TECHNICAL UNIVERSITY – SOFIA
PLOVDIV BRANCH
FACULTY OF ELECTRONICS AND AUTOMATION

SOFTWARE SYSTEM FOR AUTOMATIC RECRUITMENT OF CANDIDATES ACCORDING TO THEIR QUALIFICATIONS

Lyubomir Lambrev , fac. Num. 510259
Email: thelubo1@abv.bg

TABLE OF CONTENTS

1.1. Motivation, Main Goal, Main Tasks

Motivation:

- The motivation behind creating such a system is the labor shortage. The labor shortage could significantly impede the world's economy and the ability to recover from it.
- So our system should help companies efficiently recruit better applicants, and significantly reducing the work load .

Main Goal:

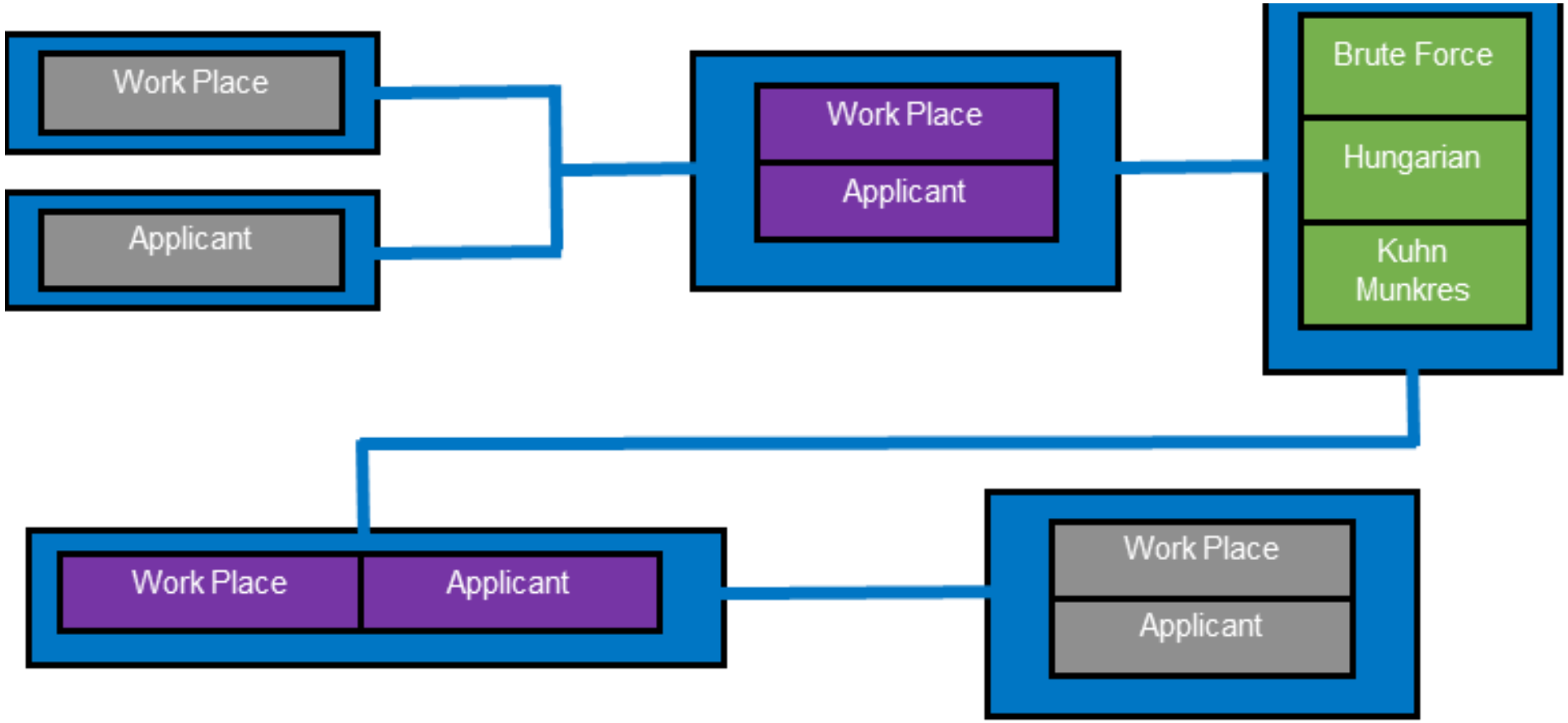
- The main goals of this system is to find the best arrangement for each applicant and the corresponding workplaces, bearing in mind the maximum efficiency.

