NEXUS

Submitted in partial fulfilment of the requirements for the award of degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

Submitted By:

KSHITIJ YADAV - 18021011451
NAVNEET KUMAR -18021011705

SCHOOL OF COMPUTING SCIENCE & ENGINEERING
Galgotias University, Greater Noida

DECLARATION

We hereby declare that the project entitled - “NEXUS”, which is being submitted as a minor project of 7h semester in Computer Science & Engineering to Galgotias University, Greater Noida (U.P) is an authentic record of our genuine work done under the guidance of Dr A Daniel **,** Dept. Computer Science & Engineering, Galgotias University**.**

NAVNEET KUMAR (18SCSE1010477)
KSHITIJ YADAV (18SCSE1010219)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Mr. Gokul Ranjan, Assistant Professor of the department of Information Technology, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he took in advising us, for the books and reference materials provided for the moral support extended to us.
Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staff for the gracious hospitality they offered us.

NAVNEET KUMAR
KSHITIJ YADAV

ABSTRACT

A blockchain is essentially a distributed database of records, or public ledger of all transactions or digital events that have been executed and shared among participating parties. Each transaction in the public ledger is verified by consensus of a majority of the participants in the system . Once entered, information can never be erased. The blockchain contains a certain and verifiable record of every single transaction ever made. Bitcoin, the decentralized peer-to-peer digital currency, is the most popular example that uses blockchain technology. The digital currency bitcoin itself is highly controversial but the underlying blockchain technology has worked flawlessly and found wide range of applications in both financial and non-financial world.

The main hypothesis is that the blockchain establishes a system of creating a distributed consensus in the digital online world. This allows participating entities to know for certain that a digital event happened by creating an irrefutable record in a public ledger. It opens the door for developing a democratic open and scalable digital economy from a centralized one. There are tremendous opportunities in this disruptive technology, and the revolution in this space has just begun.

Table of Contents

1. What is Nexus?

Nexus Crypto (NCR) is an established group of highly motivated people focused on bringing positive change to the crypto sphere through the continuous development of innovative tools.

NCR is creating the infrastructure to enable operators in the businesses to integrate crypto into their business operations where applicable & to be the ultimate safe zone for new & advanced enthusiasts.

NCR was founded by KSHITIJ YADAV that after having surpassed the high entry barrier and faced a steep learning curve lost a considerable amount of money falling for scams, rug pulls, and all sorts of mischiefs created by ill intentional people.

For that reason, we believe that every crypto enthusiast (noob or experienced) should be equipped with cutting-edge tools and highly curated information to make the best decision while investing in tokens, onboarding on projects or trading, and creating NFTs. The Nexus Ecosystem is divided into 3 sectors:

Nexus Folio: A set of tools designed to improve the trend experience in a decentralized market

- **Kre8:** Our platform that empowers artists and creators to mint their own NFT and provides collectors with a safe means to backup and secures their possessions

EDU: A learning center to lower the entry barriers and reduce the effort needed to start on crypto.

2. The Problems

Problem #1 - Asset Management in the DeFi Market

Crypto markets are decentralized and never stop. With multiple swaps and liquidity providers spread all over the blockchain by design, someone who is trying to make the best with crypto can pose a hard problem, and more often than not hard questions emerge

How much is your portfolio worth?

What happens when a new and very promising project is launching a new token?

What if a good buy/selling point appears and you are not near the closest computer/cell phone?

How about when that whale starts to dump signaling a crash?

How has it been performing and what is your best/worst asset?

All those questions crypto traders and enthusiasts try to answer one way or the other to make the best decisions. There are multiple tools in the market that can help derive the information needed but there is no tool prepared to give a comprehensive view of portfolio performance, with advanced trading tools in a truly multi-chain manner.

Problem #2 - NFT governance

There is nothing more popular in the market than NFT right now. Being a new pattern crafted not long ago the NFTs face a serious governance problem.

First, the artists need to rely on limited standard NFT minters or spend many hours learning how to code an NFT contract what caps the Artist's potential. Additionally, these tools are often bound with a single market limiting the NFT art's reach.

Second, there is a lurking problem regarding the continued existence of an NFT item due to the fact that today the all the Art and the blockchain contract are only associated with a weak integer ID. Right now most NFTs are hosted on private or public service and have a contract that points and recognizes the hosted Art. Therefore, the NFT can simply vanish if the service that is hosting it is not working anymore leaving the collector with only a very expensive ID in his wallet.

Problem #3 - The complexity of Crypto

Initiate in crypto is no easy task. Setting up metamask, learning the lingo, understanding core concepts takes a lot of energy and time. On top of that, it is very difficult to spot bad projects and scams. That makes newcomers' life really difficult discouraging them to start crypto or sometimes give up after the first bad experience.

