Preparing Students for the Software Industry New Demands

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

Diogo Pacheco
Do iT Lean
Leiria, Portugal
e-mail: diogo.pacheco@doitlean.com

***Abstract*—A solid preparation in terms of soft skills and state-of-the-art technical skills in Software Engineering (SE) is a goal for the academy. It also contributes to reducing the gap between Software Engineering education and the software industry's new demands. Generally, in computer science or computer engineering courses, there are separate subjects to teach requirements engineering, analysis, design, coding, or validation. However, integrating all these subjects usually requires experience in developing a complete project. This article describes aspects of an active and collaborative learning approach involving academia and industry actors. The approach presented in this article involved staff from a software company in collaboration with staff from an academic institution. It resulted in a student being involved in an entire software development project. The student was involved in an agile team of faculty and Information Technology (IT) professionals. The Scrum agile framework was followed, and the product was developed using a Low-code development platform. This article presents the approach, details of the project design and implementation, results achieved, lessons learned, and guidelines for the future. The results show that this agile, full-stack approach allows students to develop cutting-edge technical and non-technical skills.**

Keywords- agile software development; cognitive services; form recognizer; Scrum; software engineering; software industry.

I. INTRODUCTION

Preparing students' performance and technical skills for the software industry's new demands is challenging. If, on the one hand, they must have deep knowledge of specific technical subjects (databases, programming languages, requirements analysis, Web development, mobile development.), 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 accelerate development performance and obtain 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, SE gains importance here because its inherent abstraction requires following good development practices.

Another essential aspect nowadays is the high possibility of integration and interconnection between various systems. This aspect makes it increasingly crucial for new IT professionals to know the services available and what mechanisms to integrate them into their applications. This holistic knowledge can be acquired in theory, but nothing is 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.

In a previous article [1], we share a case study that aims to prepare students for this reality, still in the academic environment, and close collaboration with partners from the software development industry. We described an overview of our approach to prepare better students for the labor market in collaboration with software industry members. In this work, we also approached the importance of full-stack development and preparing students for the entire spectrum of full-stack technologies. 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 soft skills. These aspects can be tackled 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 recognizing data from existing invoices in digital format (pdf, photo) using cognitive services.

In this article, we extend our description of the approach, including more detailed technical issues considered in this

case study, to foster an embracing and holistic preparation of the students in this learning ecosystem. To achieve this goal, we went beyond the usual development of an academic project. We included issues about security, UI/UX, performance, data synchronization, or integration with external cognitive services.

In this case study, an important fact was that a company that develops software for the international market was involved. This collaboration helps to prepare students for the software industry demands, and with a holistic point of view about SE, involving several aspects that are important to foster effective implementations of active learning (allowing students to acquire knowledge in a practical way) and collaborative learning (creating environments that emulate as much as possible future work environments, whereas software engineers will work together to produce software products). The company defined the product/goal. A student from a higher education institution (academy), integrated into a distributed team, developed it, using Scrum [2] 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 reducing the gap that sometimes exists between what is learned in academia and what is needed in the industry. In this article, based on the experience observed in a successful case, we share an approach to prepare students for the software industry competently. The results were very positive, both in technical terms (software product and scope of the student's technical skills) and SE terms (student preparation in terms of useful soft skills for the labor market).

The remainder of this article is organized 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 optimized and contributing to reducing software development costs. These professionals should be able to work both on 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 challenges in preparing students to work proactively in these high-performance teams. Many approaches have been proposed to teach and learn SE subjects. Some attempt to motivate students to take a more active role in their training and provide them with more realistic experiences by replicating the settings used in the software development profession has been made. Project-based Learning (e.g., [3]), flipped classroom (e.g., [4]), and gamification (e.g., [5], [6]) are some of these strategies that are frequently used to teach SE. Some other strategies promote a closer involvement of software companies to reduce the gap between SE education and the needs and practice in the software industry. For instance, in the approach described in [7], the industry actively supervises software product development. Another approach is to create additional training programs that aid in screening qualified candidates, as presented in [8]. 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, potential advantages are networking with students and other corporate sponsors, building ties with faculty, and promoting their business and products among college students.

From a different perspective, SE teaching has adapted to new developments and trends, namely agile methodologies. Frequently, teaching agile methodologies has focused on teaching a specific framework like Scrum (e.g., [9]–[11]) or Extreme Programming (e.g., [12], [13]). A study on using Agile Methods in SE Education [14] concluded that using Agile practices would positively influence the teaching process, stimulating communication, good student relationships, active team participation, and motivation for present and future learning.

