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

STUDENT: Lyubomir Lambrev

SUPERVISOR: Vanya Markova , PhD

PLOVDIV 2022

DECLARATION OF ORIGINALITY OF THE FINAL YEAR PROJECT

I, the undersigned Lyubomir Lambrev**,** a student in the Industrial Engineering degree course in the Faculty of Electronics and Automation at the Technical University of Sofia, Plovdiv Branch, graduating during the 2021/2022 academic year,

faculty No: 510259

declare that the foregoing implementation of the specific tasks related to my final year project, entitled: Software system for automatic recruitment of candidates according to their qualifications.

with supervisor: Vanya Markova, PhD

in the volume of 44 text pages , including number of figures: 20 is the result of my own work.

Date: :.....................   Signature: ....

A REVIEW OF THE FINAL YEAR PROJECT BY THE SUPERVISOR

The final year project is performed according to the assignment in full volume.

and may be admitted to presentation.

Motivation: The work was performed in sufficient volume and with good quality.

Signature: **................................**

(Supervisor)

Selected reviewer: ....

Signature: ....

(Head of Department)

Table of Contents

Introduction....................4
Chapter 1....................5
1.1 State of the Art....................5
1.2 Labor Shortage....................6
Chapter 2....................13
2.1 Basic Theory....................13
2.2 Formulation of design structure....................20
2.2.1 Conceptual model....................20
2.2.2 System Architecture....................21
2.2.3 Data Structures....................22
2.2.4 Methods and algorithms....................23
2.3 Software tools....................29
Chapter 3....................33
3.1 Experiments....................33
3.2 Results....................34
Conclusions....................36

INTRODUCTION

In this work, an application is developed to allocate the most suitable applicants to certain positions. The selection is made on the basis of the highest score presented qualifications and skills of the applicant.

A program is written to find the best arrangement for each applicant and the corresponding workplace.

The list of jobs is provided by the employer, and the list of applicants by the employment agency. We will consider evaluating each applicant's qualification (competence) as the number of years of experience of the applicant.

The main objective of this work is to build a software system for assigning candidates to their respective jobs in regards of required competencies. In this system, there should be a user interface, a sub-system for loading the initial data and storing the results and control logic.

The tasks that are decided to achieve these set of goals are: an overview of the state of the art, design and construction of a software system, researching the performance of the Hungarian algorithm.

In the implementation of the software solutions, modern software technologies such as Python, Anaconda, Spyder, Streamlit, etc. are used.

This thesis is organized as follows: In the first chapter, the necessity of this work and the relevance of the tasks are discussed. In the second chapter basic theoretical questions are addressed, designed are the conceptiual model, the system architecture, basic data structures and basic methods and algorithms. In the third chapter, experiments and results are shown from the work of the main algorithms.

CHAPTER 1

1.1 State-of-the-Art and trends

The use of technology has increased enormously in the field of personnel recruiting, selection and assessment. In its digital form often referred to as e-recruiting [10] also known as the term digital selection procedures to denote "any procedure that makes use of digital communication technology (i.e. computer-, internet- or mobile-based) for the purposes of assisting organizations during recruitment and selection". Digital selection procedures such as application submission via web-based platforms or technology-mediated interviews are nowadays established alternatives to traditional, analog applications via letter post or on-site interviews and enable organizations to handle larger applicant populations more efficiently, to reach a more globalized labor market, to save time, mail, and travel costs, and to implement environmentally sustainable practices compared to traditional procedures [1].

In recent years, tremendous progress in AI could be witnessed, a field "concerned with the automation of intelligence and the enablement of machines to achieve complex tasks in complex environments" [9]. Organizations around the globe try to take advantage of these technological progresses, leading to the development of new AI-based applications that allow to automate decision-making processes that were previously carried out by humans in organizations. [2] Thus, after the transition from traditional, analog procedures to digital procedures in recruiting and selection, the next step seems to be AI-based, automated procedures. Advantages that organizations expect from automated recruiting and selection procedures are, for example, even greater savings regarding time and money compared to traditional or digital procedures and more valid and bias-free selection decisions.

Popular programs for selecting the best candidates are used in big companies becouse they create a series of steps in the hiring process. For example “Harver”[harver.com] which is a candidate selection tool that helps companies to predict job performance and select the best people for the job. It uses data and science to predict the quality of a candidate by their aptitude and skills. Another popular tool for hiring is “GoHire”[gohire.io] which is an online pre-employment platform that tests candidates before moving them onto the next phase of the hiring process. At it’s core. the highlights are it’s testing features of the personality and skills of the candidate.

Recruiting and selection involves a multi-stage mutual decision-making process in which job-seekers and hiring organizations both acquire information about each other. At each stage, they evaluate whether they still find the other party attractive and decide whether they want to proceed with the process or not.[4] Reading job advertisements is often the first instance when job-seekers and hiring organizations get in contact and thus the first critical point in the process as job-seekers decide at this stage whether or not they will enter the recruiting process [8]. Job-seekers’ so-called pre-process expectations and perceptions affect important outcomes like application intention. Thus, it is important for organizations to understand what influences these initial expectations, perceptions, and corresponding intentions, in order to design job advertisements and recruiting procedures accordingly [8].

1.2 Labor Shortage

The biggest trend over the last years was Covid-19 and it would be impossible to discuss the current labor without acknowledging the global pandemic’s role on it. While the impact of the current labor shortage varies by location and sector, it’s undoubtedly one of the biggest challenges in modern history. An ongoing labor shortage could significantly impede the world’s ability to fully recover in a post-pandemic market. Typically, a labor shortage occurs when there are not enough available workers participating in the labor market to meet the demand for employees. For example, as of early 2022, employers in Europe were struggling to fill over 1.2 million open job roles.

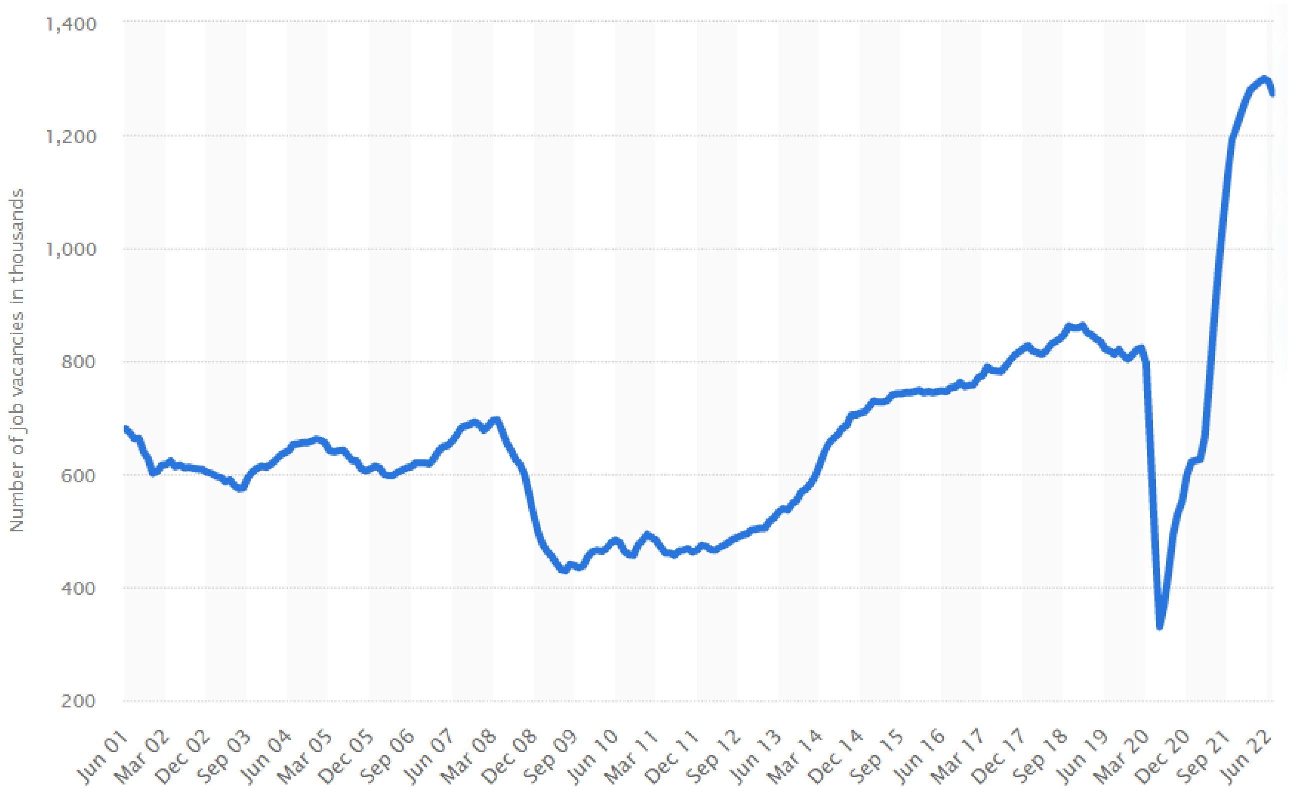


Fig.1. Number of job vacancies in the United Kingdom from 2001 to 2022. [11]

This factor alone has caused major disruptions to workplaces around the world. The lingering global pandemic has spurred a number of challenges for employers and employees, including:

- Mental health issues

Early into the pandemic, mental health professionals started to express concerns regarding the global pandemic’s impact on workers’ mental health. Today it looks like these warning are proving true.

