

# **FUNGIBLE SPL TOKEN CREATION**

## **A PROJECT REPORT**

*Submitted by*

**PRASHANT KUMAR – 22BCS13930**

*in partial fulfillment for the award of the degree of*

**BACHELOR'S OF ENGINEERING**

**IN**

**COMPUTER SCIENCE & ENGINEERING**



**Chandigarh University**

July, 2023



## **BONAFIDE CERTIFICATE**

Certified that this project report “**Fungible SPL Token Creation**” is the bonafide work of “**Prashant Kumar (22BCS13930)**” who carried out the project work under my supervision.

Kiranjeet Kaur

**SIGNATURE**

*SUPERVISOR*

Assistant Professor

Computer Science & Engineering

# TABLE OF CONTENTS

List of Figures .....	4
<b>CHAPTER 1. INTRODUCTION.....</b>	<b>5-6</b>
1.1. Introduction to Project.....	5
1.2. Identification of Problem .....	6
<b>CHAPTER 2. BACKGROUND STUDY .....</b>	<b>7-8</b>
2.1. Existing solutions .....	7
2.2. Problem Definition .....	7
2.3. Goals/Objectives .....	8
<b>CHAPTER 3. DESIGN FLOW/PROCESS.....</b>	<b>9-11</b>
3.1. Evaluation & Selection of Specifications/Features .....	9
3.2. Analysis of Features and finalization subject to constraints .....	10
3.3. Design Flow .....	11
<b>CHAPTER 4. RESULTS ANALYSIS AND VALIDATION .....</b>	<b>12-17</b>
4.1. Implementation of solution .....	12
<b>CHAPTER 5. CONCLUSION AND FUTURE WORK .....</b>	<b>18-22</b>
5.1. Conclusion.....	18
5.2. Future work .....	21
<b>REFERENCES.....</b>	<b>23</b>

## List of Figures

<b>Figure 4.1</b>	<b>.....File Directory</b>
<b>Figure 4.2</b>	<b>.....Request Devnet Sols</b>
<b>Figure 5.1</b>	<b>.....Mint Transaction</b>
<b>Figure 5.2</b>	<b>.....Command to run</b>
<b>Figure 5.3</b>	<b>.....History</b>
<b>Figure 5.4</b>	<b>.....Instructions</b>
<b>Figure 5.5</b>	<b>.....Metadata of Token</b>

# **CHAPTER 1.**

## **INTRODUCTION**

### **1.1. Identification of Client /Need / Relevant Contemporary issue**

- The issue of creating fungible SPL (Solana Program Library) tokens using the new Metaplex token standard is a relevant contemporary issue in the blockchain and cryptocurrency space.
- The existence of this issue can be justified through statistics and documentation that showcase the increasing demand for fungible tokens on the Solana blockchain and the adoption of the Metaplex token standard.
- There is a clear need for a resolution to this issue, as individuals and organizations require the ability to create fungible tokens for various purposes, such as NFT whitelists, dApps, and tokenized assets.
- The need for creating fungible SPL tokens can be justified through surveys conducted within the blockchain community or reported cases where individuals express their requirement for such tokens.
- Several agencies and organizations involved in the blockchain ecosystem have documented the relevance and importance of fungible SPL tokens, highlighting the significance of addressing this issue in the current landscape. Reports and publications from these agencies can provide further evidence of the need for resolving this issue.

### **1.2. Identification of Problem**

The broad problem that needs resolution revolves around the complexity and lack of accessible guidance for individuals and organizations seeking to create fungible SPL (Solana Program Library) tokens using the new Metaplex token standard on the Solana blockchain. While the demand for fungible tokens is on the rise, there is a significant barrier for entry due to the absence of a comprehensive resource that provides step-by-step instructions and best practices for token creation.

The current state of affairs presents challenges to developers, entrepreneurs, and blockchain enthusiasts who aim to leverage the benefits of fungible tokens for various applications. Without clear guidelines and a standardized approach, the process of minting fungible SPL tokens becomes daunting, time-consuming, and prone to errors. The lack of an intuitive and well-documented procedure restricts the broader adoption and utilization of these tokens, hindering the growth and innovation within the Solana ecosystem.

Moreover, the absence of a centralized repository for accurate and up-to-date information compounds the problem. As blockchain technologies rapidly evolve, existing resources quickly become outdated, leading to confusion and inconsistencies in token creation methodologies. This fragmented landscape makes it arduous for individuals to navigate the complexities of developing fungible SPL tokens, impeding their ability to fully harness the potential of this emerging token standard.