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

III. THE CASE STUDY

A. Context

The work described here was done by a student who attended a computer engineering course's third curricular year (fifth and sixth semesters). In parallel, the student had to attend other curricular units, which are part of the course. The final evaluation of the work done by the student, in academic terms and for diploma purposes, was done by a jury composed of a supervisor (one of those who followed the work) and two teachers external to the project. As for the role of the others involved in this approach, information concerning this is presented in section C.

B. The Elicitation Stage

At the beginning of the course/project, the start of the first semester, the student was guided to be prepared for research and analysis skills. This guidance happened briefly before Sprint 1 (as presented in the timeline in section D). He was led to research and analyze the results of a search in scientific

citation libraries and online sources related to the focus of the application/product goals. This way, the student experimented with the demands and techniques of searching and citing relevant information, and he was challenged and guided to analyze the results. In this case study, state of the art was built based on the following two crucial sources of information:

Systematic Review of Scientific Publications

For analysis of the related work, articles addressing automatic data extraction from invoices and receipts were studied. The Scopus platform was used as a data source. Scopus is one of the most complete databases in several areas and provides an advanced search that allows to configure search words in different fields such as titles, keywords, and full text, among others. It also allows the addition of logical operators: AND, OR, and NOT. This way was possible to access a significant number of scientific articles in computer science related to the theme of this project.

To collect the first sample of articles, the following keywords were defined for the Scopus search fields: "Extraction," "Recognition," "Invoice," "Invoice fields," "Algorithm," "Software," "Application," and "Program." The search was applied to the fields, title, keywords, and abstract: ("extraction" OR "recognition") AND ("Invoice*" OR "Invoice fields") AND (algorithm* OR software OR application OR program*).

Applying this query and the restriction to consider articles published after 2010 (inclusive) led to a total of seventy-one results.

After reading the title and abstract of each article, the articles that did not fit were eliminated, obtaining, in the end, a total of eleven articles.

With the full reading of the resulting eleven articles, they were further divided into two groups, mainly because the focus of some was not what was desired. In the end, of the eleven articles selected, four shared the same focus as the proposed project, and seven carried out only part of the proposal to be implemented. These seven articles that correspond only to a part of the implementation perform data extraction or comparisons between the algorithms used to do the extraction. That is, they only try to show various plausible algorithms, and afterward, the information is not allocated to any system.

In summary, we are left with four articles to analyze whose focus is equivalent to the application intended in this work.

Each of the four selected articles was analyzed and described considering the following criteria:

- Year of publication.
- Brief description of the objective of the work.
- What are the most common problems or scenarios intended to be solved by extracting data from invoices?
- What is the application's target audience (domestic, industrial, commercial)?
- What is the form of user interaction (web, mobile, web + mobile)?
- What is the detection algorithm (proprietary, how it learns, gives to various types of invoices, external API, Azure, among other considerations)?
- What is the result of the work (accuracy, types of invoices (several or just one), among other aspects)
- Main conclusions identified.

An analysis of the four resulting articles considering these criteria was then performed. This allowed all team members to clearly view related works and position the desired goals within those scopes.

Online APPs Overview

Also, the analysis of existing applications/systems whose purpose is identical to this work was carried out. The applications were found based on searches in the Google and Apple stores by applying filters such as keywords ("expense management", "expenses" and "financial control") or by selecting a particular category in which these applications fit (Ex. Finance). Here five applications were selected based on the following criteria: the platforms where it is possible to run these applications and the cost of acquiring them. Besides the five applications selected from the stores, three other applications were also considered available on the Internet but outside the stores. These applications were found by searching on the Google search engine using two strings: "capture data from invoice software" and "capture data from invoice."

All eight selected apps were analyzed and summarized. This, as done in the case of articles, allowed us to know better what products already exist for similar purposes and the most important features of related works.

Main Conclusions for the Design of the New Application

The four articles, the five applications selected from stores, and the three applications selected from additional searches allowed, at an early stage of the project, the identification of relevant functionalities, some of which were incorporated in the design and implementation of the Household Accounting project. In addition, it was possible to observe through the analysis of the article's different types of technologies and algorithms for invoice information recognition that could have been used to implement the new application and to understand better how necessary it is to have applications whose purpose to automate the reading of invoice information both at a business and personal level.

All this research work allowed the student, and the team, to have a better view of the state of the art. One of the main contributions of this project stage was to prepare the student for research and stimulate analysis and criticism skills. This allowed a clear and deep understanding of possible paths for the solution and opened a pathway for a better definition of the functional requirements. This led to a better definition of the functional and non-functional requirements defined after that by the product owner (explained below) also. In addition, it was essential to contribute to the team's cohesiveness from an early stage. The student showed interest in the subject, in knowing about it, and understood that the objectives were open and that everyone could contribute with improvement

proposals whenever this added more value to the product and aligned with the objectives defined by the company.