- Immigration disruptions

Migrant workers make up 5% of the global workforce. Countries such as the United States, Saudi Arabia, United Arab Emirates, Canada, Germany and the United Kingdom depends heavily on these workers to meet production demands. The pandemic significantly hindered this dependency as countries set stricter immigration policies to control the spread of the virus within their borders.[15]

- Shift in workers expectations

Throughout the pandemic, many employees faced additional pressure at work, such as sudden layoffs and lockdowns and extraordinary personal challenges, including homeschooling their children and caring for ageing parents. These stressors have spurred a shift in workers' expectations. At the forefront of these expectations is the desire to maintain a healthy work-life balance.Furthermore, some workers are willing to change jobs to get the flexibility they need or leave the workforce altogether if they can't find it. [12]

- Low wages

While some workers are leaving the workforce altogether, the majority are simply changing jobs due to better job opportunities. Some are leaving for higher salaries. The ongoing labor shortage has created a candidate-driven market in most areas of the world. Many employees and jobseekers are requesting higher wages and improved benefits. However, these wage increases vary across the globe.[3]

While nearly every industry is affected in some way by the growing labor shortage, there are a few sectors where the impact is larger:

- Healthcare

The healthcare industry was hit hard during the pandemic. Not only did these essential workers risk their lives, as well as those of their families, by going to work every day, but many also had to work long hours due to staffing shortages.

- Manufacturing

For the past years, the global manufacturing industry's main concern has been labor shortages. These shortages are due in large part to a lack of workers with technical skills. Other factors include increasing retirement rates, growing complexity in the global supply chain. [13]

- Supply chain

Logistics is another sector that is struggling to attract workers before and after the pandemic. This labor shortage is not isolated to just one region of the world.

Global supply chains are the networks between companies and their suppliers needed to turn materials into the products they sell. Massive supply chain shortages emerged in the wake of pandemic lockdowns that shut businesses around the world and kept workers at home.

Looking at any industry from the outside, it might seem like everything is working as it's meant to. However, behind the scenes, the problem becomes glaringly apparent. Foodservice and hospitality-related businesses experienced a 6.6% quit rate in September 2021.

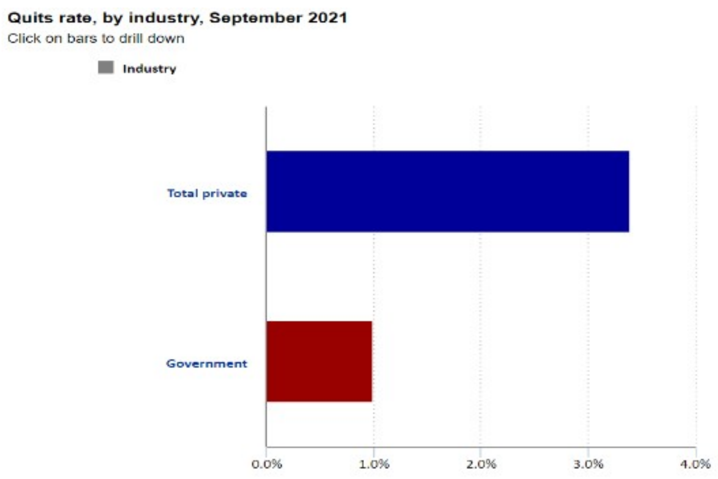


Fig.2. Foodservice and hospitality-related businesses experienced a 6.6% quit rate in September 2021. [14]

Industries worldwide have lost millions of workers due to COVID-19 and the Great Resignation that followed. Many of those who left the workforce during the pandemic decided that taking early retirement was a better option than returning to work once things returned to normal. The best choice for employers would be to make the necessary changes to bring in new workers and retain those already employed.

As one can probably picture, automating the most repetitive and time-consuming tasks brings benefits for the overall recruitment system. Here are just some of the primary benefits from implementing an automated recruitment system. By automating one of the most common tasks, it is ensured that there will never be a backlog of work sitting on the desk, holding up the entire hiring process. This is especially important if people work as a recruiter for a high-turnover industry such as call centers. One of the primary factors that contribute to inefficient recruitment funnels, and lost opportunities to hire top talent, is lengthy delays in moving applicants through the pipeline. [10]

Automation can help prevent these situations by ensuring that all of the most important, but simple, tasks aren’t piling up somewhere. Instead, candidates are able to move smoothly through the pipeline, and recruiters are able to easily meet deadlines and make decisions on who to hire.

Some of the ways that an automated recruitment system can help avoid bottlenecks include:

- Filtering applicants quickly
- Avoiding overloading
- Notifying team immediately when actions need to be taken
- Automatically flagging at-risk deadlines

Together, these automated actions can help to ensure that the recruitment funnel is always running smoothly.[1]. Which brings us to the second benefit.

Eliminating bottlenecks and improving the overall efficiency of the pipeline helps to eliminate time and resource waste that can lead to the overall poor performance of the recruitment process. Delays and missed steps due to task overload can cause lost productivity and revenue by failing to fill a skills gap when it’s needed.

Additionally, sourcing and screening candidates takes both monetary and human resources, both of which can be very expensive for the organization. Wasting that money and effort due to a lack of automation, or because of task overload, is simply not a good business move for companies.

Some examples of how automation can help to eliminate time and waste include:

- Eliminating the need to manually call non-responsive or unqualified candidates
- Freeing recruiters from repeating the same task over and over again
- Automatically identifying areas of inefficiency and waste

By automating the most time-consuming tasks, and continuously monitoring the efficiency of the recruitment process, companies will drastically reduce the amount of costly waste.

An automatic recruitment system that eliminates bottlenecks, and missed communications, contributes to an overall better candidate experience for the applicants. If there’s one thing that can ruin an applicants’ opinion of organizations, it’s the poor communication, and not following through decisions.

A recruiter’s job should mostly focus on finding and engaging with high potential applicants who will make a real difference at the organization.

Implementing an automated recruitment system will allow recruiters to focus on doing just that, rather than tasks that don’t really move the dial for the organization.

Naturally, each of the benefits above will ultimately contribute to better hiring results overall. A systematic approach of hiring that’s driven by analytics and continuous improvement means that the recruitment can continuously hone.

As candidates continue to audition the process will improve and the overall efficiency and quality of the system will also improve.

Automatic recruitment systems also help to eliminate much of this inherent human bias. Rather than relying on a subjective reading of a resume to determine if a candidate moves on to the next round, for example, automation can make that decision without any inherent bias. [7]

CHAPTER 2

2.1 Basic Theory

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. This is done to separate internal representations of information from the ways information is presented to and accepted from the user.

Components of MVC:

Model – The central component of the pattern. It is the application's dynamic data structure, independent of the user interface. It directly manages the data, logic and rules of the application.

View - Any representation of information such as a chart, diagram or table. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.

Controller - The Controller is that part of the application that handles the user interaction. It interprets the inputs from the user, informing the model and the view to change as appropriate to the input and convert it to commands for the model or view.

