Intelligent Task Management based on a Software Application

AMEL HANOOSH[1], IULIANA MARIN[2]
[1]Ministry of Education Directorate of Almuthanna Education,
Al-Muthana,
IRAQ

[2] Faculty of Engineering in Foreign Languages,
National University of Science and Technology,
Politehnica Bucharest,
ROMANIA

Abstract: - The paper outlines the creation and execution of a web application designed to manage and monitor tasks within software development projects. This application will be known as 'Scrummer' throughout the documentation. This study will demonstrate that there is a significant demand for such a software tool, as IT companies continuously search for solutions that can enhance team efficiency, accelerate time to market, or meet customer demands. The Agile Manifesto encompasses all these solutions and more, operating according to its principles and values.

Key-Words: - Scrummer, software tool, web application, Agile, task management, software development projects, team efficiency, customer demands, time to market.

Received: July 25, 2024. Revised: February 27, 2025. Accepted: April 26, 2025. Published: July 30, 2025.

1 Introduction

In the last decade, software development teams have focused on shortening up the time frame from the moment a product starts as a concept to the moment it reaches the market. Mostly, these software development teams have accomplished this goal in impressive ways, and this short time frame is possible due to Agile. Considering that nowadays IT companies are open to change from a fixed mindset to an agile mindset, the present study proposes a software solution that helps them, offering an easier way of getting successfully through the Agile Transformation. Since Agile is only an umbrella for several software development frameworks, this article follows only one of them, which implements and amplifies the principles and values of the Agile Methodology. This framework is called Scrummer, and it represents the foundation of this study.

The software tool will provide a smoother workflow for going through all the phases of a software project by incorporating elements of iterative and continuous development. The system is formed of a web-based application that can be accessed from any browser, and it automates the interactions between the members of a software development project by facilitating the management of the tasks that occur when analyzing, implementing, and testing new software.

The present study is structured into four chapters that are also split into subchapters, to efficiently organize the information and facilitate the understanding of the text. The first chapter is entitled “Presentation of the Domain” and is focused on describing the area the present paper is part of, and a comparison is made between this new tool and other similar tools on the market. The second chapter, “Scrummer Application”, describes all the functionalities of the tool and the technologies used in developing the web application. The third chapter has in scope validations that support this study, and a survey that was conducted to better understand this article. Lastly, the fourth chapter provides the conclusions and future work that can be done to enhance the present Application.

2 Methodology

2.1 Presentation of the Domain Agile Methodology

Before defining what Agile Methodology is, we need to travel back in time to understand where the agility concept came from and what the reasons behind its creation were. One of the main reasons is change. The need for change in the software development cycle arose because, no more than 30

years ago - changes or additions along the way were not possible. The Waterfall Methodology that was used in the past did not allow any changes as the team had to stick to the fixed requirements that were set from the beginning. This led to problems not only on the client side but also on the development side. On the client side, the product could take years before going out to the market and when it finally did, most of the time, it was outdated and no longer a strong fit for the market. On the development side, a lot of teams just abandoned their projects on the way, leading to numerous unfinished products and a waste of resources.

Based on the previous issues, software development teams started to change their way of planning and delivering new products back in the 1990s. Throughout this period, various development methods like Scrum, Extreme Programming, Feature-Driven Development, Pragmatic Programming, and Rapid Application Development were introduced. The common principles between these methods are flexibility and less planning beforehand. All these models are the first and earliest approaches in Agile history.

Finally, in the 2000s, a group of software developers came up with some new ideas on how to build new software and launch it to market faster and more successfully.

The two main matters that they thought would achieve these goals were gathering feedback from the users to continuously improve the product and shortening the gap between the moment a product is developed and the moment it is delivered to the client. The second important moment in Agile history was when the same group of software developers gave rise to the "Agile Manifesto", which was a great decisive moment. At this moment, the four values and the twelve principles of Agile were born.

The roadmap of a product is alive and dynamic as it is continuously analyzed and reviewed to prioritize its work based on strategic needs, [1].

The four values are:

1."Individuals and interactions over processes and tools".

The most important entity in a new project is the team. The team is the one that determines the success of the product. Therefore, the individuals and the interactions between them are more valuable than the processes and the tools.

2."Working software over comprehensive documentation" In the Waterfall approach, full documentation was required before starting the development itself. But when it comes to Agile, the highest priority is giving the client the requested software.

3."Customer collaboration over contract negotiation".

Agile preaches customer collaboration for the continuous gathering of feedback and implicating the client more in the delivery process.

4."Responding to change over following a plan" The roadmap of a product is alive and dynamic as it is continuously analyzed and reviewed to prioritize its work based on the strategic needs.

The twelve principles are:

- "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software."
- "Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage."
- "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale. "This principle gives a better definition of what delivering continuously means.
- "Business people and developers must work together daily throughout the project. Agile believes that having the development side and the business side working together, closely, can somehow reduce the risks. These risks include misunderstandings or miscommunications.
- "Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done."
- "The most efficient and effective method of conveying information to and within a development team is face-to-face conversation." Direct communication and feedback are valuable because the risks of misinterpreting or miscommunicating are reduced.
- "Working software is the primary measure of progress," Agile states that the primary measure of progress is working software. Working software means that the product has been shipped and is ready for use. If it is working, but it is not delivered, it cannot be counted as progress.
- "Working software is the primary measure of progress." Agile states that Working software means that the product has been shipped and is ready for use. If it is working, but it is not delivered.
- "Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely." This meaning that the amount of time and effort of the team should also be constant.

- "Continuous attention to technical excellence and good design enhances agility." If the technical design is neglected for too long, the delivery process will be slowed down, making the team less agile.

- "Simplicity–the art of maximizing the amount of work not done–is essential." Unnecessary steps in the development of the product should be removed, focusing the attention on more valuable work. For example, replacing the manual work with automated.

- "The best architectures, requirements, and designs emerge from self-organizing teams." When given more freedom in the decision process, teams become self-organizing by structuring their own work and taking more responsibility than just writing the code because programming is much more than that: gathering requirements, communicating with the business, understanding the business.

