TECHNICAL UNIVERSITY – SOFIA

PLOVDIV BRANCH
FACULTY OF ELECTRONICS AND AUTOMATION

FINAL YEAR PROJECT

BACHELOR'S DEGREE

TITLE:

SOFTWARE SYSTEM FOR AUTOMATIC RECRUITMENT OF CANDIDATES ACCORDING TO THEIR QUALIFICATIONS

Lyubomir Lambrev , fac. Num. 510259
Email: thelubo1@abv.bg
Phone number: 0988372440

TABLE OF CONTENTS

Chapter 1

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.

Main Goal:

- The main goal behind this system is simple. It is to help companies efficiently recruit better applicants, and significantly reducing the work load .

Main Tasks:

- The main tasks of this system are to help find the best arrangement for each applicant and the corresponding workplace for the maximum efficiency.

Chapter 2

2.1. Basic theory , Conceptual model

Basic theory:

The Model–View–Controller (MVC) is a software architectural pattern commonly used for developing user interfaces that divide the related program logic into three interconnected elements. Which are as follows:

- Model- It directly manages the data, logic and rules of the application.
- View- It represents the information such as charts, diagrams or tables.
- Controller-It handles the user interactions. It interprets the inputs from the user.

Conceptual model:

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

2.1. Conceptual model:

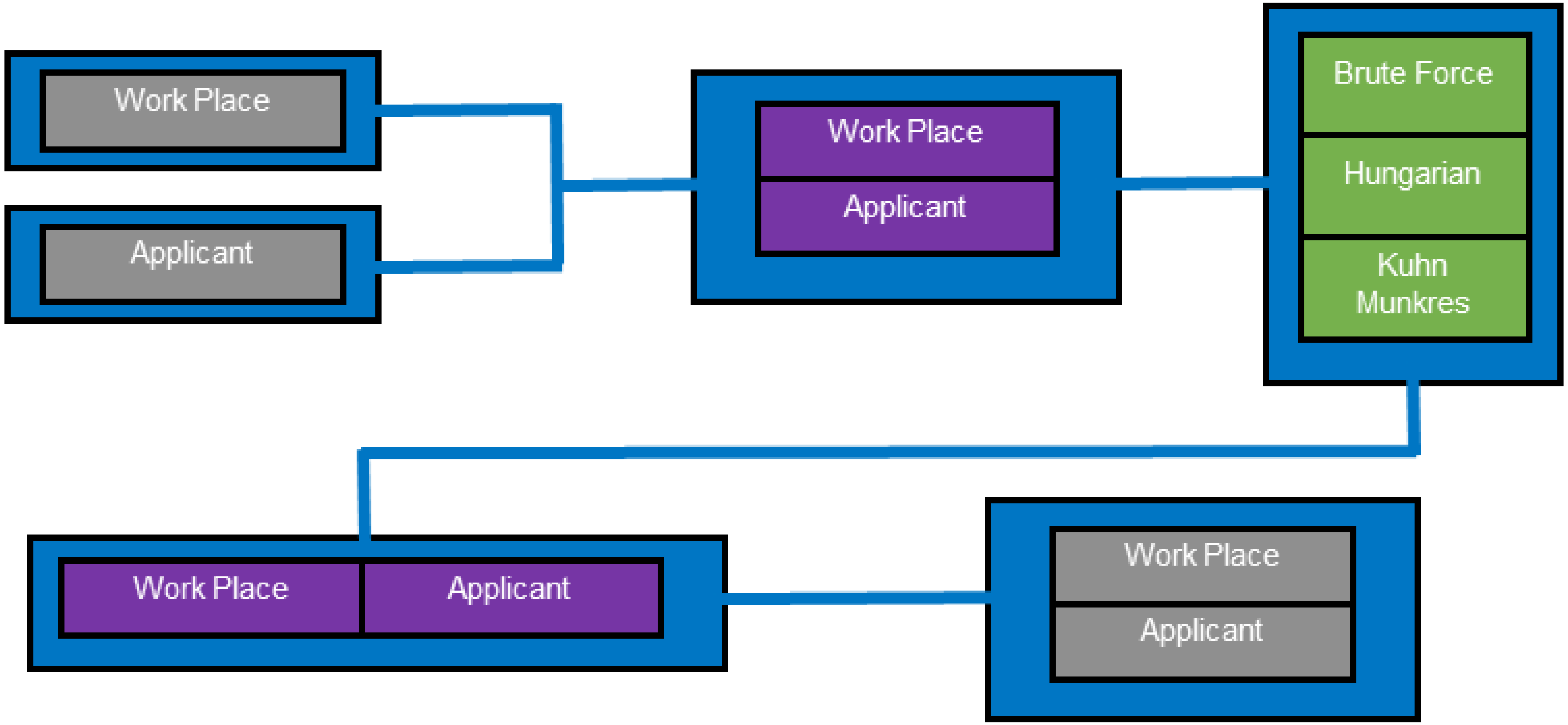
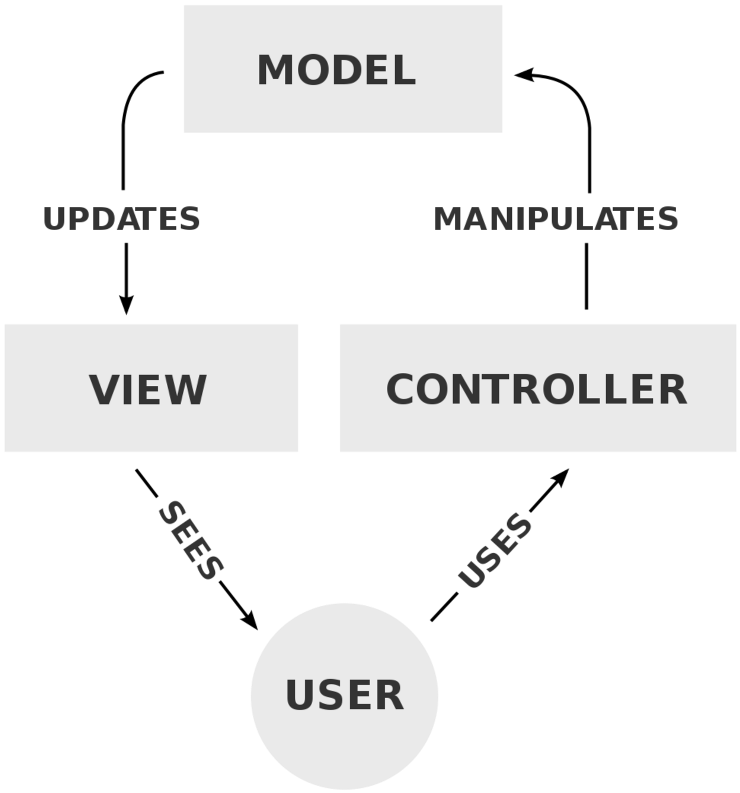