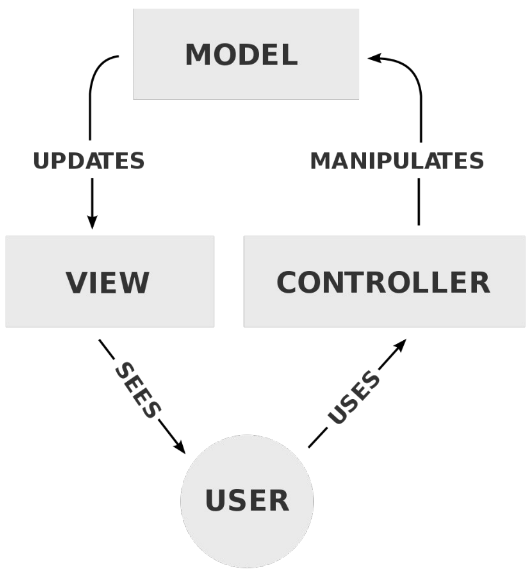


Fig. 3. Model-View-Controller

Hungarian Algorithm

Harold W. Kuhn, in his celebrated paper entitled The Hungarian Method for the assignment problem, described an algorithm for constructing a maximum weight perfect matching in a bipartite graph. In his delightful reminiscences, Kuhn explained how the works (from 1931) of two Hungarian mathematicians, D. Konig and E. Egervary, had contributed to the invention of his algorithm. The Hungarian method is a combinatorial optimization algorithm that solves the assignment problem in polynomial time and which anticipated later primal–dual methods. It was developed and published in 1955 by Harold Kuhn.

- Combinatorial optimization is a subfield of mathematical optimization that consists of finding an optimal object from a finite set of objects, where the set of feasible solutions is discrete or can be reduced to a discrete set. Typical combinatorial optimization problems are the travelling salesman problem, the minimum spanning tree problem, etc.

In many such problems, such as the ones previously mentioned, exhaustive search (Naive approach) is not tractable, and so specialized algorithms that quickly rule out large parts of the search space or approximation algorithms must be resorted to instead.

In order not to describe a lot of theory with mathematical terms and definitions, we consider a couple of options for constructing an assignment problem, which we will immediately understand in which cases the Hungarian method is applicable:

- The task of assigning employees to positions. It is necessary to distribute employees to positions so that maximum efficiency is achieved, or there are minimum costs for work.
- Assignment of machines to production sections. The distribution of machines so that when they work, production is as profitable as possible, or the cost of their maintenance is minimal.
- Selection of candidates for various vacancies according to their estimates.

There are many options for which the Hungarian method is applicable, while similar tasks arise in many areas of activity. As a result, the problem must be solved in such a way that one executor (man, machine, tool, etc.) can perform only one job, and each job is performed by only one executor.

A necessary and sufficient condition for solving a problem is its closed type, when the number of performers = the number of works (N=N). If this condition is not met, then fictitious performers, or fictitious works, can be added for which the values in the matrix will be zero. This will not affect the solution of the problem in any way, it will only give it the necessary closed type.

Example of a problem: Let an important scientific conference be planned. To conduct it, you need to set up sound, light, images, register guests and prepare for breaks between performances. There are 5 organizers for this task.

Each of them has certain ratings for the performance of a particular job (suppose that these ratings are set as the arithmetic average of the reviews of their employees). It is necessary to distribute the organizers so that their total score is maximum.

	Sound Settings	Light Settings	Image Sellings	Register Guests	Break orginizer
Maria	7	3	6	9	5
Peter	7	5	7	5	6
Aleksandra	7	6	8	8	9
Rocksan	3	1	6	5	7
Ivan	2	4	9	9	5

Fig. 4. Initial matrix.

In each row of the matrix, it is necessary to find the maximum element, subtract it from each element of the corresponding row and multiply the entire matrix by -1.

$$\begin{array}{|c|c|c|c|c|c} \hline 7 & 3 & 6 & 9 & 5 & \mathbf{9} \\ \hline 7 & 5 & 7 & 5 & 6 & \mathbf{7} \\ \hline 7 & 6 & 8 & 8 & 9 & \mathbf{9} \\ \hline 3 & 1 & 6 & 5 & 7 & \mathbf{7} \\ \hline 2 & 4 & 9 & 9 & 5 & \mathbf{9} \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|c|c|c|} \hline 2 & 6 & 3 & 0 & 4 \\ \hline 0 & 2 & 0 & 2 & 1 \\ \hline 2 & 3 & 1 & 1 & 0 \\ \hline 4 & 6 & 1 & 2 & 0 \\ \hline 7 & 5 & 0 & 0 & 4 \\ \hline \end{array}$$

Fig. 4.1. Matrix after transformation.

There must be only one selected zero per row and per column. (i.e., when zero is chosen, then the remaining zeros in this row or in this column are no longer taken into account).

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. 4.2. Selection of zeroes in the matrix.

The next step is Matrix reduction by rows (we look for the minimum element in each row and subtract it from each element, respectively):

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

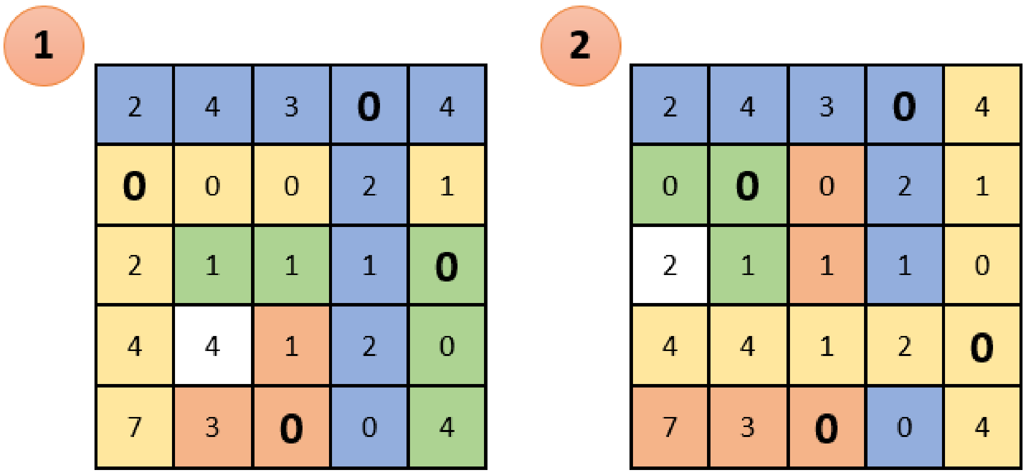
Fig. 4.3. Matrix reduction by rows.

Because all minimal elements are zero, then the matrix has not changed. We carry out the reduction by columns:

$$\begin{array}{ccccc} 2 & 6 & 3 & 0 & 4 \\ 0 & 2 & 0 & 2 & 1 \\ 2 & 3 & 1 & 1 & 0 \\ 4 & 6 & 1 & 2 & 0 \\ 7 & 5 & 0 & 0 & 4 \\ \hline \mathbf{0} & \mathbf{2} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array} \rightarrow \begin{array}{ccccc} 2 & 4 & 3 & 0 & 4 \\ 0 & 0 & 0 & 2 & 1 \\ 2 & 1 & 1 & 1 & 0 \\ 4 & 4 & 1 & 2 & 0 \\ 7 & 3 & 0 & 0 & 4 \end{array}$$

Fig. 4.4. Matrix reduction by columns.

Again, we look to ensure that in each column and in each row, there is only one selected zero. There are two options for choosing zeros, but none of them gave the desired result.

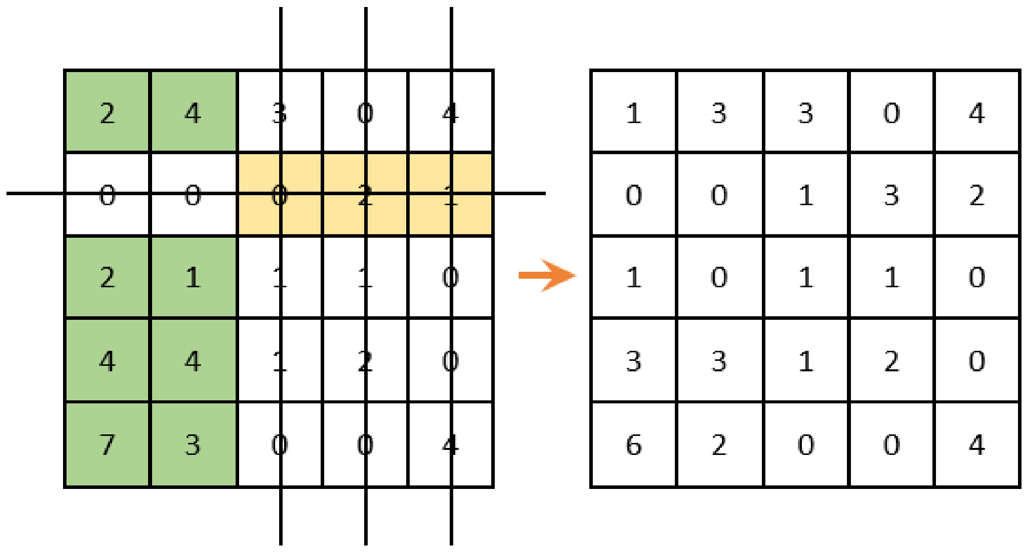


2	4	3	**0**	4
0	0	0	2	1
2	1	1	1	**0**
4	4	1	2	0
7	3	**0**	0	4

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

Fig. 4.5. Matrix with two options for choosing zeroes.

