Forming Software Development Team: Machine-Learning Approach

Zeina Tanbour, Doha Khudarieh, Hadeel Abuodeh, Amjad Hawash*
Computer & Information Systems Dept.
An-Najah National University
Nablus, Palestine, P.O. Box:7
zanbour.ttg@gmail.com, dohak181@gmail.com, hadeelabuodeh.1111@gmail.com, amjad@najah.edu

***Abstract*—Software development team formation is a task done by skilled persons who have enough experience in mapping crew members to project tasks. Choosing software development team members according to their experience and within the amount of available budget/time is a vital task. However, the availability of a tool to suggest the best team members to the different tasks in some projects will definitely help project managers in their selections. This work is related to suggesting the most suitable skilled software development professionals to projects tasks based on some machine learning technique (Random Forest Classifier). The project manager just feeds the tool with the required tasks, and the latter suggests a ranked list of the most suitable professionals that fit each task which reflects positively on the team formation process. The experimental results conducted at the end of the work reflect the improvement of software development team formation gained comparable with the ordinary, self-experience-based one.**

***Index Terms*—Software Development, Machine Learning, Members Selection, Members-Tasks Mapping.**

I. Introduction

The success of any business has several factors such as a good plan, reasonable motivation, estimating resources, and the proper selection of team members. The suitable selection of members is considered a key factor if not the most in the project success journey [1]. Software development is a leading business these days because it provides an important solution for several domains. It promotes businesses, improves sales and services, provides direct communication with customers, increases customers engagement, and helps marketing businesses [2].

However, the improvement of some businesses depends mainly on the success of software solutions (tools) involvement in that business. In order for the software to be successful, assembling software development team members is considered a vital factor. It is the software project managers [1] role in mapping the project members with the tasks involved in the software development project. The proper selection of team members improves its outcomes [3].

Usually, when starting a software project, the project manager needs enough data and pays huge effort and time to choose the right members for the project tasks [4], [5]. The wrong selection of team members leads to problems in quality and productivity. Scalability in the team is also vital and needs some consideration.

From another point of view, globalization is considered to drive competitiveness in a lot of disciplines in which geographical borders have no impact on doing global activities. Here comes the software development outsourcing in front of businesses that needs no direct contact between team members. Software development project managers are able to form teams members and distribute tasks remotely. Bringing domain knowledge, focusing on core experts, meeting customers' expectations, and reducing tasks are all some examples of outsourcing benefits [6].

Despite the benefits of software development outsourcing, forming the proper team is a challenging factor with respect to software development contractors [7]. Automation the process of team formation or relying on an expert system in mapping project tasks with the most suitable members is a key factor in forming a dependable and successful software project team.

Our contribution to this work is related to developing a web-based tool (employed a machine learning techniques) that makes it easier for the project managers to better form software development crews by specifying the set of needed tasks during the project's lifetime and mapping these tasks with the most suitable available experts all ranked with respect to the degree of their suitability per task.

The idea is related to developing a machine learning technique able to better map a given software development project tasks with the most suitable software development experts available in the tool database. To do this, a machine learning component is developed and fed with sample data taken from a free data source [2] in order to train the machine. Around 600 experts' information related to software development persons with enough information about their selections in the software development projects available in the database are all used as training data for the machine.

Machine learning techniques can be achieved by several models. In our work, we tested 3 different models to build up the software team formation model: **Decision Tree Classifier**, **Logistic Regression** and **Random Forest Classifier**. We tested and calculated the accuracy percentage for each model in order to choose the better one as in Table I. From

*Corresponding Author: amjad@najah.edu

[1]From now and on, the terms **project managers** and **software contractors** will be used interchangeably. The same for the terms **team members** and **software development experts**.

[2]https://data.world/

these models, our selection goes with the *Random Forest Classifier*.

TABLE I
ACCURACY VALUES FOR THE THREE CLASSIFIERS.

Model	Accuracy
Decision Tree Classifier	76.92%
Logistic Regression	83.02%
Random Forest Classifier	83.51%

The results of the conducted experimental tests show an improvement in the process of matching between crew members and the list of project tasks. The amounts of mapping correctness between members and the list of examined tasks promise a real improvement over the manual mapping. 5 software development projects were involved in the experimental tests and we got more than 85% as a correctness percentage on average in the process of automating the mapping between members and tasks over the manual one.

The rest of this paper is organized as follows: Related Works is proposed in SectionII. Section III discusses the system architecture of the implemented work while Section V discusses the conducted experimental tests. Finally, Section VI concludes this paper.