Fig. 5. Conceptual model

2.1. Model–View–Controller (MVC):

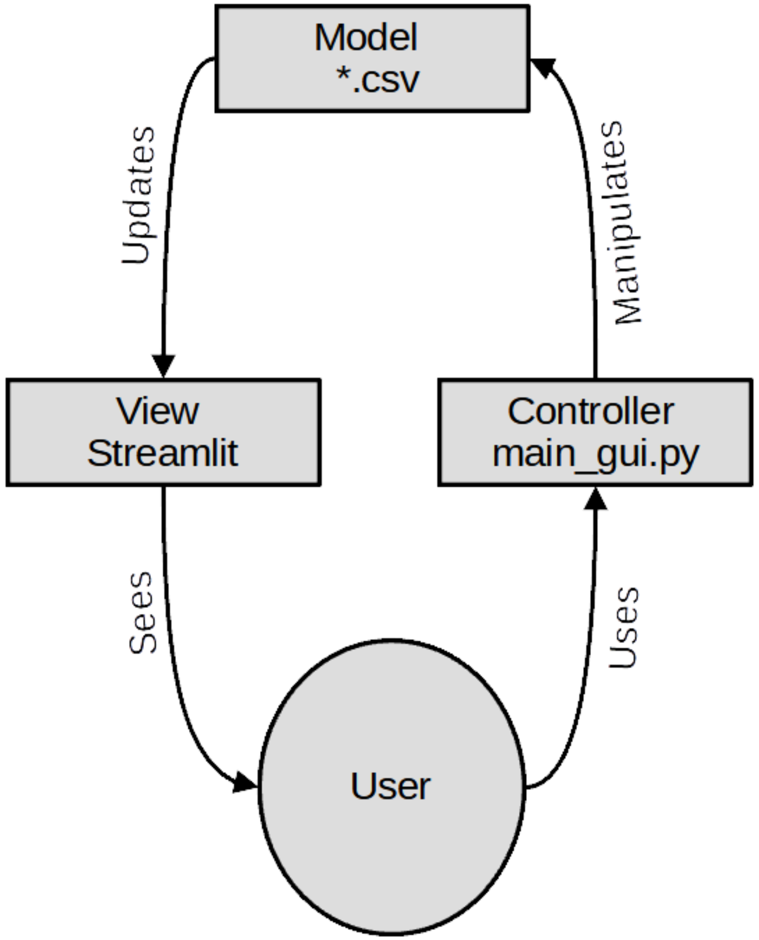


2.2. System Architecture

The system architecture is a MVC architectural pattern which means that the code will be split into 3 part:

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

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



2.3. Data Structures

The list of jobs is provided by the employer and the respective competence required for the job. We will consider evaluating each jobs requirement (competence) as the number of years required experience.

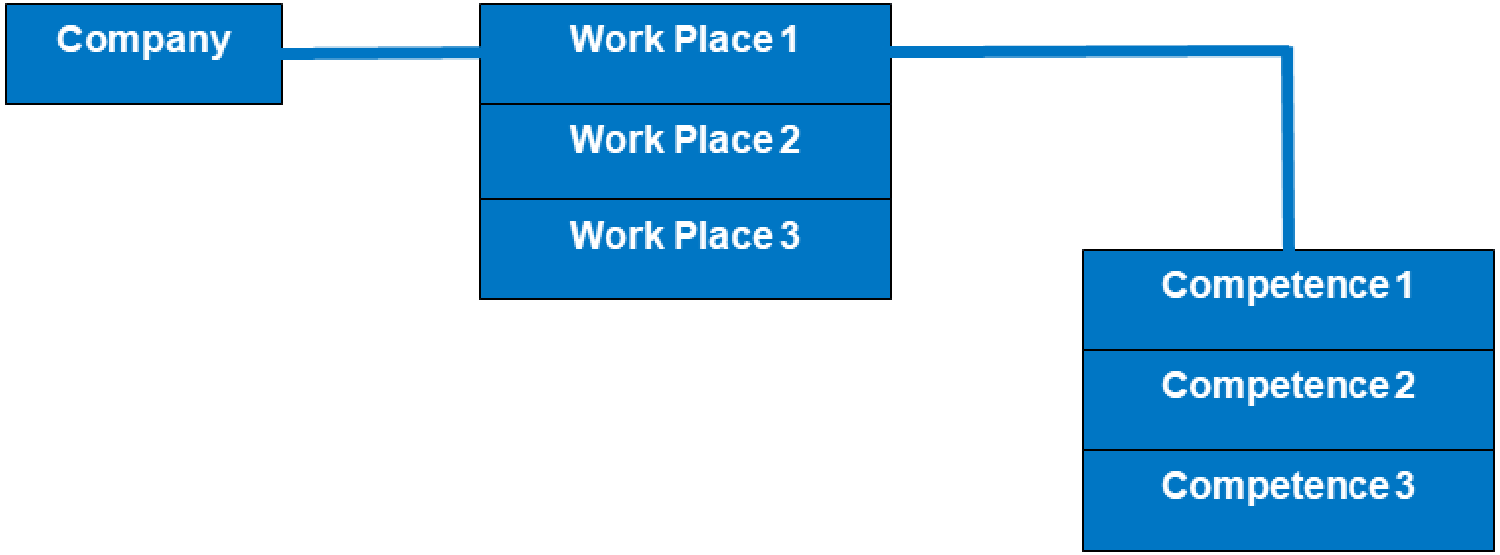


Fig. 1. Data structure represented by the employer

2.3. Data Structures

The list of applicants is provided by the employment agency with the respective competence possessed by the applicants. We will consider evaluating each applicant's qualification (competence) as the number of years of experience of the applicant.