The DeFi concepts can be so hard that experienced crypto investors and traders have a bad time fully understand dipper concepts such as liquidity pools, liquidity providers, market cap, impermanent loss, front-run, and others. Additionally, the constant invention of new terms and technology in the crypto space makes it difficult to keep up.

NEXUS UTILITY TOKEN

3. Nexus Utility Token

The tools developed by the Nexus team have premium and paid features and such transactions will be facilitated through the Nexus Utility Token or $NEXUS. Those who decide to utilize Nexus inside the platform will have different benefits to be specified in a timely manner. $NEXUS can be obtained through a set of liquidity pools in the partner swaps. Every transaction of nexus will be free except the transaction of returning the token to the liquidity pool that will be charged a 5% usage fee. This fee can be later used to generate more liquidity or be redistributed to token holders via staking or to be used in another manner that the community decides.

Tokenomics

Total Supply: **10,000,000**

Tax: 5% on buy/sell

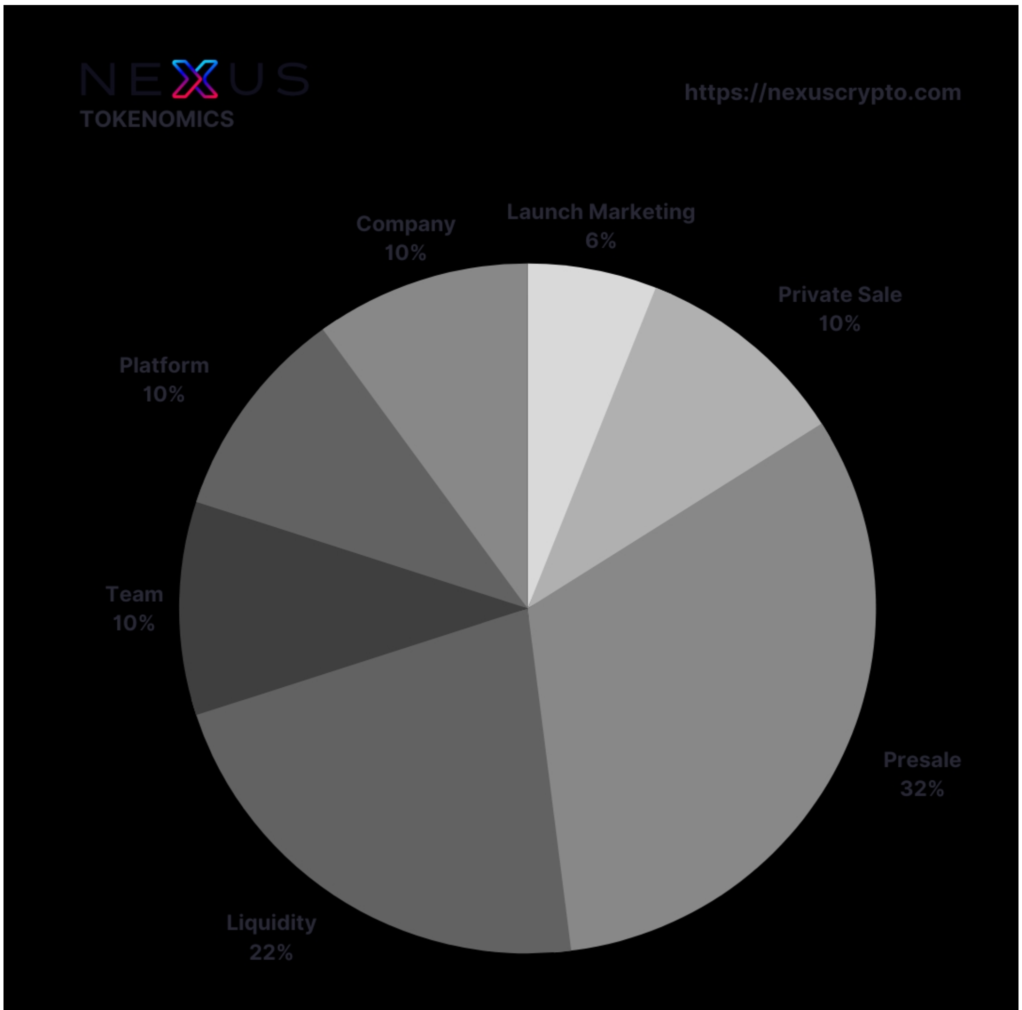
3% to Marketing

2% for Super Nova

Private Sale

We raised an initial seed

All goes straight to marketing (Certik, Influencers)



Formed strategic marketing partnerships

400 BNB Presale

Max 2BNB, min 0.1 BNB

6,750 tokens / BNB

80% of presale goes to LP, the rest for marketing

Marketing

Certik audit

AMA tour

Influencer marketing

Security, Locks & Transparency

We've learned from the past, other projects and have taken an approach that aims to protect ourselves from malicious actors and human error. We are always transparent and do what's best for the community and holders first, weighing in the long-term aspects of the project.

