**User Guide for Minswap Cardano in Hummingbot**

**Table of Contents**

[Connecting to Minswap Gateway 1](#__RefHeading___Toc3_2757360289)

[Checking Balance on Minswap Gateway 2](#__RefHeading___Toc5_2757360289)

[Adding a Token to the Gateway Balance Check 2](#__RefHeading___Toc7_2757360289)

[Creating a Script to Fetch Price of ADA-MIN from Minswap Liquidity Pool 3](#__RefHeading___Toc9_2757360289)

[Starting and Fetching Price of ADA-MIN from Minswap Liquidity Pool 4](#__RefHeading___Toc11_2757360289)

[Trading on Minswap Liquidity Pool 5](#__RefHeading___Toc13_2757360289)

[Create a Trade Script 5](#__RefHeading___Toc15_2757360289)

[Start the Trade Script 5](#__RefHeading___Toc17_2757360289)

[Result of the Trade 6](#__RefHeading___Toc19_2757360289)

[Trading on Minswap Liquidity Pool with User Specified price 6](#__RefHeading___Toc13_2757360289_Copy_1)

[Create a Trade Script with user specified price 6](#__RefHeading___Toc15_2757360289_Copy_1)

[Start the Trade Script 7](#__RefHeading___Toc17_2757360289_Copy_2)

[Result of the Trade 8](#__RefHeading___Toc13562_3020575491)

[Add Liquidity on Minswap 10](#__RefHeading___Toc13_2757360289_Copy_1_C)

[Create a Add Liquidity Script with user specified amount 10](#__RefHeading___Toc15_2757360289_Copy_1_C)

[Start the Liquidity Script 11](#__RefHeading___Toc17_2757360289_Copy_2_C)

[Result of the Add Liquidity Action 11](#__RefHeading___Toc13564_3020575491)

[Remove Liquidity on Minswap 13](#__RefHeading___Toc13_2757360289_Copy_1_1)

[Create a Remove Liquidity Script with user specified percentage 13](#__RefHeading___Toc15_2757360289_Copy_1_1)

[Start the Liquidity Script 13](#__RefHeading___Toc17_2757360289_Copy_2_1)

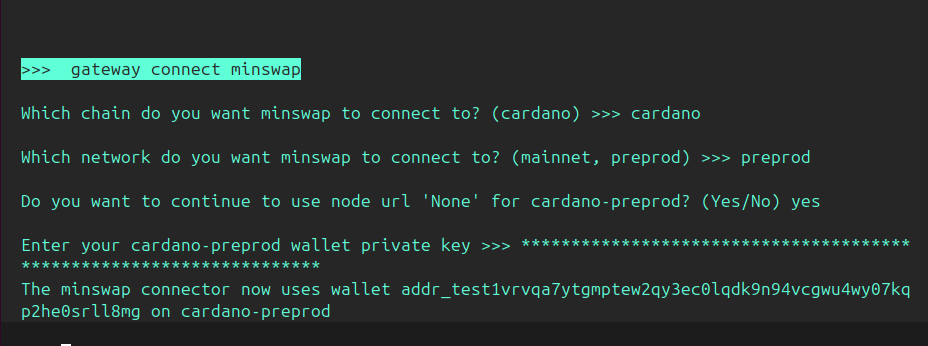
[Result of the Liquidity Removal Trade 14](#__RefHeading___Toc13720_3020575491)

### ****Connecting to Minswap Gateway****

1. **Run the Command**:

gateway connect minswap

1. **Select Blockchain**: Type cardano.
2. **Choose Network**:
   * For mainnet: mainnet
   * For pre-production: preprod
3. **Confirm Node URL**: Type yes to use the default node.
4. **Enter Private Key**: Provide your Cardano wallet private key (keep it secure).
5. **Success Message**: You'll see confirmation with your wallet address connected.

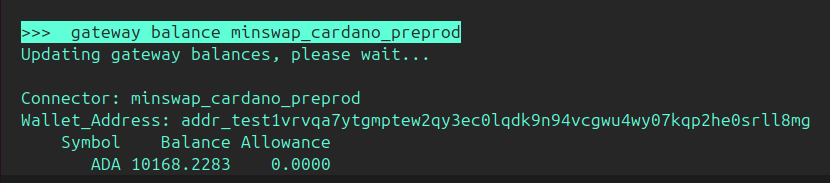


### ****Checking Balance on Minswap Gateway****

1. **Run the Command**:

gateway balance minswap\_cardano\_preprod

1. **Wait for the Update**: The gateway will fetch your wallet balance.
2. **View Results**:
   * **Wallet Address**: Displays the connected wallet address.
   * **Symbol**: The token (e.g., ADA).
   * **Balance**: Available balance in the wallet