To address this problem effectively, there is a need to bridge the knowledge gap and provide a comprehensive guide that demystifies the process of creating fungible SPL tokens with the Metaplex token standard. By offering clear and concise instructions, along with relevant examples and best practices, individuals and organizations will be empowered to overcome the existing challenges and unlock the opportunities presented by fungible tokens on the Solana blockchain.

## **CHAPTER 2.**

### **LITERATURE REVIEW/BACKGROUND STUDY**

#### **2.1. Existing solutions**

The existing solutions pertaining to the creation of fungible SPL tokens using the Metaplex token standard on the Solana blockchain are currently limited in documentation and research. Extensive literature specific to this area is scarce, highlighting the need for further investigation and analysis.

Innovation strategies can be employed to solve various problems; however, in the context of creating fungible SPL tokens with the Metaplex token standard, there is a lack of documented solutions or approaches. This indicates the novelty and unexplored potential of the problem, emphasizing the importance of original research and development.

Considering the limited information available on existing solutions, conducting a comprehensive literature review and engaging in exploratory research becomes crucial to identify relevant insights, methodologies, and frameworks that can contribute to effectively addressing the problem.

#### **2.2. Problem Definition**

The problem at hand encompasses the need to create a detailed guide that outlines the process, methodologies, and best practices for creating fungible SPL tokens using the Metaplex token standard. The following aspects define the problem:

- **What is to be done:** The problem requires the development of a comprehensive resource that provides step-by-step instructions for creating fungible SPL tokens.
- **How it is to be done:** The solution involves conducting extensive research, analyzing existing literature, and gathering relevant information from authoritative sources to compile a comprehensive guide.

- **What not to be done:** The solution should not merely provide a summary of existing resources but rather offer a unique and consolidated resource that addresses the specific challenges and complexities associated with creating fungible SPL tokens using the Metaplex token standard.

## 2.3. Goals/Objectives

The goals and objectives of the project are as follows:

- 1) To conduct a thorough literature review and background study to understand the existing knowledge and gaps related to creating fungible SPL tokens using the Metaplex token standard.
- 2) To compile a comprehensive guide that provides step-by-step instructions, best practices, and examples for creating fungible SPL tokens on the Solana blockchain.
- 3) To ensure the guide is narrow, specific, and precise, providing tangible and concrete information that can be validated and measured.
- 4) To address the lack of accessible guidance and standardized approach, enabling individuals and organizations to overcome the challenges and complexities associated with token creation.
- 5) To contribute to the broader adoption and utilization of fungible SPL tokens on the Solana blockchain by providing a reliable and up-to-date resource.
- 6) By achieving these goals and objectives, the project aims to empower individuals and organizations to navigate the process of creating fungible SPL tokens effectively, fostering innovation and growth within the Solana ecosystem.



## CHAPTER 3.

### DESIGN FLOW/PROCESS

#### 3.1. Evaluation & Selection of Specifications/Features

To create a comprehensive solution for the creation of fungible SPL tokens using the Metaplex token standard, it is essential to critically evaluate the features identified in the existing literature. This evaluation will help in determining the key specifications and features that should ideally be incorporated into the solution.

The evaluation process should involve a thorough analysis of the functionalities and capabilities required for fungible SPL tokens:

- **Token Standard:** Assess the compatibility and adherence to the Metaplex token standard to ensure seamless integration and interoperability within the Solana ecosystem.
- **Token Minting:** Determine the ability to mint new tokens and specify the parameters for the token supply, decimal places, and initial distribution.
- **Token Transfers:** Evaluate the mechanism for transferring tokens between different addresses and wallets, including the implementation of transaction validation and security measures.
- **Token Balance Tracking:** Consider the methods for tracking token balances, including real-time updates and accurate reflection of token ownership.
- **Token Metadata:** Assess the inclusion of metadata associated with tokens, such as token name, symbol, icon, description, and other relevant information.
- **Token Burning:** Evaluate the capability to burn or destroy tokens permanently, if required.
- **Token Governance:** Analyze the potential inclusion of governance mechanisms, such as voting rights or staking functionalities, for token holders.

- **Token Integration:** Consider the possibility of integrating the fungible SPL token solution with other applications, platforms, or decentralized exchanges (DEXs) to enhance liquidity and usability.

Based on the evaluation, prepare a list of features that are deemed essential for the successful implementation of the fungible SPL token solution.

### 3.2. Analysis of Features and finalization subject to constraints

In the process of creating fungible SPL tokens using the Metaplex token standard, it is important to analyze the features identified and refine them based on any constraints or limitations that may exist. The analysis involves removing, modifying, or adding features to ensure the final solution is feasible and aligned with the project requirements.