Corporate Multi-Sig

After careful review, we will likely go with Gnosis Safe to handle our proxy contract. If any changes need to be made, 2 out of 3 co-founders need to sign for it.

Team and Marketing Locks

Every core member's wallet will be public, and their allocation is vested over 10 months after a 2-month cliff.

The marketing wallet will start vesting at launch 10% every month.

All company wallets are public, we will have some kind of table on the website or in Telegram and update the white paper with significant updates.

Antibot

The Nexus Anti-bot is designed to prevent bot attacks with an automatic cool-down timer that is activated after each transaction (per wallet). This prohibits front running and from consecutive sales in a short period of time.

LP and Liquidity Management

We have some unique ideas, some borrowed from the space, and some innovations on how to manage liquidity. We want to be able to react to the market when a bear market hits, so we'll re-lock liquidity every month to 3 months based on the level of trust with the community. At launch, we are locking for at least 3 months based on a vote.

Reasons we need to be flexible:

We might want to unwind our pair with BNB and pair with BUSD if we enter a bear market.

We might want to rework our LP ratio if the LP to MCAP ratio is too soft or stiff.

Super Nova

We are working on a contract that is like an evolution of swapAndLiquify from safe moon. When LP is below a certain threshold, profit from the eco-system will be used to add to the LP pool and buy our token in an automated buy-back campaign. If the LP to MCAP ratio then becomes too stiff, this contract would then unwind some of the LP, burn the tokens or reflect them as rewards.

We're still working on some of the details and will ask for community input when the time comes to implement this concept.

Inspiration for this concept comes from Safemoon, Everrise, and Useless Tokens furnace.

Staking and Rewards

Long term, the ecosystem we're building will all rely on the token and have multiple reward layers based on amount of stake, amount of holding, and community engagement (yes, you can earn by being big in the community, not only whales will benefit as we're community first always).

Some details:

Depending on your "level" profit from ads, sniper bot, limit orders, nft minter, education course purchases, premium subscriptions... all ecosystem tools will add to a revenue pool that will be split amongst holders

Staking will give you a multiplier on the revenue rewards and early access to new features, as well as larger discounts

Holders will get discounts and rewards based on % of holdings

We have special things planned for buy backs once we hit revenue goals

All these details will be ironed out and voted on by the community and our advisors.

4. Nexus Ecosystem

We are taking the DAO approach for the development of our ecosystem. In that sense, all the features and tools decisions will be made by the community. We like a multi-pronged approach to product creation. We will vet the features that we think are viable to actually build, and add them to a list the community can vote on.

Our core values are increasing adoption and making the space easier and friendlier to engage with. As a company, our core strategy is to support consumer brands and then resell and repackage the tech in some way that automates and scales. These two guiding principles inform every major decision and keep us aligned and focused. It has to be good and it has to make money.

Nexus Folio

A cross-chain multi-wallet token tracker and discovery portal. We feel that this is our market entry product, but not claim to fame. In order to be "sticky" enough, we need to provide at least a basic trading platform that is forward-thinking.

With that, we aim to be in the "top 5" in this space. We are the "nexus" of your crypto experience, and you will need a solid portfolio tracker with an integrated workflow. We hope that when this product reaches maturity we'll decrease the number of browser tabs you need open to be an effective trader, NFT collector, student, educator, or all of the above.

Features at a glance

Cross-chain multi-wallet capable

Track tokens and NFTs

Easy to use, great design, mobile-friendly

Integration with trading tools

Sniper bot

Stop loss

Limit orders

Social media/news aggregation per token

Market IQ

Smart events (poocoin promoted ad, CMC listing)

Rule-based notification system (email, SMS, push notification)

Price alerts

Whale alerts

KRE8

KRE8 is our suite of NFT tools aimed to empower the artist. With our first product in this vertical being our white label NFT minter, we're enabling a loot box-style minter to be quickly spun up on your own .com with your logo and color scheme. Simply upload all the images that will be used to generate the NFTs and we can assist in auto-generating the images and metadata. We can also take care of all the hosting and infrastructure and just charge a flat monthly fee.

We are in a pilot program now and will have a more polished revenue and business model after we finish the first 10 projects.