At regular intervals, the team reflects on how to become more effective and then tunes and adjusts its behavior accordingly, [2]. Retrospective sessions play a major part in an agile team because they determine how the process went and where mistakes have been made. These four values and twelve principles still hold and are still valid today and they are applied by all the software development teams that follow an Agile methodology. Another basic notion that is important in Agile is the Software Development Life Cycle model, simply known as SDLC, as displayed in Figure 1. This process is continuous, and it is present from the moment a product is decided to launch to it decommission. Since the main approach of Agile is iteration, some actions are going to be sequentially repeated until the goal is met.

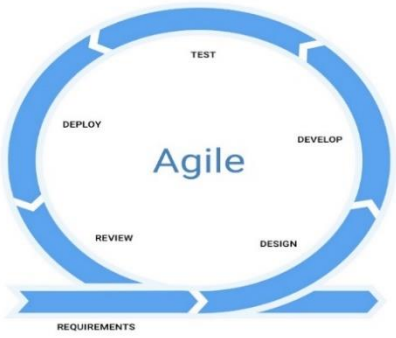

Fig. 1: The Agile SDLC

There are six phases in the Agile SDLC. Firstly, the creation of the initial documentation is made, by generating a list of what the product is going to represent. The second phase refers to the design of the product. It can be visual (UI/UX) design or architectural design. In this phase, matters like programming languages, components, or libraries are chosen. Phase three is development itself (coding). The requirements and the design are converted into actual software. Testing follows development, so it represents the fourth phase of the SDLC. A suite of tests is conducted by the QA team, so it is guaranteed that the requirements have been met. The fifth phase is deployment. The product is sent to the production environment and presented to the customers. With every iteration, new features are released, along with bug fixes. Finally, there is the review phase which represents the sixth and last phase of the SDLC. In this phase, the progress of the whole iteration is reviewed by the software development team. After the six phases are successfully completed, the team can improve the ideas for the upcoming iterations and put them on the table, [3].

Every iteration of the SDLC must be supported by specific Agile project management tools. These tools must be specialized in the agile ways for them to give the needed support to the team. To have noticeable results with respect to team communication and resource allocation, a specially conceived tool is recommended because it will facilitate the developers' work.

Using the right tool is important because of the following reasons, [4]:

- Agile projects are dynamic and tend to change all the time, making effort coordination challenging within the team.

- Another challenge is keeping the project aligned with its objectives.

- Using a tool gives better transparency within the team, as anyone can view the progress at any time.

- It maximizes the flow of work and efficiency (the structure of the project is more fluid and considerably simplified).

- The environment becomes more dynamic because the plan is continuously refined and updated.

- A better tracking of the delivery of working software against the project's roadmap.

- Team collaboration is emphasized because the entire team shares the tool.

- The data in the tool is continuously updated by every member of the team which leads to better tracking of the progress and better coordination.

Hence the previous reasons, tools can be critical in an Agile project. Such a tool should be as simple as possible to give the developers more freedom to focus on product delivery. When choosing the agile tool, the key factors should be taken into

consideration, including the needs of the company, [5].

2.2 Scrum Framework

Scrum is one of the most popular and most used Agile frameworks. It was first introduced in 1996, after which it took part in creating the Agile Manifesto a few years later, [6]. The term "Scrum" comes from the rugby game, where the player must pass the ball in multiple steps to score. This action is like the one where a project moves in iterations to reach its objectives, [6]. Unlike Waterfall, Scrum delivers features one at a time. Since the waterfall approach is sequential and the project brings value only in the end, Scrum is somehow the opposite because it constantly delivers new features (every few weeks) and it does not only focus on one big future release. Moreover, Scrum splits the complex workload into easier parts, large companies into smaller teams, and complicated projects into a set of short time frames, called sprints. Organizations can deliver products to the customers more rapidly and more efficiently if they for an incremental and iterative approach.

Scrum is only one of the multiple agile software development processes that emphasize the early delivery of the software. Under the Agile umbrella, many of these processes can be found, out of which Scrum, Extreme Programming, and Kanban are the most popular. All the processes under Agile share the same values and principles. Scrum is a lightweight framework that is easy to understand, but difficult to master.

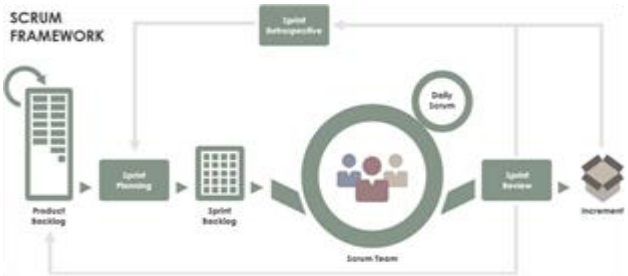

Fig. 2: Scrum software delivery

To successfully make use of the Scrum framework, one must have knowledge of the general guidelines, including roles, artifacts, rules, and ceremonies (events). All these components form the Scrum Framework, which may sound simple, but they can be difficult to master.

The diagram in Figure 2 shows some of the core components of the Scrum Framework:

- The Team - every member of the team has a Scrum role. The roles are Scrum Product Owner, Scrum Master, and Scrum Team.
- The most common Scrum artifacts - Product Backlog, Sprint Backlog, Increment.
- The Scrum Ceremonies - Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective.

The Scrum process and the most important elements are represented in Figure 3.

Fig. 3: The Scrum process

2.3 The Scrum Roles

The roles inside a Scrum team are different from the ones inside a traditional team. Only three main roles can be found in the Scrum Framework.

The first Scrum development role is the Product Owner. The PO represents the bridge between the client and the team. In a Scrum team, there is only one individual that has the role of PO. He is close to the client and is responsible for determining the features of the product. Besides this main responsibility, there are also other attributes and responsibilities that rest in the PO's hands, [7]:

1. Ownership and responsibility over the Product Backlog.
2. Setting up the strategy and the goals of the product.
3. Gather and prioritize the upcoming requirements.
4. Understand the client has needs and share them with the Development Team.
5. Collaborate with the development team.
6. Accept or reject the Product Increment at the end of every iteration.