C. People/Team

In this case, the agile team was composed of 6 members, following the recommendations of Scrum [2]. 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, among others). In this process, the student, the central element of the approach, interacted with other people. Besides getting support for developing the project/product, he also gained experience in teamwork (soft skills), realizing the difficulties and aspects 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.

D. Process

Tools were adopted for this purpose to carry out the development process. Thus, Jira [15] 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.

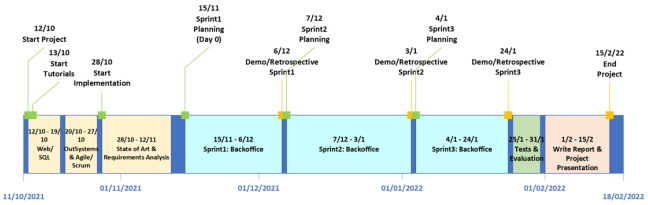


Figure 1. Timeline.

Scrum's artifacts [2] 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 optimize the project; for example, how to integrate Azure [16] cognitive services into the product under development. In addition, the definition of the sprint periods was separate from the academic activity, which took place in parallel so that the student could also fulfill the academic requirements in his other subjects. Thus, there were different sprint periods as there were also different workloads.

E. Project

Since an agile approach was adopted, it followed the value [17]:

"Working software over comprehensive documentation."

In terms of requirements documentation and modeling, 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 analyze 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 and documented the initial requirements in the product backlog. The user stories' acceptance criteria were defined, which helped design the test cases and thus contributed to a robust application.

Although the student had general knowledge about Artificial Intelligence (AI), it required them to be prepared to take advantage of the resources provided by Azure in terms of parameterization and integration and 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 synchronization (Web and mobile), security, agile Scrum framework, teamwork, 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 [17]:

"*Individuals and interactions over processes and tools*"

All these aspects mentioned above were considered, and the project includes documentation on user stories, database modeling, 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 [18].

This choice, teamwork, and adopting the agile framework (Scrum) allowed us to design from scratch and implement and test a complex and challenging software product during the standard school year.

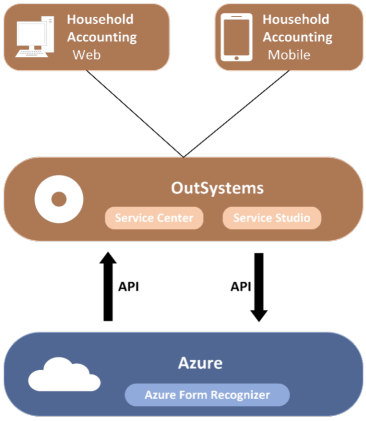


Figure 2. Systems Architecture.

F. 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 apply mechanisms to recognize and extract data from invoices and store, organize, and manage these data.

Using the OutSystems platform to develop the current project was also 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 [18].

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 to keep the improvement process constant from sprint to the next sprint, fostering a continuous pace. This demonstrates to the student the importance of the third and fourth values of [17]:

"Customer collaboration over contract negotiation"

and

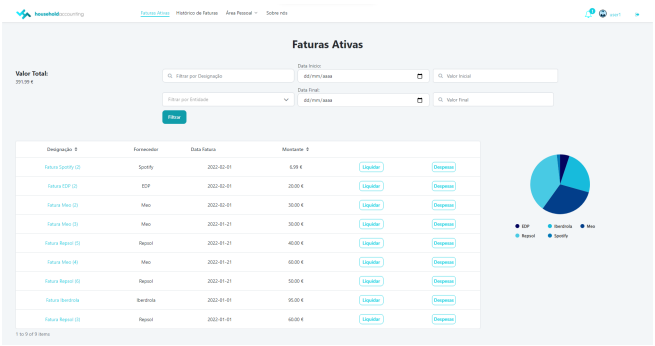
"Responding to change over following a plan"

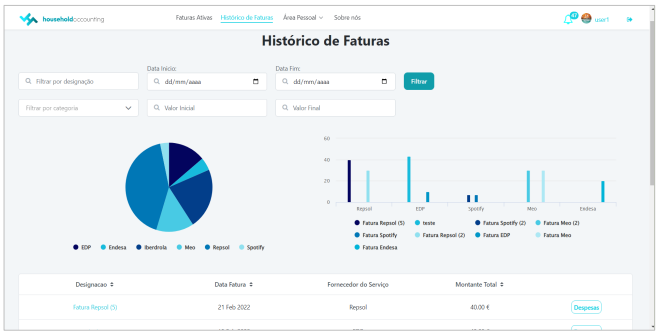
The final product was developed on time, and all goals were achieved. In other words, at the end of the project, the resulting product was an application (Web and mobile) that was developed in OutSystems with the integration of Azure cognitive services (Azure form recognizer [19]) that allows (among other functionalities) the user to:

- Register invoices automatically.
- Process invoices (recognize 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 synchronize both applications (Web and mobile) were implemented.