Features at a glance

Artist is in complete control of the entire minting experience

Instead of releasing your entire collection at once, let people mint them one by one or in batches on your site, greatly increasing their value

With Nexus hosting your NFTs will never disappear and you don't have to worry about any of the technical details

We can assist you in generating your NFT images so you don't have to create 1000s by hand from scratch

We have a strategic partner that will assist you in setting up your Telegram, Discord, and take care of all the marketing that only a crypto veteran really knows how to pull off

EDU

Crypto is far too complex, noisy, and daunting for most people to consume. Sadly, on some level, a lot of people will never learn about the wonderful technical details we nerds at Nexus have come to level and interact with.

However, there's a huge demographic of people that will onboard nonetheless and get sucked into defi in the next ten years. As these concepts become more mainstream "AMM" and "liquidity" could become common vernacular.

With this in mind, and our personal journeys into defi we wanted to have a single place that connects it all together (a nexus, if you will) with a sophisticated roadmap.

You can go self-study but where do you start? Should you learn what liquidity is first? Or what about the fact that Solana uses Rust as its programming language? What about the differences between Polygon and ETH, let alone EVM compatible blockchains? Is BSC even decentralized?

Try approaching any of these topics with the uninitiated and you'll quickly find yourself in a pit of quicksand hopping backward in time trying to find some common ground where you can start from in their mental associations.

The Course Creation Process

For these reasons, we've created structured lesson plans to take the beginner to advanced. We start with no assumptions and use our journeys as research. We then focus group it with people that actively dislike crypto and finance and see what clicks.

With this approach, we've come up with a foundational set of lessons that we can build our ecosystem on. The long-term vision is to have educators able to publish their courses on our platform and get paid like any other e-learning platform. In the launch phases, we want to control the narrative and set the bar in terms of production and information quality.

The Lesson Plan So Far

Crypto Security (free): Don't your money!

Crypto 101: What is all this?

Intermediate Crypto: Your first trade!

Advanced Trading: Portfolio Management

NFTs for Artists: How to become an NFT artist

Each course is designed to pair with existing content and has supporting articles and videos on our site. All of our educational material feeds right back into the tools we're building.

We hope that all Nexus students will identify us as the "nexus" that connects them to crypto in an inviting way (thanks to our awesome UX, branding, and community experience). The educational piece with embedded and cross-linked content that is bite-sized is a big part of that strategy.

Secret Idea #001: We love "in-app" contextual help, so we'll be cross-linking inside a tooltip of "what is liquidity" next to the LP field in the tracker. Are you a pro? Just turn off contextual help inside the app.

5. Marketing Plan

Prelaunch Hype

Partnered with some crypto whale groups, shill armies and scheduled an AMA tour with engaged crypto groups.

Applied for Cretic audit before launch.

We have a whitelist competition and incentive long-term holding.

The focus is building a community of holders that will provide feedback for building the dApps.

Private sale of 100BNB is going directly to marketing.

80% of Presale is going to liquidity, and the other 20% is going directly to marketing post-launch.

Post Launch - Build, Polish, Focus

Taking the feedback loop we've created in the community, we will prioritize the most sought-after features in our roadmap and release app versions quickly.

During this build phase, we will polish our dApps while we grow organically.

We will be forming strategic partnerships with 5-10 quality NFT projects that will inform us how to scale the NFT SaaS.

Our white label token tools team is also going to follow the same model and seek 5-10 upcoming projects that we can provide technical support for and figure out where and when to automate our white label services.

These first 10 pilot program projects will all have their own marketing efforts with backlinks to Nexus, establishing us as an authority and powerful voice in the space.

At this time our first education courses will be released as well.

Each of these services will rely heavily on word of mouth and affiliate campaigns built into the supporting dApps to support the telegram and Twitter kind of referrals that are already happening.

During this build phase, we will continue a moderate AMA schedule, but in a structured format that aligns with dApp release milestones and community targets (10k holders, 20k telegram users, etc).

2022 Q1 Push - Automate and Scale

Taking the revenue from our first pilot programs, we should be able to finish converting the existing tech into a SaaS product.

We will leverage the existing community and affiliate structure to now scaling our customer acquisition across the entire ecosystem.

As revenue starts to flow we will be able to now demonstrate the revenue sharing reward structure of holding and staking in a concrete way, and this should attract even more whales and long-term holders that will benefit the valuation of the token in an organic way.

With stable cash flow, we can begin larger marketing efforts on more traditional media platforms like sponsored Coin Telegraph articles, Brave ads, and a-ads.

We will start courting large influencers and try to turn heads from major players in the space on YouTube and Twitter. Two names on our vision board are Bitboy and Gary Vee (he loves NFTs and we want to pitch an idea we'll be ready to execute on at this point).

2022 Q2 Push - Mainstream Adoption

At this time our ecosystem should reach some level of maturity and stability. We should have a brand name in education, mobile app complete, and be able to assess what we need to exponentially grow.