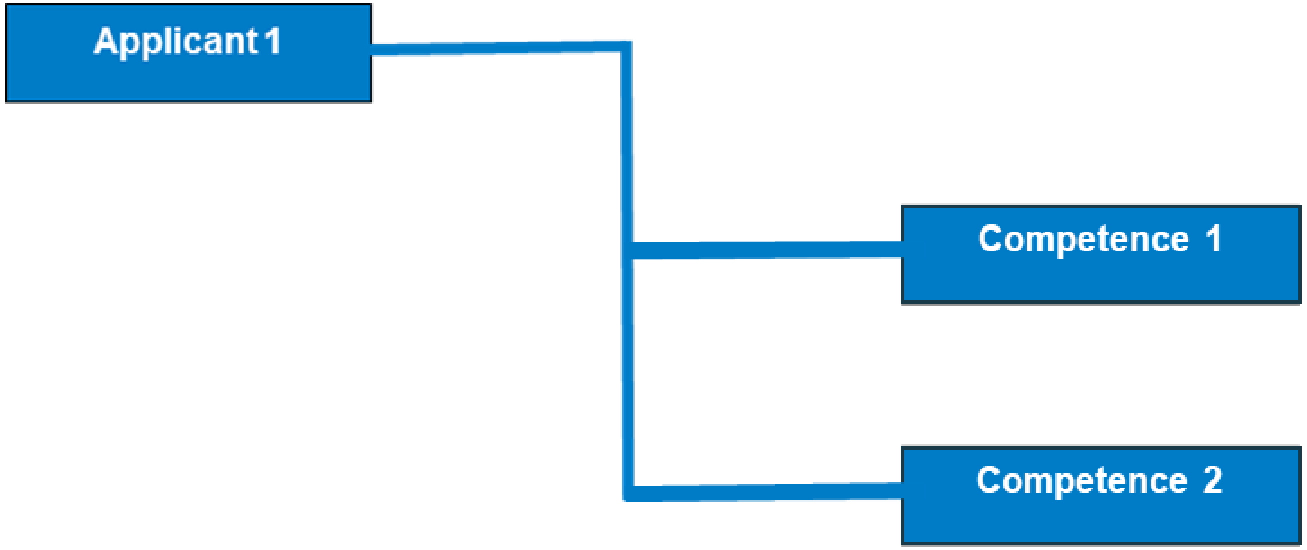


Fig. 2. Data structure represented by the applicant

2.4. Method and Algorithms

- 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. Method and Algorithms

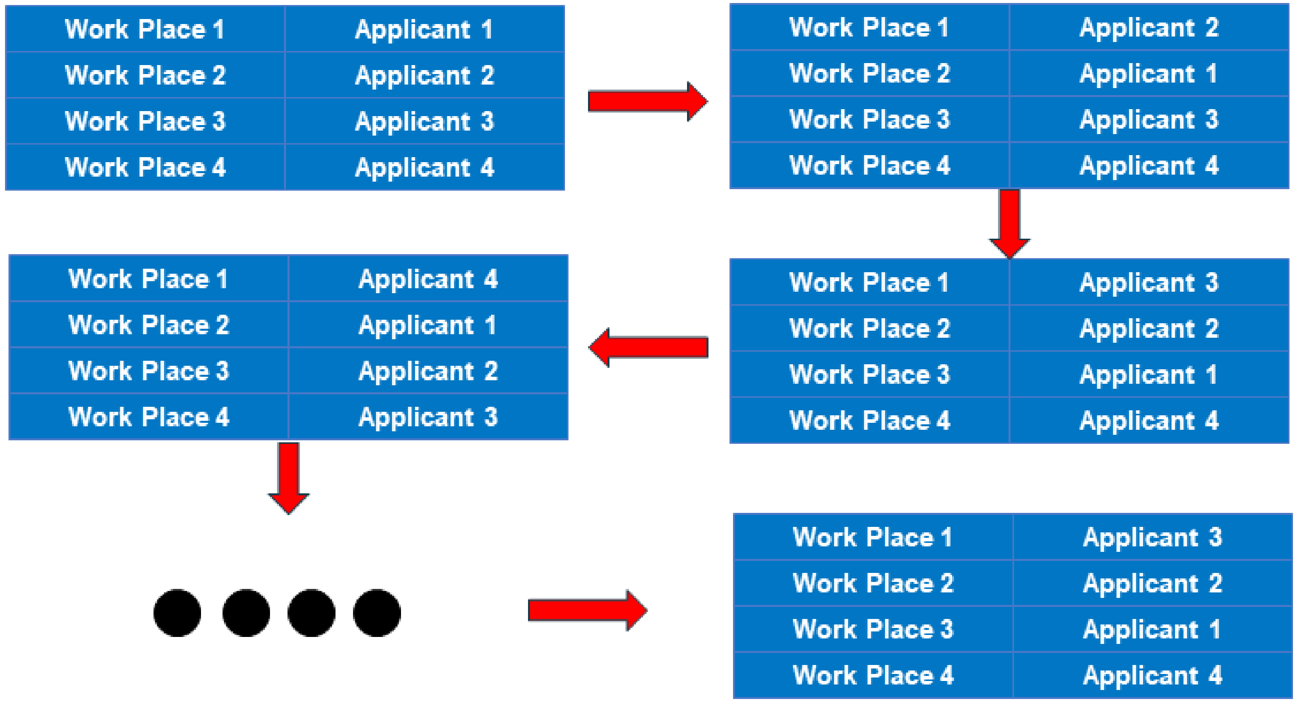
- Utility function- This function returns the sum of all fitting values of the ordered pair (Applicant,Workplace). Which is then stored in Utility_value

