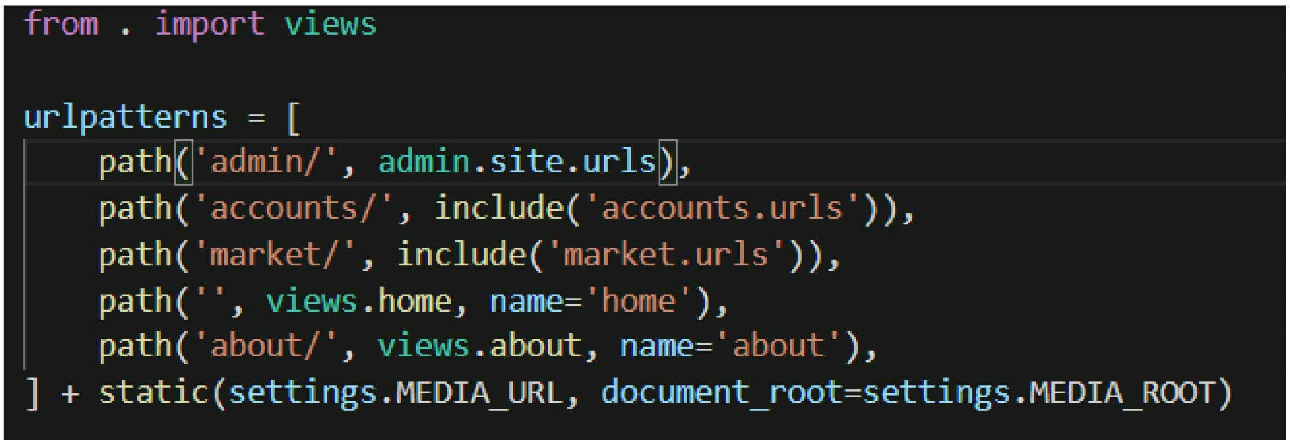
4.1.3 Market

This app handles the primary objective of assisting the user to find NFTs, buy, sell and resell them and maintains the additional metadata of the Smart Contract as well.

4.1.4 Home and About

About is a static page describing the details of the developers.

Home dynamically gathers the NFTs and showcases them to the user as well as a pretty page to attract people.