Main Tasks:

- Load the data
- Find the optimal allocation
- Save and present the results
- Find the optimal algorithm

2.1. Conceptual model:

It consists of concepts used to help understand or simulate the process of the system.

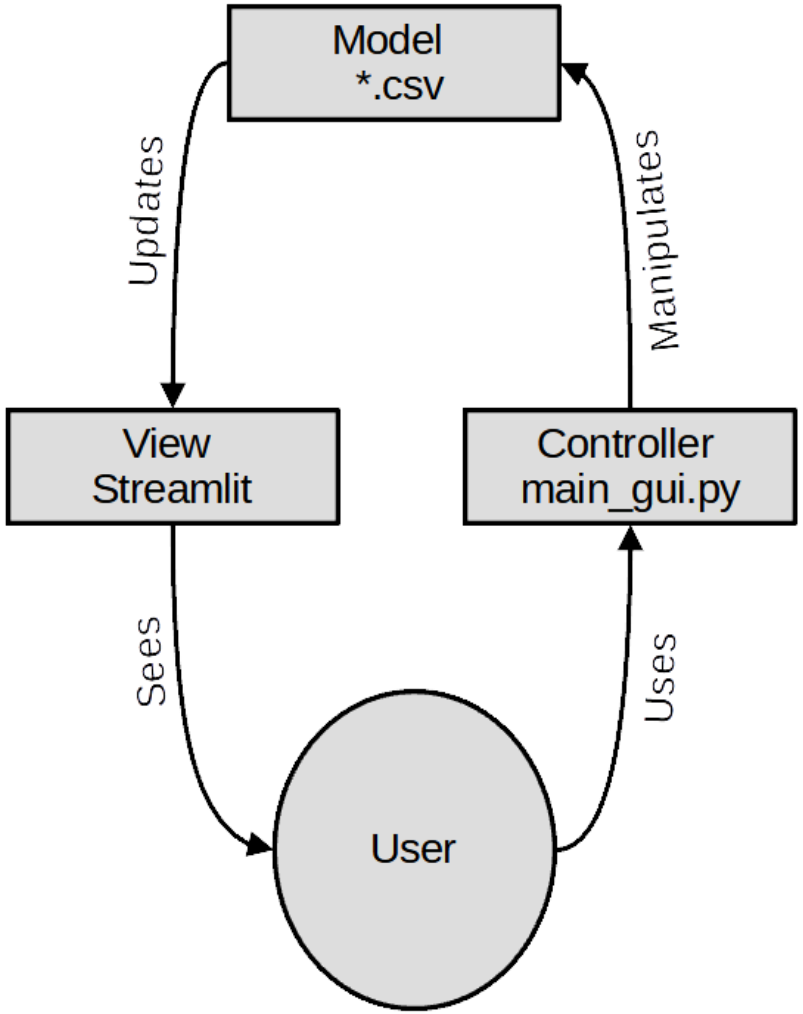


2.2. System Architecture

We use MVC as a system architecture which means that the code will be split into 3 part:

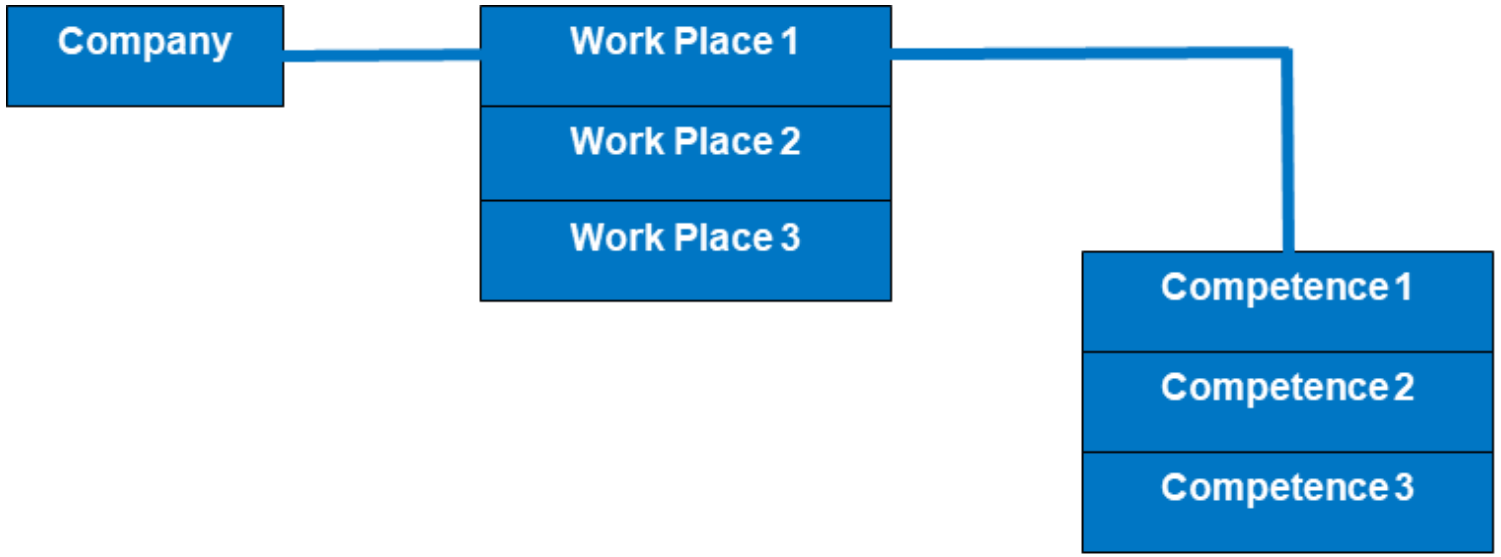
- Model- Works with data that is written/read in CSV files.
- View- the command line is used to enter arguments via Streamlit.
- Controller- Implements the algorithms.