  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 ordinances
  4. We calculate the utility_value for each item in this list
  5. We find the max utility_value from the list

2.4.1. Naïve Approach

- The Naïve Approach solution is simply to calculate the total value for every possible combination and then select the most efficient one.
It should not be confused with backtracking,
where large sets of solutions can be discarded.



Work Place 1	Applicant 1
Work Place 2	Applicant 2
Work Place 3	Applicant 3
Work Place 4	Applicant 4

Work Place 1	Applicant 2
Work Place 2	Applicant 1
Work Place 3	Applicant 3
Work Place 4	Applicant 4

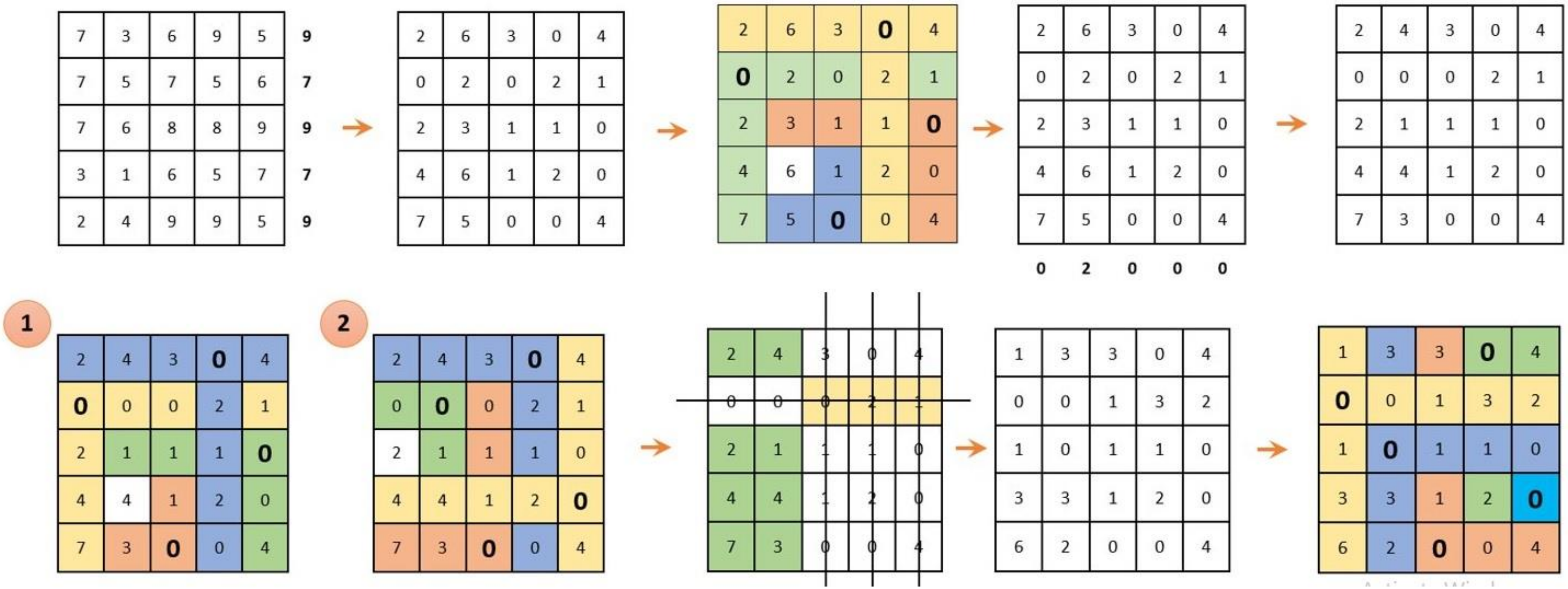
Work Place 1	Applicant 3
Work Place 2	Applicant 2
Work Place 3	Applicant 1
Work Place 4	Applicant 4