II. RELATED WORK

The literature contains a lot of works related to software development team formation. Each of which concentrates on the methodologies and benefits of the proper forming of team members. However, few of these works are related to exploiting the machine learning techniques in the process of team members formation in order to reduce the possible risks of the wrong formation [8].

The work published in [9] discusses the problems of manual management and control of the software development process from team members' and customers' points of view. The authors of the works argue regards a set of project management tools and modules enhances the project management process during its lifecycle. However, the work published in [10] discusses the problems that face the software development process such as rescheduling, improper quality of works and exceeding of budgeted costs.

Utilizing machine learning techniques in assigning roles to employees is discussed in the work published in [11]. The work discusses the difficulties faced by administrators in assigning proper employees to suitable tasks. The main issue of the work is identifying the factors to be considered while allocating the task which ensures that important points are not missed when considering the human resource allocation in a team. This work is much related to ours in terms of the proper assignment of employees to tasks by machine learning techniques. However, our contribution is much directed to software project crew formation.

Reducing the duration and cost of projects is the most important aim for all organizations. Starting from this goal, the work of [12] considers using Genetic Algorithms to better manage the team members. Their work is related to solving many different software project scenarios and can perform structured studies on the influence the most important problem attributes have on the solutions.

Despite the improvement of assigning roles to persons introduced in the mentioned works, few of them are related to discussing the automation of assigning tasks to crew members in a software project management process as a form of team formation. Moreover, non of these works discuss the outsourcing issue and the importance of assigning team members to roles taking into account the experience of each team member. Our contribution to this work is related to these subjects and can be summarized as follows:

1) Quantifying the experiences of software development experts in a suitable way to be fed for the machine learning tool.
2) Relying on *Random Forest Classifier* model in analyzing experts' data.
3) Utilizing a machine learning technique in order to assign roles to experts.
4) Developing a website to be used by the software development manager to specify tasks to be fed to the machine learning tool in order to assign tasks to experts.

III. SYSTEM ARCHITECTURE

The following sections briefly describe the major components of the system that are depicted in Figure 1.

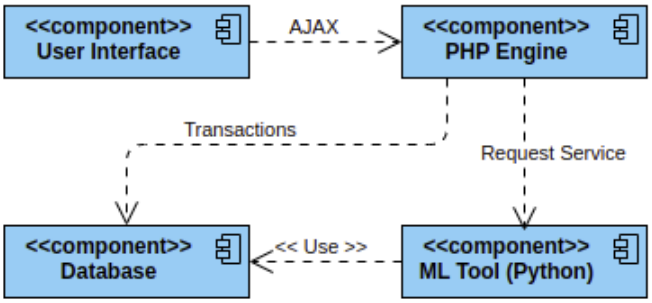

Fig. 1. System Components.

A. User Interface

The user interface is implemented as a web-based interaction in which the project manager supplies the system with the required project tasks to be sent to the *Machine Learning Tool* that executes necessary computations to map between the sent tasks and the most suitable experts. Upon the creation of these lists, the project manager is able to select the most suitable ones and when finished, the system sends invitation letters to the selected experts and upon their acceptance, the system saves all related data in a dedicated database. Of course, the same interface is used by experts to create and manage their profiles. The user interface harnesses both **DOM** (Document Object Model) and **AJAX** technologies in managing the data exchange between the front-end and back-end (*PHP Engine* component) of the system. The set of project tasks are sent from the front-end to the back-end and as a result, the list of suggested experts moves in backward after being filtered by the *Machine Learning Tool*. Figure 2 depicts this component.