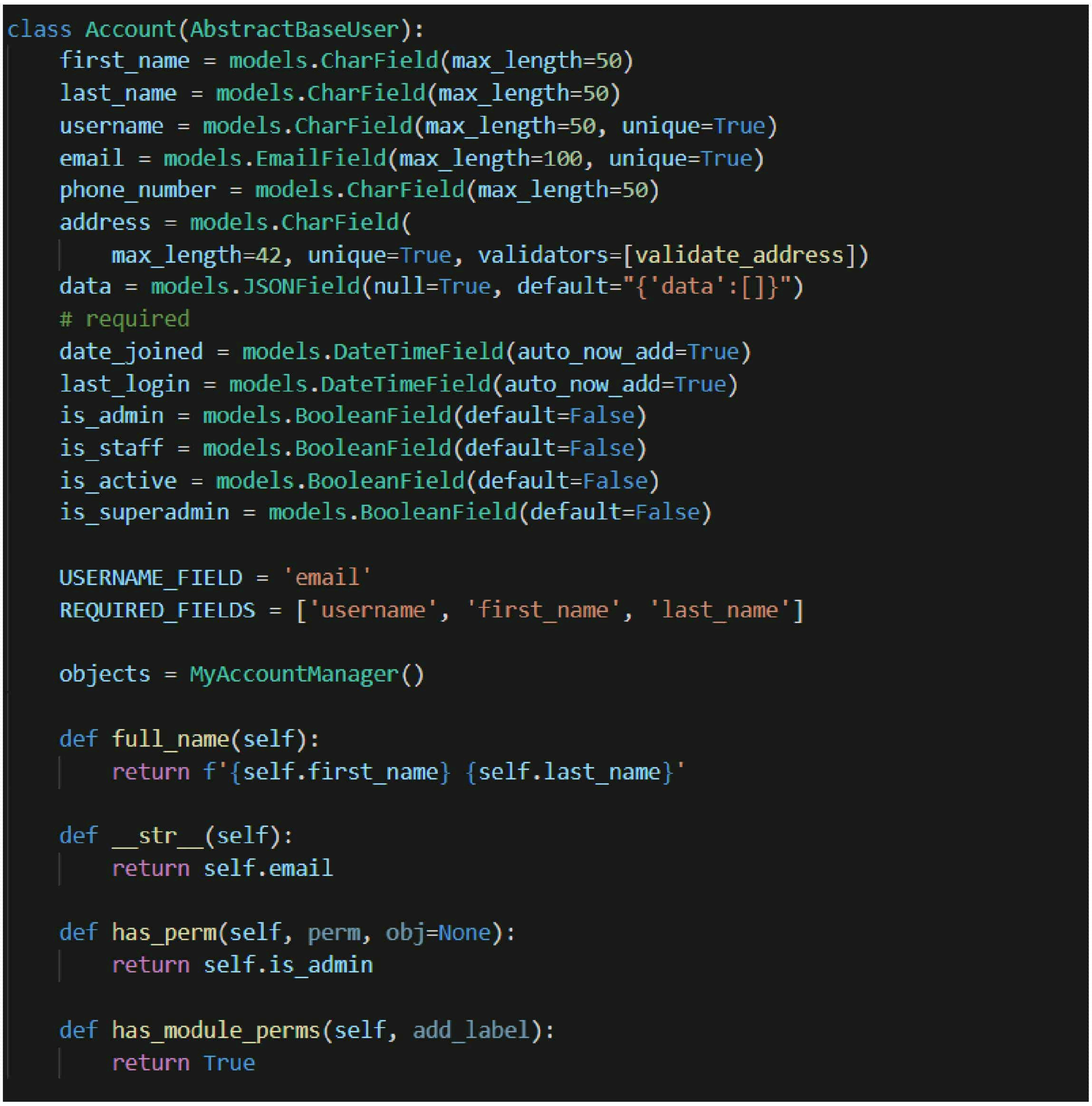


```
from . import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('accounts/', include('accounts.urls')),
    path('market/', include('market.urls')),
    path('', views.home, name='home'),
    path('about/', views.about, name='about'),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

4.2 Database Schema Design

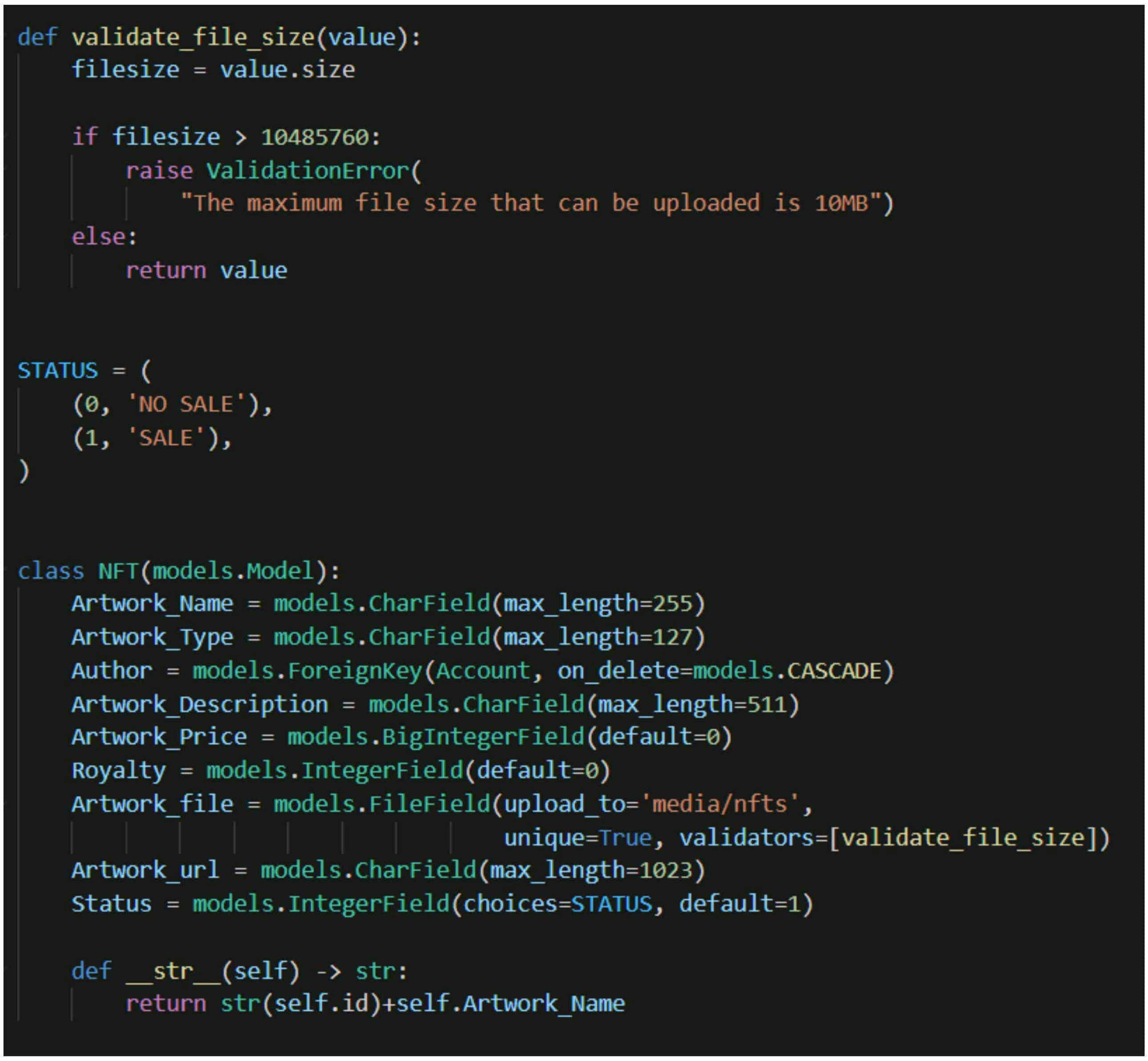
4.2.1 Accounts

The Schema definition of this app is centered around making the app as simple, uncoupled and helpful to the end user as possible.