cross out rows and columns that contain zero elements (IMPORTANT! The number of cross outs should be minimal). Among the remaining elements, look for the minimum, subtract it from the remaining elements (which are not crossed out) and add to the elements that are located at the intersection of the crossed-out rows and columns (what is marked in green - subtract there; what is marked in gold - summarize there; then, what is not painted over - do not touch):

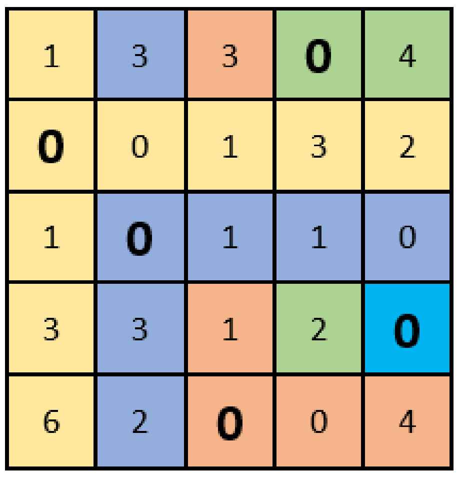


2	4	3	0	4
0	0	0	2	1
2	1	1	1	0
4	4	1	2	0
7	3	0	0	4

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

Fig. 4.6. Matrix after transformation.

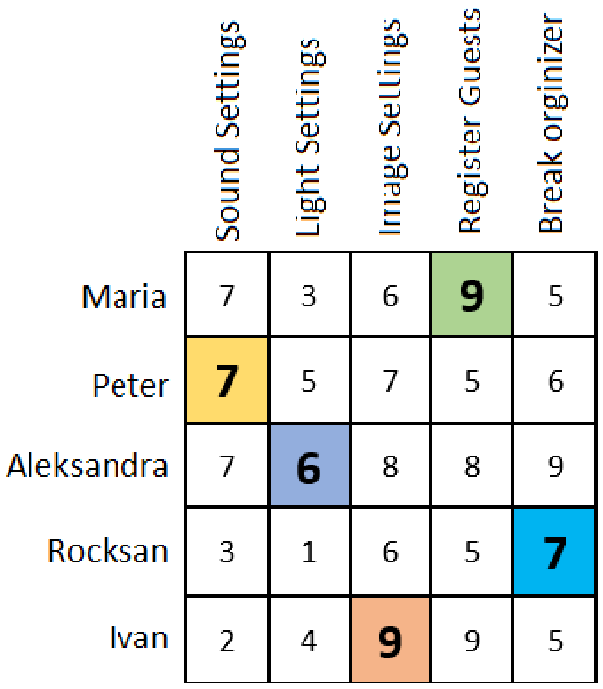
There is only one selected zero in each column and row. The solution is complete .



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. 4.7. Selection of zeroes in each column and row.

If there is a problem in the solving and it is still impossible to choose zeros so that there is only one in each column and row, we then repeat the algorithm from the place where the row reduction was carried out (the minimum element in each row).



	Sound Settings	Light Settings	Image Settings	Register Guests	Break orginizer
Maria	7	3	6	**9**	5
Peter	**7**	5	7	5	6
Aleksandra	7	**6**	8	8	9
Rocksan	3	1	6	5	**7**
Ivan	2	4	**9**	9	5

Fig. 4.8 Initial matrix with the optimal solution.

substitute the locations of the selected zeros in the initial table. Thus, we get the optimum, or the optimal plan, in which the organizers are distributed among the works and the sum of the marks is maximum.

2.2 Formulation of design structure

As previously said a program must be written to find the best arrangement for each applicant and the corresponding workplace.

2.2.1 Conceptual model

The Conceptual model is a representation of the system. It consists of concepts used to help understand or simulate the process of the system.

The conceptual model of the system is shown in Fig. 5. The input data is in the form of separate sets for jobs and candidates. At the first stage of work, these datas are combined into a structure of the same size. After that, one of the implemented methods is applied (Naïve approach, Muncres or Linear_sum_assignment).

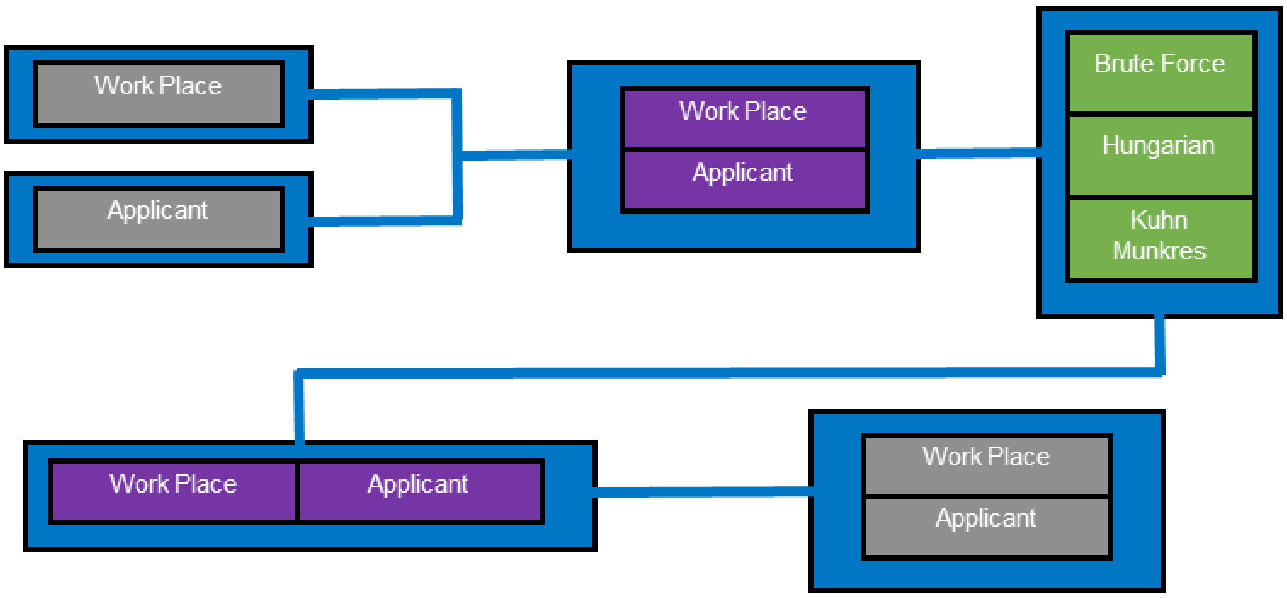


Fig.5. Conceptual model

After applying these functions, the result is combined into a result list of the distribution of candidates by job. And at the last stage, the data in this list is stored or printed.

