Prepare Students for Software Industry

A Case Study on an Agile Full Stack Project

José Carlos Metrôlho[1,2], Fernando Reinaldo Ribeiro[1,2], Rodrigo Batista[2]
[1]R&D Unit in Digital Services, Applications and Content
[2]Polytechnic Institute of Castelo Branco
Castelo Branco, Portugal
e-mail: metrolho@ipcb.pt, fribeiro@ipcb.pt, rodrigo.batista@ipcbcampus.pt

Paula Graça
DEETC of Instituto Superior de Engenharia de Lisboa
Instituto Politécnico de Lisboa
Lisbon, Portugal
e-mail: paula.graca@isel.pt

Abstract— **Reducing the gap between Software Engineering education and the needs in the software industry is a goal for Academia. Advancement in terms of cutting-edge technical skills and good soft skills preparation is the desired goal to shorten the onboarding in the labour market. Generally, in computer science or computer engineering courses, separate subjects exist to teach requirements engineering, analysis and design, coding, or validation. However, integrating all these phases normally requires experience in developing a complete project. The approach presented in this paper has involved the staff of a software company in collaboration with the staff of an academic Institution and resulted in a student's involvement in a full-stack software development project. The student was involved in an agile team composed of teachers and Information Technology (IT) professionals. Scrum framework was followed, and the product was developed using a low-code development platform. Results show that this agile and full stack approach allows students to develop cutting-edge technical and non-technical skills. The paper presents the approach, the achieved results, some lessons learned and some guidelines for the future.**

Keywords- agile software development; cognitive services; form recogniser; Scrum; software engineering; invoice.

I. INTRODUCTION

Nowadays, technology, namely software, is part of ordinary people's lives, and so there is a considerable demand for well-prepared professionals in this area of knowledge. Preparing professionals in these areas is not easy. If, on the one hand, they must have deep knowledge in specific technical subjects (databases, programming languages, requirements analysis, Web development, mobile development, etc.), it is also increasingly important that they have the skills to integrate or explore features in more complex systems. This broader view of specific software ecosystems requires a well-prepared new generation of engineers using new approaches and a more holistic experience of modern software development activity. These approaches can be enablers for accelerating development performance and obtaining better designed and high-quality software products.

In the software industry, many advances are also happening to speed up development. Examples of this are the low-code development platforms, which provide an abstraction layer that allows the developers to handle more of the inherent complexity of application development and simultaneously explore reuse and integrate different frameworks. They allow fast learning development processes, enable a more systemic view of software projects, and provide easy integration with other application endpoints. However, software engineering gains importance here because its inherent abstraction requires good development practices to be followed.

Another essential aspect nowadays is the great possibility of integration and interconnection between various systems. This makes it increasingly important that the new generation of IT professionals knows the services available and what mechanisms to use to integrate them into their applications. This holistic knowledge can be acquired in theory, but nothing better than consolidating it through developing projects that use this integration and other technologies. Cloud service providers (Amazon Web Services, Microsoft® Azure Cloud Platform or Google Cloud Platform) are cases in point.

Full stack development has changed, with new areas and skill sets becoming important. In the works published in [1], [2], in addition to the traditional definition of full stack, new scope and challenges are presented in this development field. In [1], the authors argue that students are “in a better position concerning their employment opportunities if they possess hands-on skills on the entire spectrum of full stack technologies”. Also, the authors make clear that full stack does not mean “all” technologies and that “students should learn how to recognise fundamental problems to solve them with the appropriate conceptual tools using the corresponding technology of the day”. In this paper, we share a case study that aims to prepare students for this reality, still in the academic environment and in close collaboration with partners from the software development industry.

In turn, the job market needs more technically well-prepared graduates with good soft skills. Thus, preparing the new generation of engineers requires training not only in the technical subjects that are the knowledge base, but also the vision, and more holistic experience about the paths followed today by software companies and the soft skills. Tackling all these aspects can be achieved with strategies and case studies like the one presented in this article. The product developed in this case was an application for household accounting, automatically recognising data from existing invoices in digital format (pdf, photo) using cognitive services.