```python
class Account(AbstractBaseUser):
    first_name = models.CharField(max_length=50)
    last_name = models.CharField(max_length=50)
    username = models.CharField(max_length=50, unique=True)
    email = models.EmailField(max_length=100, unique=True)
    phone_number = models.CharField(max_length=50)
    address = models.CharField(
        max_length=42, unique=True, validators=[validate_address])
    data = models.JSONField(null=True, default="{'data':[]}")
    # required
    date_joined = models.DateTimeField(auto_now_add=True)
    last_login = models.DateTimeField(auto_now_add=True)
    is_admin = models.BooleanField(default=False)
    is_staff = models.BooleanField(default=False)
    is_active = models.BooleanField(default=False)
    is_superadmin = models.BooleanField(default=False)

    USERNAME_FIELD = 'email'
    REQUIRED_FIELDS = ['username', 'first_name', 'last_name']

    objects = MyAccountManager()

    def full_name(self):
        return f'{self.first_name} {self.last_name}'

    def __str__(self):
        return self.email

    def has_perm(self, perm, obj=None):
        return self.is_admin

    def has_module_perms(self, add_label):
        return True
```

4.2.2 Market

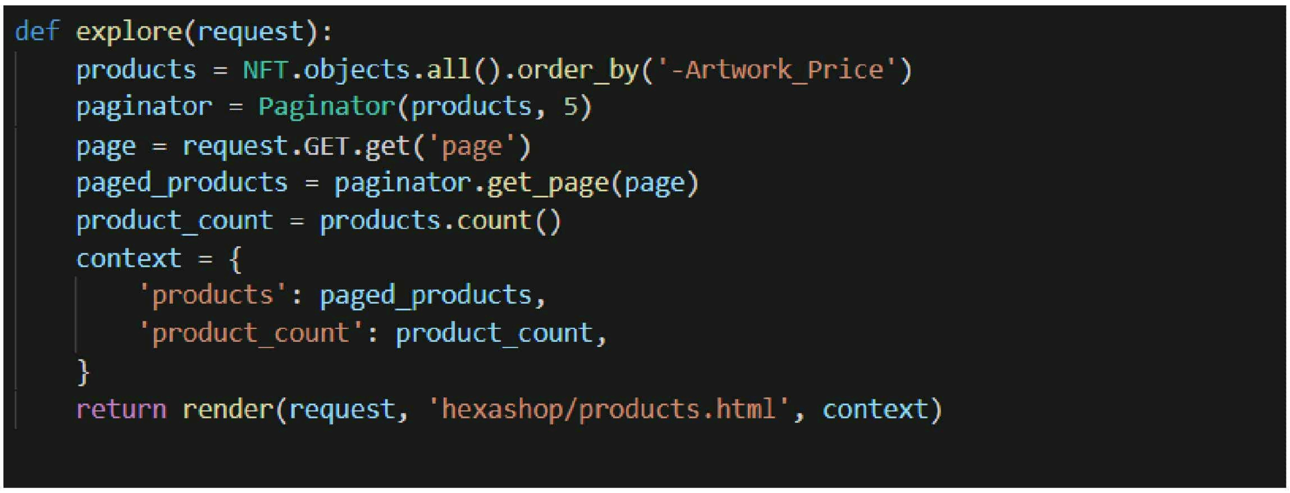
The schema of this app is to mimic the actual contract and be updated by the data there.



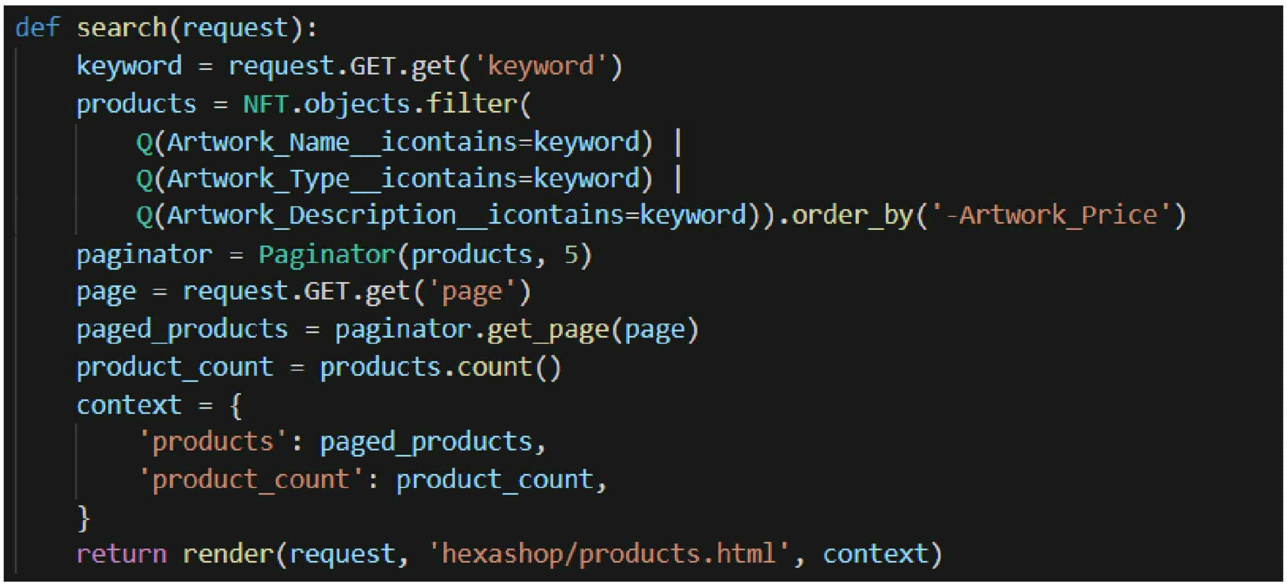
```python
def validate_file_size(value):
    filesize = value.size

    if filesize > 10485760:
        raise ValidationError(
            "The maximum file size that can be uploaded is 10MB")
    else:
        return value


STATUS = (
    (0, 'NO SALE'),
    (1, 'SALE'),
)


class NFT(models.Model):
    Artwork_Name = models.CharField(max_length=255)
    Artwork_Type = models.CharField(max_length=127)
    Author = models.ForeignKey(Account, on_delete=models.CASCADE)
    Artwork_Description = models.CharField(max_length=511)
    Artwork_Price = models.BigIntegerField(default=0)
    Royalty = models.IntegerField(default=0)
    Artwork_file = models.FileField(upload_to='media/nfts',
                                    unique=True, validators=[validate_file_size])
    Artwork_url = models.CharField(max_length=1023)
    Status = models.IntegerField(choices=STATUS, default=1)

    def __str__(self) -> str:
        return str(self.id)+self.Artwork_Name
```

4.3 Searching and Serving NFTs

4.3.1 Search Functions