Figures 3, 4, and 5 show the final layout of the Web portal User Interface (UI) and one of the mobile application's UI.





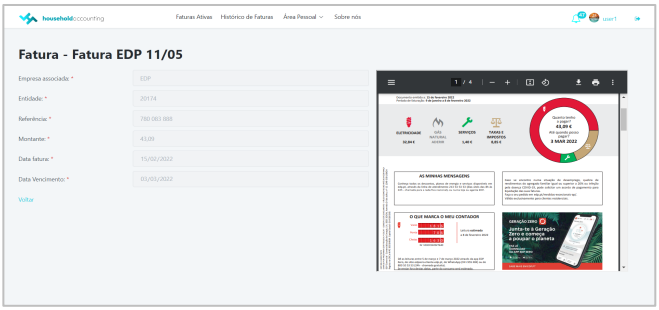


Figure 3. Examples of portal Web UIs (On top: active invoices, middle: Invoice's historical view, Bellow: Output of automatically processed invoice view).

Figure 3 (top) presents a dashboard with the active invoices (due to be paid). In Figure 3 (middle), 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 (below), we can

see the result of an automatically processed invoice view, presenting the invoice's file and the automatically captured fields.

Users can consult their invoices that are due to be paid. They can choose to visualize it through a graph organized according to the value and entities of the invoices (Figure 4- left) or to visualize it in the form of a list (Figure 4 - right).

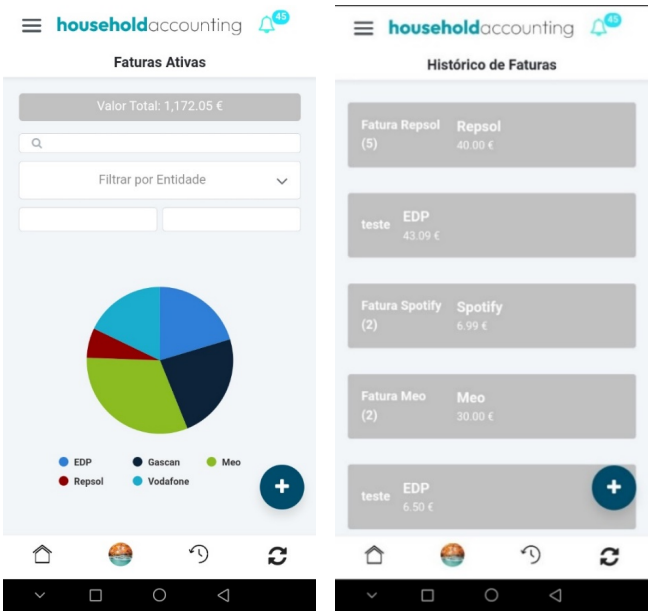


Figure 4. Examples of mobile app UIs (Left: active invoices- graph view Right: active invoices- list view).

In Figure 5, you can see the functionalities related to selecting the method to insert a new invoice (left) and a visualization of the user's expenses (right).

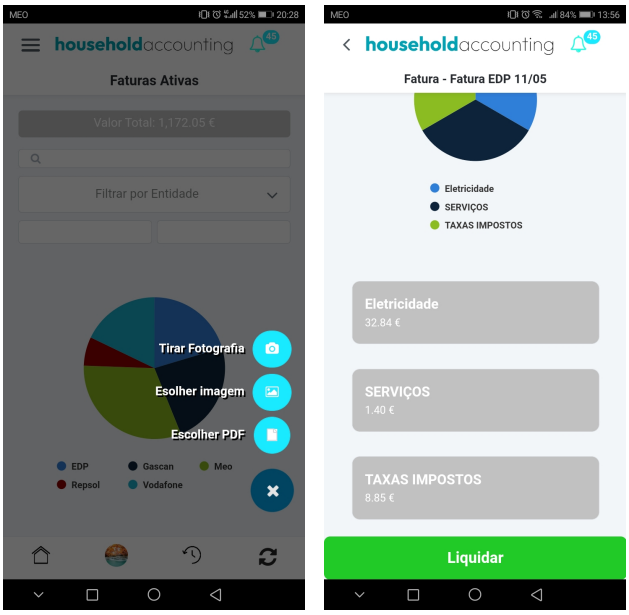


Figure 5. Examples of mobile app UIs (Left: select new invoice; Right: expenses).

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 synchronization between the Web and the mobile applications, and security issues. The performance results obtained with the recognition of invoices were also analyzed and improved.