At this time we will need to hire teams to manage the different revenue streams and will likely restructure.

With that, buffeting will become more sophisticated so it's hard to project with many details, but each vertical we've stood up should have its own team and budget.

We will use more short-term revenue positive verticals (NFT tools) to support the longer-term projects that we see taking years like the education play. Content creation and paying teachers will likely eat costs until we reach a network effect. However, the brand effect and trust factor that Nexus will gain make this a worthwhile equation and relevant to the marketing plan.

We will re-assess and update this research paper after every quarter.

Code Snippets

```
package com.fafabtc.data.data.repo.impl;

import com.fafabtc.data.data.local.dao.PortfolioDao;
import com.fafabtc.data.data.local.dao.BlockchainAssetsDao;
import com.fafabtc.data.data.local.dao.PairDao;
import com.fafabtc.data.data.repo.BlockchainAssetsRepo;
import com.fafabtc.data.model.entity.exchange.Portfolio;
import com.fafabtc.data.model.entity.exchange.BlockchainAssets;
import com.fafabtc.data.model.vo.ExchangeAssets;

import java.util.Arrays;
import java.util.List;
import java.util.concurrent.Callable;

import javax.inject.Inject;
import javax.inject.Singleton;

import io.reactivex.Completable;
import io.reactivex.CompletableSource;
import io.reactivex.Maybe;
import io.reactivex.MaybeSource;
import io.reactivex.Observable;
import io.reactivex.ObservableSource;
import io.reactivex.Single;
import io.reactivex.functions.Action;
import io.reactivex.functions.BiFunction;
import io.reactivex.functions.Consumer;
import io.reactivex.functions.Function;
import timber.log.Timber;

/**
 * Created by jastrelax on 2018/1/8.
 */
@Singleton
public class BlockchainAssetsRepository implements BlockchainAssetsRepo {


    @Inject
    BlockchainAssetsDao blockchainAssetsDao;

    @Inject
    PortfolioDao portfolioDao;

    @Inject
    PairDao pairDao;

    @Inject
    BlockchainAssetsRepository() {

    }

    @Override
    public Completable save(final BlockchainAssets blockchainAssets) {
        return Completable.fromAction(new Action() {
            @Override
            public void run() throws Exception {
```

```
            blockchainAssetsDao.insert(blockchainAssets);
        }
    });
}

@Override
public Completable update(final BlockchainAssets blockchainAssets) {
    return Completable.fromAction(new Action() {
        @Override
        public void run() throws Exception {
            blockchainAssetsDao.update(blockchainAssets);
        }
    });
}

private Observable<String> pairBases(final String exchangeName) {
    return Observable.fromCallable(new Callable<String[]>() {
        @Override
        public String[] call() throws Exception {
            return pairDao.findBases(exchangeName);
        }
    }).flatMapIterable(new Function<String[], Iterable<String>>() {
        @Override
        public Iterable<String> apply(String[] strings) throws Exception {
            return Arrays.asList(strings);
        }
    });
}
```

```
    private Observable<String> pairQuotes(final String exchangeName) {
        return Observable.fromCallable(new Callable<String[]>() {
            @Override
            public String[] call() throws Exception {
                return pairDao.findQuotes(exchangeName);
            }
        }).flatMapIterable(new Function<String[], Iterable<String>>() {
            @Override
            public Iterable<String> apply(String[] strings) throws Exception {
                return Arrays.asList(strings);
            }
        });
    }


    @Override
    public Completable initExchangeBlockchainAssets(final String assetsUUID, final
String exchangeName) {
        return pairBases(exchangeName)
                .concatWith(Observable.fromArray(pairDao.findQuotes(exchangeName)))
                .distinct()
                .flatMapCompletable(new Function<String, CompletableSource>() {
                    @Override
                    public CompletableSource apply(final String s) throws Exception {
                        return Completable.fromAction(new Action() {
                            @Override
                            public void run() throws Exception {
                                final BlockchainAssets assets = new BlockchainAssets();
                                assets.setAssetsUuid(assetsUUID);
                                assets.setName(s);
                                assets.setExchange(exchangeName);
                                blockchainAssetsDao.insert(assets);
                            }
                        }).onErrorComplete();
                    }
                });
    }

    /**
     * Restore block chain assets of an exchange from an {@link ExchangeAssets}.
     *
     * @param exchangeAssets assets belong to an exchange and an assets account.
     * @return a Completable
     */
    @Override
    public Completable restoreExchangeBlockchainAssets(ExchangeAssets
exchangeAssets) {
        return Observable.fromIterable(exchangeAssets.getQuoteAssetsList())
                .concatWith(Observable.fromIterable(exchangeAssets.getBlockchainAssets
List()))
                .flatMapCompletable(new Function<BlockchainAssets,
CompletableSource>() {
                    @Override
                    public CompletableSource apply(final BlockchainAssets
blockchainAssets) throws Exception {
                        return Completable.fromAction(new Action() {
                            @Override
                            public void run() throws Exception {
                                blockchainAssets.setAvailable(blockchainAssets.getAvailable() +
blockchainAssets.getLocked());
```

```
                        blockchainAssets.setLocked(0.0);
                        blockchainAssetsDao.update(blockchainAssets);
                    }
                }).onErrorComplete();
            }
        });
    }

    @Override
    public Single<List<BlockchainAssets>> getBaseBlockchainAssets(final String
assetsUUID, final String exchangeName) {
        return pairBases(exchangeName)
                .flatMapMaybe(new Function<String, MaybeSource<BlockchainAssets>>()
{
                    @Override
                    public MaybeSource<BlockchainAssets> apply(final String s) throws