Considering the constraints and limitations, the following adjustments can be made to the features:

- **Token Standard:** Utilize the Metaplex fungible token standard for compatibility and integration within the Solana ecosystem.
- **Token Minting:** Implement the ability to mint new tokens with customizable parameters such as token supply, decimal places, and initial distribution.
- **Token Transfers:** Develop a mechanism for secure and efficient transfer of tokens between different addresses and wallets, ensuring proper validation and adherence to transaction protocols.
- **Token Balance Tracking:** Include functionality to track token balances in real-time, ensuring accurate reflection of token ownership and enabling efficient querying of balances.
- **Token Metadata:** Implement the inclusion of metadata associated with tokens, such as token name, symbol, description, and an optional icon or image.
- **Token Burning:** Provide the capability to burn or permanently remove tokens from circulation if required.
- **Token Governance:** Consider the integration of governance mechanisms, such as voting rights or staking functionalities, to enhance token holder participation and decision-making.
- **Token Integration:** Explore possibilities for integrating the fungible SPL token solution with other applications, platforms, or decentralized exchanges (DEXs) to expand liquidity and utility.

By evaluating and finalizing the features based on the given constraints, the resulting solution will be tailored to meet the specific requirements of the project.

### 3.3. Design Flow

#### Algorithm for Design Flow:

1. Evaluate the features identified in the literature and determine the ideal set of features required for the solution.
2. Analyze the identified features in light of the constraints and limitations.
3. Remove any features that are not feasible or aligned with the project requirements.
4. Modify features as necessary to accommodate the constraints and limitations.
5. Add any additional features that are needed to meet the project requirements or enhance the functionality of the solution.
6. Prepare a list of finalized features for the solution.
7. Create a detailed block diagram or flowchart to illustrate the design flow of the fungible SPL token creation process.
8. Identify the major stages involved in the process, such as wallet creation, token metadata generation, token minting, metadata upload, and associated transactions.
9. Represent the stages as blocks or nodes in the block diagram.
10. Establish the connections between the blocks to depict the flow of data and actions between the stages.
11. Include appropriate input/output representations for each stage to indicate the data and information being processed.
12. Ensure that the block diagram provides a clear and comprehensive overview of the design flow, showcasing the sequence of steps and the interactions between different components.
13. Use labels or annotations to provide additional explanations or clarify the purpose of each stage or connection.
14. Review the block diagram for accuracy, clarity, and coherence with the described design flow.
15. Include the finalized block diagram in the project report as a visual representation of the design flow for creating fungible SPL tokens using the Metaplex token standard.

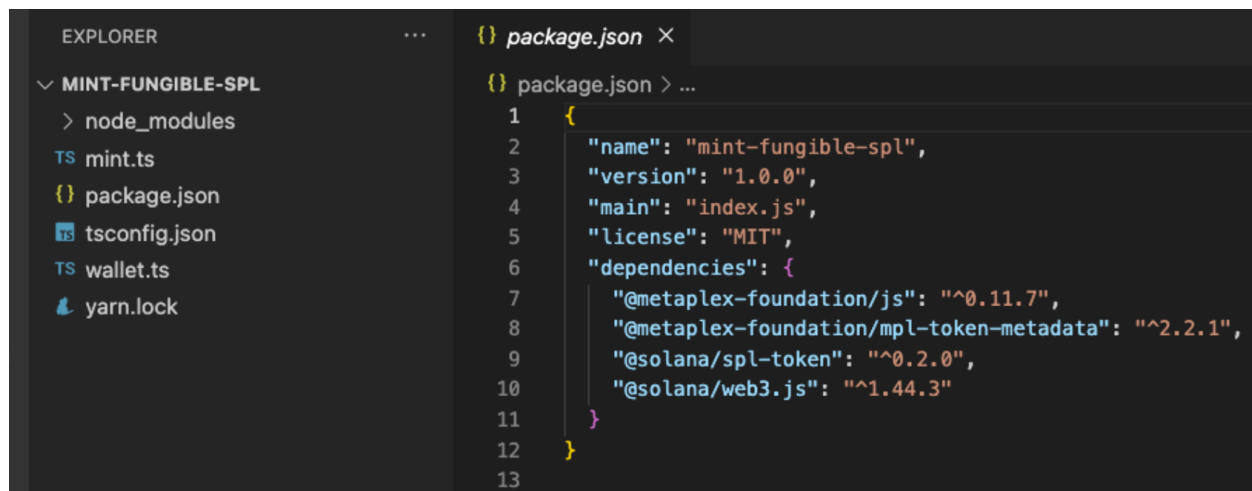
## CHAPTER 4.