2.2.2 System Architecture

We then make a MVC architectural pattern that means that the code will be split into 3 parts:

- modul- works with data that is written/read in CSV files. They are accessed also read and written by using the Pandas library.
- Controller- The Naive approach algorithm has been implemented. Then two algorithms are used for the implementation of the Hungarian algorithm from the scipy library. The two algorithms are Linear_sum_assignment and Munkres.
- View- Entering an argument from the command line with argparse (CLI main_cli.py). GUI(main_gui.py) is used to enter arguments via Streamlit

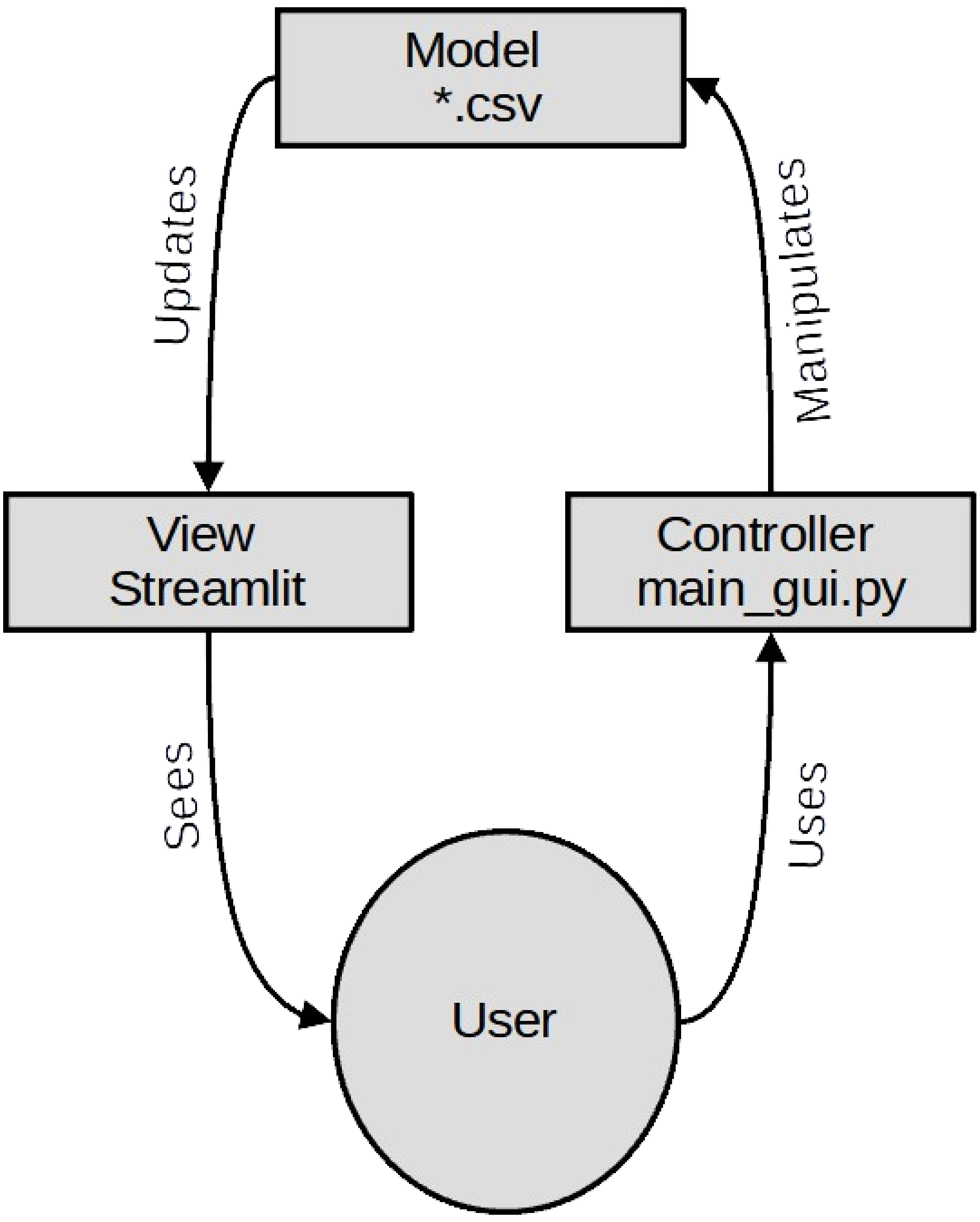


Fig. 6. Model-View-Controller in our system.

Experimental(case studies)- this part represents the experiments that have been done. We state that one of the three implementations of the algorithms is better than the others according to the criteria - algorithm execution speed and memory usage for algorithm execution.

2.2.3 Data structures

The design structures of the program are as follows:

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.

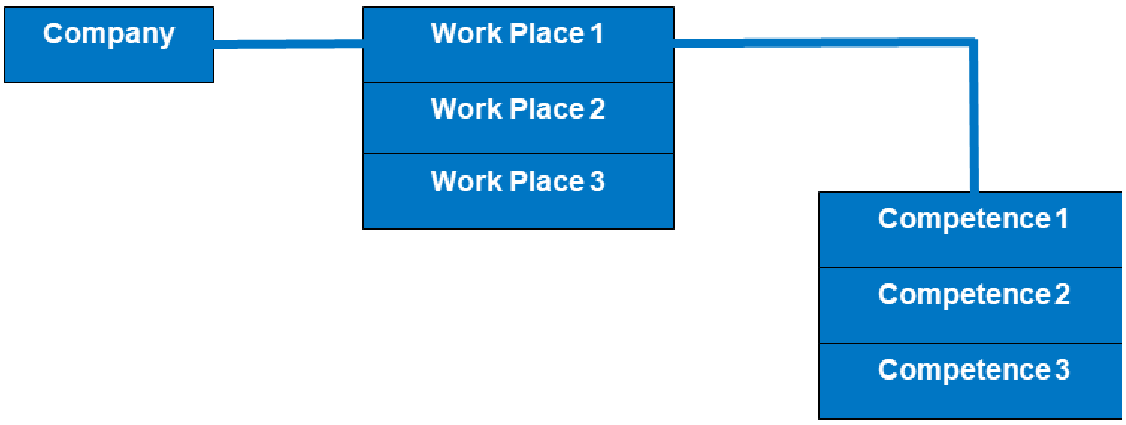


Fig.7. Data structure represented by the employer.

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.

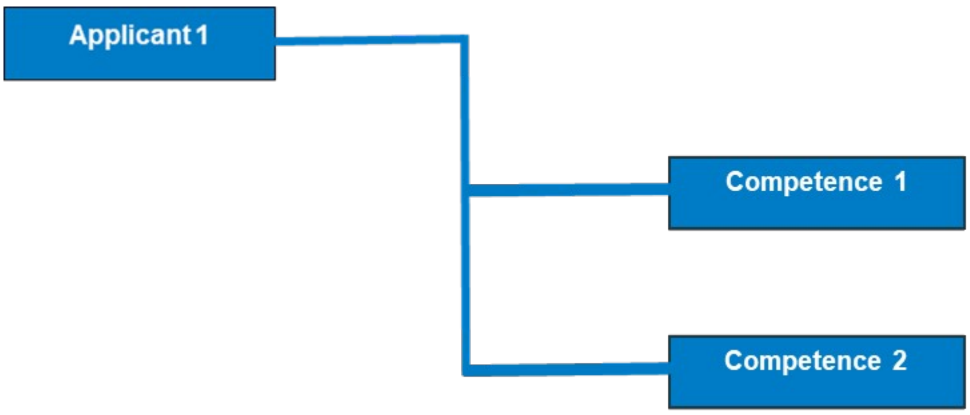


Fig.8. Data structure represented by applicant.

2.2.4 Methods and algorithms

Functions of the program:

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:

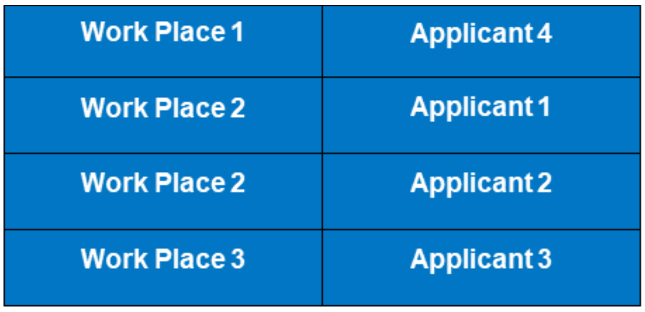
I When the competence is required by the employer and possessed by the applicant (we return the number of years of the applicant).

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.

Utility function - This function returns the sum of all fitting values of the ordered pair (Applicant, Workplace) --> utility_value.

The ordinance is a way of describing the relationship between the workplace and the applicants.



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

Fig.9. Data structure of the ordinance.

Algorithm of full enumeration:

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 this list.

This is the so-called naive approach or the method of full enumeration.