In this case study, an important fact was that a company that develops software for the international market was involved. The company defined the product/goal. A student from a higher education institution (academy), integrated into a distributed team, developed it, using Scrum [3] as a software development process. This combination of several contributions and developing a full stack project using an agile software development process allows the student to acquire the knowledge and preparation necessary for today's challenges in the modern competitive software development market. The main goal is to contribute to reduce the gap that sometimes exists between what is learned in academia and what is needed in the industry. In this paper, based on the experience observed in a successful case, we share some practices in the teaching of software engineering to best prepare students for software industry.

The remainder of this article is organised as follows. Section II presents a background and related work. In Section III, the case study is presented. In Section IV, results and discussion about lessons learned are presented. Finally, some conclusions are presented in Section V.

II. BACKGROUND

Developers often do not just play a single role in software development; they must be multifaceted, often taking on the role of designers, coders, and database specialists. Therefore, having this knowledge and multi-tasking skills is essential and allows the developer to use them to complete a project or software development independently. This is also an advantage because it will enable the developer to be more familiar with all stages of the development process, making cooperation inside and outside the team more optimised, and contributing to reducing software development costs. These professionals should be able to work both in Web and mobile platforms with also knowledge of design through the Web like Hyper Text Markup Language (HTML) and Cascading Style Sheets (CSS). In addition, they should be able to use software development tools and techniques that allow the development team to be at its highest level of productivity.

However, higher education institutions face a challenging task in preparing students to work proactively in these high-performance teams. Many approaches have been proposed to teach and learn Software Engineering subjects. Some attempt to motivate students to take a more active role in their training and provide them more realistic experiences by replicating the settings used in the software development profession. Project-based Learning (e.g., [4]), flipped classroom (e.g., [5]), and gamification (e.g., [6], [7]) are some of these strategies that are frequently used to teach Software Engineering. Some other strategies promote a closer involvement of software companies to reduce the gap between Software Engineering education and the needs and practice in the software industry. For instance, in the approach described in [8], the industry actively supervises software product development. Another approach is to create supplementary training programs that aid in the screening of qualified candidates, as presented in [9]. These approaches are essential for students because they provide real challenges, more realistic experiences, and recreating industry software development environments. Nevertheless, they are also crucial for companies. For them, networking with students and other corporate sponsors, building ties with faculty, and promoting their business and products among college students are some potential advantages.

From a different perspective, software engineering teaching has been adapting to new developments and trends, namely the agile methodologies. Frequently, teaching agile methodologies have focused on teaching a specific framework like Scrum (e.g., [10]–[12]) or Extreme Programming (e.g., [13][14]). A study on using Agile Methods in Software Engineering Education [15] concluded that using Agile practices would positively influence the teaching process, stimulating communication, good relationships among students, active team participation, and motivation for present and future learning.

Besides the good results obtained by several of these strategies, software engineering teaching and learning can still benefit from a more participative and closer involvement of software development companies in the training process. This can enable students to join distributed teams, enhance their non-technical skills, and engage themselves in the practices used in these companies.

III. THE CASE STUDY

A. *People/Team*

In this case, the agile team was composed of 6 members. This is following the recommendations of Scrum [3]. Regarding the role of each one: 1 member of the company acted as product owner; 2 members of the company (with vast experience in terms of development using the adopted platforms) acted as coaches/development technical support; 1 member was the student that acted as a developer; 2 teachers acted as Scrum masters and, for some tasks, as coaches (involved in documentation, timeline, etc.). In this process, the student, the central element of the approach, interacted with the other people. Besides getting support for the development of the project/product, he also gained experience in terms of teamwork (soft skills), realising the difficulties and aspects that are common in business projects of this type. The members' posture was demanding and methodical, continually adopting practices equal to what is done in the day-to-day business activity.

B. *Process*

Tools were adopted for this purpose to carry out the development process. Thus, Jira [16] was used to manage all stages so that details of the evolution of the project and its rhythm could be adequately monitored in an articulated manner. This choice requires everyone to follow good communication practices and compliance with activity logs and user stories in this case.

Figure 1 presents the timeline of one of the semesters, and in Figure 2, we can see the Jira interface with part of the product backlog (the content is in Portuguese because it was the language agreed by the team for it).