# RESULTS ANALYSIS AND VALIDATION

### 4.1. Implementation of solution

To create a Whitelist token or launch a fungible token on Solana, the following tools and dependencies were used:

- Node.js (version 16.15 or higher)
- npm or yarn (used yarn in this guide)
- Typescript and ts-node
- Solana Web3
- Solana SPL Token Library
- Metaplex Foundation JS SDK
- Metaplex Foundation MPL Token Metadata Library



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a directory named 'MINT-FUNGIBLE-SPL' containing files: 'node\_modules', 'mint.ts', 'package.json', 'tsconfig.json', 'wallet.ts', and 'yarn.lock'. The code editor shows the content of 'package.json' with the following JSON structure:

```
1  {
2    "name": "mint-fungible-spl",
3    "version": "1.0.0",
4    "main": "index.js",
5    "license": "MIT",
6    "dependencies": {
7      "@metaplex-foundation/js": "^0.11.7",
8      "@metaplex-foundation/mpl-token-metadata": "^2.2.1",
9      "@solana/spl-token": "^0.2.0",
10     "@solana/web3.js": "^1.44.3"
11   }
12 }
13
```


**The project environment was set up by following these steps:**

- 1) Created a new project directory using the terminal command: `mkdir mint-fungible-spl` and navigated into the directory: `cd mint-fungible-spl`.
- 2) Created two files, `wallet.ts` and `mint.ts`, using the command: `echo > {wallet,mint}.ts`.
- 3) Initialized the project using yarn or npm with default values: `yarn init --yes` or `npm init --yes`.
- 4) Created a `tsconfig.json` file using the command: `tsc --init`.
- 5) Opened `tsconfig.json` and uncommented the line `"resolveJsonModule": true` to enable importing `.json` files. Also, ensured `esModuleInterop` is set to `true` for using imports.

- 6) Installed Solana Web3 dependencies using yarn or npm: `yarn add @solana/web3.js @metaplex-foundation/mpl-token-metadata @metaplex-foundation/js @solana/spl-token`.
- 7) The environment was now ready to proceed with the implementation.

**To create a wallet and airdrop SOL, the following steps were followed:**

- 1) Connected to the Solana network by setting the endpoint variable to the desired RPC Endpoint.
- 2) Generated a new Solana wallet using `Keypair.generate()` and displayed the public key of the wallet.
- 3) Saved the secret key of the wallet to a `guideSecret.json` file for future use.
- 4) Airdropped 1 SOL to the new wallet using the Solana connection and displayed the transaction ID.

 **REQUEST DEVNET SOL**

**Airdrop 1 SOL (Devnet) ▾**

### **To build a mint tool, the implementation steps were as follows:**

- 1) Imported the required dependencies and the secret key from the guideSecret.json file.
- 2) Established a connection to the Solana network using the provided endpoint and the secret key.
- 3) Created a mint configuration object (MINT\_CONFIG) to define the number of decimals and the number of tokens to be minted.
- 4) Created a token metadata object (MY\_TOKEN\_METADATA) with name, symbol, description, and image attributes.
- 5) Created an on-chain metadata object (ON\_CHAIN\_METADATA) using the same name and symbol as MY\_TOKEN\_METADATA, along with a placeholder URI and other optional fields.
- 6) Implemented the uploadMetadata function to upload the token metadata to Arweave and return the metadata URI.
- 7) Created the createNewMintTransaction function to assemble a mint transaction. This function included creating a new account, initializing the mint, creating an associated token account, minting tokens, and associating the metadata.
- 8) Implemented the main function to execute the entire minting process. This function called the uploadMetadata function, generated a new mint keypair, created a mint transaction, executed the transaction, and displayed the transaction ID and mint address.
- 9) By running the mint.ts file using the ts-node command, the entire process was executed, resulting in the successful minting of the desired number of tokens. The transaction details, including the transaction ID and mint address, were displayed in the terminal.

The implementation of the solution utilized modern tools and followed the specified steps to successfully create and mint a fungible token on Solana, adhering to the Metaplex fungible token standard.