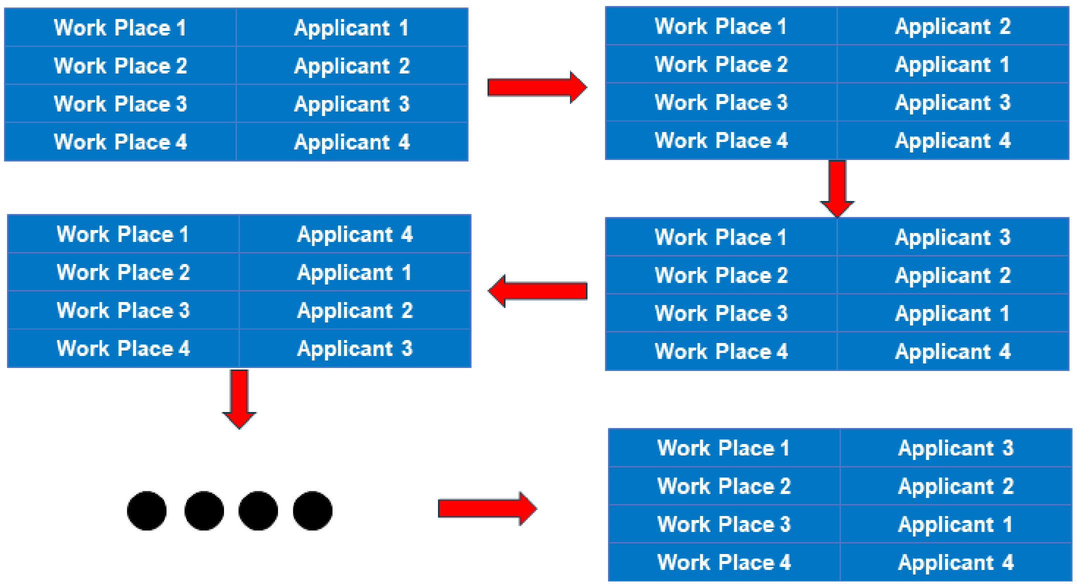


Fig.10. List of all possible Ordinance.

converting data structures and preparing them to find the optimized linear sum.

As input we have two python dictionaries(Java hasmap, C++ map, Javascript associative array),

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

	Communication	Learning	Problem solving
Lead Developer	4	4	4
Manager	6	5	6

Secretary	2	2	0

These two dictionaries need to be converted into a two-dimensional array more precisely np.array for this purpose we first make a transient array.

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

In the cells of this array we put corresponding values from the fitting function, passing input corresponding to applicant-job(tuple). We then record the results in a dictionary called fit_val. We fill this dictionary by traversing the elements of my_new_df_app first and then the elements of my_new_df_jobs in a double loop, after that we create a new np.array = m in which we again traverse in a double loop with the data from the dictionary fit_val. In the next step we pass the np.array m to scipy.optimize linear_sum_assignment, as a result we get a tuple consisting of the maximum sum of the utility function in regards to the assignment conditions. The second part of the returned result is a one-dimensional array with the indices of the corresponding jobs.
this array has the form:[1,2,0]

In an implicit form, the indexes of applicants are also present, so the assignment results can be presented as follows:[(0,1),(1,2),(2,0)]

The relationship between the explicit and implicit indices in the m matrix and the keys in the fit_val dictionary can be represented as such:

		0	1	2
		Lead Developer	Manager	Secretary
0	Josh	80	115	26
1	Maria	20	28	10
2	Michael	52	74	16

using unique_app and unique_job we convert numeric indices to strings

[(0,1),(1,2),(2,0)]–>[(“Josh”,”Manager”),(“Maria”,”Secretary”), (“Michael”,Lead Developer)]

Naive approach Algorithm

Naive approach or exhaustive search, also known as generate and test, is a very general problem-solving technique and algorithmic paradigm that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement.

A naive approach algorithm that finds the divisors of a natural number n would enumerate all integers from 1 to n, and check whether each of them divides n without remainder. The algorithm would examine all possible arrangements.

While it is simple to implement and will always find a solution if it exists, implementation costs are proportional to the number of candidate solutions – which in many practical problems tends to grow very quickly as the size of the problem increases. Therefore, the naive approach is typically used when the problem size is limited, or when there is a problem-specific that can be used to set of candidate solutions to a manageable size. The method is also used when the simplicity of implementation is more important than speed.

Indeed, the naive approach can be viewed as the simplistic. It should not be confused with backtracking, where large sets of solutions can be discarded without being explicitly enumerated. The naive approach method for finding an item in a table – namely, check all entries of the latter, sequentially – is called linear search.

Advantages and disadvantages of the naive approach algorithm

Advantages:

- The naive approach is a guaranteed way to find the correct solution by listing all the possible candidate solutions for the problem.
- It is a generic method and not limited to any specific domain of problems.
- The naive approach method is ideal for solving small and simpler problems.
- It is known for its simplicity and can serve as a comparison benchmark.

Disadvantages:

- The naive approach is inefficient, for real-time problems.
- This method relies more on compromising the power of a computer system for solving a problem than a good algorithm design.
- Naive approach algorithms are slow.
- Naive approach algorithms are not constructive or creative compared to algorithms that are constructed using some other design paradigms.

2.3 Software tools

Streamlit is a free and open-source framework to rapidly build and share beautiful machine learning and data science web apps. Users instantly develop web apps and deploy them easily using it. It’s also a Python-based library specifically designed for machine learning engineers and Data scientists. Streamlit is easier to learn and to use rather than other more complicated frameworks which will take much more time to learn, it is simple as long as it can display data and collect needed parameters for modeling.

Working with Streamlit is straightforward. First the code is written using Streamlit commands into a normal Python script, then it is run with the command “Streamlit run (name of the script)” in the Prompt. As soon as the script is running, a local Streamlit server will show up and the app will open in a new tab in the default web browser. The app is the users canvas, where users draw charts, text, widgets, tables, and more.

Development flow. Every time the user wants to update the app, the source file needs to be saved and it will automatically update. When the user does that, Streamlit detects if there is a change and asks whether the user wants to rerun the app. if Chosen "Always rerun" (at the top-right of the screen) it will automatically update the app every time there is a change in the source code. This allows users to work in a fast interactive loop: users type some code, save it, try it out live, then type some more code, save it, try it out, and so on until users are happy with the results. This tight loop between coding and viewing results live is one of the ways Streamlit makes life easier.

Data flow. Streamlit's architecture allows to write apps the same way users write plain Python scripts. Streamlit apps have a unique data flow. Whenever a callback is passed to a widget or parameter, the callback will always run before the rest of the script. And to make all of this fast and seamless, Streamlit does some heavy lifting behind the scenes. A big role plays the cache detector, which allows developers to skip certain costly computations when their apps rerun. [streamlit.io]

Python and Anaconda

Python- Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. [python.org] [6]

Anaconda- Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. Package versions in Anaconda are managed by the package management system conda. [anaconda.com]

Argparse- The argparse module makes it easy to write user-friendly command-line interfaces. It parses the defined arguments from the sys.arg .
The argparse module also automatically generates help and usage messages, and issues errors when users give the program invalid arguments. It is also a standard module. There is no need for installation.
A parser is created with ArgumentParser and a new parameter is added with add_argument. Arguments can be optional, required, or positional. [python.org]

Spyder- Spyder is a free and open source scientific environment written in Python, for Python, and designed by and for scientists, engineers and data analysts. It features a unique combination of the advanced editing, analysis, debugging, and profiling functionality of a comprehensive development tool with the data exploration, interactive execution, deep inspection, and beautiful visualization capabilities of a scientific package. It has the following components: Editor, IPython Console, Variable Explorer, Plots, Debugger, Helper. [spyder-ide.org]

Pandas- Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language**.** It is most widely used for data science/data analysis and machine learning tasks. Pandas is built on top of another package named Numpy, which provides support for multi-dimensional arrays. As one of the most popular data wrangling packages, Pandas works well with many other data science modules inside the Python ecosystem, and is typically included in every Python distribution. [pandas.pydata.org]

Numpy- Numpy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices) and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. At the core of the Numpy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. [numpy.org]

CSV- CSV stands for comma-separated values, which is a delimited text file that uses a comma to separate values. Each line of the file is a data record. Each record consists of one or more fields, separated by commas. The use of the comma as a field separator is the source of the name for this file format. A CSV file typically stores tabular data (numbers and text) in plain text, in which case each line will have the same number of fields.

The CSV file format is not fully standardized. Separating fields with commas is the foundation, but commas in the data or embedded line breaks have to be handled specially. [5]

CHAPTER 3

In this chapter, a comparative analysis of the effectiveness is made for the various solutions of the researched task:

A program is written to find the best arrangement for each applicant and the corresponding workplace. The list of jobs is provided by the employer, and the list of applicants by the employment agency. We will consider evaluating each applicant's qualification (competence) as the number of years of experience of the applicant.

Three methods are implemented in this work: The naive approach, the LSA method from scipy and the Muncres method. The initial hypothesis is that LSA is the fastest because it is written in C, the Muncres method is the next most efficient because it is written in perl, and the naive approach method is the slowest because it implements full enumeration.