2.2. Model–View–Controller (MVC) in our system:



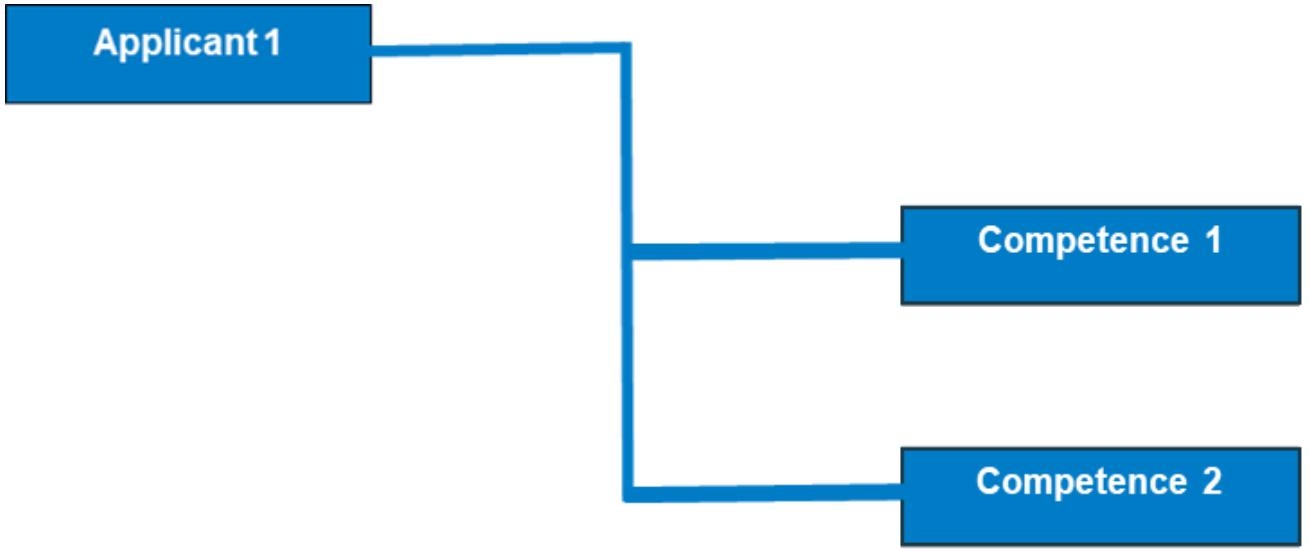
2.3. Data Structures

The dataset of the jobs is provided by the employer and the respective competence required for the job. We load the data from *CSV file as Pandas data frame and convert it to Python dictionary. We will consider evaluating each jobs requirement (competence) as the number of years required experience.



2.3. Data Structures

The dataset of the applicants is provided by the employment agency with the respective competence possessed by the applicants. We load the data from *CSV file as Pandas data frame and convert it to Python dictionary. We will consider evaluating each applicant's qualification (competence) as the number of years of experience of the applicant.



2.4. Methods and Algorithms

2.4.1 Naive Approach

- Fitting function - Determines to what extent a give competence is covered by the applicant. The fitting function brings back a numerical value (metric, grade). For that reason, there are 3 outcomes:

I. When the competence is required by the employer and possessed by the applicant (we return the number of years of the applicant).

II. When the competence is required by the employer, but the applicant does not have such a qualification, we return 0.

III. When the applicant has a qualification that is not required at the work place, we return 0.

2.4. Methods and Algorithms

2.4.1 Naive Approach

- Utility function- This function returns the sum of all fitting values of the ordered pair (Applicant,Workplace). Which is then stored in Utility_value

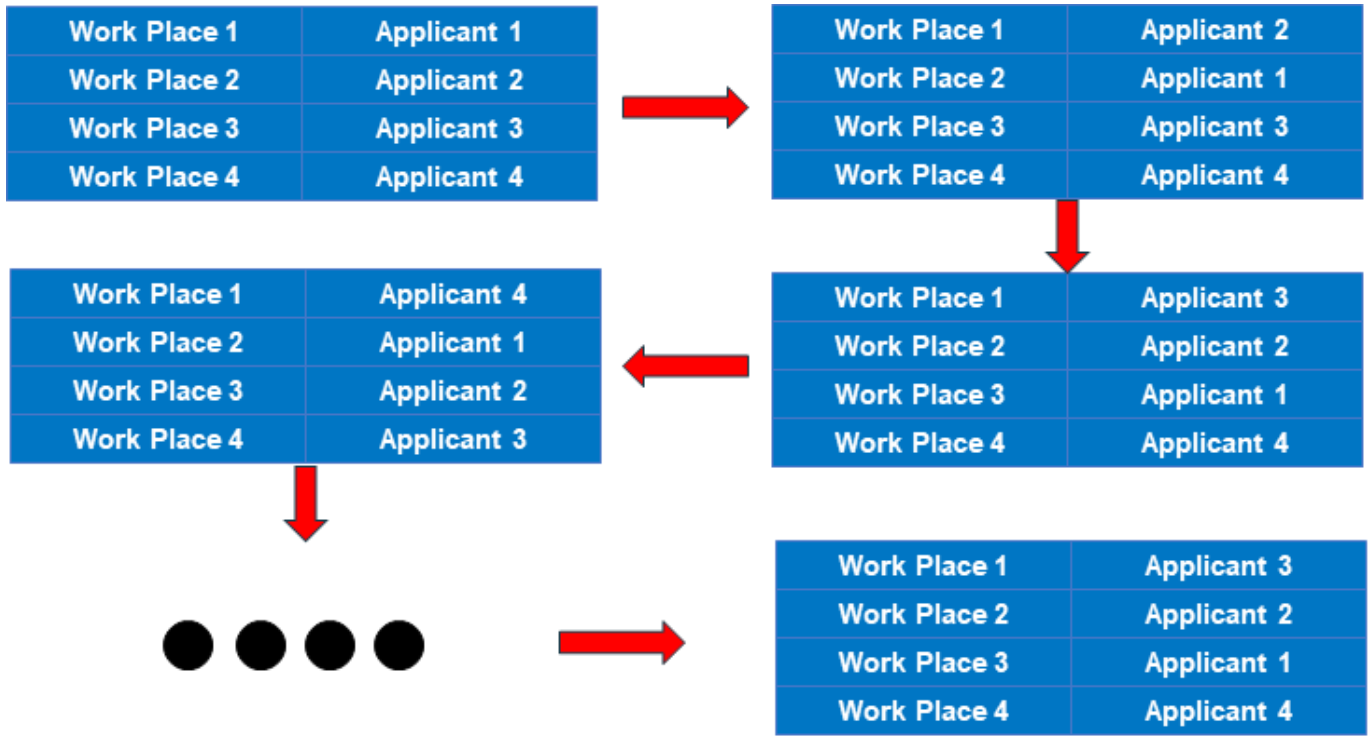
  After all the functions are created we then create our algorithm of full enumeration . Which is as follows:

  1. We load the list of workplaces
  2. We load the list of applicants
  3. We create a list of all possible permutations
  4. We calculate the utility_value for each permutation in this list
  5. We find the max utility_value from the list

2.4. Methods and Algorithms

2.4.1 Naive Approach

- The Naïve Approach solution is the simplest method to calculate the total value for every possible combination and then select the most efficient one.



2.4.2. Hungarian Algorithm

Fig. 1.

7	3	6	9	5	9
7	5	7	5	6	7
7	6	8	8	9	9
3	1	6	5	7	7
2	4	9	9	5	9

Fig. 1.1.

2	6	3	0	4
0	2	0	2	1
2	3	1	1	0
4	6	1	2	0
7	5	0	0	4

Fig. 1.2.

2	6	3	**0**	4
0	2	0	2	1
2	3	1	1	**0**
4	6	1	2	0
7	5	**0**	0	4

Fig. 1.3.

2	6	3	0	4
0	2	0	2	1
2	3	1	1	0
4	6	1	2	0
7	5	0	0	4
0	**2**	**0**	**0**	**0**

Fig. 1.4.

2	4	3	0	4
0	0	0	2	1
2	1	1	1	0
4	4	1	2	0
7	3	0	0	4

1

Fig. 1.5.

2	4	3	**0**	4
0	0	0	2	1
2	1	1	1	**0**
4	4	1	2	0
7	3	**0**	0	4

2

Fig. 1.6.

2	4	3	**0**	4
0	**0**	0	2	1
2	1	1	1	0
4	4	1	2	**0**
7	3	**0**	0	4

Fig. 1.7.

2	4	3	0	4
~~0~~	~~0~~	~~0~~	~~2~~	~~1~~
2	1	1	1	0
4	4	1	2	0
7	3	0	0	4

Fig. 1.8.

1	3	3	0	4
0	0	1	3	2
1	0	1	1	0
3	3	1	2	0
6	2	0	0	4

Fig. 1.9.