In technical terms, several aspects have been studied from scratch and implemented for this software application, demonstrating a real and active learning implementation. Some of the aspects that evidence these practices are:

Security

The constant demand to ensure the security of information and infrastructures is recurrent and essential. In this project, some measures were considered to ensure a higher level of se curity for users and the information associated with using the mobile application.

The security mechanisms implemented in the mobile application follow the Top 10 Mobile Risks identified by the OWASP organization [20] in 2016. The ten risks identified by the organization were as follows: M1: Improper Platform Usage; M2: Insecure Data Storage; M3: Insecure Communication; M4: Insecure Authentication; M5: Insufficient Cryptography; M6: Insecure Authorization; M7: Poor Code Quality; M8: Code Tampering; M9: Reverse Engineering; M10: Extraneous Functionality.

Local Data

Creating a local database becomes mandatory once it is decided to allow the user to use the application without an Internet connection (offline). As such, using the application offline requires that some of the user data and associated data be stored on the device itself. Using the application offline does not allow using all the features that the concept promises; however, it ensures most of them, such as viewing active expenses.

The information stored in this application includes data from the invoices and their respective expenses (information from the logged-in user's invoices and the expenses associated with those invoices), from the support messages (associated with the logged-in user), from the notifications (associated with the logged-in user), from the logged-in user, from the proposals of new types of invoices (information from the proposals made by the logged-in user and from the user who generated the proposals) and data related to the application itself (properties for running the application).

The information stored in local storage at the login stage usually does not correspond to all the data stored in the server database. The amount of data stored in the server database can be huge, and it is necessary to filter the amount of data to be stored on the end device, namely by how recent the data is.

The structure of the proposed local database was based on non-relational modeling, i.e., NoSQL modeling. NoSQL modeling is used to process large volumes of unstructured

and ever-changing data. This type of modeling is very efficient in data processing and fast in querying information [21]. NoSQL modeling on mobile devices is highly important because devices do not have the same computing power as a server. This type of constraint has repercussions on the use of the mobile application, such as an increase in the application's response time to the user. The relationships between tables require more computational power and, consequently, more time. That said, the structure of each table stores specific information and associated information, possibly relevant when accessing the primary information.

Although two structures are used to store the authenticated user information, the server database is the main structure that guarantees data fidelity.

Synchronization

Synchronism becomes necessary when the mobile application allows users to use features without (offline) and with an Internet connection.

For the user to use the application without being connected to the Internet, saving some user information and some associated data is necessary. Given this, the user will use the application the usual way regardless of whether he has an Internet connection, causing the data/records to change.

In offline mode, it is necessary to ensure that the data/records that have changed in the device's local memory while using the mobile application without an internet connection will be updated in the server database when reconnecting to the Internet. In short, synchronization is based on ensuring that the data in the server database and the data in local storage are always up to date with the latest version of the data that has been manipulated and ensuring that the navigation is automated so that it is not dependent on a constant Internet connection. The concept and methods inherent to synchronization can be quite complex to understand and implement, and synchronization is achieved through implementing platform-specific patterns called Offline Data Sync Patterns [22]. The data synchronization patterns used in the mobile application were the following:

- Read-Only Data [23];
- Read/Write Data Last Write Wins [24].

The Read-Only Data pattern is essential for storing auxiliary information, such as filling dropdowns.

The use of the pattern Read-Write Data Last Write Wins is important, in turn, for synchronizing the changed information in the server database and the local database. This implementation makes it possible to use the mobile application without the user being connected to the Internet.

Data synchronization is performed when a given user logs into the mobile application (the essential data is loaded at this stage), on-demand (as the application is used continuously), and when the user uses the *PullToRefresh* feature implemented on almost every screen.

UI/UX

The development of mobile applications also requires a special concern with the design of interfaces and the experience that affects the end user. In this project, there was a constant concern with the interface design and providing the users with the best usability experience. Some of the details (easily visible using the mobile application) that were considered were the following: Size of the buttons; Button position; Color of the buttons; Representative icons; Contrast between colors; Always keeping the user informed. However, some programming logics are implemented to provide the best user experience; for example, the *OnScrollEnding* mechanism is used to load more data on pages that list information. The algorithm implemented loads only some information at a time to load the page as fast as possible and does not immediately make reading the data exhausting for the user. As such, the data is loaded in blocks of fixed size according to the user's wishes.

Performance

The performance of mobile applications has, over time, increased with technological advancement. However, this advancement does not evolve in only one direction. While hardware evolves, the amount of data that users manipulate and generate has also been increasing, in addition to the resources used by the applications.

The application solutions were implemented throughout the development to provide a better user experience. Some of the solutions identified and implemented to improve the performance of the application were the following:

- Restrictions on the amount of data to be stored in the local database.
- Restrictions on the amount of data to display to the user when loading certain screens.
- Image compression.