Another important role in a Scrum team is the Scrum Master. The name indicates that this individual is an expert when it comes to the Scrum Framework. The SM is a servant-leader of the Scrum team. The definition of servant-leadership involves the individual demonstrating the characteristics of empathy, listening, stewardship, and commitment to personal growth toward others", [8]. The SM oversees promoting Scrum and Agile

practices and makes sure that everyone on the team respects and follows the Scrum principles and values. Most of the time, the SM is not a technical person, and one SM can work with multiple Agile teams. The SM should be the team's coach, guiding and coordinating them when it comes to the Scrum values and principles.

The Scrum Team, also known as the development team, is usually formed out of 3 to 9 people. They must carry out all the technical requirements to deliver the product, [9]. At the end of every Sprint, the development team is responsible for the product Increment that is supposed to be shaped after going through all the SDLC phases: analysis, design, development, testing, deployment, and review. There are certain characteristics a Scrum Team must possess and some responsibilities it must fulfil:

- Self-organized - a self-organized team is a team that manages the work on its own and it does not depend on someone to assign the tasks to them. Every team member is free to choose the desired piece of work they are going to focus on, in agreement with the rest of the team, [10].
- Cross-functional - to be cross-functional, a team must be formed of individuals that belong to different functional areas of the organization, [11]. By making the team cross-functional, the quality of the team improves to changes and has a better handle on the maintenance, [12].
- There are no titles recognized in a Scrum Team - regardless of the profession of the individual, all members of the Scrum Team are known as Developers.
- There are no sub-teams in a Scrum Team - the Scrum Team is recognized as a whole, and there is no testing, analysis, or implementation sub-team.
- The whole team is accountable for the delivery of the product - this means that the Increment at the end of every Sprint lies in the hands of the Development Team.
- The team follows a common goal - the term "Sprint Goal" appears a total of 27 times in the official Scrum Guide. If the team sets a common goal at the beginning of every iteration, it will bring more focus to the important work, [13].
- The team members show respect to each other - respect is one of the core values in the Agile methodology, therefore it should not be neglected within the team.
- The team follows the same rules and standards - by having a common set of rules and principles, [14].

2.4 The Scrum Artifacts

The Scrum Artifacts define the product that is in the development stage, with all the involved activities. These artifacts must be acknowledged and understood by all the Scrum Team members, but also by the stakeholders.

The artifacts of the Scrum Framework are:

The Sprint: At the heart of the Scrum Framework, we find the Sprint. The Sprint is a time-boxed iteration that usually lasts between one and four weeks. Once the length is set, it cannot be changed throughout the life of the project. The most common length is two weeks. During this time, a product Increment is created, which is supposed to be potentially deliverable and usable by the end of the Sprint. Immediately after a Sprint is finished, the next one starts, with a different goal and requirements. Once a Sprint is started, no changes can be made that would potentially endanger the goal and the quality must not decrease. The Sprint is a time-boxOne of the benefits of the Sprint is predictability enablement, as it limits the risks that may arise along the way.

The Sprint Goal: A Sprint Goal is an objective, a high-level summary that is set at the beginning of every Sprint, for the Scrum Team to be aware of what is in focus. It should be met by the end of the iteration, and it should be common within the team, transparent, and understood by everyone, [15].

The Product Backlog: The Product Backlog is a list owned by the Product Owner that includes everything that is known about the product that is in development, like features, requirements, and bug fixes. It is the single source of truth related to the requirements that are needed for the product. The Product Backlog must be ordered by priority, as the items towards the top are clearer and ready for development. The items towards the bottom are usually less detailed and need to be updated with the necessary information.

The Sprint Backlog: The Sprint Backlog is a subset extracted from the Product Backlog that is selected for the Sprint. It describes the work that is going to be conducted within the Sprint. The items in the Sprint Backlog need to have enough details for the Development Team to be able to gainfully perform the work.

The Increment: The Increment is the totality of the items implemented during a Sprint, added to the Increment of all the previous Sprints. Every Sprint

needs to have an Increment at the end, as it is considered a step forward toward the product's final goal.

The Definition of Done (DoD): When an Increment or an item belonging to the Product Backlog is considered "done", everyone on the team must have the same understanding of what this "done" means. Every team has a different view of the DoD, as the whole team defines it and agrees upon the definition, creating a convention, [16].

The Definition of Ready (DoR): Unlike DoD, the DoR is a set of agreements within the team that decides whether the work item in the Product Backlog meets its essential requirements and is ready to be taken into the Sprint.

The Burn-down chart: The burn-down chart is a representation of the leftover work (that is estimated in story points) with respect to time. The story points are on the Y-axis (vertically), while the time is on the X-axis (horizontally). It proved to be useful in forecasting and predictions on when the work is going to be finished, [17].

The Impediment Log: The Impediment Log is a list of the obstacles and blockers a Scrum Team may face. It is the SM's responsibility to track and make sure that the impediments are resolved.

The Epic: The Epic is a large piece of work which describes a client request, a feature, or a requirement. It describes exactly what the final output is and what are the user's needs. At first, it may not contain all the details that need to be worked on. In Scrum, the details are defined in User Stories, for which the Epic acts as a placeholder, since the Epic is the top-tier in the work hierarchy. This makes tracking of the User Stories much easier. Most of the time, Epics are too complex to be completed in one iteration, therefore they spread over multiple Sprints.

The User Story: The Epic previously mentioned is broken down into multiple User Stories. The rule of the thumb is usually applied, meaning that if there are more than 5 User Stories covering the same area of features, they are put together inside an Epic. The User Story is the basic unit of work in Scrum. They are short, easy to understand and describe a feature from a user perspective. The User Story should follow the template suggested by Mike Cohn: "As a <type of user>, I want <some goal> so that <some reason>", [17].

The Task: A task belongs to a User Story, and it is more of a technical nature. While on a User Story more team members can work on, the task has only one assignee.

The Scrum Events: Like all the other activities in the Scrum Framework, the Scrum Ceremonies are time- boxed, meaning that every meeting has a fixed duration.

Sprint Planning: Every iteration starts with planning. The whole team participates, trying to establish and commit to the work items that are going to be performed during the Sprint and delivered at the end of it.