```python
def explore(request):
    products = NFT.objects.all().order_by('-Artwork_Price')
    paginator = Paginator(products, 5)
    page = request.GET.get('page')
    paged_products = paginator.get_page(page)
    product_count = products.count()
    context = {
        'products': paged_products,
        'product_count': product_count,
    }
    return render(request, 'hexashop/products.html', context)
```



```python
def search(request):
    keyword = request.GET.get('keyword')
    products = NFT.objects.filter(
        Q(Artwork_Name__icontains=keyword) |
        Q(Artwork_Type__icontains=keyword) |
        Q(Artwork_Description__icontains=keyword)).order_by('-Artwork_Price')
    paginator = Paginator(products, 5)
    page = request.GET.get('page')
    paged_products = paginator.get_page(page)
    product_count = products.count()
    context = {
        'products': paged_products,
        'product_count': product_count,
    }
    return render(request, 'hexashop/products.html', context)
```

The Search function makes use of advanced queries and checks the three parameters Artwork_Name, Type and Description for the searched keyword to find the results and then add pagination in a way that only 5 results are there in a single page. We use GET method to send the page number and keep pagination intact.

4.3.2 Minting NFT (Two Stage)

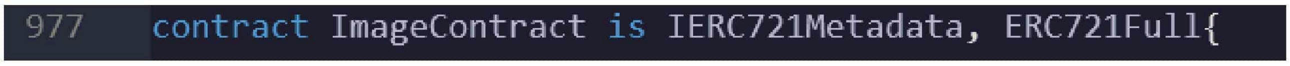
The minting portion takes place in two parts, first the NFT and all the associated data is uploaded to the web application where all the data sent is also assessed by the validators and then rendered for confirmation on the next page.

On the next page the data is validated and will be primed to be sent in the contract as a transaction in the future. This way we take care of all the background and the signed transaction gets sent for execution where sooner or later someone would mine it and add it into the block.

4.4 Smart Contract Design and Creation

4.4.1 Openzepplin

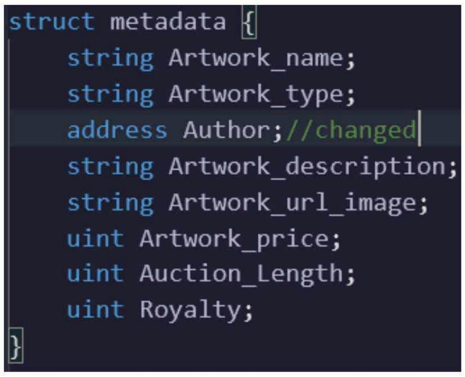
As described above, we intended to use ERC721 standard for our NFTs. We did so by implementing the complete and full IERC721Metadata and ERC721Full Smart Contracts provided by Openzepplin which is the industry standard for the same.



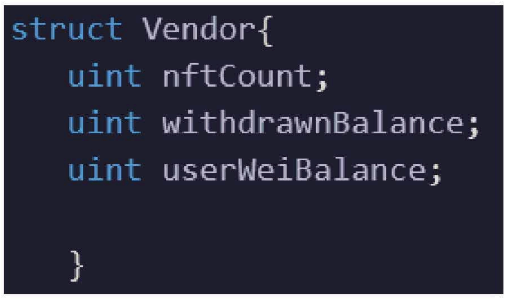
```solidity
contract ImageContract is IERC721Metadata, ERC721Full{
```

4.4.2 Structs Used

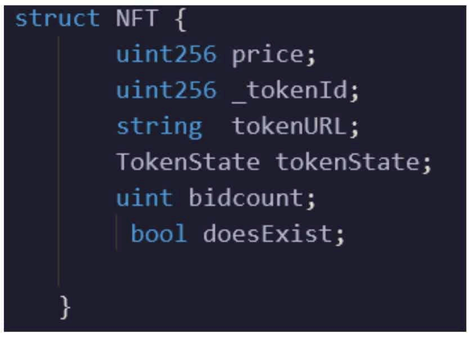
All the relavant Structs used are shown below



```
struct metadata {
    string Artwork_name;
    string Artwork_type;
    address Author;//changed
    string Artwork_description;
    string Artwork_url_image;
    uint Artwork_price;
    uint Auction_Length;
    uint Royalty;
}
```



```
struct Vendor{
    uint nftCount;
    uint withdrawnBalance;
    uint userWeiBalance;

    }
```



```
struct NFT {
        uint256 price;
        uint256 _tokenId;
        string  tokenURL;
        TokenState tokenState;
        uint bidcount;
         bool doesExist;

