|  |
| --- |
| [Default] |
| Requirements Specification (RS) |
|  |

|  |
| --- |
| Witold Zolnowski  17143853  12/06/2018 |

Requirements Specification (RS)

Document Control

Revision History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Date** | **Version** | **Scope of Activity** | **Prepared** | **Reviewed** | **Approved** |
|  | 1 |  |  |  |  |
|  | 2 |  |  |  |  |

Distribution List

|  |  |  |
| --- | --- | --- |
| **Name** | **Title** | **Version** |
| David Hamill | Lecturer |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Related Documents

|  |  |
| --- | --- |
| **Title** | **Comments** |
| Title of Use Case Model |  |
| Title of Use Case Description |  |

**Table of Contents**

[1 Introduction 3](#_Toc316977392)

[1.1 Purpose 3](#_Toc316977393)

[1.2 Project Scope 3](#_Toc316977394)

[1.3 Definitions, Acronyms, and Abbreviations 3](#_Toc316977395)

[2 Requirements Specification 3](#_Toc316977397)

[2.1 Functional requirements 3](#_Toc316977398)

[2.1.1 Use Case Diagram 4](#_Toc316977399)

[2.1.2 Requirement 1](#_Toc316977400)

[2.1.3 Requirement](#_Toc316977400) 2

2.1.4 Requirement 3

2.1.5 Requirement 4

[2.2 Non-Functional Requirements 9](#_Toc316977402)

[2.2.1 Performance/Response time requirement 9](#_Toc316977403)

[2.2.2 Availability requirement 9](#_Toc316977404)

[2.2.3 Recover requirement 9](#_Toc316977405)

[2.2.4 Robustness requirement 9](#_Toc316977406)

[2.2.5 Security requirement 9](#_Toc316977407)

[3 GUI 10](#_Toc316977414)

[4 System Architecture 13](#_Toc316977415)

[5 System evolution 13](#_Toc316977416)

# Introduction

## Purpose

The purpose of this document is to set out the requirements for ‘Crypto Scanner’ application. The application will identify arbitrage opportunities within cryptocurrency markets.

The application is intended for personal use only.

## Project Scope

The scope of the project is to develop an application that will analyse current market prices in real time, and identify arbitrage opportunities (if any). The application will be limited to data collection from external sources, and it’s analysis. This means that results will be dependent on reliability of data supplied by third party services.

The main goal of this application is to allow me to identify existing arbitrage opportunities within cryptocurrency markets.

## Definitions, Acronyms, and Abbreviations

CS Crypto Scanner

# User Requirements Definition

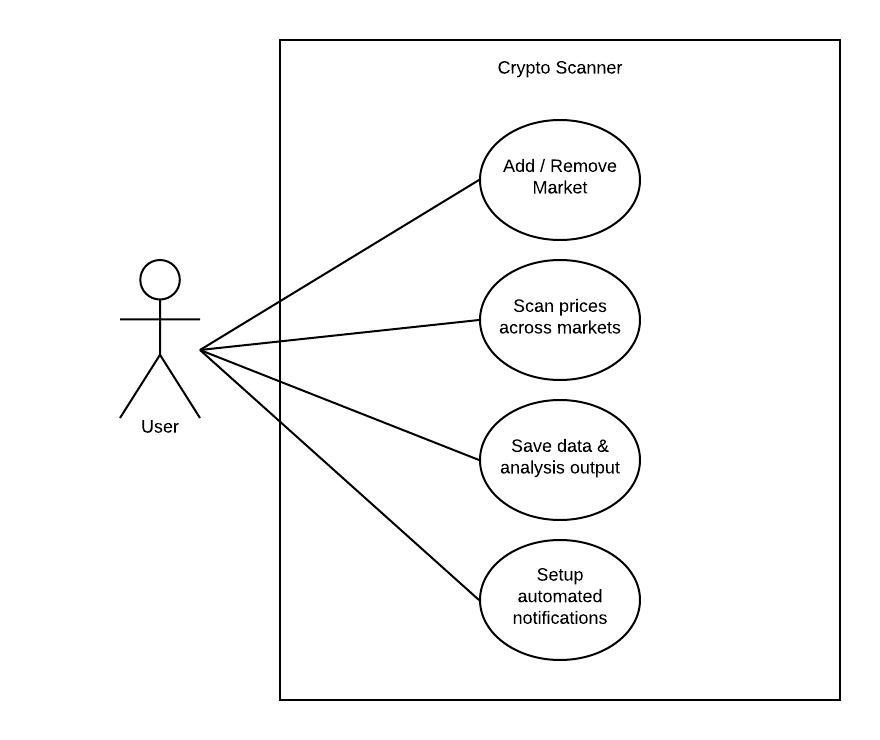
System should update data regularly & in real time. Further, data analysis should account for difference in market fees / miscellaneous costs, as well as ‘depth’ of potential price differences.