These three functions are applied to a set of data of varying volume. The main data structure that the functions work with are two dictionaries app_serie and job_serie of the same size. For each experiment, the volume of data grows. Through the operation of the system timer, the time for which each of the implemented functions is executed is counted.

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.

3.2 Results

The results of the first experiment are shown in figure 11. From here it can be seen that the naive approach method is significantly slower than the other two methods.

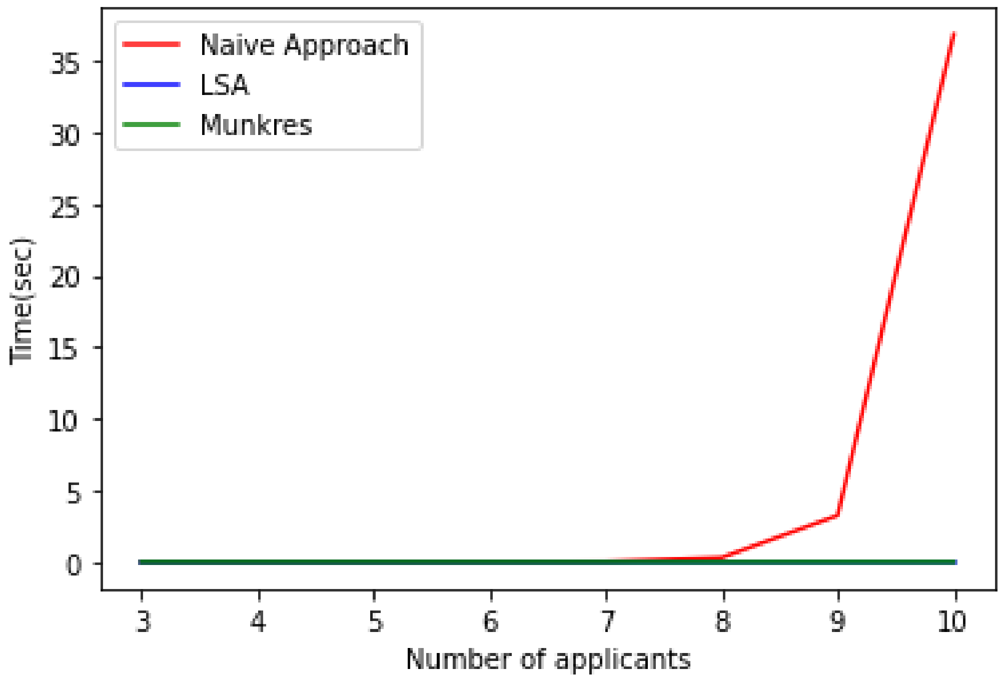


Fig. 11.

Furthermore, the execution time grows exponentially for more than 12 candidates, and exceeds 10 min for 11 candidates. This makes this method unsuitable for practical use with more than 10 candidates.

Therefore, it is necessary to compare the methods of Muncres and LSA for larger volumes of data.

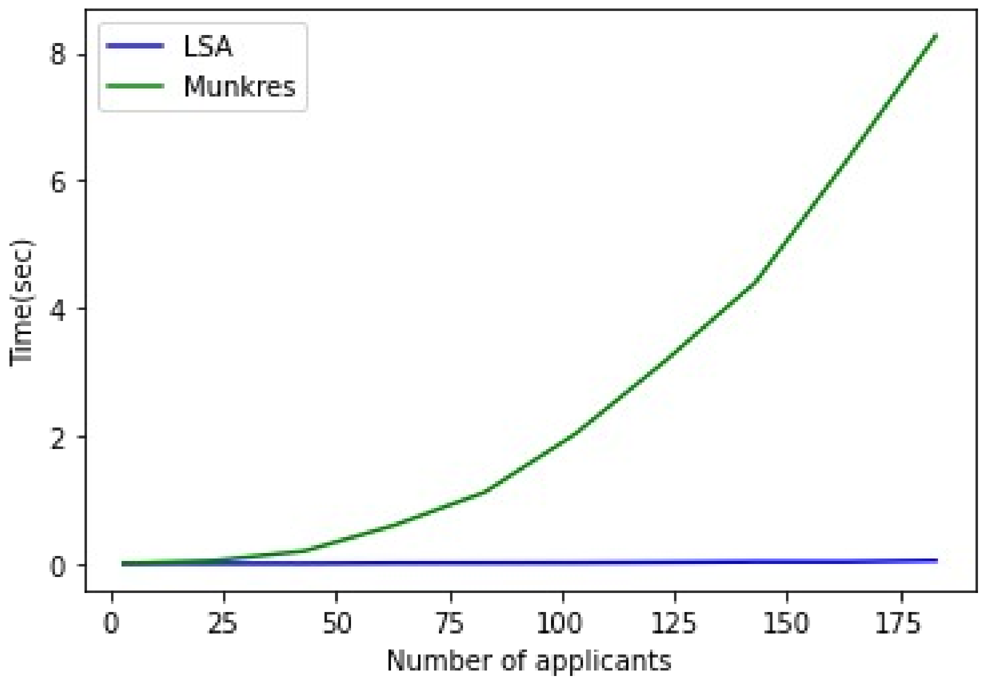


Fig 12.

From the graph of the second experiment, it can be seen that LSA is almost an order of magnitude faster than the Muncres implementation. Which is due to the fact that LSA is written in C and Muncres in perl. It can be concluded that the LSA method can solve a problem for 500 candidates in less than one second.

CONCLUSIONS

In this work it is designed and implemented a software system for automatically selecting suitable candidates according to their qualifications.

As a result of the work done, the following conclusions can be made:

- From the theoretical overview, it follows that systems of this type will find more widespread applications in the automated labor hire.
- From the practical part, it can be concluded that the streamlit environment is suitable for building a sustainable and convenient graphical interface.
- In addition, python, anaconda and spyder are suitable for developing similar software systems and modules.
- 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.

From the experiments that are made it follows that the library implementations of the Hungarian algorithm: scipy.optimize.linear_sum_assignment and Munkres are significantly more efficient than the Naive approach. The difference grows with the size of the input data. For large data sizes, the munkres implementation is more efficient.

In this work, the case is studied where the number of applicants is equal to the number of jobs. One possible development is the implementation of padding of the input data to allow selection to be made in the case where the number of applicants and jobs do not match.

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.

APPENDIX A

Installation of **Anaconda and Python** for windows 10:

Go to the Anaconda website and choose download. It will automatically download the newest version.

Once the installer is downloaded, run Anaconda installer.

When the installer is open click next, then read the licence agreement and click on I Agree.

Click install for all users(required admin privileges),then continue by clicking next.

Note the installation location and then click Next.

This is an important part of the installation process. Check the box to add Anaconda to my path. then click install.

When the installation is complete click on next.

Click skip on Microsoft VSCode.Once the installation is over click Finish.

When Anaconda is installed it comes with built in tools and applications.These tools and applications are as follows: Anaconda navigator, Anaconda powershell prompt , Anaconda prompt, Jupyter notebook, Spyder and Python.

Then by using the Windows key, we find the Anaconda folder. Click on it and a drop-down menu appears. Afterwords Anaconda Prompt is selected.

Installation of **Scipy** on windows 10:

Use the Anaconda Prompt and just type

> conda install scipy

Installation of **Numpy** on windows 10:

Use the Anaconda Prompt and just type

> conda install numpy

installation of **Pandas** on windows 10:

Use the Anaconda Prompt and just type

```
> conda install pandas
```

installation of **Munkres** on windows 10:

Use the Anaconda Prompt and just type

```
> conda install Munkres
```

installation of **Matplotlib** on windows 10:

Use the Anaconda Prompt and just type

```
> conda install Matplotlib
```

APPENDIX B

Installation and working with streamlit in windows 10:

Using the previously installed Anaconda Navigator, we start it up and then set up our environment by hovering and clicking on Environments.

It should show a base(root) Environment with a green arrow, click on the green arrow and it will pop-up a drop menu , open terminal is selected.

After the terminal has appeared we type in it

> Conda install Streamlit

then we wait for the package to install.

When the installation is finished, in the same terminal we can type

> Streamlit run code_application.py

APPENDIX C

Example of the work in a session of windows 10:

The application code will be uploaded to a Github repository for easy access.

To download the code we go to the repository and click on the name of the code.

After the page has loaded on the right side of the page we can see a button for download,

We click it and select a path in which the file to be saved.

To open the code of the script, we use Spyder as our environment.

After opening Spyder we go at the top left corner and click the file drop menu, Open is selected.