Daily Scrum: The Daily Scrum meeting is an event that occurs every day, as the name suggests. It is used for the team to synchronize and plan the upcoming 24 hours.

Sprint Review: At the end of every Sprint, the Scrum Team holds a demo of the newly developed Increment. This demo is part of the Sprint Review ceremony. The team presents a demo of the Increment, with emphasis on the DoD and the Sprint Goal. It is the PO's responsibility to review the Increment and decide whether to accept or reject it.

Sprint Retrospective: The last ceremony that takes place in a Sprint is the Sprint Retrospective. The whole Scrum Team should participate, and it is usually time-boxed for a maximum of 3 hours for a one-month Sprint. The purpose of this meeting is for everyone to discuss what went well in the previous Sprint, what could be improved and what actions are going to be taken in the next Sprint.

Backlog Refinement: The Backlog Refinement meeting, also known as "Grooming", is an important Scrum Ceremony as it helps the Scrum Team to better understand the Product Backlog. The owner of the Product Backlog, the PO, shares the items from the list with the rest of the team to discuss and refine them. After debating discussions, everyone votes according to their beliefs, and the voting is repeated until everyone comes to a consensus. This technique is proved to lead to better estimates for the items in the Product Backlog.

2.5 Similar Applications

Jira Software: Jira Software is a project management tool from Atlassian. Atlassian is a software multinational company that offers a wide range of products for software development and

project management. The most popular tools from Atlassian are Jira and Confluence. Jira supports all the Agile methodologies, but it is also customizable. It provides any Agile board, like the Scrum board or the Kanban board. By using the application, Agile teams can plan, track and manage their projects, all from inside the tool, as illustrated in Figure 4.

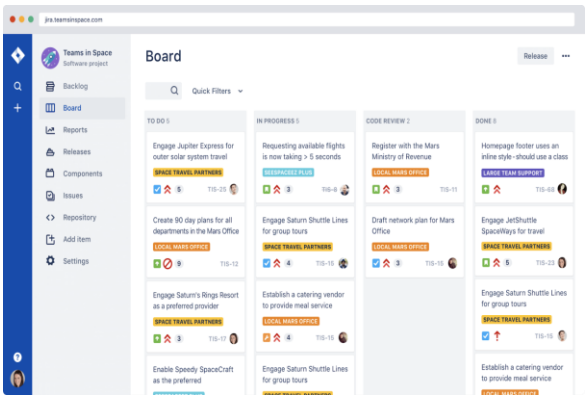

Fig. 4: Jira Scrum Board

Now, over 75.000 customers from 122 countries use Jira for Agile project management, [18]. Jira provides a wide range of reports that are generated at the end of the Sprint. These reports are specific for the Scrum Teams, and they provide an insight into how the process went, showing the areas where there is room for improvement. These reports include Burndown Chart, Velocity Chart, Release Burndown, Sprint Report, and so on, [19]. The pricing for Jira varies and depends on the number of users, [20]. One of the disadvantages of using Jira is the confusing UI. People who have used Jira reported that the UI is a bit messy and the filtering options are not easy to use.

Tuleap: Tuleap was developed by a French company called Enalean and it is a system for managing the application lifecycle, as displayed in Figure 5. It is an alternative to Jira and Confluence from Atlassian, but it is free of charge. Over 500 companies, ranging from small to big companies or just open-source projects use this tool for project management, whether it is Agile or traditional, Waterfall approach, [21].

Tuleap offers a Scrum-friendly workspace by providing a template for the project. This template contains 6 trackers: an Epics tracker, a User Story tracker, a Task tracker, a Release tracker, a Sprint tracker, and an Incident tracker. It also offers a Product Backlog that can be prioritized and the capability to add User Stories to it. Tracking progress is done through the graphical report of the Burn-down Chart. Tuleap also provides an archive of the past Sprints and their Burn-down Chart for the team to have an overview, compare, and analyze the previous Sprints.

Fig. 5: Tuleap Scrum Board

While Jira is a widely used tool supporting Agile methodologies and providing detailed Sprint reports (Burndown Chart, Velocity Chart), Scrummer, our proposed software solution, focuses on simplifying the user experience. Many users find Jira's interface overwhelming and filtering options unintuitive, which can hinder efficiency. Scrummer, in contrast, offers a streamlined, user-friendly interface tailored to reduce cognitive load and encourage adoption, particularly among smaller teams or those new to Agile frameworks.

Similarly, Tuleap's emphasis on providing trackers for every aspect of Scrum (Epics, User Stories, Tasks) and its free pricing model make it an appealing alternative to Jira for cost-conscious organizations. However, Scrummer differentiates itself by integrating real-time collaboration features, a seamless registration process, and flexible role-based access, ensuring that all Scrum roles are empowered to perform their functions without unnecessary complexity.

Scrummer improves upon existing solutions, reinforcing its strengths in delivering a simpler yet powerful Agile project management tool, as in Table 1 (Appendix).

The table illustrates how Scrummer balances simplicity with functionality, distinguishing itself through its UI and role-specific workflows while maintaining core Scrum principles.

3 Scrummer Application

"Scrummer" is our proposed web-based application that has as its focus providing an effective and user-friendly environment for Scrum Teams (Figure 6). To use "Scrummer", a constant, stable, and safe internet connection is required through which one can have real-time access to all the data inside the

application. It interacts and responds to client requests, and it can be accessed from any type of browser.

Fig. 6: Scrummer logo

3.1 Functionalities

The following diagrams have been created with Visual Paradigm Community Edition. The use cases inside the "Scrummer" application vary depending on the role of the currently logged-in user, as in Figure 7. As we know from the theory of Scrum, there are three main roles for this framework: Product Owner, Scrum Master, and Development Team. These roles also exist inside "Scrummer", and the specific role of the user must be picked when the user registers for the first time. If the user has not started a user session yet, he is considered anonymous, therefore the only two options he has are to register (if there is no account in his name) or log in (if an account already exists). Without an active session, he is not able to access other pages, like the dashboard.