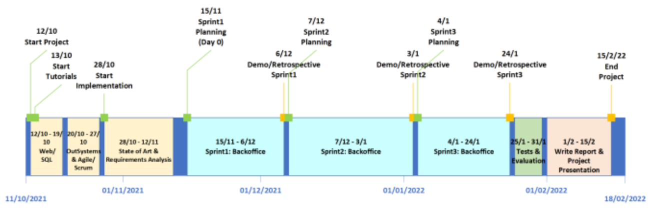
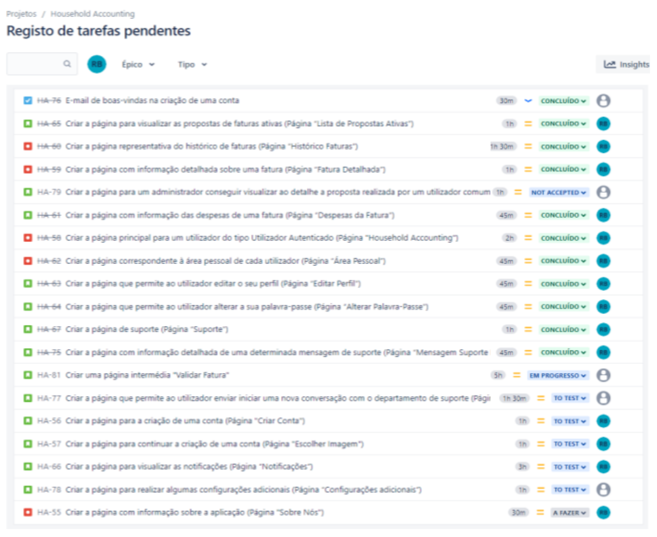


Figure 1. Timeline.


Projetos / Household Accounting

Registo de tarefas pendentes

Épico Tipo Insights

~~HA-76~~	E-mail de boas-vindas na criação de uma conta	30m	CONCLUÍDO
~~HA-65~~	Criar a página para visualizar as propostas de faturas ativas (Página "Lista de Propostas Ativas")	1h	CONCLUÍDO
~~HA-60~~	Criar a página representativa do histórico de faturas (Página "Histórico Faturas")	1h 30m	CONCLUÍDO
~~HA-59~~	Criar a página com informação detalhada sobre uma fatura (Página "Fatura Detalhada")	1h	CONCLUÍDO
HA-79	Criar a página para um administrador conseguir visualizar ao detalhe a proposta realizada por um utilizador comum	1h	NOT ACCEPTED
~~HA-61~~	Criar a página com informação das despesas de uma fatura (Página "Despesas da Fatura")	45m	CONCLUÍDO
~~HA-58~~	Criar a página principal para um utilizador do tipo Utilizador Autenticado (Página "Household Accounting")	2h	CONCLUÍDO
~~HA-62~~	Criar a página correspondente à área pessoal de cada utilizador (Página "Área Pessoal")	45m	CONCLUÍDO
~~HA-63~~	Criar a página que permite ao utilizador editar o seu perfil (Página "Editar Perfil")	45m	CONCLUÍDO
~~HA-64~~	Criar a página que permite ao utilizador alterar a sua palavra-passe (Página "Alterar Palavra-Passe")	45m	CONCLUÍDO
~~HA-67~~	Criar a página de suporte (Página "Suporte")	1h	CONCLUÍDO
~~HA-75~~	Criar a página com informação detalhada de uma determinada mensagem de suporte (Página "Mensagem Suporte	45m	CONCLUÍDO
HA-81	Criar uma página intermédia "Validar Fatura"	5h	EM PROGRESSO
HA-77	Criar a página que permite ao utilizador enviar iniciar uma nova conversação com o departamento de suporte (Pági	1h 30m	TO TEST
HA-56	Criar a página para a criação de uma conta (Página "Criar Conta")	1h	TO TEST
HA-57	Criar a página para continuar a criação de uma conta (Página "Escolher Imagem")	1h	TO TEST
HA-66	Criar a página para visualizar as notificações (Página "Notificações")	3h	TO TEST
HA-78	Criar a página para realizar algumas configurações adicionais (Página "Configurações adicionais")	1h	TO TEST
HA-55	Criar a página com informação sobre a aplicação (Página "Sobre Nós")	30m	A FAZER

Figure 2. View of backlog in Jira UI.

On the left side of the Jira interface of Figure 2, we can see the list of user stories and, on the right side, the corresponding status (finished, in progress, not accepted or to test).