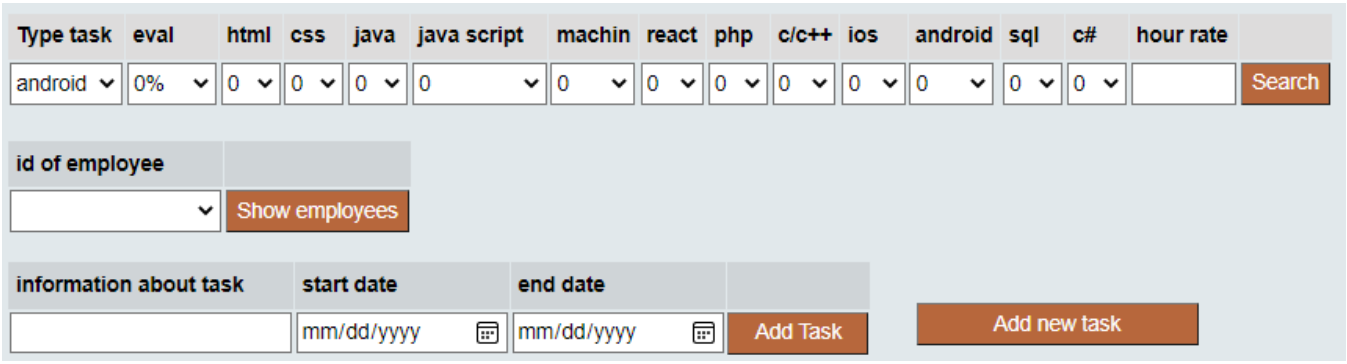

Fig. 2. The interface where the project manager determines the project tasks.

B. PHP Engine

This component is represented by a set of related PHP files written to handle the activities that take place in the back-end of the system. All activities related to handling the requests directed from the front-end go here. This component has a direct relation with both the *Machine Learning Tool* and the *Database* components. It communicates with the first component in a way to send the list of project tasks and get back the list of suggested experts. However, all database transactions related to all users in the system are also handled in the *PHP Engine* with the collaboration with the *Database* component.

C. Machine Learning Tool

This is the most important part of the system where all important activities take place. It receives its inputs (the list of project tasks) from the *PHP Engine* component and then executes a special Python code that exploits a set of libraries like **sklearn** to create the best mapping between the sent tasks and the set of experts after extracting them from the *Database* component.

The implementation of this component went in several steps. Firstly, we prepared a database of around 1000 computer development-related experts with enough data per each. The data is split into two parts: training and testing with 60% and 40% respectively. Secondly, the information of experts is prepared in two directions X and Y where X is the set of attributes for each expert, and Y is their IDs. Thirdly, We tested the training data subset with 3 mathematical models as in table I that previously introduced in the Introduction section. We relied on the *Random Forest Classifier* model that has the highest accuracy value among others. Fourthly, we saved the chosen model using the Python library *joblib* in order to be loaded later on in the real-time of the system. Definitely, we need the saved model to deal with the system database that accumulates its data from the new expert's sign-ups. So, we configured the model to take into account the new entries of the expert's database that create their accounts using the *System Interface* component. Figure 4 depicts the discussed process.

The Random Forest Classifier employed in this work takes the different properties of members and build up a

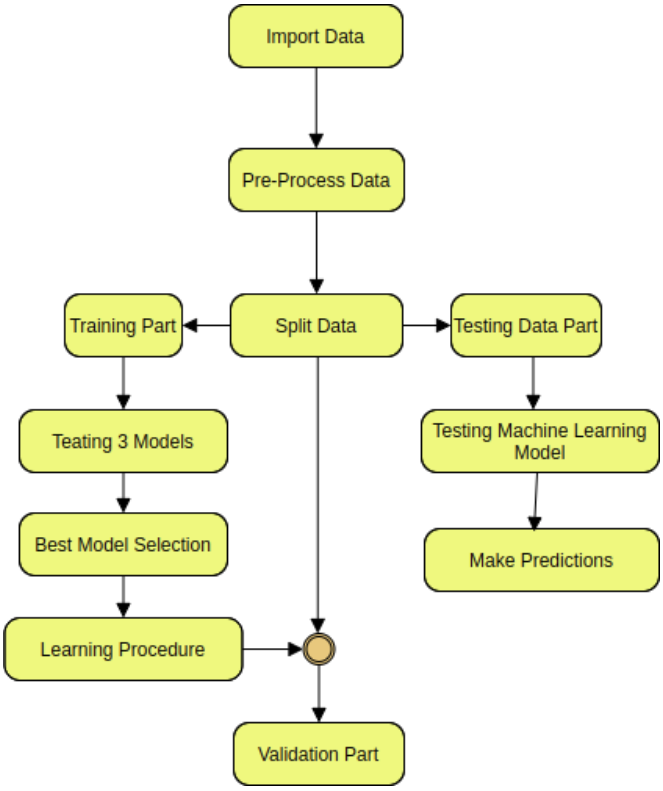

Fig. 3. Working Flow.

set of decision trees and merges them together to get a more accurate and stable predictions. Fig. 5 below depicts the process of generating two sample decision trees for the JavaScript feature in which the model builds up two branches for a given node by the selection of random values for the feature. The model ends up with a summation between all constructed decision trees to assign the accumulated values for the feature.

Fig. 6 depicts the differentiation between a set of members in terms of 3 sample features: (1) Weekly working hours (represented by background color), (2) member's evaluation (represented by font color), and (3) member's hour rate (represented by the font underline). The figure illustrates the mechanism of the decision tree filtering of members.