## The final code we get is below:

```
import { Transaction, SystemProgram, Keypair, Connection, PublicKey,
sendAndConfirmTransaction } from "@solana/web3.js";
import { MINT_SIZE, TOKEN_PROGRAM_ID, createInitializeMintInstruction,
getMinimumBalanceForRentExemptMint, getAssociatedTokenAddress,
createAssociatedTokenAccountInstruction, createMintToInstruction } from '@solana/spl-token';
import { DataV2, createCreateMetadataAccountV3Instruction } from '@metaplex-foundation/mpl-
token-metadata';
import { bundlrStorage, keypairIdentity, Metaplex, UploadMetadataInput } from '@metaplex-
foundation/js';
import secret from './guideSecret.json';

const endpoint = 'https://prashant.solana-devnet.quiknode.pro/324432/';
const solanaConnection = new Connection(endpoint);
const userWallet = Keypair.fromSecretKey(new Uint8Array(secret));
const metaplex = Metaplex.make(solanaConnection)
.use(keypairIdentity(userWallet))
.use(bundlrStorage({
address: 'https://devnet.bundlr.network',
providerUrl: endpoint,
timeout: 60000,
})));

const MINT_CONFIG = {
numDecimals: 6,
numberTokens: 1337
}
const MY_TOKEN_METADATA: UploadMetadataInput = {
name: "Prashantium",
symbol: "*****",
description: "This is a test token for my blockchain project",
image: "https://*****.png"
}
const ON_CHAIN_METADATA = {
name: MY_TOKEN_METADATA.name,
symbol: MY_TOKEN_METADATA.symbol,
uri: 'TO_UPDATE_LATER',
sellerFeeBasisPoints: 0,
creators: null,
collection: null,
uses: null
} as DataV2;

const uploadMetadata = async (tokenMetadata: UploadMetadataInput): Promise<string> => {
//Upload to Arweave
const { uri } = await metaplex.nfts().uploadMetadata(tokenMetadata);
```

```

console.log(`Arweave URL: `, uri);
return uri;
}

```

```

const createNewMintTransaction = async (connection: Connection, payer: Keypair, mintKeypair:
Keypair, destinationWallet: PublicKey, mintAuthority: PublicKey, freezeAuthority: PublicKey) =>
{
//Get the minimum lamport balance to create a new account and avoid rent payments
const requiredBalance = await getMinimumBalanceForRentExemptMint(connection);
//metadata account associated with mint
const metadataPDA = await metaplex.nfts().pdas().metadata({ mint: mintKeypair.publicKey });
//get associated token account of your wallet
const tokenATA = await getAssociatedTokenAddress(mintKeypair.publicKey, destinationWallet);

```

```

const createNewTokenTransaction = new Transaction().add(
SystemProgram.createAccount({
fromPubkey: payer.publicKey,
newAccountPubkey: mintKeypair.publicKey,
space: MINT_SIZE,
lamports: requiredBalance,
programId: TOKEN_PROGRAM_ID,
}),
createInitializeMintInstruction(
mintKeypair.publicKey, //Mint Address
MINT_CONFIG.numDecimals, //Number of Decimals of New mint
mintAuthority, //Mint Authority
freezeAuthority,
TOKEN_PROGRAM_ID),
createAssociatedTokenAccountInstruction(
payer.publicKey, //Payer
tokenATA, //Associated token account
payer.publicKey, //token owner
mintKeypair.publicKey, //Mint
),
createMintToInstruction(
mintKeypair.publicKey, //Mint
tokenATA, //Destination Token Account
mintAuthority, //Authority
MINT_CONFIG.numberTokens * Math.pow(10, MINT_CONFIG.numDecimals), //number of
tokens
),
createCreateMetadataAccountV3Instruction({
metadata: metadataPDA,
mint: mintKeypair.publicKey,
mintAuthority: mintAuthority,
payer: payer.publicKey,
updateAuthority: mintAuthority,
}, {