### ****Adding a Token to the Gateway Balance Check****

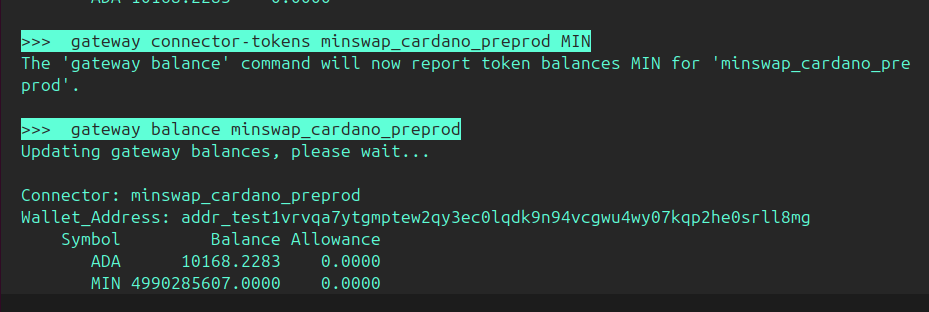
1. **Add a Token**:  
   Use the following command to add a specific token (e.g., MIN) to your balance check.

gateway connector-tokens minswap\_cardano\_preprod MIN

1. **Verify the Balance**:  
   Run the balance command to view the added token balance.

gateway balance minswap\_cardano\_preprod

1. **Output Details**:
   * **ADA**: Displays ADA balance.
   * **MIN**: Displays the balance of the added token (e.g., MIN).



### ****Creating a Script to Fetch Price of ADA-MIN from Minswap Liquidity Pool****

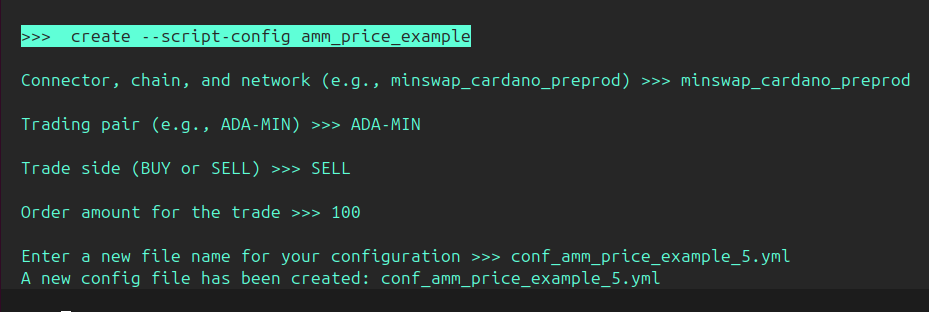
1. **Run the Command**:  
   Use the create command with the following structure:

create --script-config amm\_price\_example

1. **Provide the Required Inputs**:
   * **Connector, Chain, and Network**: Enter the connector (e.g., minswap\_cardano\_preprod).
   * **Trading Pair**: Specify the pair (e.g., ADA-MIN).
   * **Trade Side**: Choose BUY or SELL (e.g., SELL).
   * **Order Amount**: Enter the trade amount (e.g., 100).
   * **File Name**: Provide a new file name for the configuration (e.g., conf\_amm\_price\_example\_5.yml).
2. **Result**:  
   A new configuration file will be created:

A new config file has been created: conf\_amm\_price\_example\_5.yml

1. **Next Steps**:  
   Use the generated configuration file (conf\_amm\_price\_example\_5.yml) to start the script



### ****Starting and Fetching Price of ADA-MIN from Minswap Liquidity Pool****

1. **Run the Script**:  
   Use the start command to begin the script and fetch balance:

start --script amm\_price\_example.py --conf conf\_amm\_price\_example\_5.yml

1. **Execution Process**:
   * The script will initialize and fetch price information for the specified trading pair (ADA-MIN in this case).
   * Example output:

Status check complete. Starting 'amm\_price\_example' strategy...

'amm\_price\_example' strategy started.

Run 'status' command to query the progress.

1. **Stop the Script**:  
   To terminate the script, use the stop command:

stop

Output:

Winding down...

Canceling outstanding orders... All outstanding orders canceled.