    }
```

These Structs are pivotal to the functionality of the Smart Contract as will be clear in the next topic.

4.4.3 Mappings

All the relevant Mappings used are shown below

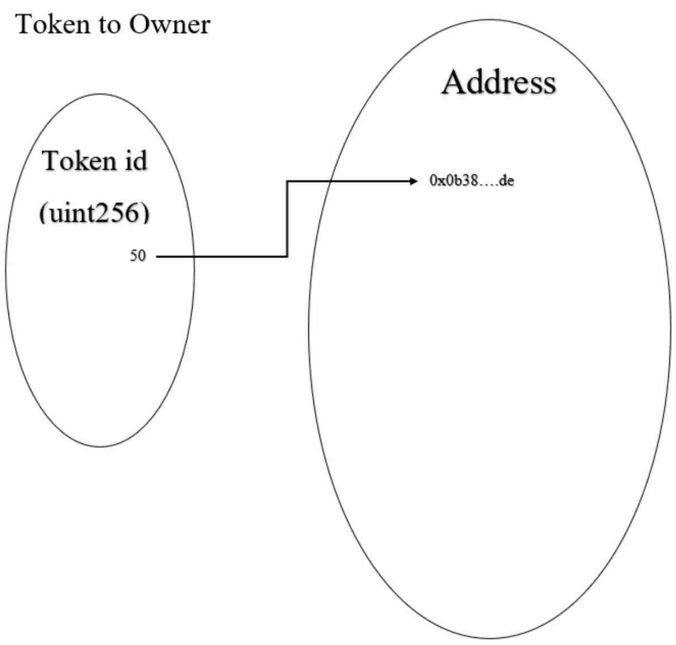


Fig 4.4.3.1 Token to Owner Mapping

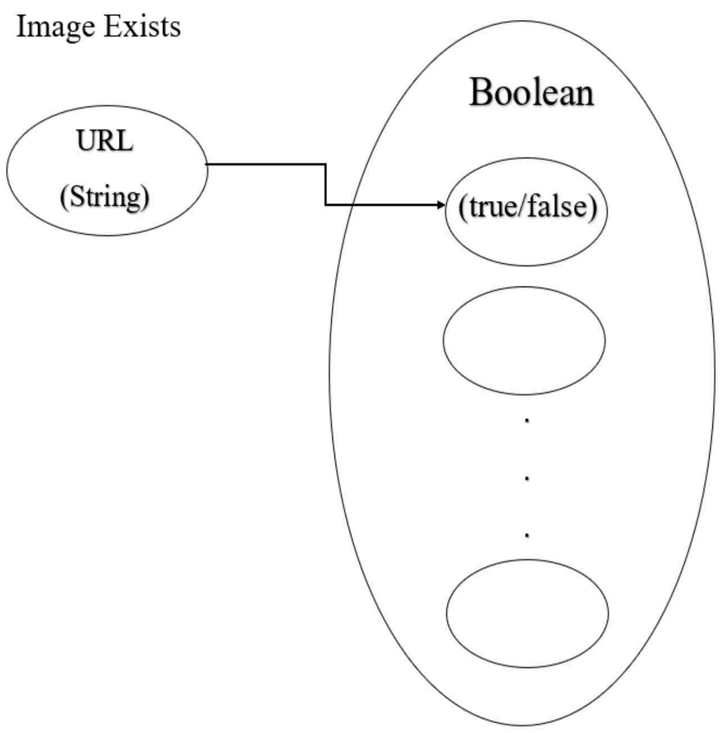


Fig. 4.4.3.2 Image Exists Mapping

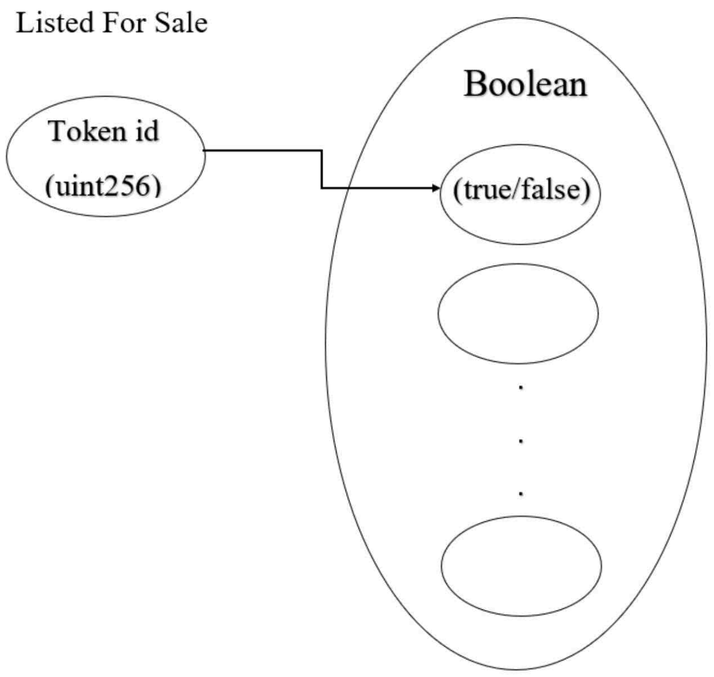


Fig. 4.4.3.3 NFT Listed for sale mapping