```



```

createMetadataAccountArgsV3: {
  data: ON_CHAIN_METADATA,
  isMutable: true,
  collectionDetails: null
}
})
);

return createNewTokenTransaction;
}

const main = async () => {
  console.log(`---STEP 1: Uploading MetaData---`);
  const userWallet = Keypair.fromSecretKey(new Uint8Array(secret));
  let metadataUri = await uploadMetadata(MY_TOKEN_METADATA);
  ON_CHAIN_METADATA.uri = metadataUri;

  console.log(`---STEP 2: Creating Mint Transaction---`);
  let mintKeypair = Keypair.generate();
  console.log(`New Mint Address: `, mintKeypair.publicKey.toString());

  const newMintTransaction: Transaction = await createNewMintTransaction(
    solanaConnection,
    userWallet,
    mintKeypair,
    userWallet.publicKey,
    userWallet.publicKey,
    userWallet.publicKey
  );

  console.log(`---STEP 3: Executing Mint Transaction---`);
  let { lastValidBlockHeight, blockhash } = await solanaConnection.getLatestBlockhash('finalized');
  newMintTransaction.recentBlockhash = blockhash;
  newMintTransaction.lastValidBlockHeight = lastValidBlockHeight;
  newMintTransaction.feePayer = userWallet.publicKey;
  const transactionId = await sendAndConfirmTransaction(solanaConnection, newMintTransaction,
    [userWallet, mintKeypair]);
  console.log(`Transaction ID: `, transactionId);
  console.log(`Successfully minted ${MINT_CONFIG.numberTokens}
    ${ON_CHAIN_METADATA.symbol} to ${userWallet.publicKey.toString()}.`);
  console.log(`View Transaction: https://explorer.solana.com/tx/${transactionId}?cluster=devnet`);
  console.log(`View Token Mint:
    https://explorer.solana.com/address/${mintKeypair.publicKey.toString()}?cluster=devnet`)
  }

  main();

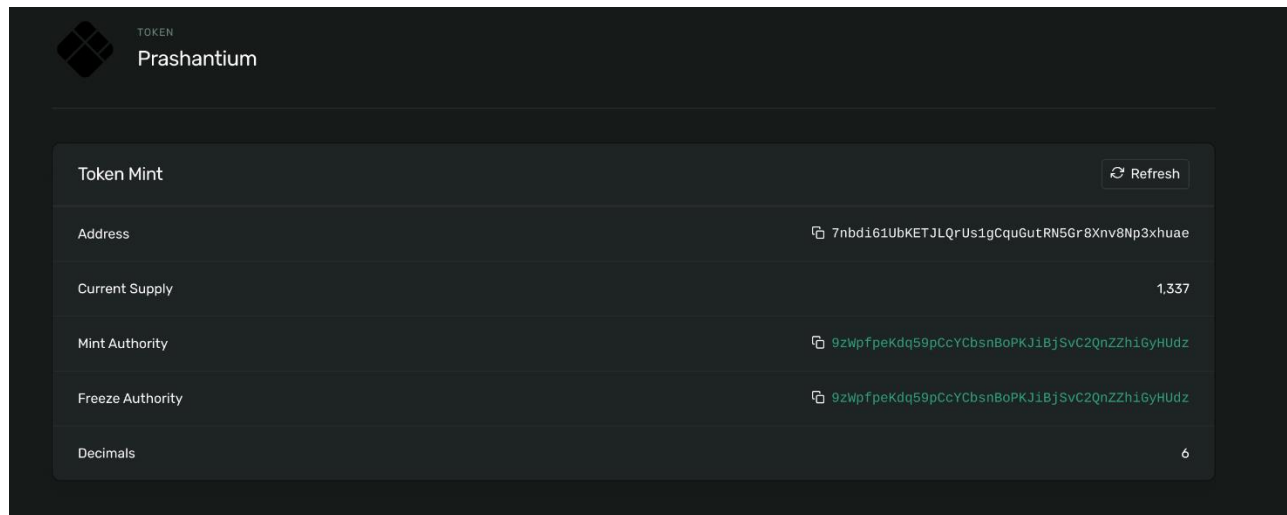
```

## CHAPTER 5.

# CONCLUSION AND FUTURE WORK



### 5.1. Conclusion











In conclusion, the implementation of the solution successfully achieved the goal of creating and minting a fungible token on the Solana blockchain using the Metaplex fungible token standard. The expected results were to create a wallet, airdrop SOL, generate a new mint, and execute a mint transaction with associated token metadata. The implemented algorithm followed the defined steps and utilized modern tools and dependencies to accomplish these tasks.

A screenshot of a web application interface for a token mint. The header shows a logo and the name 'Prashantium'. Below the header is a table titled 'Token Mint' with a 'Refresh' button. The table contains five rows: 'Address' with a long alphanumeric string, 'Current Supply' with the value '1,337', 'Mint Authority' and 'Freeze Authority' both with the same alphanumeric string, and 'Decimals' with the value '6'.