We then find the location of the script and open it.

After opening the script in Spyder we press the button F5 on the keyboard to execute the code and show the results.

Afterwords by using the Windows key, we find the Anaconda folder. Click on it and a drop-down menu appears. Anaconda Prompt is selected.

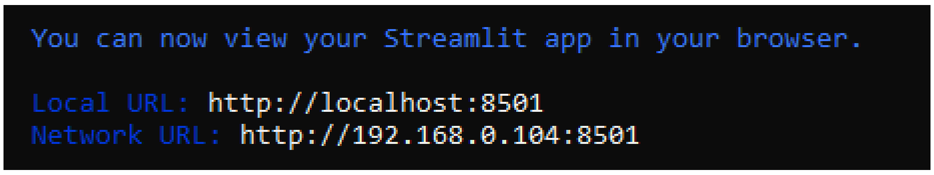
In the Anaconda Prompt we type

> Streamlit run code_application.py

As soon as the script is done running , a local Streamlit server will spin up and the app will open in a new tab in the default web browser.

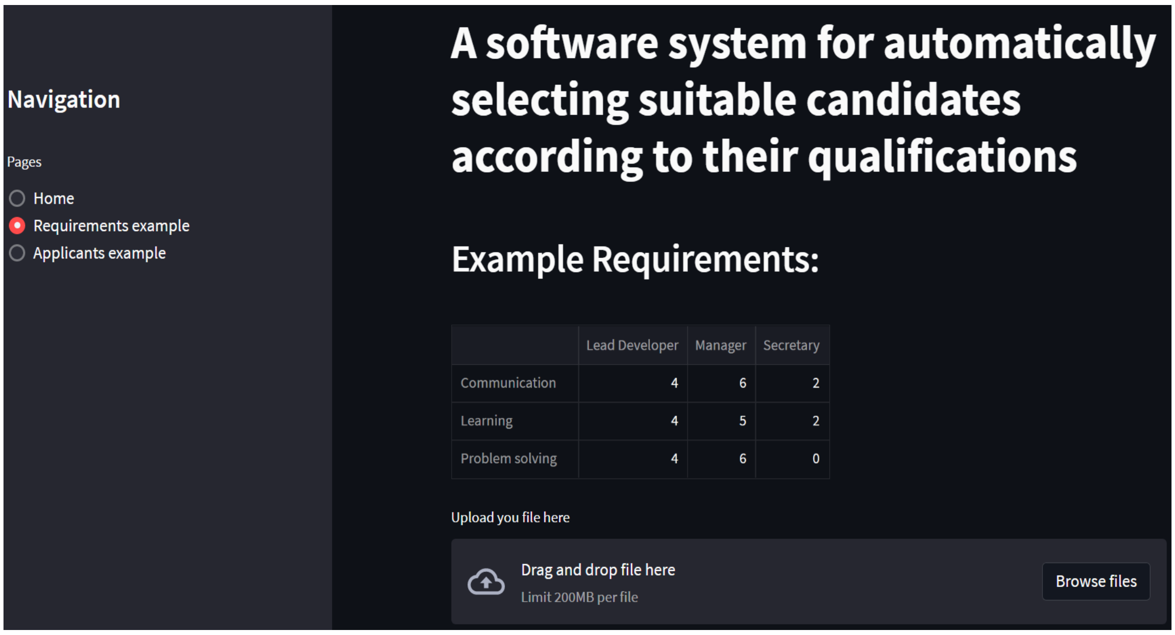
The results are shown in Streamlit.

As soon as the script is done running , a local Streamlit server will spin up and the app will open in a new tab in the default web browser.



```
You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.0.104:8501
```



Navigation

Pages

Home
[✓ Requirements example]
Applicants example

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

Drag and drop file here
Limit 200MB per file

Browse files

APPENDIX C

```
# -*- coding: utf-8 -*-
"""
Created on Sun Aug 21 12:31:17 2022

@author: thelu
"""

import pandas as pd
import argparse
import itertools
from itertools import permutations
import math
import streamlit as st
import numpy as np
from scipy.optimize import linear_sum_assignment
from munkres import Munkres
import sys

applicants = pd.read_csv ('D:\Applicants.csv')
jobs = pd.read_csv ('D:\Jobs.csv')

my_df_applicants=           applicants.pivot(index='skill
name',columns= 'applicant name',values = 'grade')
#print(my_df_applicants)

my_new_df_app = my_df_applicants.to_dict('dict')
#print(my_new_df_app)

my_df_jobs = jobs.pivot(index='skill name',columns ='job
name', values='grade')
#print(my_df_jobs)

my_new_df_jobs = my_df_jobs.to_dict('dict')
#print(my_new_df_jobs)

grade_job = my_new_df_jobs['Manager']['Learning']
#print(grade_job)

grade_applicant = my_new_df_app['Josh']['Communication']
#print(grade_applicant)

unique_applicants = set(my_new_df_app)
unique_applist=list(unique_applicants)
#print(unique_applist)
```

```
    unique_jobs = set(my_new_df_jobs)
    unique_joblist=list(unique_jobs)
    #print(unique_joblist)

    uniq_app_perm =  list(permutations(unique_applist))
    #print(uniq_app_perm)

    lst=[]
    for i in uniq_app_perm:
        lst.append(tuple(zip(i,unique_joblist)))
    #print(lst)

    def fitting(lst,my_new_df_app,my_new_df_jobs):
        res_val=0.0
        for comp in my_new_df_app[lst[0]]:
                        mult  =  my_new_df_app[lst[0]][comp]*
my_new_df_jobs[lst[1]][comp]
            if math.isnan(mult):
                mult=0.0
            res_val=res_val + mult
        return res_val


    def utility(lst,my_new_df_app,my_new_df_jobs):
        res_util=0.0
        for comb in lst:
                                        res_util=res_util     +
fitting(comb,my_new_df_app,my_new_df_jobs)
        return res_util

    #print(utility)

    def naive_approach(lst,my_new_df_app,my_new_df_jobs):
        ress=0.0
        for l in lst:
            val=utility(l,my_new_df_app,my_new_df_jobs)
            if val > ress:
                ress=val
                ress_tpl=l
        return [ress,ress_tpl]

    print(naive_approach(lst,my_new_df_app,my_new_df_jobs))
```

REFERENCES:

1.Blacksmith N, Willford JC, Behrend TS. “Technology in the employment interview: A meta-analysis and future research agenda”. Personnel Assessment and Decisions. 2016;2(1):2

2.Ferràs-Hernández X. ”The future of management in a world of electronic brains”. Journal of Management Inquiry. 2018 Apr;27(2):260-3.

3.Horbach J, Rammer C. “Labor shortage and innovation”. ZEW-Centre for European Economic Research Discussion Paper. 2020(20-009).

4.Li, X., & Song, Z. “Recruitment, job search and job choice: An integrated literature review” 2018 .

5. Matsunaga M. “How to Factor-Analyze Your Data Right: Do's, Don'ts, and How-To's”. International journal of psychological research. 2010;3(1):97-110.

6. Miller, B., & Ranum, D. (2013). “Problem solving with algorithms and data structures”

7. Newman DT, Fast NJ, Harmon DJ. “When eliminating bias isn’t fair: Algorithmic reductionism and procedural justice in human resource decisions”. Organizational Behavior and Human Decision Processes. 2020 Sep 1;160:149-67.

8. Reeve CL, Schultz L. “Job-seeker reactions to selection process information in job ads”. International Journal of Selection and Assessment. 2004 Dec;12(4):343-55.

9. Russell SJ. “Artificial intelligence a modern approach”. Pearson Education Inc.; 2010.

10. Stone, D. Deadrick, L., Lukaszewski, K. M., & Johnson, R. (2015). “The influence of technology on the future of human resource management”.Human resource management review, 25(2), 216-231.

11.https://www.statista.com/statistics/283771/monthly-job-vacancies-in-the-united-kingdom-uk/#:~:text=The%20number%20of%20job%20vacancies,when%20compared%20with%20January%202021.

12.https://hrexecutive.com/astounding-number-of-workers-looking-for-new-jobs-whats-hrs-move/

13.https://globaledge.msu.edu/blog/post/56844/global-manufacturing-labor-shortages

14.https://www.bls.gov/opub/ted/2021/quits-rate-6-6-percent-in-accommodation-and-food-services-in-september-2021.htm

15.https://www.cnbc.com/video/2021/10/15/why-rich-countries-are-so-dependent-on-migrant-workers.html