Scrum's artefacts [3] were all met, such as having a product backlog, sprint backlog, etc. In addition, there were daily meetings between the student and his mentors and checkpoints to clear any impediments to progress. The sprints were 2-4 weeks long, but every week there was a meeting (weekly meeting) between all the team members to review the progress of the work. There were sprints for development, but there were also periods when the goal was to learn how to develop or optimise the project. For example, how to integrate Azure [17] cognitive services into the product under development. In addition, the definition of the sprint periods were not unrelated to the academic activity, which took place in parallel, so that the student could also be able to fulfil the academic requirements in his other subjects. Thus, there were different sprint periods as there were also different workloads.

C. Project

Since an agile approach was adopted, it followed the value [18]:

"*Working software over comprehensive documentation.*"

In terms of requirements documentation and modelling, the requirements were documented using user stories, and in addition, we used wireframes and the Entity-Relationship (ER) model. The team did not follow an extensive and deeper documentation approach because the student knew it from previous work in other curricular units. However, taking advantage of this knowledge, the student also represented the use cases and the ER model for the final report.

The research work was demanding for the student and the other members involved. New challenges were posed that required research, pre-experimentation, and analysis. For example, to implement the synchronism between the mobile application and the backend, it was necessary to analyse several patterns and adjust them to the concrete objective of this new product. The same happened in relation to security aspects of the application or the use of Azure cognitive services. In other words, the fact that it is an application with ambitious goals also posed interesting challenges to all team members. The product owner defined the initial requirements and documented them in the product backlog. For the user stories, the acceptance criteria were defined, which helped to design the test cases and thus contribute to a robust application.

Although the student had general knowledge about Artificial Intelligence (AI), it required him to be prepared on how to take advantage of the resources provided by Azure not only in terms of parameterisation and integration, but also in training to get the best performance. This is important because what was at stake in this challenge was to implement the functional requirements and user stories and obtain a final product with the highest possible accuracy in terms of automatic detection of fields of interest present in invoices.

Thus, the project involved research, development, software integration, application synchronisation (Web and mobile), security, agile Scrum framework, teamwork, and new tools (low-code platform, integration with cloud Azure, cognitive services, Jira, etc.). A detailed report of all phases and details of each aspect covered during the implementation process was also made. In the final stage of the project, acceptance tests were done to determine if the implemented features were useful and satisfied the users' needs.

This work covered many aspects of a software project, which could hardly be contemplated in a purely academic project. In addition to the technical-scientific coverage evidence, the agile methodology was chosen, and the fact that there was permanent communication between all its members was central. This leads us to verify in practice that one more value of the agile manifesto followed results in a successful path [18]:

"*Individuals and interactions over processes and tools*"

All these aspects mentioned above were considered, and the project includes documentation on user stories, database modelling, wireframes, and systems software architecture, among other valuable and necessary documentation.

Figure 3 shows the general architecture of the implemented system.

This work involved full-stack Web and mobile development using the OutSystems low-code platform [19]. This choice, teamwork, and adopting the agile framework (Scrum) allowed us to design from scratch, implement and test a complex and challenging software product during the normal period of a school year.

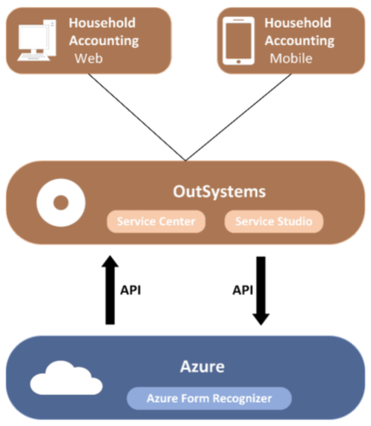


Figure 3. Systems Architecture.

D. Product

The recognition and automatic extraction of data from documents (invoices, receipts, etc.) are complex to implement and require various aspects to make it work successfully. With this project, we intended to apply mechanisms to recognise and extract data from invoices and store, organise and manage these data.

Using the OutSystems platform to develop the current project was a requirement from the company. The company proposed this product idea to develop a Web and mobile application using the OutSystems low-code platform, allowing users to manage their expenses in a digital format, independently of the users receiving the documents digitally or on paper. One of the characteristics of this platform is the speed of development and the integration with other necessary tools for the implementation of the objectives of this work (e.g., integration with Azure services). It allows to build and deploy full-stack Web and mobile applications [19].