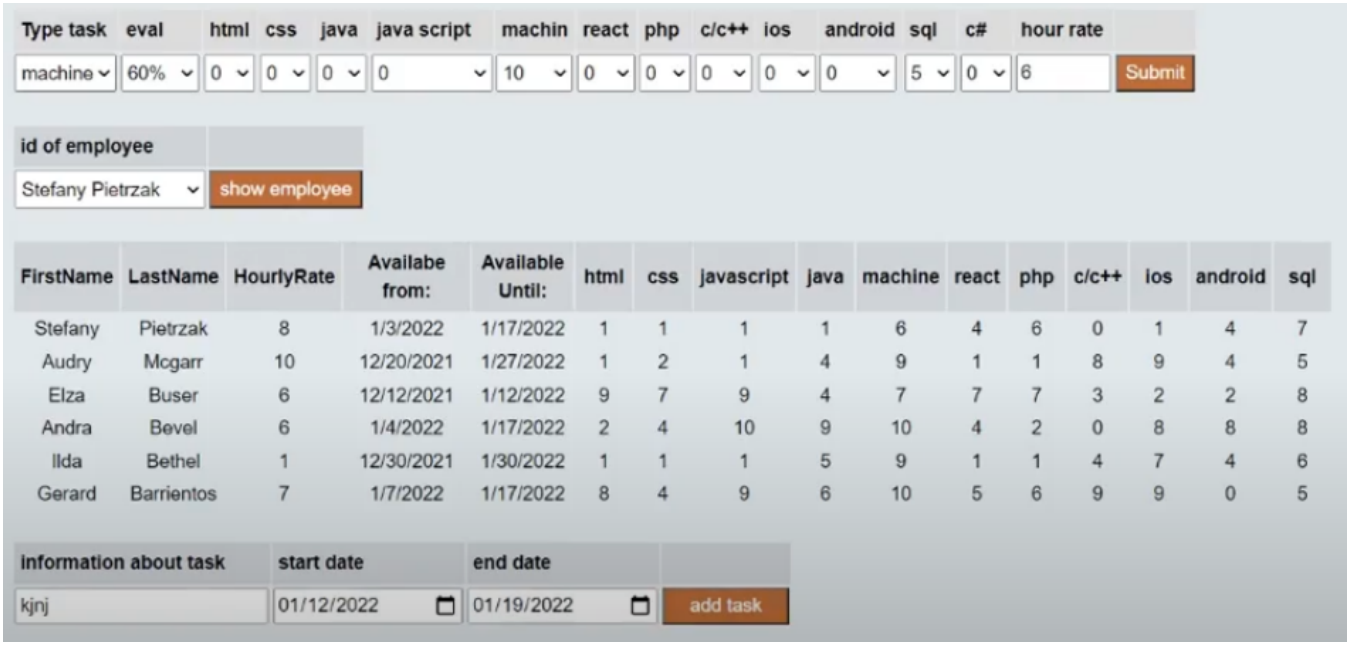

Fig. 4. Assigning tasks to experts.

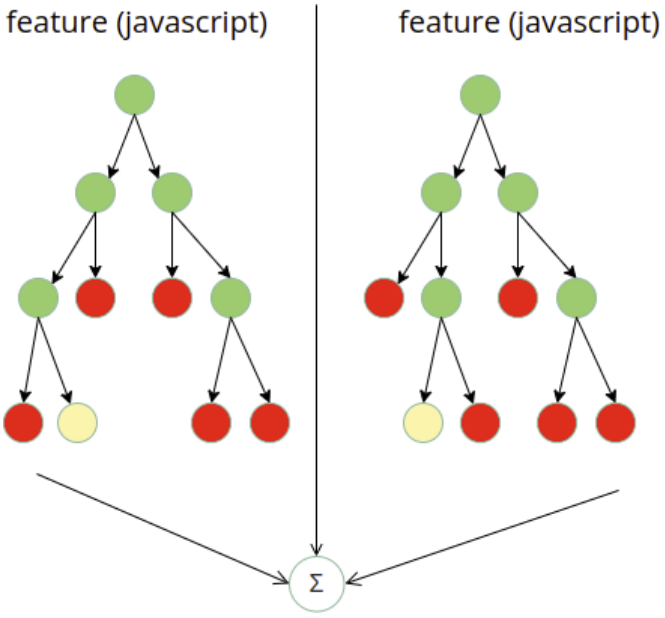

Fig. 5. Random Forest Classifier model for the JavaScript (sample feature).