1	3	3	**0**	4
0	0	1	3	2
1	**0**	1	1	0
3	3	1	2	**0**
6	2	**0**	0	4

2.4.2. Hungarian Algorithm

	Communication	Learning	Problem solving
Lead Developer	4	4	4
Manager	6	5	6
Secretary	2	2	0

	Communication	Learning	Problem solving
Josh	8	5	7
Maria	3	2	0
Michael	4	4	5

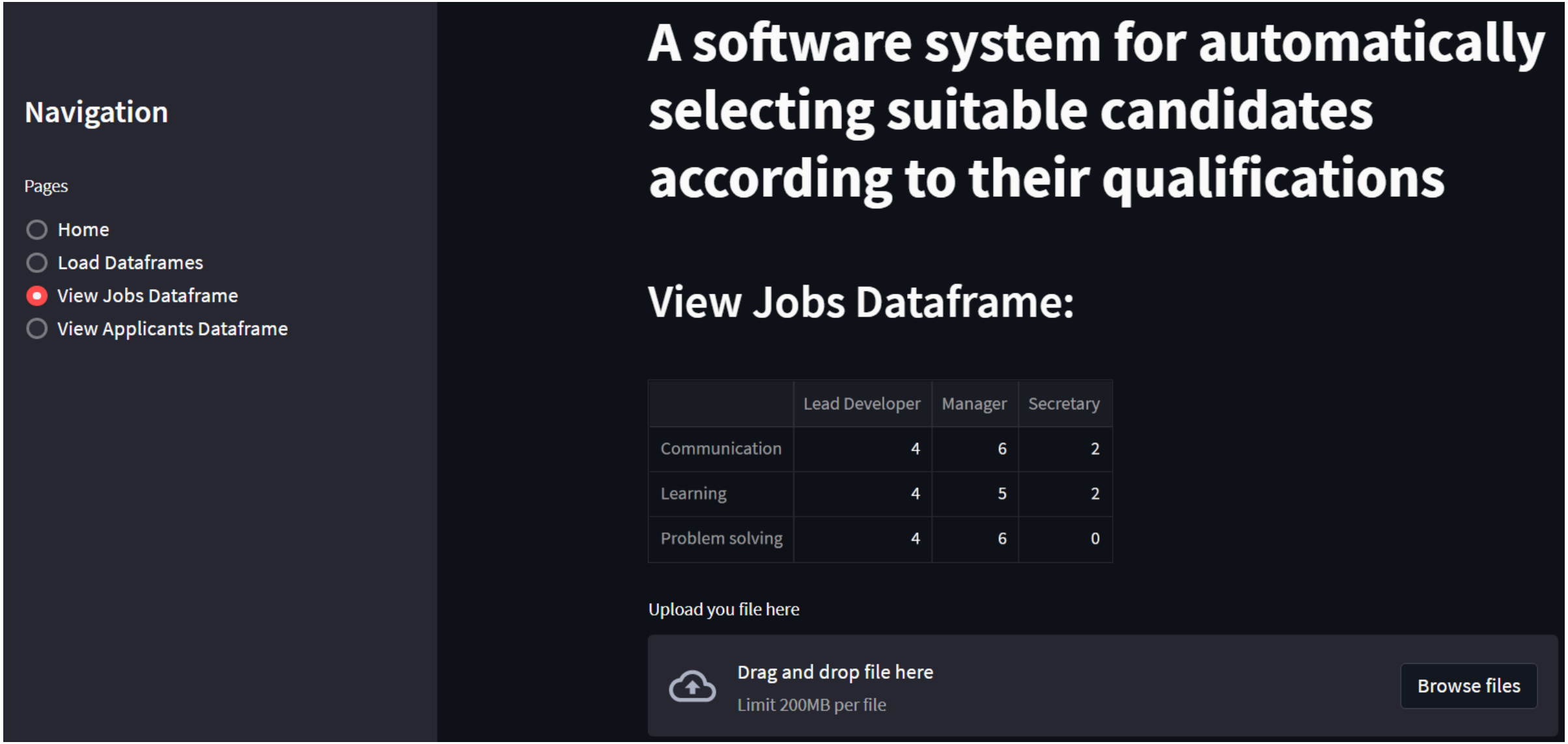
	Lead Developer	Manager	Secretary
Josh	80	115	26
Maria	20	28	10
Michael	52	74	16