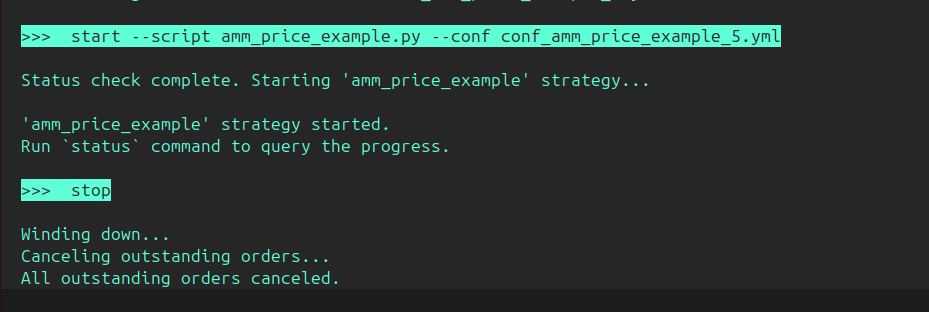
1. **Result**:  
   The script fetches the price and balance details for the trading pair. Example log:

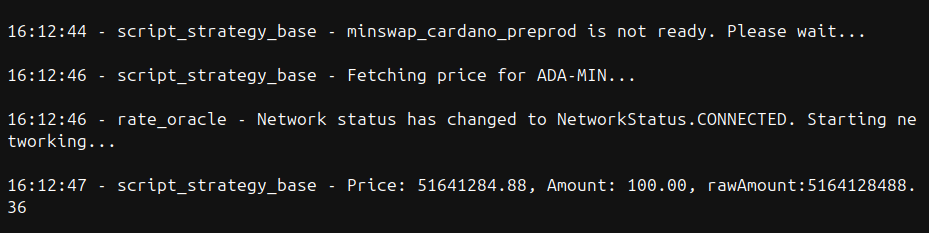
16:12:44 - script\_strategy\_base - minswap\_cardano\_preprod is not ready. Please wait...

16:12:46 - script\_strategy\_base - Fetching price for ADA-MIN...

16:12:46 - rate\_oracle - Network status has changed to NetworkStatus.CONNECTED. Starting networking...

16:12:47 - script\_strategy\_base - Price: 51641284.88, Amount: 100.00, rawAmount: 5164128488.36





### ****Trading on Minswap Liquidity Pool****

#### ****Create a Trade Script****

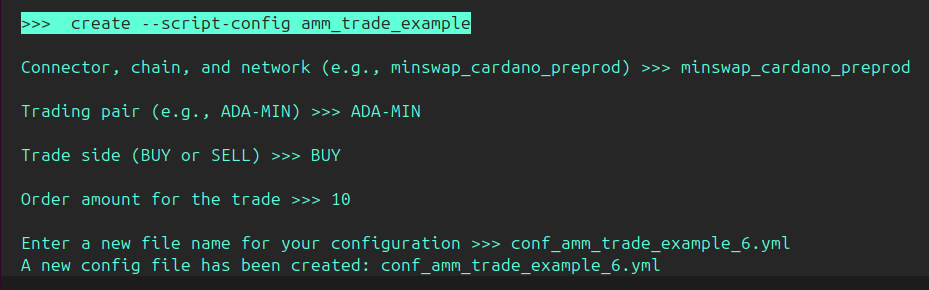
To set up a trade on the Minswap liquidity pool:

create --script-config amm\_trade\_example

* **Connector, chain, and network**: minswap\_cardano\_preprod
* **Trading pair**: ADA-MIN
* **Trade side**:
  + BUY: Buy ADA using MIN.
  + SELL: Sell MIN to receive ADA.
* **Order amount**: Example: 10
* **Configuration file**: Example: conf\_amm\_trade\_example\_6.yml

Output:

A new config file has been created: conf\_amm\_trade\_example\_6.yml



#### ****Start the Trade Script****

Start the script to execute the trade:

start --script amm\_trade\_example.py --conf conf\_amm\_trade\_example\_6.yml

* The script fetches the price, adjusts it with slippage, and executes the trade.
* Example output:

Status check complete. Starting 'amm\_trade\_example' strategy...

'amm\_trade\_example' strategy started.

Run 'status' command to query the progress.