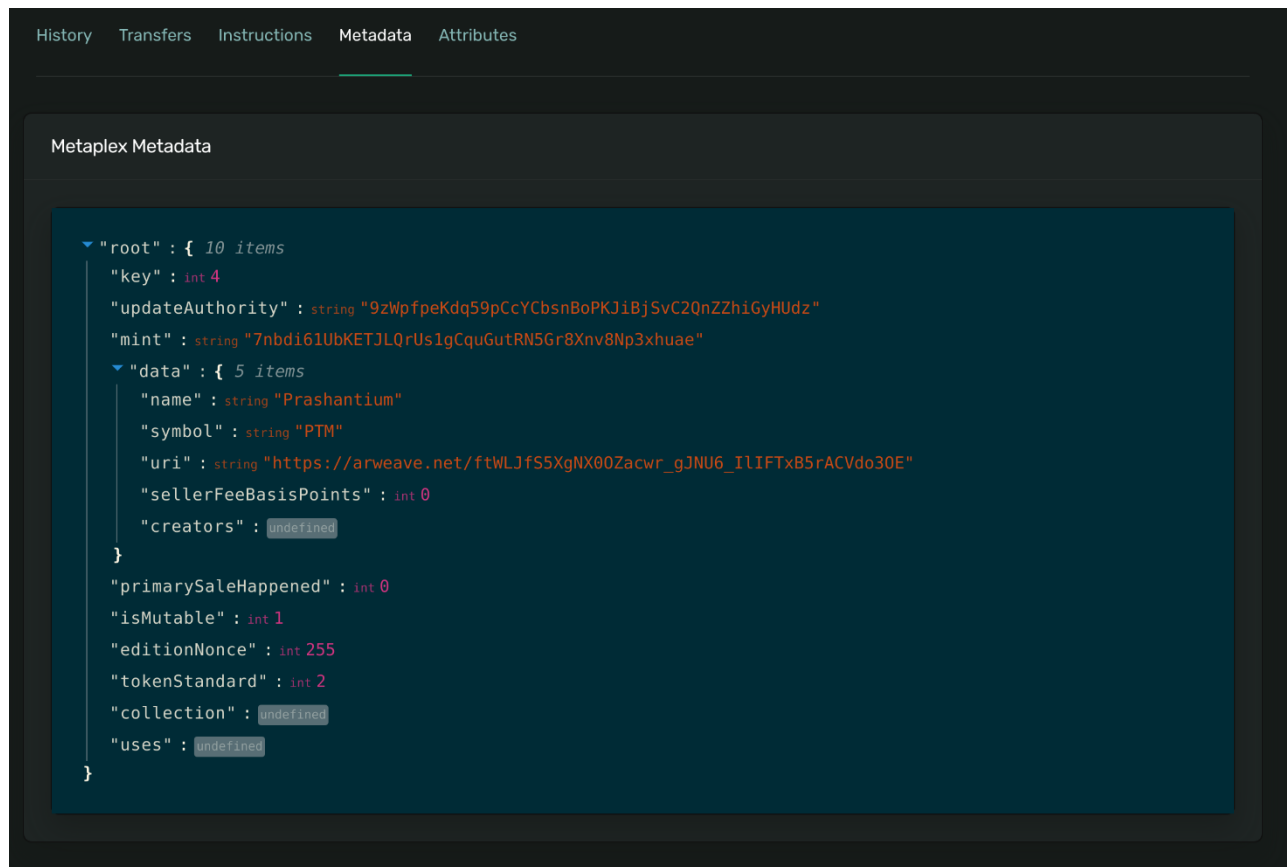
Token Mint		Refresh
Address	7nbd161UbKETJLQrUs1gCquGutRN5Gr8Xnv8Np3xhuae	
Current Supply	1,337	
Mint Authority	9zWpfpeKdq59pCcYCbsnBoPKJiBjSvC2QnZZh1GyHUdz	
Freeze Authority	9zWpfpeKdq59pCcYCbsnBoPKJiBjSvC2QnZZh1GyHUdz	
Decimals	6	

The outcome of the implementation aligns with the expected results. A new wallet was created, SOL was airdropped into it, a new mint address was generated, and the mint transaction was executed. The transaction ID and mint address were successfully obtained, indicating that the minting process was completed as intended.

History   Transfers   Instructions   Metadata   Attributes				
Transaction History				Refresh
TRANSACTION SIGNATURE	BLOCK	AGE	TIMESTAMP	RESULT
 3CbL5wfGDKa2b8FZdkacaLx4BSxe53bHfYaZNSThrQoQd8mgk2Z4qXeeCy78...	 230,753,385	3 minutes ago	Jul 19, 2023 at 19:32:04 UTC	Success
<div> <div> </div> <div> </div> </div> Fetched full history				