## Functional requirements

Application should:

* Allow user to add and remove markets that he/she is interested in scanning for data
* Query for data, analyse it, and display results for the user
* Be able to able to save both market and output data
* Allow user to set up automatic price scans and email notifications with the results

### Use Case Diagram



### Requirement 1 “Add/Remove Market”

#### Description & Priority

User can add/remove specific cryptocurrency market

#### Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

**Scope**

The scope of this use case is to add/remove market to/from the ‘tracked list’

**Description**

This use case describes the process of adding / removing a market

**Precondition**

The API has been set up for the specific market

User is on the ‘Markets’ page

**Activation**

This use case starts when a User clicks add/remove button

**Main flow**

1. The User clicks the add button near the market name
2. The System tests the API by sending a query to the market
3. The System receives success response from the market
4. The System adds the market to the main page list

**Exception flow**

A2 : The System tests the API by sending a query to the market

1. The System receives no response from market or receives error response
2. The System displays appropriate error message to the User

**Termination**

1. Based on the outcome, The System displays appropriate message for the User

**Post condition**

The system goes into a wait state

### Requirement 2 “Scan Prices across markets”

#### Description & Priority

The user initiates scan, and receives analysis of potential arbitrage opportunities

#### Use Case

**Scope**

The scope of this use case is to allow user to start the price scan

**Description**

This use case describes the process of configuring and running a price scan

**Precondition**

There are at least two markets added to the markets list

All required scan options have been selected

**Activation**

This use case starts when a User specifies which cryptocurrencies to include

**Main flow**

1. The User selects cryptocurrency/ies to be tracked
2. The User specifies minimum arbitrage threshold (expressed in terms of %)
3. The User specifies minimum arbitrage value threshold (expressed in US dollars)
4. The User hits the ‘Scan’ button
5. The System checks the scan options
6. The System gets the price data from listed markets
7. The System collects all data into one file and saves it
8. The System analyses the data using file created in step 7

**Alternate flow**

A5.1 : The User didn’t select any currencies to be tracked

1. The System uses default option ‘all’
2. The use case continues to position 6

A5.2 : The User didn’t select any arbitrage threshold

1. The System uses default option ‘5%
2. The use case continues to position 6

A5.3 : The User didn’t select min arbitrage value threshold

1. The System uses default option ‘10’
2. The use case continues to position 6

A6 : The System didn’t obtain data from one of the markets

1. The System displays the appropriate message to the user.
2. If the System obtained data from minimum 2 markets, the use case continues to position 7

**Termination**

1. The System displays all the arbitrage opportunities that have met the minimum criteria

**Post condition**

The system goes into a wait state

### Requirement 3 “Save analysis output”

#### Description & Priority

This is an optional feature that doesn’t affect main functionalities

#### Use Case

**Scope**

The scope of this use case is to save output obtained from analysis of data

**Description**

This use case describes the process of saving the data

**Flow Description**

**Precondition**

The System has completed a Scan

**Activation**

This use case starts when a User clicks the ‘Save’ button

**Main flow**

1. The User hits the ‘Save’ button
2. The System asks the User to specify name and location of the file to be saved
3. The System saves the output of the last scan to the file

**Exceptional flow**

E3 : The System fails to save the file

1. The system displays an error message to the User

**Termination**

The system displays message to the User

**Post condition**

The system goes into a wait state

### Requirement 4 “Set automated Scan and Notifications”

#### Description & Priority

This is an optional feature that is not essential to the main functionality of the application. It allows user to set times for scans to run automatically, and to send the output via email.

#### Use Case

**Scope**

The scope of this use case is to set times at which scan will be executed and email address to which output will be sent to

**Precondition**

User is on the ‘Automation’ page

**Activation**

This use case starts when a User clicks ‘Schedule Scan’ button

**Main flow**

1. The User clicks the ‘Schedule Scan’ button
2. The System generates 3 input fields: ‘Time’, ‘Repeat’, ‘Send to’
3. The User fills out all fields
4. The User clicks ‘Add Schedule’ button
5. The System validates the fields
6. The System adds a schedule

**Alternate flow**

A5.1 : The date/time field has invalid input ( out of range/ date in past)

1. The System highlight ‘Time’ field and displays message
2. The use case moves to position 3

A5.2: The User didn’t fill out ‘Repeat’ field

1. The System uses default option ‘No repeat’
2. The use case continues to position 6

A5.3 : The User didn’t fill out ‘Send to’ field

1. The System uses default option ‘don’t email’
2. The use case continues to position 6

**Termination**

1. The System displays message that the schedule has been created

**Post condition**

The system goes into a wait state

## Non-Functional Requirements

### Performance/Response time requirement

Scan shouldn’t take more than 1 minute, inclusive of obtaining of data.

### Availability requirement

Application is desktop based, therefore it is available as long as user has access to it

### Recover requirement

No database is planned for this application

### Robustness requirement

The System has to be able to handle errors without system closing.