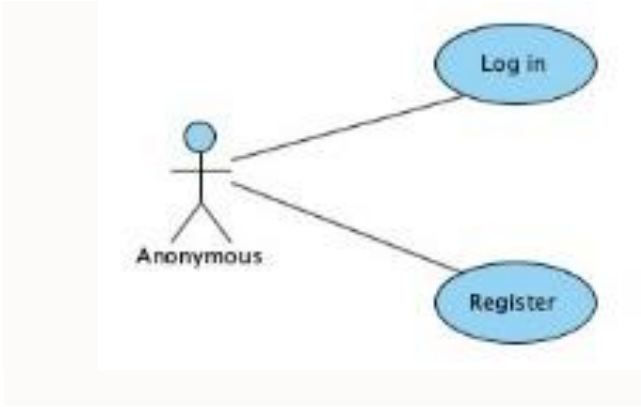

Fig. 7: Initial application use cases

When registering, the user must fill in a registration form, that includes the mandatory inputs: First name, Last name, Email address, Password, and Role (Product Owner, Scrum Master, or Development Team). Once the registration is completed, the user is redirected to the login page, where the email address and password are required to successfully access the account. If the password is incorrect or the email address does not exist in the database, the user won't be allowed to access the application. Initially, there is no project that the new user is assigned to Depending on the user's role, there are certain actions one may have access to.

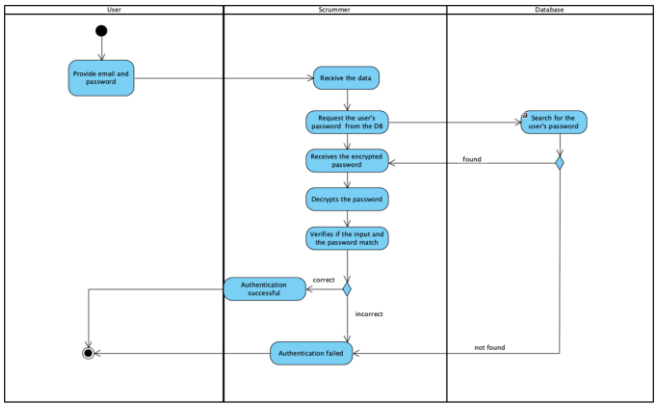

Fig. 8: Activity diagram for the authentication process

Figure 8 describes the authentication process in the "Scrummer" application. The first role that is going to be analyzed from a use case point of view is the Product Owner.

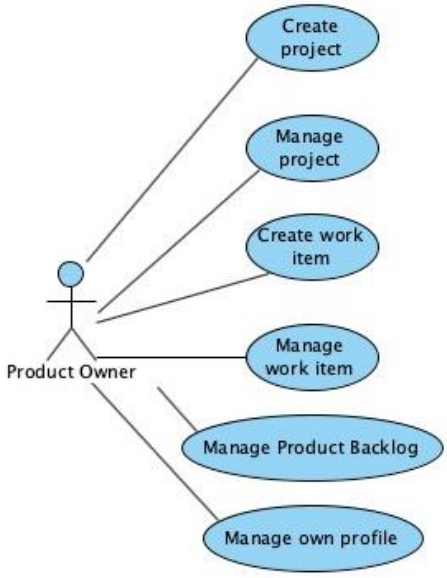

Fig. 9: Use Case Diagram for PO

The PO is the only user that is allowed to create new projects, as illustrated in Figure 9. He is also able to manage the project by editing the project's information, adding/removing users from a project, or completely deleting a project. Once the PO adds a new user to the project, that user is going to be able to see everything related to that project, ensuring full transparency.

The project requires the following mandatory information to be successfully created: Name, Description, The Scrum Master, and The Development Team members.

Figure 10 presents the activity for the use case when the PO creates a new project for the team. A similar activity is performed when the SM creates a new Sprint. Subsequently, other information related

to the project, like the Definition of Ready and Definition of Done is added by the PO, in agreement with the rest of the team.

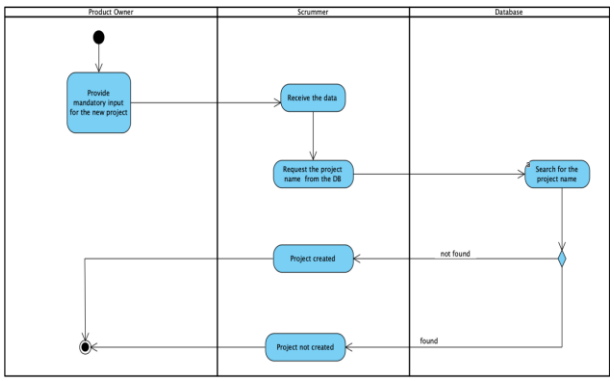

Fig. 10: Activity diagram for creating a new team project

Work items (Epics, User Stories) are initially created by the PO and they are automatically placed in the Product Backlog, the list of all the upcoming work items, owned by the PO. The tasks under a User Story usually have a technical nature and are created by the Development Team, but the PO has no restriction in creating tasks as well. All the work items in the Product Backlog can be edited or deleted by any member of the Scrum Team, despite their role, but only the PO is allowed to prioritize the items according to the product's needs.

Like any other user, the PO can access his user's profile, allowing him to edit personal information related to his account. The personal information includes: First name, Last name, Email, and Password.

The second role is the role of the Scrum Master, as in Figure 11.

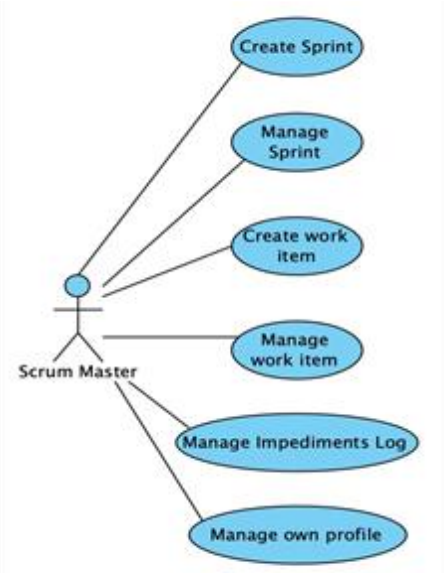

Fig. 11: Use Case Diagram for SM

The second role is that the newly created Sprints will be categorized as "inactive", until, during the Sprint Planning meeting, the upcoming Sprint will start and will be set to "active". Then, the SM can edit the Sprint Backlog, by adding or removing items the team agrees upon. On the last day of the Sprint, the SM is the one who changes the status of the Sprint from "active" to "finished". Any unfinished work will be sent back into the Product Backlog. The impediments that the Scrum Team faces are added to the Impediments Log, which is managed by the SM. The SM creates a new impediment whenever one comes up, by completing a form that contains the following inputs: Name and Description.