#### 

#### ****Result of the Trade****

1. **Fetching Price and Adjustments**:
   * Current price: 51641284.88
   * Adjusted price (with slippage): 52157697.728800006
2. **Balances Before Trade**:

ADA: 10168.22826

MIN: 4990285607

1. **Executing Trade**:  
   Transaction ID: 524276816558701e2ca48db7bab0dcead59515c6602d43ca139daf61b12d2b6d
   * Status: Confirmed
   * Block: 14df4442a1c4ae1cdd1c799683fc3b424bfcfef4198b27fc4fc1d95d70980019
   * Block height: 3062051
   * Fees: 186973
2. **Balances After Trade**:

ADA: 10176.541287

MIN: 4472134049

### ****Trading on Minswap Liquidity Pool with User Specified price****

#### ****Create a Trade Script with user specified price****

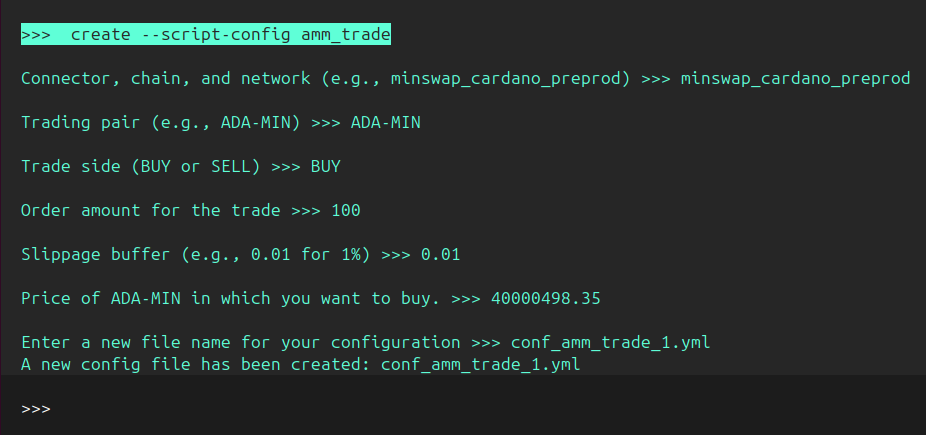
**To set up a trade on the Minswap liquidity pool:**

**create --script-config amm\_trade**

* **Connector, chain, and network**: minswap\_cardano\_preprod
* **Trading pair(e.g, ADA\_MIN)**: ADA-MIN
* **Trade side**:
  + BUY: Buy ADA using MIN.
  + SELL: Sell MIN to receive ADA.
* **Order amount**: Example: 100
* **Slippage buffer (e.g., 0.01 for 1%): 0.01**
* **Price of ADA-MIN in which you want to buy: 40000498.35**
* **Configuration file: conf\_amm\_trade\_1.yml**

Output:

A new config file has been created: **conf\_amm\_trade\_1.yml**



#### ****Start the Trade Script****

Start the script to execute the trade:

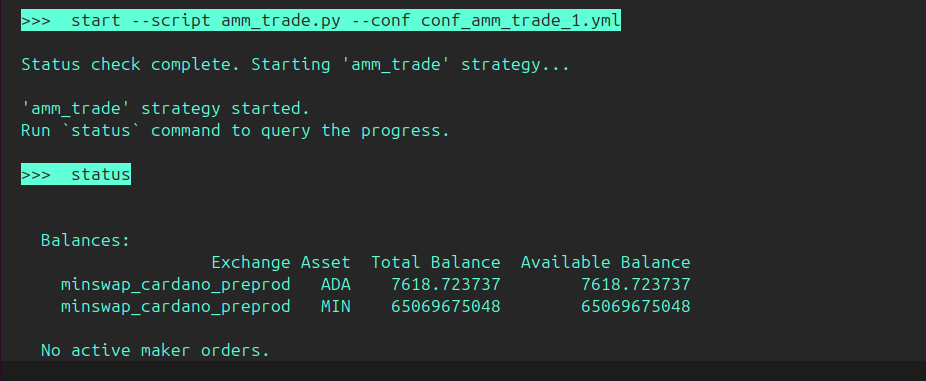
**start --script amm\_trade.py --conf conf\_amm\_trade\_1.yml**

* The script fetches the price, adjusts it with slippage, and executes the trade.
* Example output:

**Status check complete. Starting 'amm\_trade' strategy...**

**'amm\_trade' strategy started.**

**Run 'status' command to query the progress.**



#### ****Result of the Trade****

1. **Fetching Price and Adjustments:**

* **Initial Attempt (14:15:55):**
  + Current Price: 40014498.35
  + Adjusted Price (with Slippage): 40414643.333500006
  + User Specified Price: 40000498.35
  + **Trade Status:** Conditions not met. Trade skipped.
* **Subsequent Attempt (14:29:38):**
  + Current Price: 39765640.1
  + Adjusted Price (with Slippage): 40163296.501000006
  + User Specified Price: 40000498.35
  + **Trade Status:** Conditions met. Executing trade...