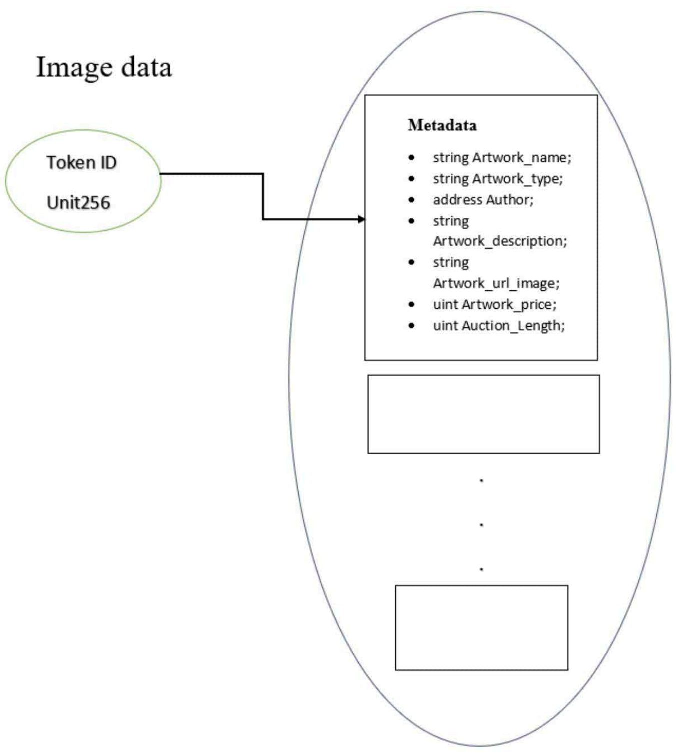


Fig. 4.4.3.4 Image data

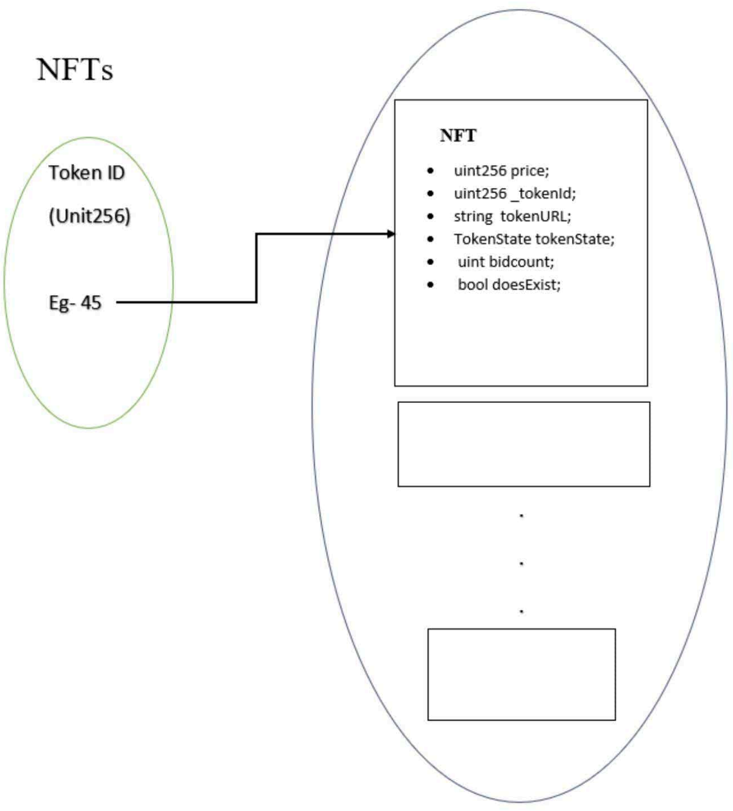
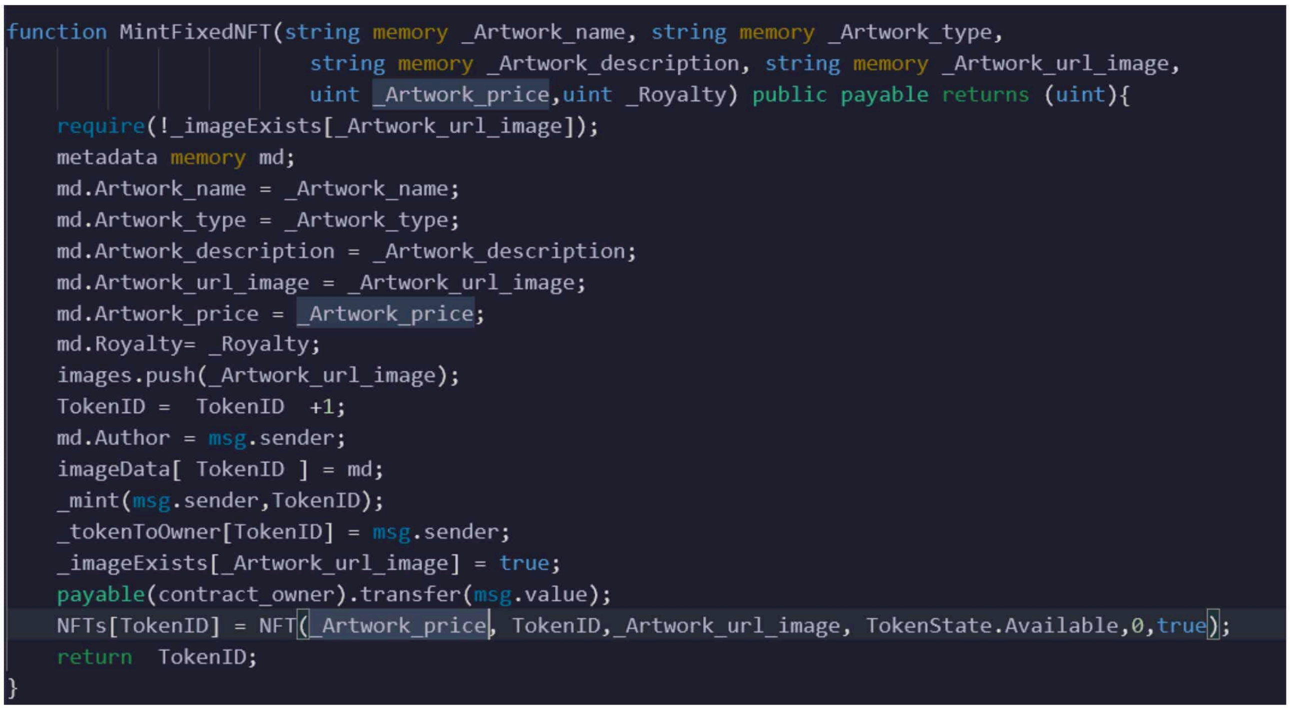


Fig. 4.4.3.5 NFTs Metadata

4.4.4 Functionality

4.4.4.1 Mint Function

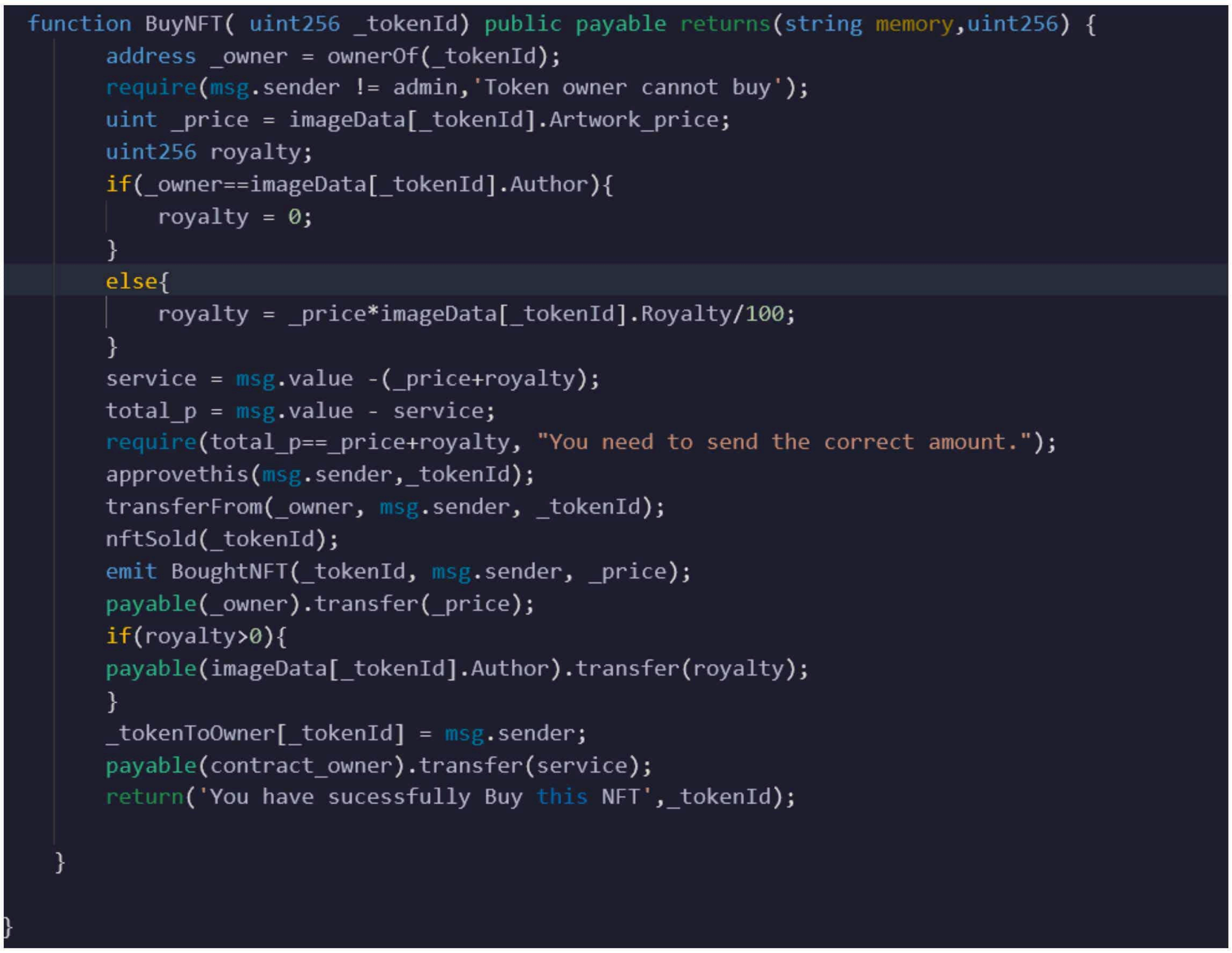
Mint function initializes the metadata, updates the global token id counter by one and invokes the minting functions to generate the NFT of the data sent in the function parameter.