2.5. Software tools

- **Streamlit** - is a free and open-source framework which builds web apps.
- **Python and Anaconda** - Python is an interpreted, object-oriented, high-level programming language. Anaconda is a distribution of the Python programming language.
- **Spyder** - is a free and open source scientific environment for Python.
- **Pandas** - Pandas is a fast, flexible and easy to use open source data analysis and manipulation tool.
- **Numpy** - is a Python library that provides a multidimensional array object, various derived objects, etc. It is a fundamental package for computing.
- **Argparse** - is module which makes it easy to write user-friendly command-line interfaces.
- **CSV** - stands for comma-separated values, which is a delimited text file. In Bulgaria we use as delimiter semi - column instead of comma.

2.5. Software tools

- Streamlit screenshots



3.1. Experiments

Two experiments are performed:

In the first experiment, the performance of the three implemented algorithms is compared: Naive Approach,Munkres and LSA. The input data set consists of a list of app_serie and job_serie dictionaries. The first item in the list includes dictionaries with three candidates and three jobs. The size of the dictionaries grows in steps of 2. Thus, the list of job sizes has the following form:

```
[3,5,7,9,11]
```

- In the second experiment, Munkres and LSA are compared. As the size of the items in the dictionary list start in the range 3 to 200 with a step of 20.