2. **Balances Before Trade:**

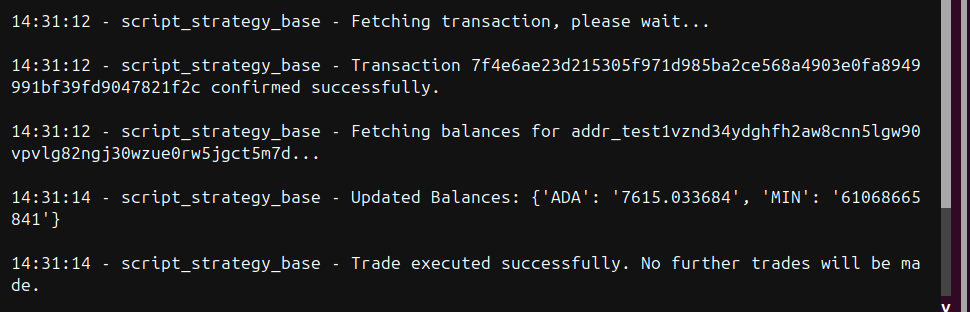
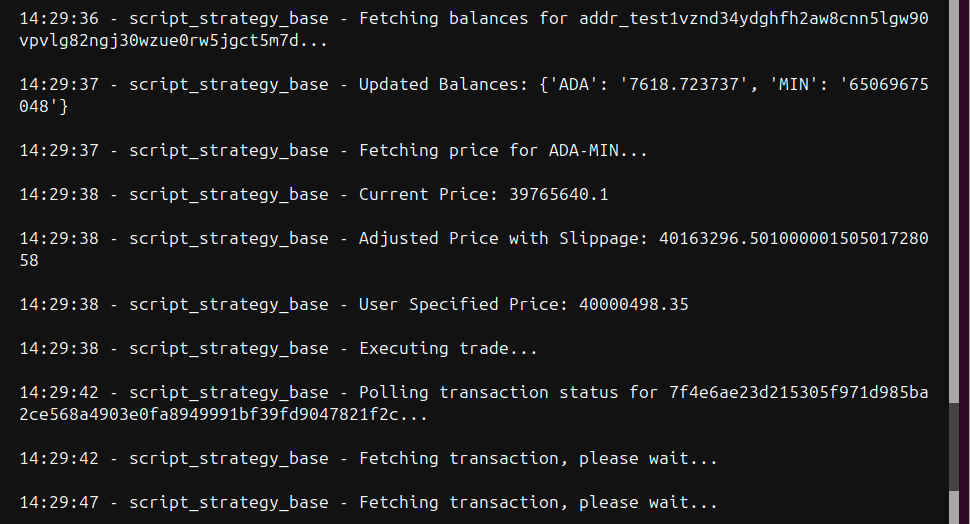
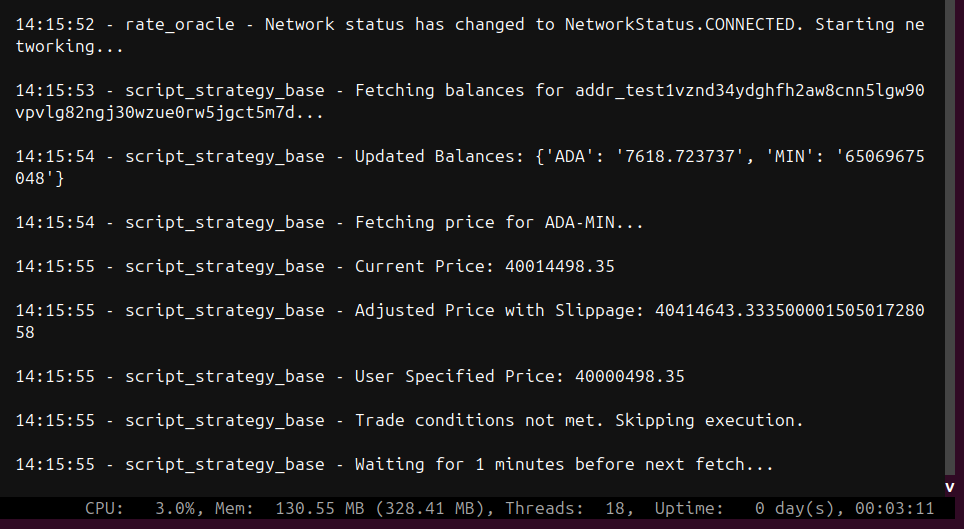
* **ADA:** 7618.723737
* **MIN:** 65069675048