```
function MintFixedNFT(string memory _Artwork_name, string memory _Artwork_type,
                        string memory _Artwork_description, string memory _Artwork_url_image,
                        uint _Artwork_price,uint _Royalty) public payable returns (uint){
    require(!_imageExists[_Artwork_url_image]);
    metadata memory md;
    md.Artwork_name = _Artwork_name;
    md.Artwork_type = _Artwork_type;
    md.Artwork_description = _Artwork_description;
    md.Artwork_url_image = _Artwork_url_image;
    md.Artwork_price = _Artwork_price;
    md.Royalty= _Royalty;
    images.push(_Artwork_url_image);
    TokenID =  TokenID  +1;
    md.Author = msg.sender;
    imageData[ TokenID ] = md;
    _mint(msg.sender,TokenID);
    _tokenToOwner[TokenID] = msg.sender;
    _imageExists[_Artwork_url_image] = true;
    payable(contract_owner).transfer(msg.value);
    NFTs[TokenID] = NFT(_Artwork_price, TokenID,_Artwork_url_image, TokenState.Available,0,true);
    return  TokenID;
}
```

4.4.4.2 Buy Function

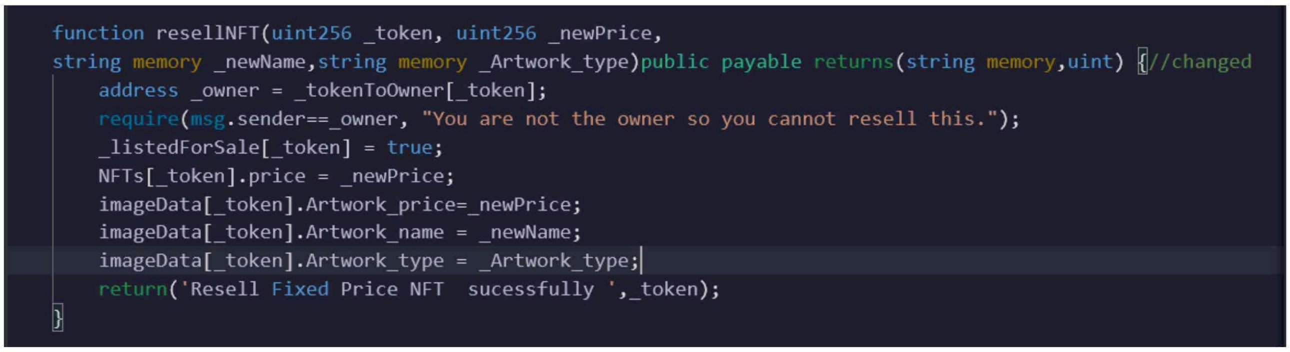
The buy function calculates the necessary values such as Royalty and validates the amount sent by the buyer is correct or not for the invoked token number. Then it proceeds to pay the Author the proper royalties, the previous owner the listed price and transfers the ownership to the new owner.