Work Place 1	Applicant 4
Work Place 2	Applicant 1
Work Place 3	Applicant 2
Work Place 4	Applicant 3

Work Place 1	Applicant 3
Work Place 2	Applicant 2
Work Place 3	Applicant 1
Work Place 4	Applicant 4

Fig. 4. List of all possible Ordinance

2.4.2. Hungarian Algorithm



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**

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

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

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**

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

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

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

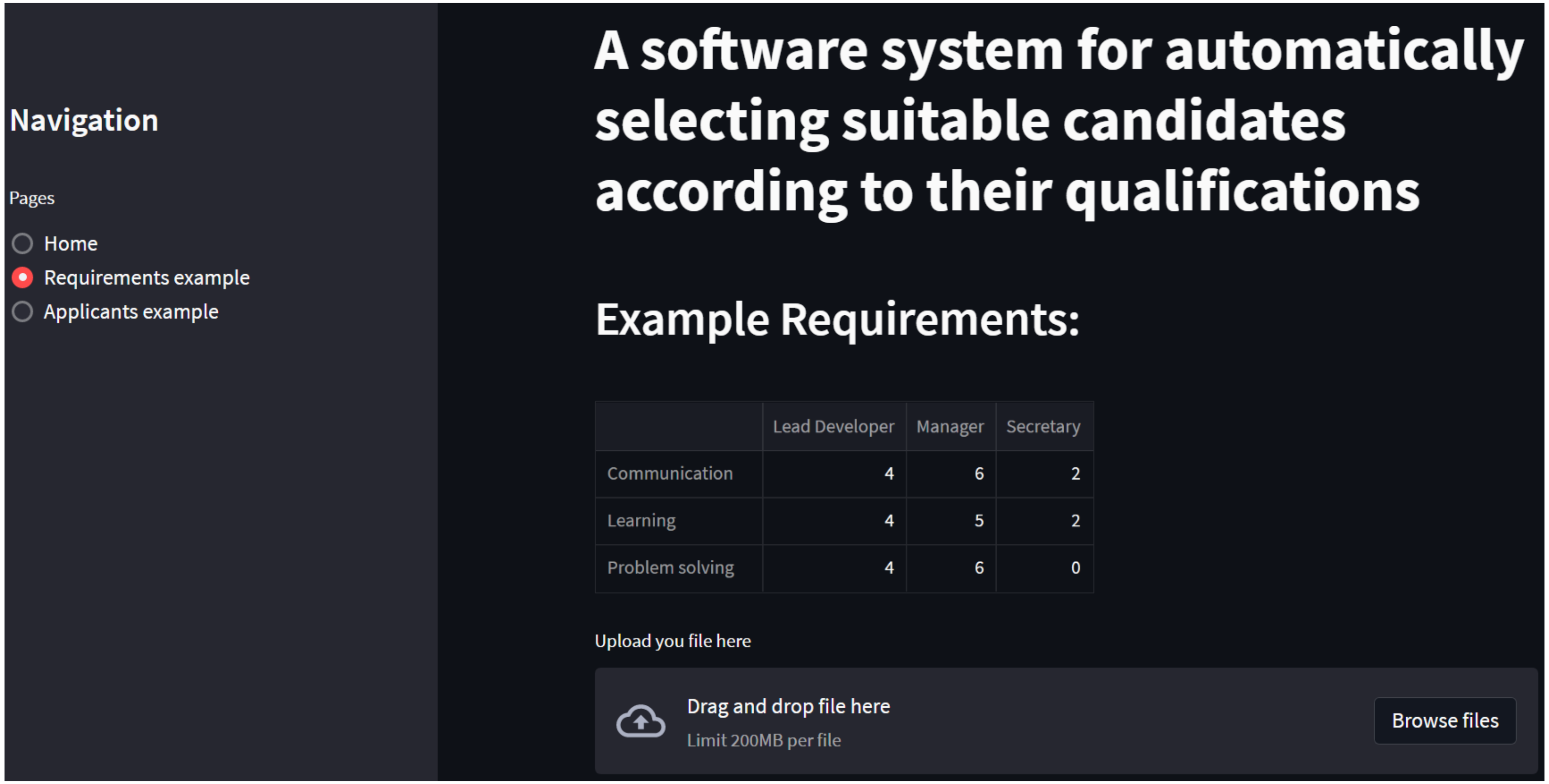
1	3	3	**0**	4
0	0	1	3	2
1	**0**	1	1	0
3	3	1	2	**0**
6	2	**0**	0	4