The amount of data associated with a given user can be immense, and one must restrict the amount of data to be stored in the local database without compromising the normal operation of the application without the device being connected to the Internet. A practical example implemented in the Household Accounting application corresponds to the information on outstanding invoices and the respective expenses stored in the local database.

The main page of the mobile application lists the outstanding invoices of the authenticated user; however, the invoices are not displayed all at once so as not to cause significant delay when building the page. That said, fixed-sized blocks are loaded at a time (depending on the user's willingness to scroll the page). First, the page is filled with data in local storage, and if all the data in local storage has already been loaded, it is checked to see if there are any new records in the server database that do not exist in the local database. If there are new records, they are shown on the screen.

The main functionality implemented in the mobile application is the automatic reading of relevant information from an invoice. Reading this information from invoices can be done by uploading PDF files, uploading gallery images taken previously from invoices in physical format, and taking a picture of the pages of a physical invoice at the time. Using this functionality in the last-mentioned format caused some inconvenience regarding the size of the files being generated,

which would have to be handled by the application and the resource. In the first prototypes, the usage of the application without performing any "treatment" on the generated files led to the closing of the application after a waiting period (3-4 minutes) for the application's response to the reading of the invoice by the resource. The application generated files with about 8 MB (for each page of an invoice). This made the application unable to handle such a large amount of data and sending data to the Azure resource with a size exceeding 15 MB (in the case of an invoice consisting of two pages) because analyzing the invoice information takes time, taking up to 4 minutes and/or even closing the application suddenly. Considering this, it was necessary to implement some compression mechanisms to solve this problem. In the first instance, a comparison was made between two methods provided by two different OutSystems components to proceed with the compression of the generated files. To do this, several factors were considered, such as the time of performance, the size of the files obtained because of the methods, and the image quality lost with each one. The comparison was made between the following methods provided by the following OutSystems components:

- *Image_Utils_Reactive*.
- *ImageCompress*.

To select the most appropriate method, some initial experiments were done. Table 1 presents the results of the experiments using these two methods applied in the same file.

TABLE I. OUTSYSTEMS COMPONENTS COMPRESSION.

Image_Utils_Reactive Before (original)		Image_Utils_Reactive After (60% quality)	
Size	8 MB	Size	629 KB
Width	2268	Width	2268
Height	3024	Height	3024

ImageCompress Before (original)		ImageCompress After (60% quality)	
Size	8 MB	Size	739 KB
Width	2268	Width	2268
Height	3024	Height	3024

Initial experiments start by compressing the captured image to 60% of its initial quality. These experiments showed that the images compressed through the *Image_Utils_Reactive* component are smaller than those compressed by the *ImageCompress* component (Table 1 up-right/down-right). It was also verified that the use of the component did not result in such a long delay for the final insertion of a new invoice.

However, although the component generated a smaller file size, it was noticed that the algorithm's accuracy had been compromised when the file was compressed too much. Then, more experiments with distinct compression rates (considering 70% and 80% of the image's original quality) were performed with the *Image_Utils_Reactive* component. These experiments made it possible to realize that file compression below 70% is the threshold for correctly understanding invoice information.

Integration with Azure Services

Azure Form Recognizer [25] is an artificial intelligence (AI) service provided in the cloud that uses AI models to extract information from documents such as text and tables from our documents. In this case, the documents are invoices, and extracting the desired information from the invoices was possible.

The configuration of the template to be used in this system involved the use of a tool called Form OCR Testing Tool [26]; however, the template's configuration required the existence of some previously obtained keys because of the configuration on the Azure side. Invoices uploaded to Azure containers (in various formats) appear in the editor automatically, ready to be configured. It is then necessary to run a function on each document to highlight which fields are possible to select to be identified by the template.

Afterward, it is necessary to create tags (as you can see on the right side of Figure 6) for each piece of information you want to extract from the invoices. The tags considered in this implementation were 14.

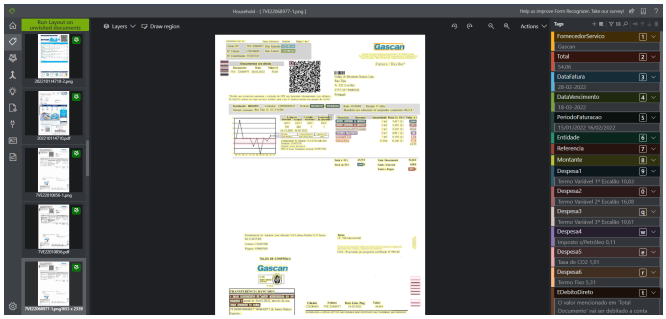


Figure 6. OCR Testing Tool GUI.