Initially, the status of the impediment will be considered "open", but will be switched to "resolved" by the SM, once the issue has been closed. Just like the PO, the SM can also create new work items (Epics, User Stories, Tasks) and manage their personal account.

The last role is the Development Team Member (Developer), as in Figure 12.

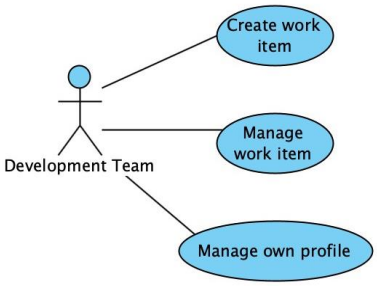

Fig. 12: Use Case Diagram for Development Team

The Development Team can only create work items for themselves and update them according to their needs. If a new Epic is created, the form for the Epic must include Name, Description, Acceptance Criteria, Priority (low, medium, high), Project, and Assignee (the user that oversees the new Epic).

If a new User Story is created, the form contains the same fields as for the Epic, but it also contains a field for the Epic the new User Story belongs. The Developer is supposed to divide the User Story into multiple tasks, that are more of a technical nature, as in Figure 13. In addition to the User Story creation form, the Task form also includes a field for the User Story the Task belongs to.

As a self-organized team, every team member is free to choose the work items that suit them best. When updating work items, the Development Team should have a clear understanding of the Definition of Done and the Definition of Ready.

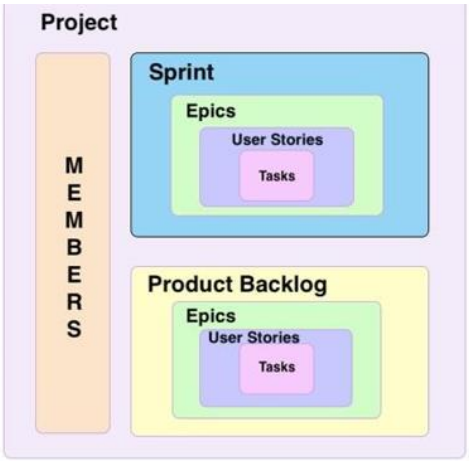

Fig. 13: Models relationship

The functionalities outlined, including role-specific features for the Product Owner, Scrum Master, and Development Team, aim to ensure that Scrummer aligns with Agile principles like collaboration, transparency, and flexibility. By requiring a stable internet connection and browser-based access, the tool facilitates real-time interaction, making it easier for teams to adhere to Agile practices no matter where they are.

The diagrams Use Case and Activity further illustrate how the application supports these roles and processes. The PO's ability to prioritize tasks or the SM's management of Sprint statuses directly reflects the Agile emphasis on iterative planning and team alignment. Similarly, the Development Team's capacity to self-organize and work with a clear Definition of Done underscores the tool's alignment with Scrum values.

3.2 Design and Implementation

The implementation of "Scrummer" has been accomplished using the Webstorm IDE from JetBrains, with a free educational license. Webstorm offers JavaScript and TypeScript development.

When it comes to web applications, there is always a growing demand for offering enhanced and smoother user experiences. The responsiveness and interactivity qualities are in high demand when considering a web application.

To successfully fulfil these requests, developers are approaching advanced full-stack technologies, like the MERN stack. "Scrummer" is also built using the MERN approach. MERN is a JavaScript stack that combines 4 technologies: M - MongoDB, E - Express JS, R - React JS, and N - Node JS, as in Figure 14. These 4 technologies offer a full stack development framework, meaning that it provides backend components, frontend components, and a database system.

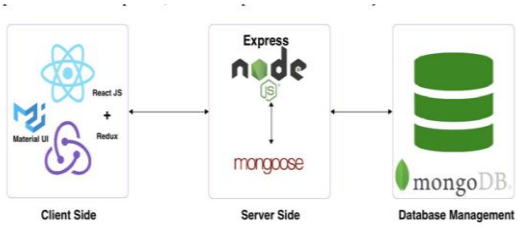

Fig. 14: Architecture of "Scrummer"

The technology stack uses the REST API, which handles the client-server relationship by making use of reusable components when high performance is required. These REST APIs are based on HTTP methods (GET, PUT, POST, PATCH, DELETE), which is an advantage for the MERN stack, as multiple frontends can be easily connected to the same backend system. The client sends the request to the server with the help of HTTP, as a web URL. The response from the server is most of the time a JSON format, as JSON is the most frequently used resource format [22]. For example, "Scrummer" uses an HTTP GET method when accessing the Dashboard page, by sending a GET request through the URL "/dashboard" that will return an HTTP response code of 200 if the data has been successfully retrieved or a code of 500 if an internal server error has occurred. A POST method is performed when a user creates a new work item, for example, a User Story. A POST request is sent to the server, through the URL "/create-user-story" that will return a response code of 201 if the resource has been successfully created or a code of 500 if there is an internal server error.

MongoDB: MongoDB is classified as a NoSQL database. Unlike SQL, a NoSQL database is non-relational and non-tabular. It stores the data in other formats, like documents or key-value models. Even if it is a non-relational database, it also stores relational data, only differently.