2.5. 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 and R programming languages.
- 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 that uses a comma to separate values.

2.5. Software tools

- Streamlit screenshots



A software system for automatically selecting suitable candidates according to their qualifications

Example Requirements:

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

Upload you file here

Chapter 3

3.1. Experiments

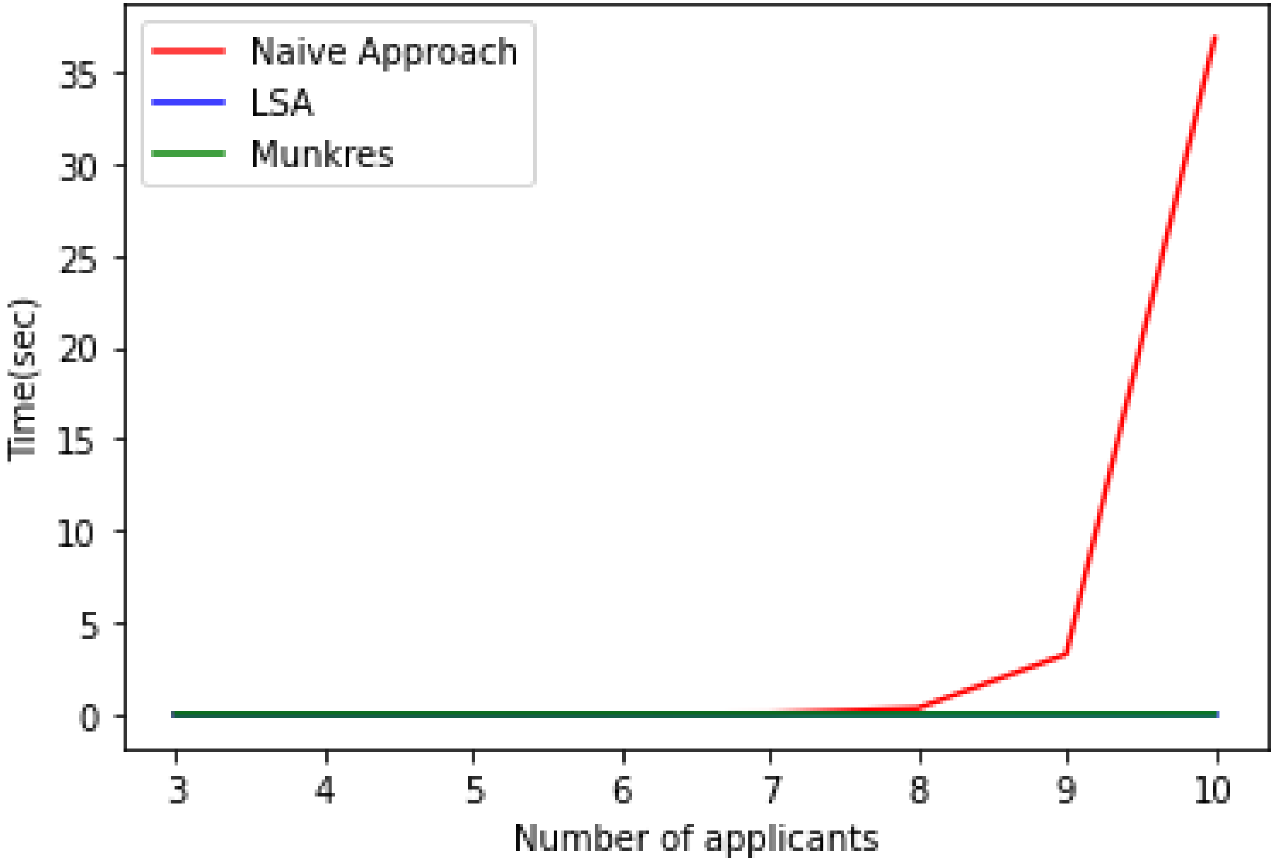
Two experiments are performed:

- In the first experiment, the performance of the three implemented functions is compared. 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, Muncres 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.

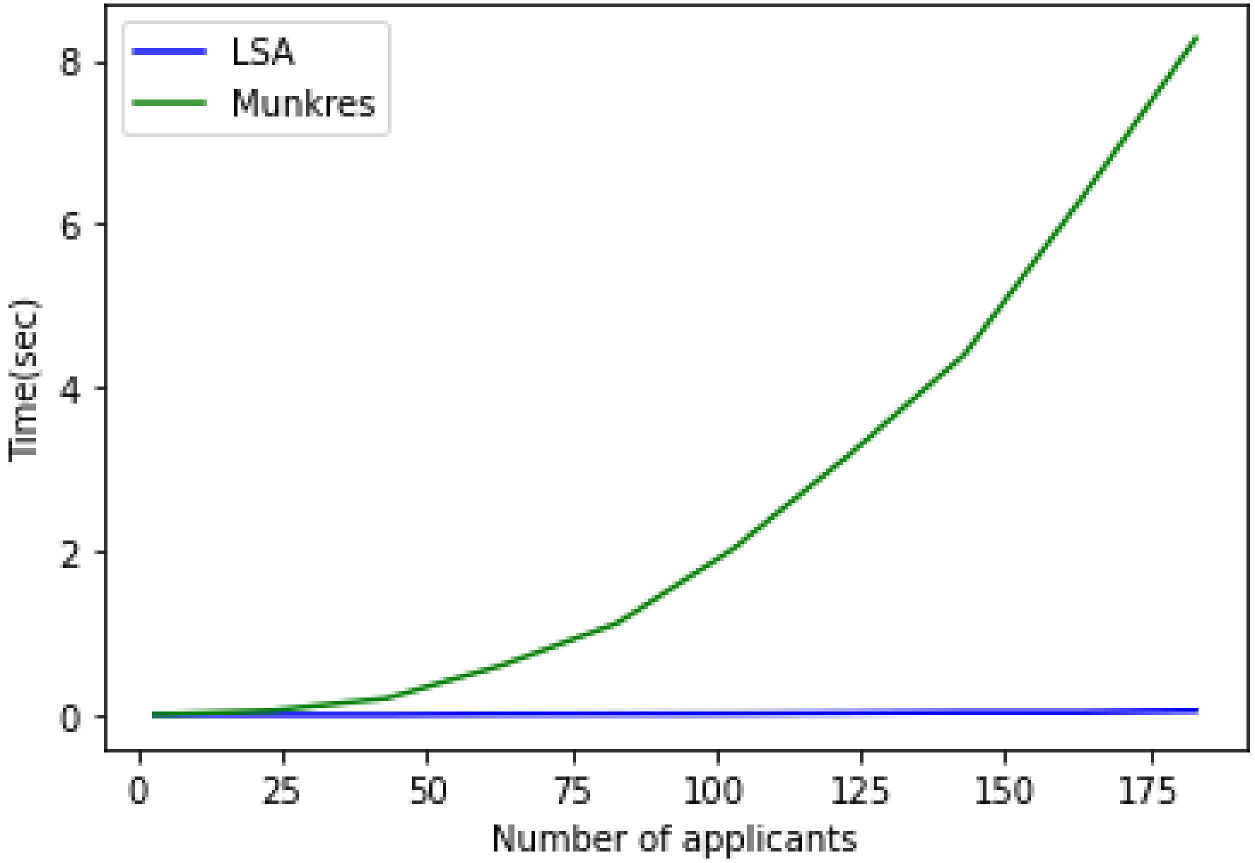
3.1. Results

This is the result from the first experiment:



3.1. Results

The result from the second experiment:



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.

- 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!