Finally, the model was trained based on the configurations performed previously. It required the configuration of at least five invoices to be able to be performed, and obviously, the more invoices inserted and configured, the better the training results.

The results of the training vary, considering the number of files inserted and configured on the platform, the concern with the tags that are created and how general they can be to respond to all types of invoices, and, finally, the insertion of files of various types and obtained in various ways. PDF files and PNG files were used and converted from invoices in PDF format. PNG images obtained from photographs of paper invoices were also inserted.

During the study stage, it was noticed that the resource's accuracy in analyzing invoice photographs is lower than in analyzing PDF invoices. For example, when using a PDF file consisting of photographs taken from the camera of an end device, information near the edges of the pages was identified with flaws. This pattern is easily justified by the fact that the focus of the devices' cameras is by default on the center, so the quality of the remaining areas of the photos will be lower.

Integration with *OneSignal*

Another deployed issue was using *OneSignal* [27], which provides push notifications for mobile devices and web,

messages within the application, SMS, and email. The use of these services is possible through the free consumption of the API by REST. Despite being the free version, the version of *OneSignal* used in our application guarantees the data protection imposed by the General Data Protection Regulation (GDPR). However, they use some statistical values for personal marketing purposes, among others. This tool was useful in integrating the native notifications of users' end devices with our application, providing further rigor to the mobile application.

Above all, it was another tool to integrate into our application that, consequently, led to learning how it works and how it is possible to integrate it into the most diverse systems. Furthermore, it was also important to demonstrate how the functionality works and observe the intended goal with the integration of *OneSignal*.

These are some of the several (others exist like database modeling and implementation or learning new tools (ex. balsamic)) aspects involved in this project's preparation, design, and implementation. This demonstrates the wide range of software aspects and tools involved throughout the project and at various levels (in terms of technical knowledge and soft skills). The project's success was due to the student's commitment and dedication and to the entire collaborative environment that was present from the beginning. We believe it was clearly a success story that we intend to replicate in subsequent editions and with other students.

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 software developed 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 organized 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 recognized automatically from the original pdf invoices received by email from the service providers (e.g., a gas company, an energy company, or a telecommunications provider). A lower accuracy rate was achieved if the invoices were digitized using the smartphone camera (even so, in the performed experiments, at least 46.5% of the fields were well recognized, and the user manually entered the remaining fields). After having the 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, and 80% rated it as easy or very easy to use. As shown in Figure 7, all the respondents were satisfied or very satisfied with the automatic reading of invoice information.

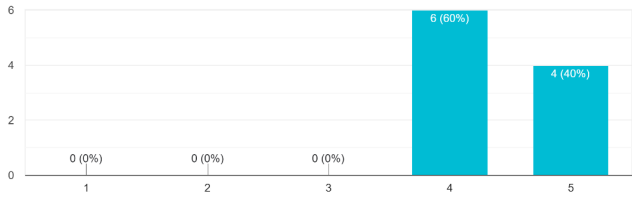


Figure 7. Users' satisfaction related to automatic reading of invoice information.

The functionalities that allow the import of invoices from PDF files or photos are among the most relevant in the application. Regarding these features, user opinions were very positive (see Figure 8). Importing invoices in PDF format was considered very important by 80% of respondents (Figure 8).

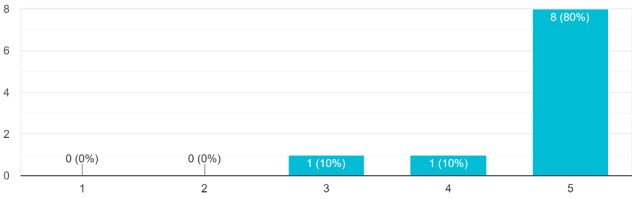


Figure 8. Importance of the "importing invoices in pdf format" feature.

Users recognize that the feature import of invoices from a photo was important or very important by 60% (Figure 9). However, despite the recognition of the importance of this feature, user opinions are more divided. Only 40% of them recognize this functionality as important or very important.

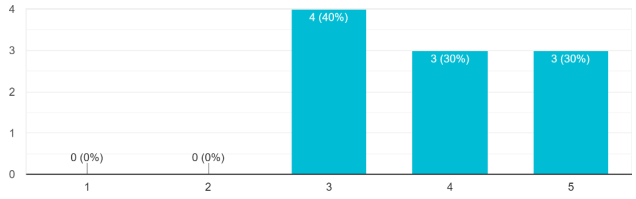


Figure 9. Importance of the "importing invoices from photo" feature.

How data is identified (extracted from the invoice) and presented to users was also to the liking of respondents, as can be seen in Figure 10. All respondents rated how the data are presented as good or very good.