```
  function BuyNFT( uint256 _tokenId) public payable returns(string memory,uint256) {
        address _owner = ownerOf(_tokenId);
        require(msg.sender != admin,'Token owner cannot buy');
        uint _price = imageData[_tokenId].Artwork_price;
        uint256 royalty;
        if(_owner==imageData[_tokenId].Author){
            royalty = 0;
        }
        else{
            royalty = _price*imageData[_tokenId].Royalty/100;
        }
        service = msg.value -(_price+royalty);
        total_p = msg.value - service;
        require(total_p==_price+royalty, "You need to send the correct amount.");
        approvethis(msg.sender,_tokenId);
        transferFrom(_owner, msg.sender, _tokenId);
        nftSold(_tokenId);
        emit BoughtNFT(_tokenId, msg.sender, _price);
        payable(_owner).transfer(_price);
        if(royalty>0){
        payable(imageData[_tokenId].Author).transfer(royalty);
        }
        _tokenToOwner[_tokenId] = msg.sender;
        payable(contract_owner).transfer(service);
        return('You have sucessfully Buy this NFT',_tokenId);

    }

}
```

4.4.4.3 Resell NFT

This function allows the user to resell their NFT and even give it a new name. Comparatively it is simpler because it just has to essentially update a ledger saying that this NFT is available for Sale.



```
    function resellNFT(uint256 _token, uint256 _newPrice,
    string memory _newName,string memory _Artwork_type)public payable returns(string memory,uint) {//changed
        address _owner = _tokenToOwner[_token];
        require(msg.sender==_owner, "You are not the owner so you cannot resell this.");
        _listedForSale[_token] = true;
        NFTs[_token].price = _newPrice;
        imageData[_token].Artwork_price=_newPrice;
        imageData[_token].Artwork_name = _newName;
        imageData[_token].Artwork_type = _Artwork_type;
        return('Resell Fixed Price NFT  sucessfully ',_token);
    }
```

4.5 Limitations

- The Smart Contract cannot yet handle Auction functionality
- The Smart Contract and the Web Application need further development to interface with each other more seamlessly.

- Optimization and simplification of the interface and the underlying contract can be done upon deployment and public beta testing of the platform which can improve the user experience and ease.

Chapter 5 Conclusion & Future Scope

The purpose of this application was to find a way to make NFTs more accessible and easier to get into for an average Joe and we have achieved that with a user-friendly interface and a lot of abstraction of the underlying tech while still staying true to our goal of staying transparent and open to those who want to investigate deeper. Since this is a new field and uncharted territory in a manner of speaking, there is a lot of trust deficit in the field leading to a lot of people missing out on the opportunities.

As more and more people get into this, we can tie these NFTs to more than just artwork but rather to other real-life assets in a way that tampering or manipulating the record becomes impossible and we achieve true meaning and honesty of contracts. For example, if government institutions developed a way to maintain the exact records of assets of each and every individual on a decentralized system spanning across millions of devices then the ambiguous nature and corruption opportunities that arise because of a centralized control and vulnerability to manipulation will be essentially eliminated and will thus lead to a better and brighter nation in general. The current application may just be a proof of concept or a valuable but not the only place someone can benefit from the existence of this technology- maybe like how Botox was given only to people who underwent eye surgery before people started using it to avoid wrinkles.

References

- Hands-On Smart Contract Development with Solidity and Ethereum: From Fundamentals to Deployment
  Book by David H. Hoover, Kevin Solorio, and Randall Kanna
- Web Development with Django: Learn to Build Modern Web Applications with a Python-based Framework
  Book by Andrew Bird, Ben Shaw, and Saurabh Badhwar
- Token Economy: How the Web3 Reinvents the Internet
  Book by Shermin Voshmgir
- L Ante
  "Smart Contracts on the Blockchain -A Bibliometric Analysis and Review", Telemat. Informatics, volume 57
  Posted: 2021
- Solidity — Solidity 0.8.13 documentation
- Ethereum development documentation