One may find a NoSQL database easier to use than a SQL one, as the relational data doesn't need to be split into tables, in case of many-to-many situations. Moreover, NoSQL databases give developers more flexibility when working with the Agile Methodology, as changes in the database model can be easily performed in a NoSQL system. MongoDB is a document-oriented database that stores the data in JSON-type documents, instead of tables, columns, and rows, which provides an easier and faster integration, as in Figure 15. Using the JSON format also allows a better and faster data exchange from the client to the server and back. MongoDB also offers cloud services (MongoDB Atlas), that are "built for agile teams who'd rather

spend time building apps than managing databases", [23]. MongoDB Atlas is a DBaaS that provides a free version for early development or for learning purposes.

"Scrummer" uses the MongoDB Atlas free edition called M0 Sandbox that provides 512 MB of data storage, shared RAM, and vCPU, has several 100 maximum connections, and a limit to one cluster per project.

Mongoose is a MongoDB ODM library that translates between the objects in the code and the objects in the database, manages the relationships, and offers schema validations. A schema is an object where the key name is the same as the property name in the desired collection and it defines the document's fields.

Mongoose's API is very flexible and offers numerous ways to accomplish a query. It provides basic operations like creating, fetching, updating, and deleting records (CRUD operations).

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

const userSchema = new Schema({
    firstName: {
        type: String,
        required: true,
    },
    lastName: {
        type: String,
        required: true,
    },
    email: {
        type: String,
        required: true,
    },
    password: {
        type: String,
        required: true,
    },
    role: {
        type: String,
        enum: ['PRODUCT OWNER', 'SCRUM MASTER', 'TEAM MEMBER'],
        required: true,
    },
    projects: [{
        type: Schema.Types.ObjectId,
        ref: 'ProjectModel'
    }],
    epics: [{
        type: Schema.Types.ObjectId,
        ref: 'EpicModel'
    }],
    userStories: [{
        type: Schema.Types.ObjectId,
        ref: 'UserStoryModel'
    }],
    tasks: [{
        type: Schema.Types.ObjectId,
        ref: 'TaskModel'
    }],
    comments: [{
        type: Schema.Types.ObjectId,
        ref: "CommentModel",
    }],
});

const UserModel = mongoose.model( name: 'user', userSchema);
module.exports = UserModel;
```

Fig. 15: Code Snippet of the User Model

Node JS: Node JS is an asynchronous, event-driven JavaScript runtime environment that is designed to use JavaScript on the server side. It has access to Node Package Manager (NPM) is the world's largest software registry and is automatically installed with Node JS. It contains over 800.000 code packages and a lot of companies use NPM in private development, even if it is open source. To install the desired package, one can use the NPM CLI. All the installed NPM packages are defined in a file "package.json", where the content of the file needs to be JSON formatted, as in Figure 16.

```
{
  "name": "server",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "node index.js",
    "build": "cd client && npm run build",
    "install-client": "cd client && npm install",
    "startAll": "(cd client && npm run start) & nodemon --exec babel-node index.js",
    "heroku-postbuild": "npm run install-client && npm run build"
  },
  "engines": {
    "node": "13.9.0",
    "npm": "6.13.7"
  },
  "author": "Cristina Sirb",
  "license": "ISC",
  "dependencies": {
    "@babel/core": "^7.8.4",
    "@babel/node": "^7.8.4",
    "@babel/preset-env": "^7.8.4",
    "bcrypt": "^4.0.0",
    "body-parser": "^1.19.0",
    "config": "^3.2.6",
    "connect-flash": "^0.1.1",
    "cookie-parser": "^1.4.5",
    "cookie-session": "^1.4.0",
    "cors": "^2.8.5",
    "dotenv": "^8.2.0",
    "ejs": "^3.0.1",
    "esm": "^3.2.25",
    "express": "^4.17.1",
    "express-ejs-layouts": "^2.5.0",
    "express-session": "^1.17.0",
    "http-proxy-middleware": "^1.0.0",
    "jsonwebtoken": "^8.5.1",
    "method-override": "^3.0.0",
    "mongoose": "^5.9.2",
    "morgan": "^1.9.1",
    "nodemon": "^2.0.2",
    "passport": "^0.4.1",
    "passport-local": "^1.0.0",
    "react-particles-js": "^3.2.0",
    "v-response": "^1.0.0"
  },
  "devDependencies": {
    "babel-cli": "^6.26.0"
  }
}
```

Fig. 16: Code Snippet from the server side "package.json" file

The European Computer Manufacturers Association (ECMA) has overseen the JavaScript changes and the biggest update on JavaScript was in 2015, when ECMAScript 6 (also known as ES6) was approved and later, in 2016, released. ES6 is a standardized version of JavaScript, and it brings a whole new syntax and features that make the code easier to read. It provides features like arrow functions, destructuring, spread operators, modules, etc. Not all browsers offer support for ES6; therefore the code needs to be converted to ES5. The tool that transpiles the code into ES5 before running in the browser is called Babel.

Express JS: Express JS is a minimalist Node JS backend framework (server side) used in building web and mobile applications. It runs as a thin layer on top of Node JS, and it manages the routes and servers by providing a vigorous set of features. It was first introduced in 2010 by TJ Holowaychuck and it was inspired by a web framework based on Ruby, Sinatra. The latest version of Express is 4.17.1 and it is available through the npm registry. To successfully install Express, one needs to have Node JS installed on their local machine. The following npm install command is performed so that

Express is added to the Node JS modules: "npm install express". Once Express is saved into the dependency list, it is ready to be imported and used, as illustrated in Figure 17, [24].