However, each feature is given a special weight so that at the leaf of a tree there will be the summation of the preceding weights. If the filtering process outcomes with a set of members, the members' total weights will be used to generate a ranked list of members. The size of the list is predefined by the project manager.

D. Database

The constructed database contains personal and technical information about employees, the projects they are involved in, and the tasks per employee. Figure 7 below depicts the entity-relationship diagram for the entities included in the database. The *Expert* entity is used to save information about experts including their *Major* and *Hour Price*. The technical profiles experts contain extra information about the skills they have as well as their availability periods. These information can be taken from the *Experts* relations with both *Availability* and *Skills* entities. The first one contains information about the available times per expert so that the system is able to include/exclude a given expert from the searching results. While the second contains a list of experiences of experts. The *project* entity is used to save data related to the projects like start and end dates, information about the owner of the project, and the costs. The connectivity between experts and projects is done through the entity *Role* where information about the roles of all experts involved in a given project is saved in that entity.

IV. System in Action

The following sections are describing the activity of the previous components and how they collaborate to accomplish the mapping between experts and projects' roles.

A. Supplying Roles

The project manager supplies a list of the project's roles using the system interface component represented by an HTML form that is connected using AJAX with the *PHP Engine* component as in Figure 4 (the existence of zeros for some tasks means that the project does not contain these tasks). The list of roles is then transmitted from the *PHP Engine* to the *Machine Learning* component by preparing a suitable string containing all roles. Two PHP functions are then executed: **escapeshellcmd()** to escape the sent string in order to prevent the sell command from executing arbitrary commands and the **shellexec()** function to make the actual

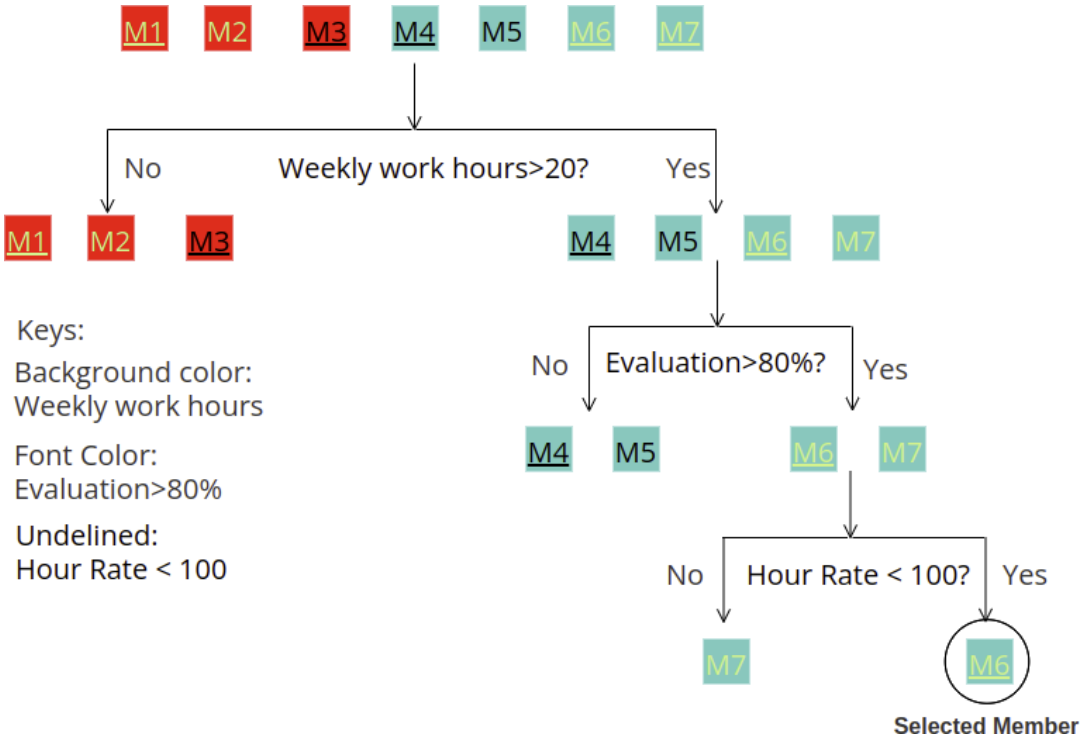

Fig. 6. Member selection example.