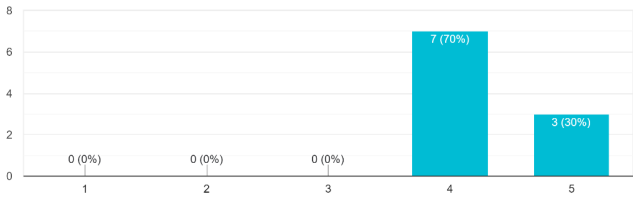


Figure 10. How the data extracted from the invoice is presented.

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 hour/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, including teachers, students, and staff involved from the partner company.

This approach has been successfully carried out in this case study and it was also implemented in previous editions with other students with different characteristics [28]. In both scenarios very positive results were achieved which suggest that this approach can contribute to overcome the gap academia-industry, mentioned in the introduction of this article. This approach fosters students to develop state-of-the-art technical and non-technical skills. However, these case studies also showed that this is a demanding approach that depends on the availability of all participants: teachers, industry partners and students. Scaling up the application of this methodology to a large number of students will be a challenge as it will require availability and a lot of work on the part of teachers and professionals from the companies involved. In this sense, to ensure the success of this methodology, an important aspect is that all parties (industry and academia) should be committed to apply this approach and to guarantee the necessary time availability. Also, students enrolled in this approach must be aware that they must maintain a constant pace and commitment with the established goals.

V. Conclusions

In this work, we presented a case study that shares an agile approach to preparing students for the job market regarding SE practices. The main goal of this approach was to prepare students' performance and technical skills for the software industry's new demands and thus contribute to reducing the gap between SE education and practice in the software industry. We followed an approach that promotes active and collaborative learning to achieve this goal. The case study presented illustrates the work methodology, the approach, details of the project design and implementation, the results achieved, including the resulting product, some lessons learned, and some guidelines for the future. And the resulting product.

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. These industry-academia partnerships help students become better and more 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 perform better in software development teams. These partnerships are also advantageous for the other partners involved. Hiring qualified human resources is good for the companies, as well as for the participating higher education institutions (contributes to improving their employability rate).

In this case study, an Agile methodology was followed. It allows closer and more frequent monitoring of all members of the team and therefore has advantages for student learning. In situations where companies use other methodologies or development tools, the involvement of companies in the student's teaching/learning process will always benefit the student. However, to assess these benefits or to compare the benefits when different methodologies are followed, further studies will be needed.

Acknowledgment

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

References

[1] J. Metrôlho, F. Ribeiro, and P. Graça, “Prepare Students for Software Industry. A case study on an agile full stack project,” in *Seventeenth International Conference on Software Engineering Advances*, 2022, pp. 75–80.

[2] 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).

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

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

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

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

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

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

[20] OWASP, "Top 10 Mobile Risks - Final List 2016," 2016. https://owasp.org/www-project-mobile-top-10/2016-risks (accessed Mar. 21, 2023).

[21] Microsoft Corporation, "Base de Dados NoSQL – O que é NoSQL?" https://azure.microsoft.com/pt-pt/overview/nosql-database/ (accessed Mar. 21, 2023).

[22] OutSystems, "Offline Data Sync Patterns," 2023. https://success.outsystems.com/Documentation/11/Developing_an_Application/Use_Data/Offline/Offline_Data_Sync_Patterns (accessed Mar. 20, 2023).

[23] OutSystems, "Read-Only Data," 2023. https://success.outsystems.com/Documentation/11/Developing_an_Application/Use_Data/Offline/Offline_Data_Sync_Patterns/Read-Only_Data (accessed Mar. 20, 2023).

[24] OutSystems, "Read/Write Data Last Write Wins," 2023. https://success.outsystems.com/Documentation/11/Developing_an_Application/Use_Data/Offline/Offline_Data_Sync_Patterns/Read/Write_Data_Last_Write_Wins (accessed Mar. 20, 2023).

[25] OutSystems, "Get started with the Form Recognizer Sample Labeling tool," 2023. https://docs.microsoft.com/en-us/azure/applied-ai-services/form-recognizer/quickstarts/try-sample-label-tool (accessed Mar. 20, 2023).

[26] Microsoft, "Microsoft, "Editor - Form OCR Testing Tool." https://fott-2-1.azurewebsites.net/ (accessed Jun. 22, 2022).

[27] OneSignal, "OneSignal." https://onesignal.com (accessed Jun. 13, 2022).

[28] J. Metrôlho, F. Ribeiro, P. Graça, A. Mourato, D. Figueiredo, and H. Vilarinho, "Aligning Software Engineering Teaching Strategies and Practices with Industrial Needs," *Computation*, vol. 10, no. 8, 2022, doi: 10.3390/computation10080129.