Exception {
                        return Maybe.fromCallable(new Callable<BlockchainAssets>() {
                            @Override
                            public BlockchainAssets call() throws Exception {
                                return blockchainAssetsDao.find(assetsUUID, exchangeName, s);
                            }
                        });
                    }
                })
                .toList();
    }

    @Override
    public Single<List<BlockchainAssets>> getQuoteBlockchainAssets(final String
assetsUUID, final String exchangeName) {
        return pairQuotes(exchangeName)
                .map(new Function<String, BlockchainAssets>() {
                    @Override
                    public BlockchainAssets apply(String s) throws Exception {
                        return blockchainAssetsDao.find(assetsUUID, exchangeName, s);
                    }
                })
                .toList();
    }

    @Override
    public Single<List<BlockchainAssets>> getAllFromExchange(final String
assetsUUID, final String exchangeName) {
        return Single.fromCallable(new Callable<List<BlockchainAssets>>() {
            @Override
            public List<BlockchainAssets> call() throws Exception {
                return blockchainAssetsDao.findByAccountWithExchange(assetsUUID,
exchangeName);
            }
        });
    }

    @Override
    public Single<BlockchainAssets> getFromCurrentPortfolio(final String
exchangeName, final String name) {
        return Single
                .fromCallable(new Callable<Portfolio>() {
                    @Override
                    public Portfolio call() throws Exception {
                        return portfolioDao.findCurrent();
                    }
                })
                .map(new Function<Portfolio, BlockchainAssets>() {
                    @Override
                    public BlockchainAssets apply(Portfolio portfolio) throws Exception {
                        return blockchainAssetsDao.find(portfolio.getUuid(), exchangeName,
name);
                    }
                });
    }

    @Override
    public Single<List<BlockchainAssets>> getFromAccountByName(final String
assetsUUID, final String name) {
        return Single
                .fromCallable(new Callable<Portfolio>() {
                    @Override
                    public Portfolio call() throws Exception {
                        return portfolioDao.findByUUID(assetsUUID);
                    }
                })
                .map(new Function<Portfolio, List<BlockchainAssets>>() {
                    @Override
                    public List<BlockchainAssets> apply(Portfolio portfolio) throws
Exception {
                        return
blockchainAssetsDao.findByAccountWithName(portfolio.getUuid(), name);
                    }
            });
    }

    @Override
    public Single<Double> getUsdtBalanceFromAccount(String assetsUUID) {
        return getFromAccountByName(assetsUUID, "usdt")
                .flattenAsObservable(new Function<List<BlockchainAssets>,
Iterable<BlockchainAssets>>() {
                    @Override
                    public Iterable<BlockchainAssets> apply(List<BlockchainAssets>
blockchainAssets) throws Exception {
                        return blockchainAssets;
                    }
                })
                .reduce(0.0, new BiFunction<Double, BlockchainAssets, Double>() {
                    @Override
                    public Double apply(Double aDouble, BlockchainAssets
blockchainAssets) throws Exception {
                        return aDouble + blockchainAssets.getAvailable() +
blockchainAssets.getLocked();
                    }
                });

    }

    private Single<String[]> quotesAsBalance = Single.fromCallable(new
Callable<String[]>() {
        @Override
        public String[] call() throws Exception {
            return pairDao.findQuotesAsBalance();
        }
    });

    @Override
    public Single<List<BlockchainAssets>> getBalanceFromAccount(final String
assetsUUID) {
        return quotesAsBalance.flattenAsObservable(new Function<String[],
Iterable<String>>() {
            @Override
            public Iterable<String> apply(String[] strings) throws Exception {
                return Arrays.asList(strings);
            }
        }).flatMap(new Function<String, ObservableSource<BlockchainAssets>>() {
            @Override
            public ObservableSource<BlockchainAssets> apply(String s) throws
Exception {
                return
Observable.fromIterable(blockchainAssetsDao.findByAccountWithName(assetsUUID
, s));
            }
        }).toList();
    }
}
  @Override
    public Single<List<BlockchainAssets>> getQuoteBlockchainAssets(final String
assetsUUID, final String exchangeName) {
        return pairQuotes(exchangeName)
                .map(new Function<String, BlockchainAssets>() {
                    @Override
                    public BlockchainAssets apply(String s) throws Exception {
                        return blockchainAssetsDao.find(assetsUUID, exchangeName, s);
                }
            })
            .toList();
    }


    @Override
    public Single<List<BlockchainAssets>> getAllFromExchange(final String
assetsUUID, final String exchangeName) {
        return Single.fromCallable(new Callable<List<BlockchainAssets>>() {
            @Override
            public List<BlockchainAssets> call() throws Exception {
                return blockchainAssetsDao.findByAccountWithExchange(assetsUUID,
exchangeName);
            }
        });
    }


    @Override
    public Single<BlockchainAssets> getFromCurrentPortfolio(final String
exchangeName, final String name) {
        return Single
            .fromCallable(new Callable<Portfolio>() {
                @Override
                public Portfolio call() throws Exception {
                    return portfolioDao.findCurrent();
                }
            })
            .map(new Function<Portfolio, BlockchainAssets>() {
                @Override
                public BlockchainAssets apply(Portfolio portfolio) throws Exception {
                        return blockchainAssetsDao.find(portfolio.getUuid(), exchangeName,
name);
                    }
                });
    }

    @Override
    public Single<List<BlockchainAssets>> getFromAccountByName(final String
assetsUUID, final String name) {