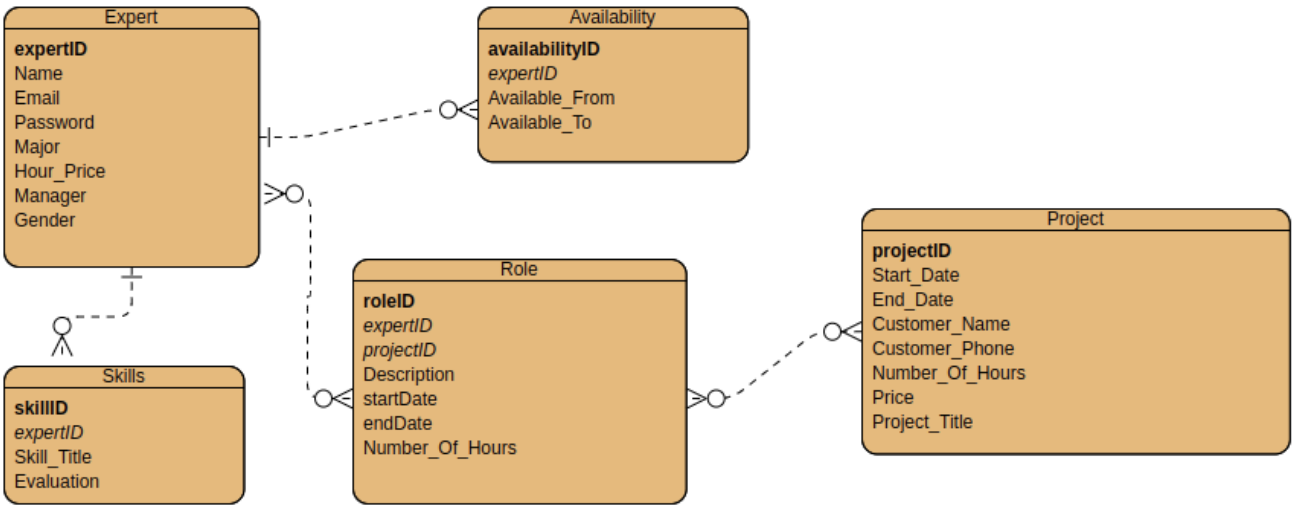

Fig. 7. Entity Relationship Diagram.

execution of the python code and to return the result of the execution (the list of the most suitable experts IDs). This process excludes those experts that are not available during the project lifetime.

B. Experts Selection

Upon the completion of the previous step, the project manager is listed with the most suitable experts per project task. The project manager then selects the most suitable ones and asks the system to send invitations for the selected ones. Upon the acceptance of proposed tasks (by the experts), task information is saved in the database.

C. Tasks acceptance/Rejection

when tasks proposals are sent to experts, they have the choice to accept or reject them. Upon acceptance, the project manager is notified and the task is mapped to the expert and saved in the database. In the case of rejection, the project manager is also notified in order to select another expert from the list already generated by the machine. Figure 8 depicts

the proposal sent to some expert and the action s/he might take.

Task Proposal

Project Title	Task Type	Start Date	End Date	Information	Decision
E-Health	Android	15/2/2022	20/6/2022	Writing Java code	yes ok

Fig. 8. An expert accepts some proposal.

D. Updating Experts Scores

After the completion of the project period, the project manager updates the scores of the employees who participated in the project. This process is important in order to adjust the machine selection process of experts for future projects.

Figure 9 depicts the sequence diagram of the system where the execution begins when the *Project Manager* executes the function *supplyProjectTasks()* that takes *project* title and the list of tasks *tasks[]* as parameters to be executed by the object *Web Page* that in tern executes the function *send()* on the *PHP* object with the same parameters. The *PHP* executes the function *supply* with the tasks as parameter using the object *Forming Team Machine* that contacts the dataase engine by executing the function *getEmployees()* that returns the list of filtered *employees[]* in which the the object *Forming Team Machine* uses this list of wmployees in the funciton *Execute()* in order to extract the employees who are the best fit each task. The execution of the system then returns back to the object *PHP* that accepts the list if ranked employees per task in the list *rankedEmployees[]* that in tern returned to the object *Web Page* in which notifies the *Project Manager* by executing the function *notify()*. After that the *Project Manager* selects a list of most fit employees per task by executing the function *selectEmployees()* on the object *Web Page* that in tern contacts the *PHP* object to save the data in the database by executing the function *saveData()* and then executes the function *notifyAllSelectedEmployees()* to notify those selected by the *Project Manager* about their involvement in the project being managed.