The product owner proposed the product backlog. However, in each sprint review meeting, there were adjustments to the user stories. A sprint retrospective was always carried out so that the improvement process was constant from sprint to the next sprint, fostering a continuous pace. This demonstrates to the student the importance of the 3rd and 4th values of [18]:

"Customer collaboration over contract negotiation"

and

"Responding to change over following a plan".

The final product was developed on time, and all goals were achieved. In other words, at the end of the project, the resulting product was an application (Web and mobile) that was developed in OutSystems with the integration of Azure cognitive services (Azure form recogniser [20]) that allows (among other functionalities) the user to:

- Register invoices automatically.
- Process invoices (recognise and extract) data from pdf or an image captured by a smartphone.
- See spending statistics of a specific type and period.

The mobile application was implemented to be used even when it is offline. Because of that, mechanisms to synchronise both applications (Web and mobile) were implemented.

Figures 4 and 5 show the final layout of both the Web portal User Interface (UI) and one of the mobile applications UI.

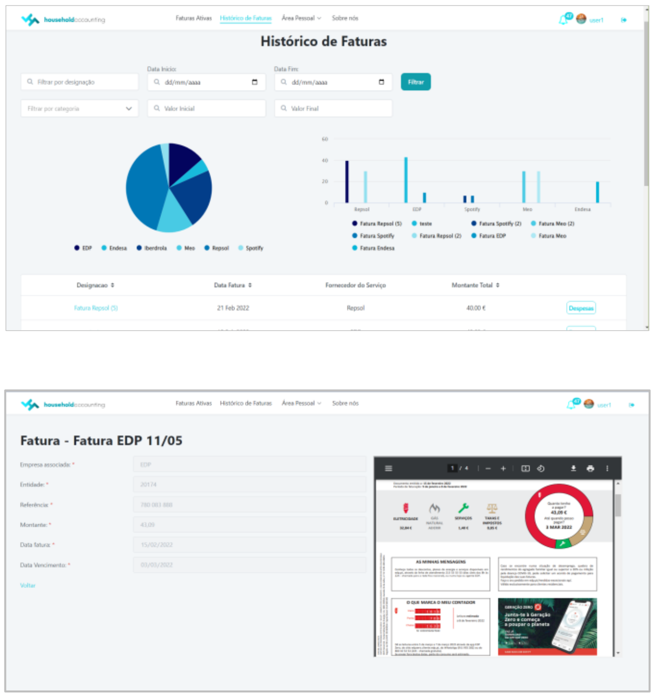


Figure 4. Examples of portal Web UIs (On top: Invoice's historic view; Bellow: Output of automatic processed invoice view).

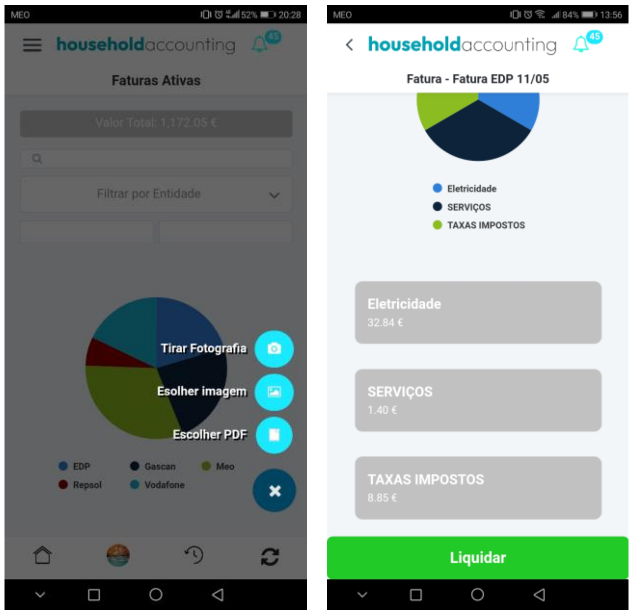


Figure 5. Examples of mobile app UIs (Left: select new invoice; Right: expenses).

In Figure 4 (top), we can see a dashboard to consult invoice's historic, with filters, charts presenting the collected data of different service providers (Telecommunications companies, electricity suppliers, etc.) and the list of stored invoices by designation, date, service provider, and monetary value. In Figure 4 (bellow), we can see the result of an automatically processed invoice view, presenting the invoice's file and the automatically captured fields.