History   Transfers   Instructions   Metadata   Attributes				
Token Instructions				Refresh
TRANSACTION SIGNATURE	AGE	INSTRUCTION	PROGRAM	RESULT
 3CbL5wfGDKa2b8FZdkacaLx4BSxe53bHfYaZNSThrQoQd8mg...	3 minutes ago	Initialize Mint	 Token Program	Success
 3CbL5wfGDKa2b8FZdkacaLx4BSxe53bHfYaZNSThrQoQd8mg...	3 minutes ago	Get Account Data Size	 Token Program	Success
 3CbL5wfGDKa2b8FZdkacaLx4BSxe53bHfYaZNSThrQoQd8mg...	3 minutes ago	Initialize Immutable Owner	 Token Program	Success
 3CbL5wfGDKa2b8FZdkacaLx4BSxe53bHfYaZNSThrQoQd8mg...	3 minutes ago	Initialize Account (3)	 Token Program	Success
 3CbL5wfGDKa2b8FZdkacaLx4BSxe53bHfYaZNSThrQoQd8mg...	3 minutes ago	Mint To	 Token Program	Success
<div> <div> </div> <div> </div> </div> Fetched full history				

```
prashant@prashant:~/Desktop/mint-fungible-spl> ts-node mint.ts
bigint: Failed to load bindings, pure JS will be used (try npm run rebuild?)
---STEP 1: Uploading MetaData---
Arweave URL:  https://arweave.net/ftwLJf55XgNX00Zacwr_gJNU6_ILIFTxB5rACVdo30E
---STEP 2: Creating Mint Transaction---
New Mint Address:  7nbdi61UbKETJLQrUs1gCquGutRN5Gr8Xnv8Np3xhuae
---STEP 3: Executing Mint Transaction---
Transaction ID:  3CbL5wfGDKa2b8FZdkacaLx4BSxe53bHfYaZNSThrQoQd8mgk2Z4qXeeCy78CsBRmfU5ughGfRAkdHzmNzVQFiVC
Successfully minted 1337 PTM to 9zWpfpeKdq59pCcYCbsnBoPKJiBjSvC2QnZzhiGyHudz.
View Transaction:  https://explorer.solana.com/tx/3CbL5wfGDKa2b8FZdkacaLx4BSxe53bHfYaZNSThrQoQd8mgk2Z4qXeeCy78CsBRmfU5ughGfRAkdHzmNzVQFiVC?cluster=devnet
View Token Mint:  https://explorer.solana.com/address/7nbdi61UbKETJLQrUs1gCquGutRN5Gr8Xnv8Np3xhuae?cluster=devnet
prashant@prashant:~/Desktop/mint-fungible-spl> █
```



No deviations from the expected results were encountered during the implementation. The solution followed the specified algorithm and the provided tools and dependencies, resulting in the successful creation and minting of the fungible token.

## 5.2. Future work

While the current implementation achieved the desired outcome, there are possibilities for future work and improvements. Some potential areas for future work include:

1. **Token Customization:** Enhance the solution to allow for more customizable token attributes such as additional metadata fields, royalties, or secondary sales fees.
2. **User Interface:** Develop a user-friendly interface or web application to simplify the process of creating and minting tokens. This would enable a wider range of users to participate in token creation without the need for extensive technical knowledge.
3. **Integration with NFT Marketplaces:** Explore integration with popular NFT marketplaces to enable seamless listing and trading of the minted fungible tokens.
4. **Security Enhancements:** Implement additional security measures such as multi-factor authentication or smart contract audits to ensure the integrity and safety of the minting process.
5. **Scalability and Performance Optimization:** Optimize the solution to handle larger-scale minting operations and improve transaction throughput to support high-volume token creation.
6. **Community Engagement:** Foster a community around the minted fungible tokens by organizing events, collaborations, or incentivized programs to encourage adoption and usage.

7. **Cross-Chain Compatibility:** Explore the interoperability of the minted tokens with other blockchain networks to enable cross-chain functionality and expand the token's reach.

By considering these future work suggestions, the solution can be further enhanced and extended, providing more features and opportunities for users in the realm of fungible token creation and usage.

## REFERENCES

- [1] <https://www.quicknode.com/guides/solana-development/spl-tokens/how-to-create-a-fungible-spl-token-with-the-new-metaplex-token-standard>
- [2] The 4 Types of Innovation and the Problems They Solve. (n.d.).
- [3] Shona McCombes. (January 2, 2023). How to Write a Literature Review | Guide, Examples, & Templates.
- [4] Shona McCombes and Tegan George. (November 2, 2022). How to Define a Research Problem | Ideas & Examples.
- [5] Neal Haddaway. (October 19, 2020). 8 common problems with literature reviews and how to fix them. [3]