V. Experimental Tests

In order to measure the amount of correctness of the automatic mapping between software development project tasks and experts, we conducted an experimental test where we prepared a set of projects by collecting the requirements per each. After that, we asked a set of experienced project managers to prepare project plans by asking them to manually (according to their experience) assign each role in all projects to the most suitable experts. Then we fed our system with the same tasks and asked it to make the mapping between the roles and the experts. The tool then generates a raked list of suggested experts per role per project. These lists are displayed in the front-end part of the tool.

Table II shows the similarity ratio between the manual selection and the automated one. The values appear in the table are measured by comparing the list of experts suggested by the tool with the list suggested by the expert for every role per project. For example, the value 92% for Project 2 role HTML represents the correctness value between the list of members suggested by the tool compared with the list of members suggested by the expert. In order to have accurate results of these measures, we asked both the tool and the experts assign 100 members per role. So, the value 92% means that there are 92 similar members in both lists.

TABLE II
Correctness percentage per task.

Role	Project 1	Project 2	Project 3	Project 4	Project 5
HTML	90	92	87	91	93
CSS	85	84	89	90	92
Java	92	85	86	81	90
Javascript	78	85	89	85	86
Machine	92	80	85	86	87
React	95	92	94	93	91
PHP	86	84	87	89	88
C++	92	94	82	75	85
IOS	85	86	84	85	86
Android	90	82	84	86	82
SQL	74	78	95	78	85
C#	95	84	87	85	86
Average	87.8	85.5	87.4	85.3	87.6

TABLE III
Machine Learning vs. Genetic Algorithm Features Comparison.

Criteria	Machine Learning	Genetic Algorithm
Algorithm	Random Forest Classifier.	Genetic Algorithm.
Selection of Team Members	Selecting the most related person to the task.	Selecting the best person without the care of his/her tasks suitability.
Project Budget	Taken into account	Does not taken into account.
Constant Learning	There is learning criteria that depend on the selection of employees.	No learning phase, just searching for the best.

Table II depicts a comparison between our work and the work presented in [12] in terms of a set of features. The table indicates the selection criteria between the two methods.

VI. Conclusion

Forming a software project team is considered the essential success factor of software development projects. Assigning the most experienced experts to project roles is the major task in which project managers have to pay enough effort. However, it is very hard for project managers to form the perfect team in the case of outsourcing projects. The aim of this work is to make it easier for outsourcing project managers to better form project teams by harvesting machine learning tools. We computed the accuracy of three main classifiers: Decision Tree, Logistic Regression and Random Forest. Our decision went with the last one that has the most accuracy.

Comprehensive experiments were conducted in order to test the amount of correctness in the implemented model. We got more than 85% correctness as an average in the process of comparing the suggested members between the implemented tool and the participating experts.

Our future work is related to trying more classification techniques for comparison studies. We are planning to use Naïve Bayes, Stochastic Gradient Descent, and K-Nearest Neighbours.