        return Single
                .fromCallable(new Callable<Portfolio>() {
                    @Override
                    public Portfolio call() throws Exception {
                        return portfolioDao.findByUUID(assetsUUID);
                    }
                })
                .map(new Function<Portfolio, List<BlockchainAssets>>() {



private Single<String[]> quotesAsBalance = Single.fromCallable(new
Callable<String[]>() {
        @Override
        public String[] call() throws Exception {
            return pairDao.findQuotesAsBalance();
        }
    });
public Completable save(final BlockchainAssets blockchainAssets) {
        return Completable.fromAction(new Action() {
            @Override
            public void run() throws Exception {
```

```
            blockchainAssetsDao.insert(blockchainAssets);
        }
    });
}

@Override
public Completable update(final BlockchainAssets blockchainAssets) {
    return Completable.fromAction(new Action() {
        @Override
        public void run() throws Exception {
            blockchainAssetsDao.update(blockchainAssets);
        }
    });
}

private Observable<String> pairBases(final String exchangeName) {
    return Observable.fromCallable(new Callable<String[]>() {
        @Override
        public String[] call() throws Exception {
            return pairDao.findBases(exchangeName);
        }
    }).flatMapIterable(new Function<String[], Iterable<String>>() {
        @Override
        public Iterable<String> apply(String[] strings) throws Exception {
            return Arrays.asList(strings);
        }
    });
}
```

```
   private Observable<String> pairQuotes(final String exchangeName) {
      return Observable.fromCallable(new Callable<String[]>() {
         @Override
         public String[] call() throws Exception {
            return pairDao.findQuotes(exchangeName);
         }
      }).flatMapIterable(new Function<String[], Iterable<String>>() {
         @Override
         public Iterable<String> apply(String[] strings) throws Exception {
            return Arrays.asList(strings);
         }
      });
   }


   @Override
   public Completable initExchangeBlockchainAssets(final String assetsUUID, final
String exchangeName) {
      return pairBases(exchangeName)
            .concatWith(Observable.fromArray(pairDao.findQuotes(exchangeName)))
            .distinct()
            .flatMapCompletable(new Function<String, CompletableSource>() {
               @Override
               public CompletableSource apply(final String s) throws Exception {
                  return Completable.fromAction(new Action() {
                     @Override
                     public void run() throws Exception {
                        final BlockchainAssets assets = new BlockchainAssets();
                        assets.setAssetsUuid(assetsUUID);
                        assets.setName(s);
                        assets.setExchange(exchangeName);
                        blockchainAssetsDao.insert(assets);
                    }
                }).onErrorComplete();
            }
        });
    }

    /**
     * Restore block chain assets of an exchange from an {@link ExchangeAssets}.
     *
     * @param exchangeAssets assets belong to an exchange and an assets account.
     * @return a Completable
     */
    @Override
    public Completable restoreExchangeBlockchainAssets(ExchangeAssets
exchangeAssets) {
        return Observable.fromIterable(exchangeAssets.getQuoteAssetsList())
                .concatWith(Observable.fromIterable(exchangeAssets.getBlockchainAssets
List()))
                .flatMapCompletable(new Function<BlockchainAssets,
CompletableSource>() {
                    @Override
                    public CompletableSource apply(final BlockchainAssets
blockchainAssets) throws Exception {
                        return Completable.fromAction(new Action() {
                            @Override
                            public void run() throws Exception {
                                blockchainAssets.setAvailable(blockchainAssets.getAvailable() +
blockchainAssets.getLocked());
```

```
                        blockchainAssets.setLocked(0.0);
                        blockchainAssetsDao.update(blockchainAssets);
                    }
                }).onErrorComplete();
            }
        });
    }

    @Override
    public Single<List<BlockchainAssets>> getBaseBlockchainAssets(final String
assetsUUID, final String exchangeName) {
        return pairBases(exchangeName)
                .flatMapMaybe(new Function<String, MaybeSource<BlockchainAssets>>()
{
                    @Override
                    public MaybeSource<BlockchainAssets> apply(final String s) throws
Exception {
                        return Maybe.fromCallable(new Callable<BlockchainAssets>() {
                            @Override
                            public BlockchainAssets call() throws Exception {
                                return blockchainAssetsDao.find(assetsUUID, exchangeName, s);
                            }
                        });
                    }
                })
                .toList();
    }

    @Override
```

```
   public Single<List<BlockchainAssets>> getQuoteBlockchainAssets(final String
assetsUUID, final String exchangeName) {
       return pairQuotes(exchangeName)
               .map(new Function<String, BlockchainAssets>() {
                   @Override
                   public BlockchainAssets apply(String s) throws Exception {
                       return blockchainAssetsDao.find(assetsUUID, exchangeName, s);
                   }
               })
               .toList();
   }
 @Override
                   public CompletableSource apply(final BlockchainAssets
blockchainAssets) throws Exception {
                       return Completable.fromAction(new Action() {
                           @Override
                           public void run() throws Exception {
                               blockchainAssets.setAvailable(blockchainAssets.getAvailable() +
blockchainAssets.getLocked());
                               blockchainAssets.setLocked(0.0);
                               blockchainAssetsDao.update(blockchainAssets);
                           }
                       }).onErrorComplete();
                   }
               });
   }
```