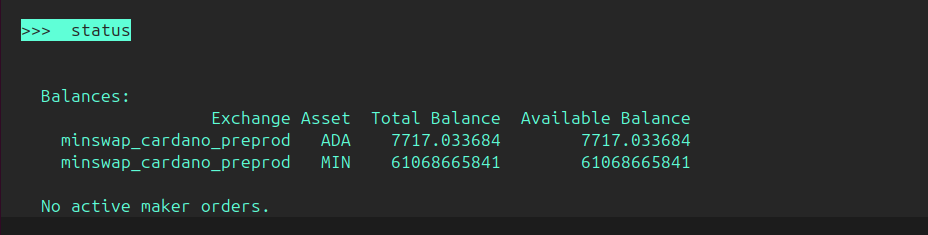
3. **Executing Trade:**

* **Transaction ID:** 7f4e6ae23d215305f971d985ba2ce568a4903e0fa8949991bf39fd9047821f2c
* **Status:** Confirmed
* **Block:** Not Specified
* **Block Height:** Not Specified
* **Fees:** Not Specified

4. **Balances After Trade:**

* **ADA:** 7717.033684
* **MIN:** 61068665841





### ****Add Liquidity on Minswap****

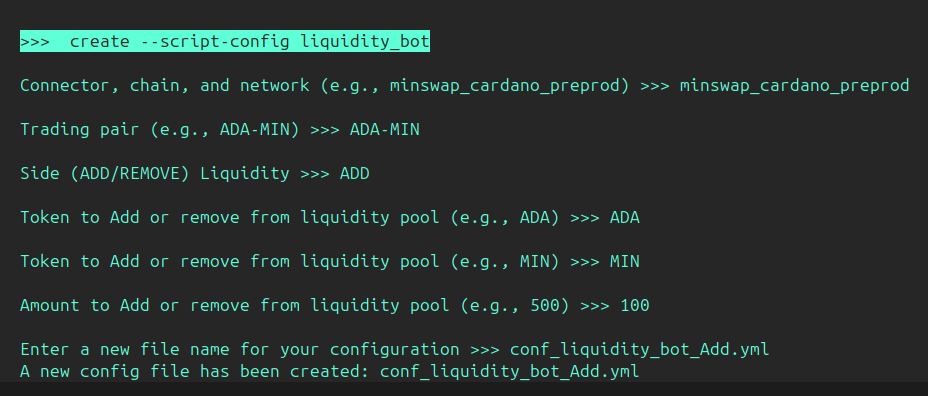
#### ****Create a Add Liquidity Script with user specified amount****

**create --script-config liquidity\_bot**

* **Connector, chain, and network**: minswap\_cardano\_preprod
* **Trading pair(e.g, ADA\_MIN)**: ADA-MIN
* ****Side (ADD/REMOVE) Liquidity****:
  + ADD: Add liquidity to minswap liquidity pool
  + REMOVE: Remove liquidity from minswap liquidity pool
* **Token to Add or remove from liquidity pool (e.g., ADA): ADA**
* **Token to Add or remove from liquidity pool (e.g., MIN): MIN**
* **Amount to Add or remove from liquidity pool (e.g., 100) : 100**
* **Configuration file: conf\_liquidity\_bot\_Add.yml**

Output:

A new config file has been created: **conf\_liquidity\_bot\_Add.yml**

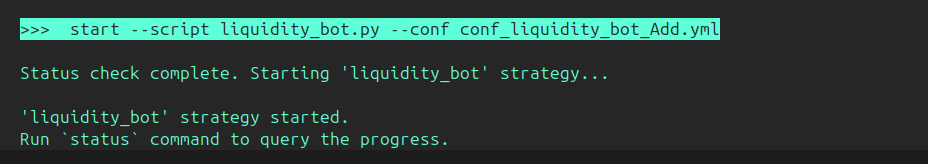


#### ****Start the Liquidity Script****

Start the script to Add liquidity:

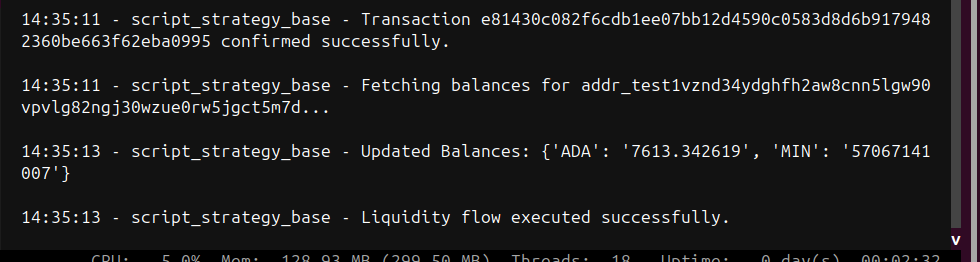
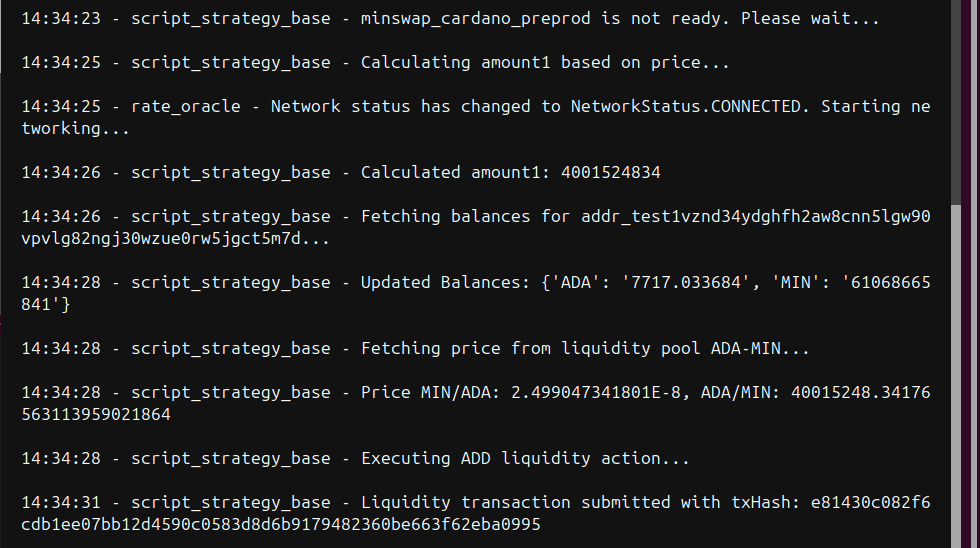
**start --script liquidity\_bot.py --conf conf\_liquidity\_bot\_Add.yml**

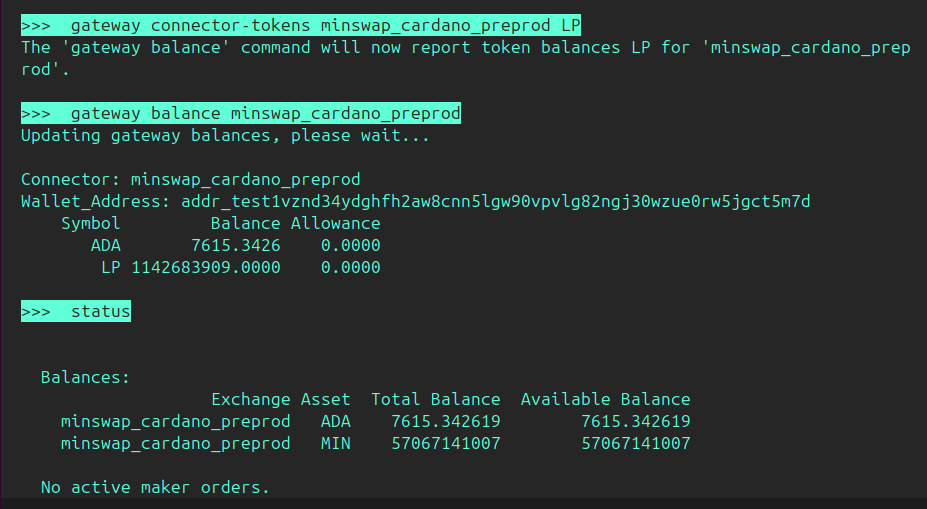
* The script calculates the amount1 required for trade, and fetch the wallet balance and if the wallet has sufficient balance then it executes the trade.
* Example output:
  + **Status check complete. Starting 'liquidity\_bot' strategy...**
  + **'liquidity\_bot' strategy started.**
  + **Run `status` command to query the progress.**