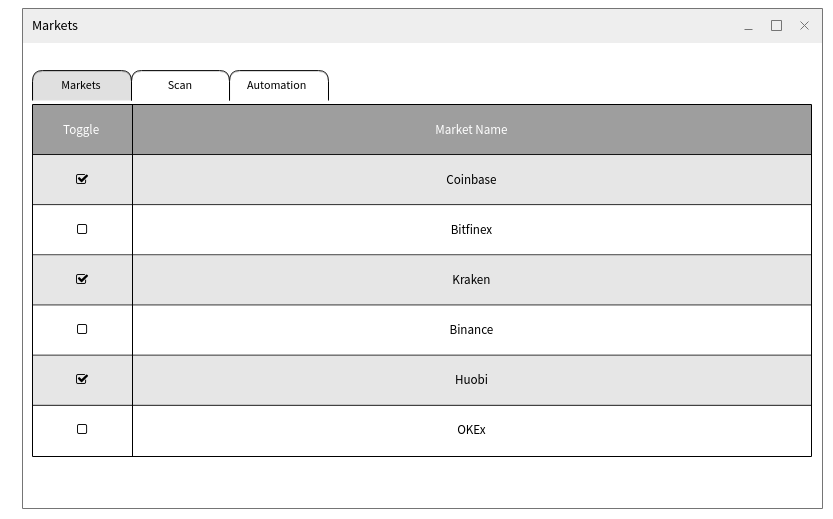
### Security requirement

Because the application is desktop based and no private/ confidential data is required, security is minimal

# GUI

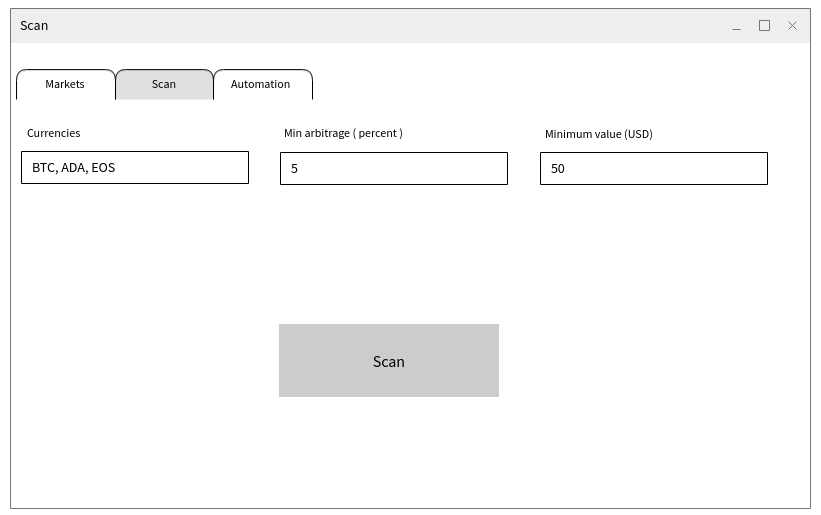
**Markets page**

User can toggle on / off markets to be scanned. Markets and their relevant API queries are preset for the user.



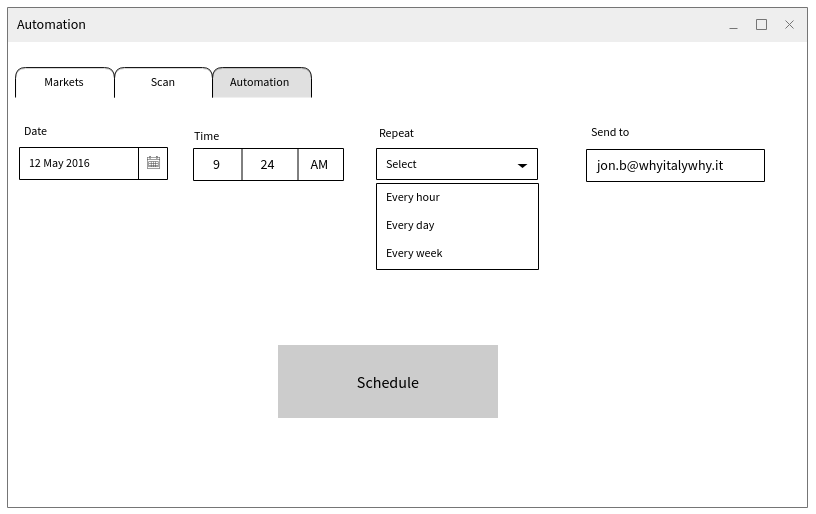
**Scan page**

User can specify which crypto currencies are to be included in the analysis, as well as what are the minimal thresholds in absolute and relative terms.



**Automation page**

User can schedule Scans to run at specific day & time. Further, loop option is provided. Finally, User can send output of the Scan to his/her email.



# System Architecture

# C:\Users\a589799\Downloads\CS_ClassDiagram.jpeg

# System Evolution

Once the application is finished, I’m planning to create a bot that will be able to take action on the results of the scan. For example, if all pre specified conditions are met, and opportunity for beneficial transaction has been identified, bot will take action on this information and will execute a transaction (provided it has sufficient funds etc).