Conclusion

Nexus Crypto (NCR) is an established group of highly motivated people focused on bringing positive change to the crypto sphere through the continuous development of innovative tools.

NCR is creating the infrastructure to enable operators in the businesses to integrate crypto into their business operations where applicable & to be the ultimate safe zone for new & advanced enthusiasts.

NCR was founded by KSHITIJ YADAV that after having surpassed the high entry barrier and faced a steep learning curve lost a considerable amount of money falling for scams, rug pulls, and all sorts of mischiefs created by ill intentional people.

For that reason, we believe that every crypto enthusiast (noob or experienced) should be equipped with cutting-edge tools and highly curated information to make the best decision while investing in tokens, onboarding on projects or trading, and creating NFTs. The Nexus Ecosystem is divided into 3 sectors:

Nexus Folio: A set of tools designed to improve the trend experience in a decentralized market

- **Kre8:** Our platform that empowers artists and creators to mint their own NFT and provides collectors with a safe means to backup and secures their possessions

EDU: A learning center to lower the entry barriers and reduce the effort needed to start on crypto.

Future Work

A blockchain is essentially a distributed database of records, or public ledger of all transactions or digital events that have been executed and shared among participating parties. Each transaction in the public ledger is verified by consensus of a majority of the participants in the system . Once entered, information can never be erased. The blockchain contains a certain and verifiable record of every single transaction ever made. Bitcoin, the decentralized peer-to-peer digital currency, is the most popular example that uses blockchain technology. The digital currency bitcoin itself is highly controversial but the underlying blockchain technology has worked flawlessly and found wide range of applications in both financial and non-financial world.

The main hypothesis is that the blockchain establishes a system of creating a distributed consensus in the digital online world. This allows participating entities to know for certain that a digital event happened by creating an irrefutable record in a public ledger. It opens the door for developing a democratic open and scalable digital economy from a centralized one. There are tremendous opportunities in this disruptive technology, and the revolution in this space has just begun.

REFERENCES

- https://towardsdatascience.com/emotion-detection-a-machine-learning-project-f7431f652 b1f
- https://www.rcciit.org/students_projects/projects/it/2018/GR8.pdf
- https://web.stanford.edu/class/ee368/Project_Autumn_1617/Reports/ report_pao.pdf