#### ****Result of the Add Liquidity Action****

1. **Fetching Price and Calculations:**
   * Network Status: Connected.
   * Price MIN/ADA: 2.499047341801E-8.
   * Price ADA/MIN: 40,015,248.34176563113959021864.
   * Calculated Amount1 (MIN): 4,001,524,834.
2. **Balances Before Liquidity Addition:**
   * **ADA:** 7,717.033684
   * **MIN:** 61,068,665,841
3. **Executing Add Liquidity Action:**
   * Liquidity Transaction Submitted Successfully:
     + **Transaction ID:** e81430c082f6cdb1ee07bb12d4590c0583d8d6b9179482360be663f62eba0995.
4. **Balances After Liquidity Addition:**
   * **ADA:** 7,615.3426
   * **MIN:** 57,067,141,007
   * **Liquidity Pool Tokens (LP):** 1,142,683,909





### ****Remove Liquidity on Minswap****

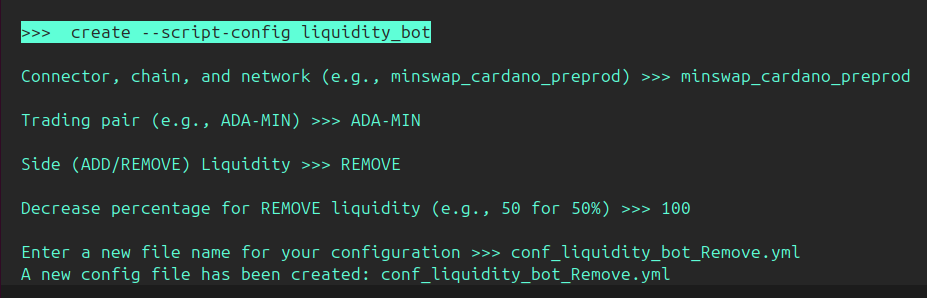
#### ****Create a Remove Liquidity Script with user specified percentage****

**create --script-config liquidity\_bot**

* **Connector, chain, and network**: minswap\_cardano\_preprod
* **Trading pair(e.g, ADA\_MIN)**: ADA-MIN
* ****Side (ADD/REMOVE) Liquidity****:
  + ADD: Add liquidity to minswap liquidity pool
  + REMOVE: Remove liquidity from minswap liquidity pool
* ****Decrease percentage for REMOVE liquidity (e.g., 50 for 50%)**: 100**
* ****Enter a new file name for your configuration**: conf\_liquidity\_bot\_Remove.yml**

Output:

A new config file has been created: **conf\_liquidity\_bot\_Remove.yml**

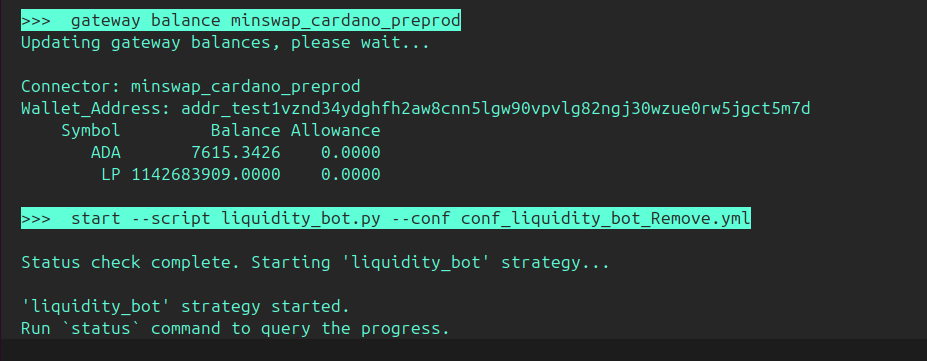


#### ****Start the Liquidity Script****

Start the script to Remove liquidity:

**start --script liquidity\_bot.py --conf conf\_liquidity\_bot\_Remove.yml**

* The script takes the L.P token from user wallet and calculates how much ADA and MIN user will get and executes the trade.
* Example output:
  + **Status check complete. Starting 'liquidity\_bot' strategy...**
  + **'liquidity\_bot' strategy started.**
  + **Run `status` command to query the progress.**



#### Result of the Liquidity Removal Trade

1. **Fetching Price and Adjustments:**
   * Current Price (ADA-MIN): 40015248.34176
   * Adjusted Price (with slippage): 40015248.34176
2. **Balances Before Trade:**
   * ADA: 7615.342619
   * MIN: 57067141007
3. **Executing Liquidity Removal:**
   * Transaction ID: e2afa987ebf569178695c2e5ab8f7c2993cd49d307089cda8409002a016b3e3b
   * Status: Confirmed
   * Transaction Hash: e2afa987ebf569178695c2e5ab8f7c2993cd49d307089cda8409002a016b3e3b
4. **Balances After Trade:**
   * ADA: 7813.340493
   * MIN: 65057690701