Invoice templates can be configured in the administration portal for different service providers. The training and configuration of the recognition algorithms, using Azure cognitive services, can be configured through a dedicated Web portal.

The company's representatives validated the visual aspects (UI/User Experience (UX)), the synchronisation between the Web and the mobile applications, and security issues. The performance results obtained with the recognition of invoices were also analysed and improved.

IV. Results and lessons learned

According to the opinion of all those involved, the result of the project was very positive. In addition to completing the entire system within the planned period of 2 semesters, a high-quality software product was developed (all requirements implemented and good acceptance from potential external end-users). In other words, in addition to providing all the intended features identified throughout the project, the developed software also performs with good performance results. After several tests with invoices from various service providers, the performance was excellent in all cases. As in any of these cases, the better organised the information on the invoice is (input), the easier it will be to train the system and, obviously, the better the accuracy of the data obtained (output). In the tests carried out, in most cases, all data was recognised automatically from the original pdf invoices received by email from the service providers (e.g., gas company, energy company or telecommunications providers). A lower accuracy rate was achieved if the invoices were digitised using the smartphone camera (even so, in the performed experiments, at least 46.5% of the fields were well recognised, and the user manually entered the remaining fields). After having a first version of the system available (Web and mobile), several potential users were asked to install and use the application and to respond to a survey. The survey included 14 questions. Twelve questions were answered using the Likert Scale (1–5), and one question asked for a numerical answer. The other question was an optional free response question where respondents could include any information. The results obtained at this stage serve three crucial goals: 1) to provide a better insight into the platform, which may identify novel issues/problems to consider; 2) to obtain initial feedback on potential users' acceptance and perception of the platform's key features and 3) to evaluate the usefulness of the proposed system. Twelve users completed the survey. To exclude the outliers, the survey with the best evaluation and the survey with the worst evaluation were excluded. This resulted in 10 valid answered questionnaires.

The analysis of the responses shows that:

- 90% of respondents rated the application as useful or very useful.
- 80% of respondents rated the application as easy or very easy to use.
- All the respondents were satisfied or very satisfied with the automatic reading of invoice information.
- Importing invoices in pdf format was considered very important by 80% of respondents. The import of invoices from a photo was considered important or very important by 60% of the respondents.
- In the open answer question, it was possible to obtain some feedback on usability improvements and the reporting of some bugs.

In terms of lessons learned, this approach requires a dedication of at least one h/week (average) from the teachers and the company's members. In the case of the mentor, this period was longer due to all the daily meetings. The dedication paid off because the result (resulting product, preparation of the student (technical and non-technical skills)) was very positive. The fact that everyone was engaged in developing a comprehensive project that involved all stages and components of the proposed architecture was challenging, motivating and clearly beneficial for all parties, that is, for teachers, students, and staff involved from the partner company.

V. Conclusions

The presented case study shares an agile approach to preparing students for the job market regarding Software Engineering (SE) practices in the context of final year projects (in the 5th and 6th semesters of the course curricular plan to complete the degree). This approach reduces the gap between SE education and practice in the software industry. The student was involved in a distributed team with teachers and IT professionals from a software house to develop a product that demanded full stack development and agile best practices. The case study presented illustrates the work methodology and the resulting product. In other words, the paper described people, the process, the project, and the product.

These industry-academia partnerships helps students become better and quickly prepared to work in high-performing teams. They raise students´ employment opportunities by preparing them in cutting-edge fields and improving their soft skills to have better performance in software development teams. These partnerships are also advantageous for the other involved partners. Hiring qualified human resources is good for the companies, as well as for the participating higher education institutions (contributes to improve their employability rate).

Acknowledgment

We thank the members of the company Do iT Lean who were also involved in this case study providing technical support.

REFERENCES

[1] A. Taivalsaari, T. Mikkonen, C. Pautasso, and K. Systä, "Full Stack Is Not What It Used to Be," in *International Conference on Web Engineering,* 2021, pp. 363–371.

[2] A. R. C. Akshat Dalmia, "The New Era of Full Stack Development," *Int. J. Eng. Res. Technol.*, vol. 9, no. 4, pp. 7–11, 2020, doi: 10.17577/IJERTV9IS040016.

