**Requirements specification**

***Automatically add missing data***

*8vance Matching Technologies BV*

*Venlo*

|  |
| --- |
| **Date : 25-02-2016** |
| **Version : 0.1** |
| **Status : Concept** |
| **Document name : Data exploration v0\_1** |
| **Author : Tim Hermens** |

**Version**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Changes** | **Status** |
| 0.1 | 25-02-2016 | Tim Hermens | First concept. | Concept |
|  |  |  |  |  |

**Distribution**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Recipient** |
|  |  |  |

**Approval**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date of approval** | **Name** | **Signature** |
|  |  |  |  |

**Table of contents**

1. Introduction 4

1.1 Introduction 4

1.2 Goal of this document 4

2. Requirements regarding the server-side algorithm functionality 5

3. Requirements regarding the stand-alone application 6

# Introduction

## Introduction

The company 8vance Matching Technologies BV wants to have two products:

* Product B1: Server-side algorithm functionality. An algorithm that supplements the missing KSC data for profiles that can be integrated into the server as a piece of Python code.
* Product B2: Stand-alone application. A stand-alone application to test various algorithms on correct input and output, and performance.

Whenever profiles are mentioned within this document, it refers *only* to the LinkedIn profiles.

Product B1 is just a piece of functionality, meaning it won't have any interaction with users. This product needs to be integrated into the server but should also be usable for analysis in product B2.

There're three users for product B2:

* Data scientists (algorithm developers). They'll want to use the product to test the algorithm on correct input and output, and performance. If the analysis overall negative, it's a sign the algorithm needs to be improved.
* Data analysts. They'll want to use the product to look for correct and incorrect predictions of the algorithm to find possible flaws they need to fix in their models.
* Other users. The previous two users are the main users of the product. There're also other users who don't want to use the product with a problem-solution mindset, but who're just interested what predictions the algorithm comes up with and how well it performs.

## Goal of this document

All of the requirements of these two products are discussed within this document. Every requirement in this document have a certain urgency and importance factor.

The urgency factor indicates how early the requirement needs to be implemented.

The importance factor indicates which requirements have higher priority than others. Requirements with a high priority indication need to be implemented no matter what, as soon as possible (this doesn't mean they have to be rushed!). Requirements with a medium priority also need to be implemented, but not necessarily straight away. Requirements with a low priority are things that would be nice to have (like an enhancement of the user experience/interface, additional features, etc.). This roughly translates to the MoSCoW notation as follows: high priority = must have, medium priority = should have, low priority = could have.

# Requirements regarding the server-side algorithm functionality

The following table contains the requirements for product A.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Requirement description** | **Urgency** | | | **Importance** | | |
| **High** | **Medium** | **Low** | **High** | **Medium** | **Low** |
|  | The algorithm is created in Python. |  |  |  |  |  |  |
|  | The algorithm can be integrated in the server's existing Django framework. |  |  |  |  |  |  |
|  | The input data type for the algorithm must have a JSON format. |  |  |  |  |  |  |
|  | The input data must contain all the necessary data fields (specified in the research document) of the profiles. |  |  |  |  |  |  |
|  | The algorithm can parse the profile data of a selection of profiles to the wanted syntax. |  |  |  |  |  |  |
|  | The algorithm can write the parsed profile data of a selection of profiles to a external target (database/file). |  |  |  |  |  |  |
|  | The algorithm can predict missing skills for a selection of profiles. |  |  |  |  |  |  |
|  | The algorithm can write the predicted missing skills of a selection of profiles to a external target (database/file). |  |  |  |  |  |  |
|  | The algorithm can calculate a certainty score for every predicted skill in a profile. |  |  |  |  |  |  |
|  | The algorithm can calculate a certainty score for every skill for a profile. |  |  |  |  |  |  |
|  | The algorithm can determine whether or not user-specified skills are correct. |  |  |  |  |  |  |
|  | The algorithm can flag and exclude the incorrect user-specified skills. |  |  |  |  |  |  |
|  | The algorithm can write the flagged incorrect user-specified skills to a external target (database/file). |  |  |  |  |  |  |
|  | The algorithm can calculate an certainty score for every determination of whether or not a user-specified skill is correct or incorrect. |  |  |  |  |  |  |
|  | The algorithm can predict missing skills for every profile in 1 second or less. |  |  |  |  |  |  |

# Requirements regarding the stand-alone application

The following table contains the requirements for product B.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Requirement description** | **Urgency** | | | **Importance** | | |
| **High** | **Medium** | **Low** | **High** | **Medium** | **Low** |
|  | The user can import an algorithm he wants to analyze. |  |  |  |  |  |  |
|  | The user can select/specify a data source for the algorithm. (it's important that the structure of the data source is the same as the one used to create the algorithm) |  |  |  |  |  |  |
|  | The application runs locally (without any network connection). |  |  |  |  |  |  |
|  | The user can select/specify a dump target which is used to dump algorithm-specific data (for instance, for pre-processing). |  |  |  |  |  |  |
|  | The user can save the algorithm analysis results. |  |  |  |  |  |  |
|  | The user can load and view previous algorithm analysis results. |  |  |  |  |  |  |
|  | If an algorithm is selected, a data source is specified and no analysis is running (see b8), the user can run an analysis of the algorithm. |  |  |  |  |  |  |
|  | The user can kill the analysis process at any given time. |  |  |  |  |  |  |
|  | The user can see the analysis' progress. |  |  |  |  |  |  |
|  | The user can run one analysis at a time. (heavy cpu load) |  |  |  |  |  |  |
|  | The application should be horizontally scalable. (GUI tab control so more viewpoints of the analysis can be supported?) |  |  |  |  |  |  |
|  | The result of the analysis shows the execution time of the algorithm. (ticking clock when analysis starts and ends?) |  |  |  |  |  |  |
|  | The result of the analysis shows the cpu cost per line in the algorithm. |  |  |  |  |  |  |
|  | The result of the analysis shows the memory cost per line in the algorithm. |  |  |  |  |  |  |
|  | The result of the analysis shows the predictions. (predicted skills per profile, in a simple table) |  |  |  |  |  |  |
|  | The result of the analysis shows the certainty score per prediction. (predicted skill per profile, all skills per profile, certainty bar graph per profile?, overall certainty bar graph?, can be sorted) |  |  |  |  |  |  |
|  | The result of the analysis shows information about the algorithm's code structure (lines of code, comments, statements, classes, cyclomatic complexity) |  |  |  |  |  |  |
|  | When the analysis is done, the user can request the analysis results for a particular profile. |  |  |  |  |  |  |
|  | When viewing the analysis results of a particular profile, the user can search for a particular prediction (predicted skill / owned skill). |  |  |  |  |  |  |
|  | When an analysis is running, the user can't do any actions apart from killing the analysis process. (to save cpu load) |  |  |  |  |  |  |