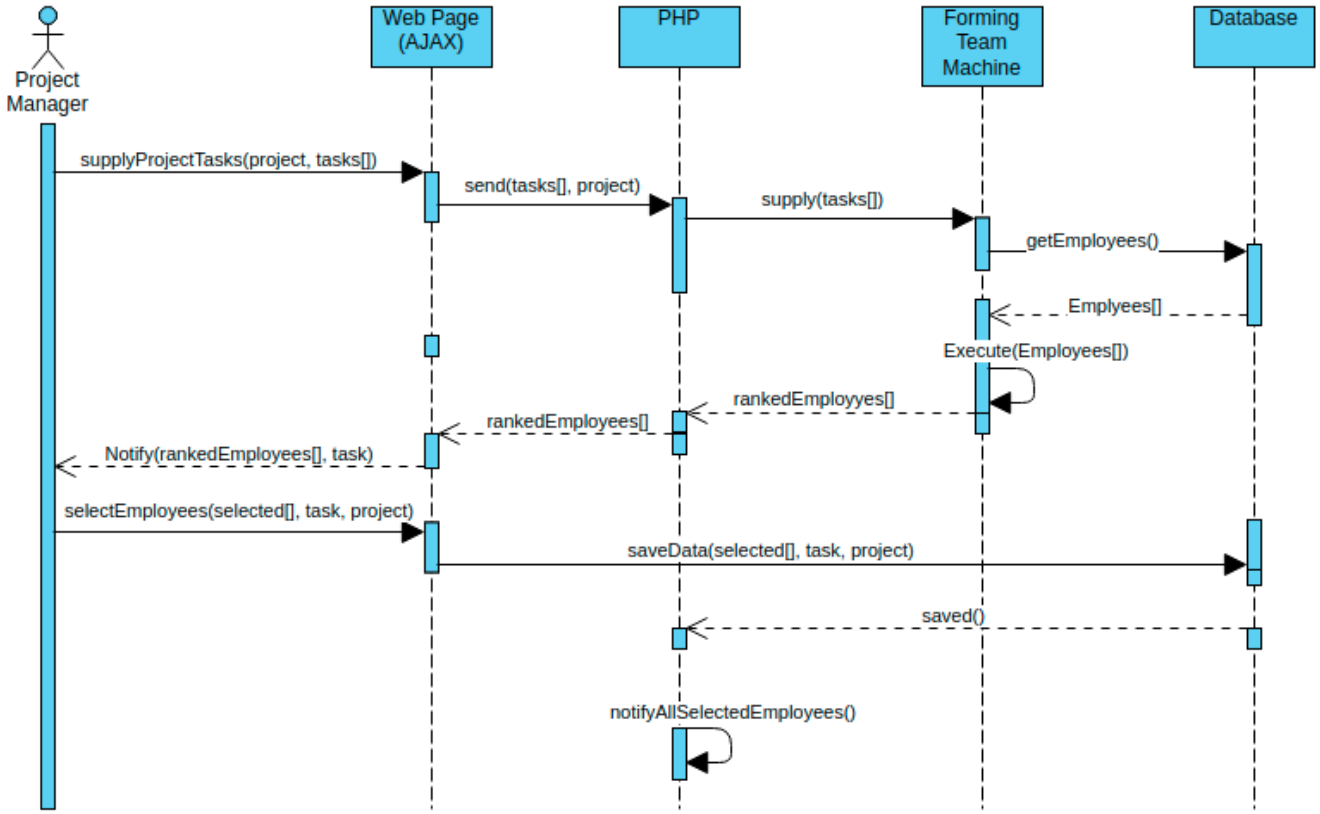

Fig. 9. Requesting a list of related experts sequence diagram.

REFERENCES

[1] F. Alamsjah, "Key success factors in implementing strategy: Middle-level managers' perspectives," *Procedia - Social and Behavioral Sciences*, vol. 24, pp. 1444–1450, 2011. The Proceedings of 7th International Strategic Management Conference.

[2] L. M. Maruping and S. Matook, "The evolution of software development orchestration: current state and an agenda for future research," *European Journal of Information Systems*, vol. 29, no. 5, pp. 443–457, 2020.

[3] A. Medina and A. J. Francis, "What are the characteristics that software development project team members associate with a good project manager?," *Project Management Journal*, vol. 46, no. 5, pp. 81–93, 2015.

[4] G. Sudhakar, "Understanding software development team performance," *Scientific Annals of the Alexandru Ioan Cuza University of Iasi : Economic Sciences Series*, 01 2010.

[5] A. B. Soomro, N. Salleh, E. Mendes, J. Grundy, G. Burch, and A. Nordin, "The effect of software engineers' personality traits on team climate and performance: A systematic literature review," *Information and Software Technology*, vol. 73, pp. 52–65, 2016.

[6] C. Harland, L. Knight, R. Lamming, and H. Walker, "Outsourcing: Assessing the risks and benefits for organisations, sectors and nations," *International Journal of Operations & Production Management*, vol. 25, pp. 831–850, 09 2005.

[7] W. Prashandi and A. Kirupananda, "Automation of team formation in software development projects in an enterprise: What needs to improve?," pp. 16–22, 11 2019.

[8] M. Z. M. H, M. N. Mahdi, M. S. Mohd Azmi, L. K. Cheng, A. Yusof, and A. R. Ahmad, "Software project management using machine learning technique - a review," in *2020 8th International Conference on Information Technology and Multimedia (ICIMU)*, pp. 363–370, 2020.

[9] S. Chandrashekar, B. Mayfield, and M. Samadzadeh, "Towards automating software project management," *International Journal of Project Management*, vol. 11, no. 1, pp. 29–38, 1993.

[10] O. Rostova, S. Shirokova, and N. Sokolitsyna, "Management of project for automation of investment control at industrial enterprise," *IOP Conference Series: Materials Science and Engineering*, vol. 497, p. 012017, apr 2019.

[11] R. Mamatha and K. Suma, "Role of machine learning in software project management," *Journal of Physics: Conference Series*, vol. 2040, p. 012038, oct 2021.

[12] E. Alba and J. Francisco Chicano, "Software project management with gas," *Information Sciences*, vol. 177, no. 11, pp. 2380–2401, 2007.