[3] K. Schwaber and J. Sutherland, "The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game.," 2016. https://www.scrum.org (accessed Jul. 20, 2022).

[4] R. Brungel, J. Ruckert, and C. M. Friedrich, "Project-Based Learning in a Machine Learning Course with Differentiated Industrial Projects for Various Computer Science Master Programs," in *2020 IEEE 32nd Conference on Software Engineering Education and Training, CSEE and T 2020*, 2020, pp. 50–54, doi: 10.1109/CSEET49119.2020.9206229.

[5] L. Gren, "A Flipped Classroom Approach to Teaching Empirical Software Engineering," *IEEE Trans. Educ.*, vol. 63, no. 3, pp. 155–163, 2020, doi: 10.1109/TE.2019.2960264.

[6] P. Rodrigues, M. Souza, and E. Figueiredo, "Games and gamification in software engineering education: A survey with educators," in *2018 IEEE Frontiers in Education Conference,* 2018, vol. 2018-Octob, pp. 1–9, doi: 10.1109/FIE.2018.8658524.

[7] R. Malhotra, M. Massoudi, and R. Jindal, "An innovative approach: Coupling project-based learning and game-based learning approach in teaching software engineering course," in *Proceedings of 2020 IEEE International Conference on Technology, Engineering, Management for Societal Impact Using Marketing, Entrepreneurship and Talent, TEMSMET 2020*, 2020, pp. 1–5, doi: 10.1109/TEMSMET51618.2020.9557522.

[8] W. E. Wong, "Industry Involvement in an Undergraduate Software Engineering Project Course: Everybody Wins," in *120th ASEE Annual Conference and Exposition*, 2013, pp. 23.742.1-23.742.12, doi: 10.18260/1-2--19756.

[9] E. Tuzun, H. Erdogmus, and I. G. Ozbilgin, "Are Computer Science and Engineering Graduates Ready for the Software Industry? Experiences from an Industrial Student Training Program," in *2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 2018, pp. 68–77.

[10] A. Heberle, R. Neumann, I. Stengel, and S. Regier, "Teaching agile principles and software engineering concepts through real-life projects," in *2018 IEEE Global Engineering Education Conference (EDUCON)*, 2018, pp. 1723–1728, doi: 10.1109/EDUCON.2018.8363442.

[11] G. Wedemann, "Scrum as a Method of Teaching Software Architecture," in *Proceedings of the 3rd European Conference of Software Engineering Education*, 2018, pp. 108–112, doi: 10.1145/3209087.3209096.

[12] I. Bosnić, F. Ciccozzi, I. Čavrak, E. Di Nitto, J. Feljan, and R. Mirandola, "Introducing SCRUM into a Distributed Software Development Course," in *Proceedings of the 2015 European Conference on Software Architecture Workshops*, 2015, pp. 1–8, doi: 10.1145/2797433.2797469.

[13] J. J. Chen and M. M. Wu, "Integrating extreme programming with software engineering education," in *38th International Convention on Information and Communication Technology, Electronics and Microelectronics*, 2015, pp. 577–582, doi: 10.1109/MIPRO.2015.7160338.

[14] B. S. Akpolat and W. Slany, "Enhancing software engineering student team engagement in a high-intensity extreme programming course using gamification," in *27th Conference on Software Engineering Education and Training*, 2014, pp. 149–153, doi: 10.1109/CSEET.2014.6816792.

[15] S. Al-Ratrout, "Impact of using Agile Methods in Software Engineering Education: A Case Study," in *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2019, pp. 1986–1991, doi: 10.1109/CoDIT.2019.8820377.

[16] ATLASSIAN, "Jira Software." https://www.atlassian.com/br/software/jira (accessed Oct. 13, 2022).

[17] Microsoft Corporation, "AZURE. INVENT WITH PURPOSE. Learn, connect, and explore." https://azure.microsoft.com/en-us/ (accessed Oct. 13, 2022).

[18] "Manifesto for Agile Software Development," 2001. https://agilemanifesto.org (accessed Jul. 20, 2022).

[19] OutSystems, "OutSystems Developers: Develop more. Ship more. Get more done." https://www.outsystems.com/developers/ (accessed Jul. 26, 2022).

[20] Microsoft Corporation, "Azure Form Recognizer." https://azure.microsoft.com/en-us/services/form-recognizer (accessed Jul. 20, 2022).