3.1. Results

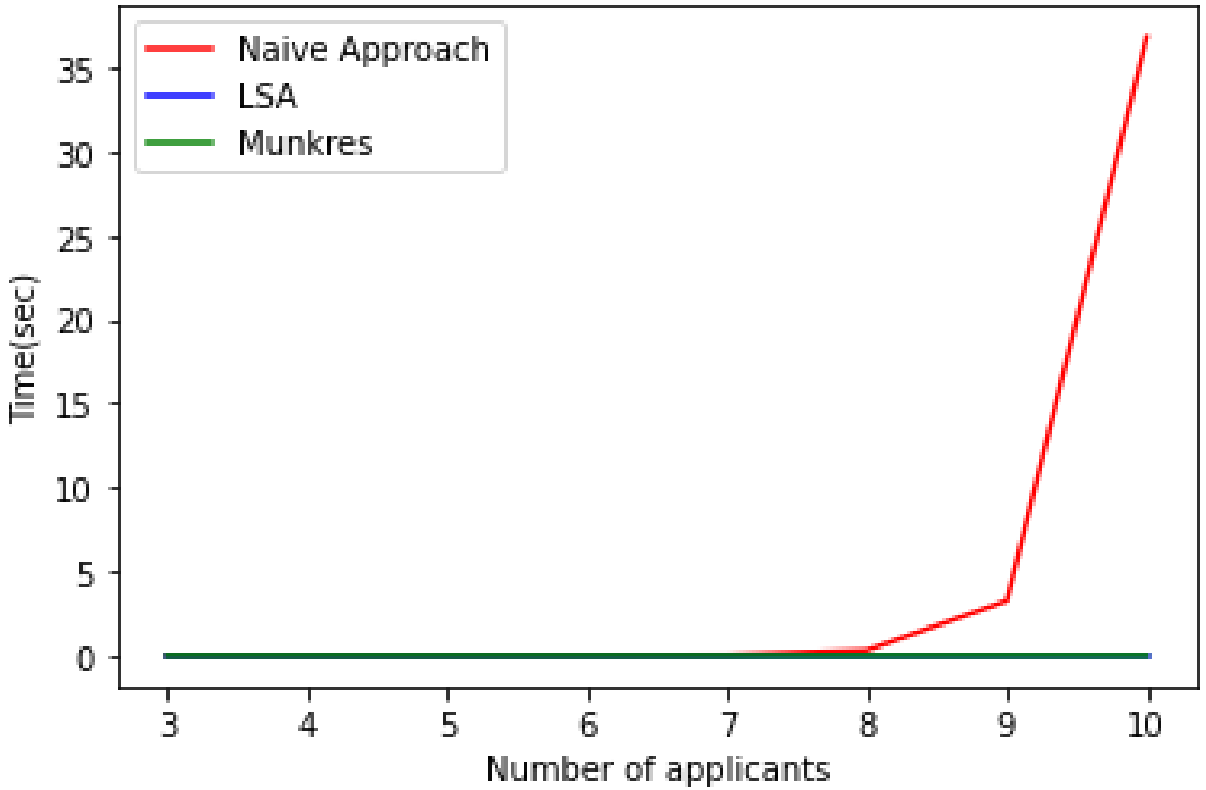


Fig. 2.

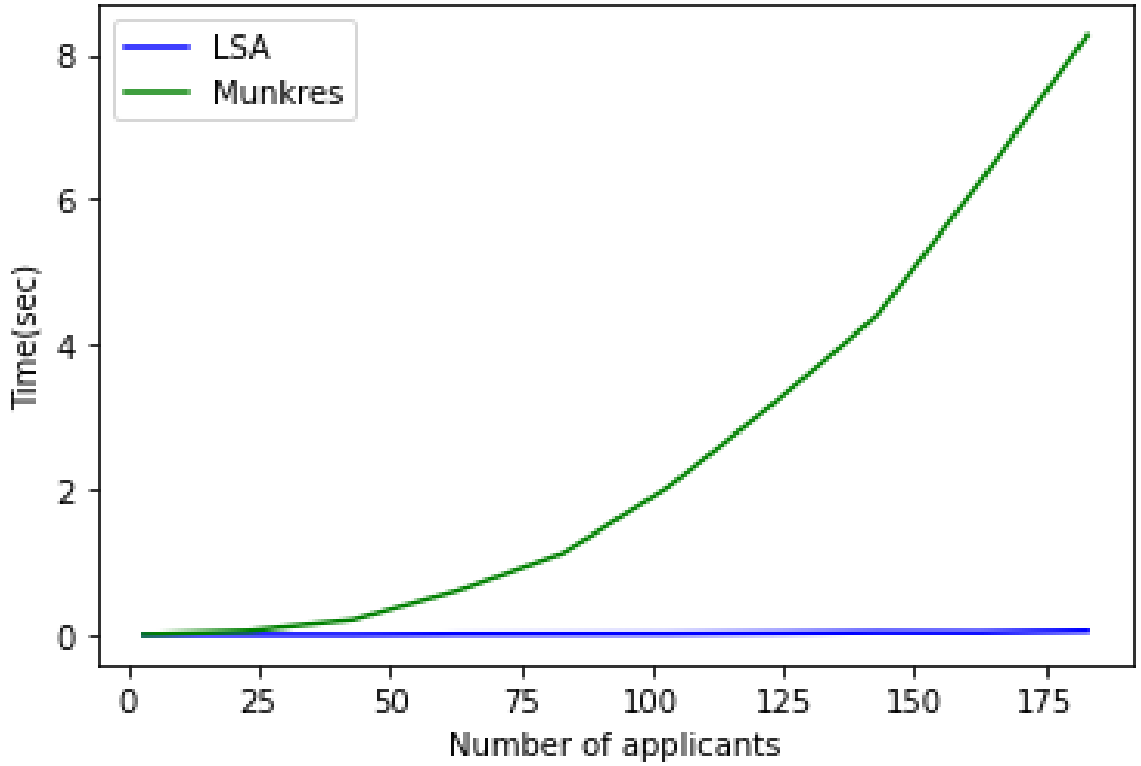


Fig. 2.1.

3.2. Conclusion and future work

- From the experimental part it follows that the naive approach can only be used in very small volumes of data. For working with data in practice, one of the existing implementations of the Hungarian algorithm should be used. We propose the LSA implementation from scipy library.

- Another aspect of future expansion of the functionality of such systems would be to replace csv files for data storage with a Database Management System. The architecture of the system allows this to be done easily.

Thank you for the attention!