```
const express = require('express' 4.17.1 ) // import Exppress into the file
const app = express(); // start server
const port = process.env.PORT; // listening on the specified port

app.use('/', DashboardRoute); // '/dashboard' route
app.use('/', LoginRoute); // '/login' route
app.use('/', RegisterRoute); // '/register' route
```

Fig. 17: Code Snippet from "index.js" file on the server side

React JS: React JS is a very popular JavaScript library used for building interactive UIs. It was created at Facebook to resolve some issues related to data-driven, large-scale websites and it was first released in 2013. One of the main advantages of React is its capability to only re-render the components that have been updated, which makes this library fast and scalable. React is component-based, meaning that every UI component is encapsulated, having its own state. At first sight, it seems like the HTML code is written in the same file as the JavaScript code, but the HTML tags need some processing before running in the browser. This syntax extension to JavaScript is called JSX and it basically describes what the component is going to look like, using HTML tags. JSX needs to be transpired to work with the browser, as in converted to JavaScript, [25].

Since the introduction of React Hooks back in 2018, one can use the local state and other features without writing a class, [26]. "Scrummer" UI is purely composed of Functional components, it does not have any classes. The most popular and widely used React Hooks are: useState() and useEffect(). The first one allows adding a local state to the component by returning a pair formed of the current state value and the function that updates it. The second one covers the React lifecycle methods in a functional component (componentDidMount, componentWillUnmount, componentDidUpdate).

Redux JS: Redux is a JavaScript library that manages the application state. It was created by Dan Abramov and Andrew Clark in 2015, and it only contains 99 lines of code. The main idea behind Redux is separating the state of the application from the application itself. The state is stored in the Redux Store and is a simple JS hierarchical object.

The workflow cycle of Redux is relatively simple:

- A user generates an event inside the application.
- The event handler calls the dispatch() function that sends an action and the state to a Reducer.
- The Reducer recognizes the action type and generates a new state.
- After the Redux Store receives the updated state, it re-renders the application.

Material-UI: Material-UI is an open-source component library for React and it implements Google's Material Design. It was created in 2014, not long after React started and it is one of the most popular UI libraries for React. It provides higher performance to applications because it only loads the styles for the components that are used. Material-UI components can also be customized by overriding styles with class names.

Other JavaScript libraries: Besides the libraries mentioned earlier, there are also other smaller libraries that had their part in implementing "Scrummer". All these libraries, packages, and modules are available through the npm registry.

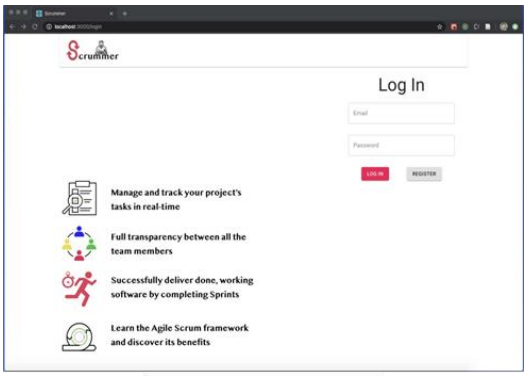

Fig. 18: "Scrummer" login page

The authentication system for "Scrummer" is based on the local strategy from Passport JS. This strategy authenticates the user based on username (in this case, email) and password, as in Figure 18. The authenticate() method handles the login request inside the LoginController. In Node JS, the controller is a function that separates the code containing the route requests from the code that processes them.

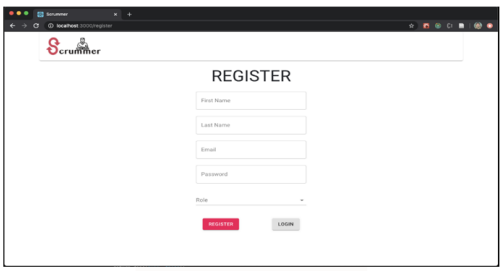

Fig. 19: "Scrummer" register page

The register page is a simple form with mandatory fields, as in Figure 19. The fields include the first name, last name, email address (which will be the login username after a successful registration), password, and role. The password is encrypted using the crypt function. The registration will not be successful unless all the fields are filled, and the email is already in the database.

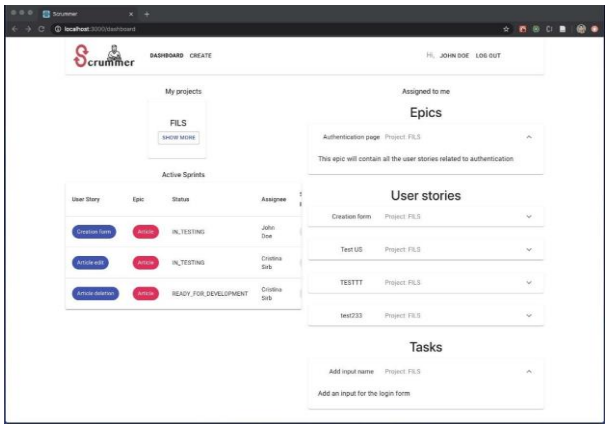

Fig. 20: "Scrummer" dashboard page

The dashboard page is the main page of the application, as in Figure 20. On the upper side, we have the navigation bar that contains the logo of the application and a few buttons to help the user navigate through it. The first button is the button that takes the user back to the dashboard page, like a home button. The next button is the "create" button, which has a submenu with options that depend on the role of the current user.

If the user is a PO, the "create" submenu will contain the following options: project, epic, user story, and task. In the case of the SM, it will contain: Sprint, impediment, epic, user story, and task. For development team members, only the options: epic, user story, and task are available. Moving more to the right in the navigation, we find a button with the full name of the currently logged-in user. By clicking on that button, the application will take the user to their profile, where they can update their profile accordingly. The last button in the navigation is the "logout" button which will "kill" the active session and will redirect the user to the login page.

Below the navigation, the user finds the content of this page. The first section is the "My Projects" section, which will list the projects the user is currently enrolled in, with the possibility to navigate to the specific project through a "Show more" button. The second section is the "Assigned to me" section, which is a list of accordion-type items that represent the categories of the work items that are assigned to the currently logged-in user (epics, user stories, or tasks). The user can see more details about the items by expanding each one.

The dashboard page is the main page of the application. On the upper side, we have the navigation bar that contains the logo of the application and a few buttons to help the user navigate through it. The first button is the button that takes the user back to the dashboard page, like a home button. The next button is the "create" button, which has a submenu with options that. The last section on this page is the "Active Sprints" section. In this section, the user can see the active sprints and the associated list of user stories (the Sprint Backlog).

4 Results and Discussions

4.1 Validation

Scrummer provides an intuitive interface for planning sprints, allowing the Scrum Master to create sprints, assign tasks, and set deadlines with minimal setup time. For instance, the drag-and-drop feature for assigning tasks from the backlog to the sprint backlog simplifies team collaboration during planning sessions. The Product Owner can easily prioritize backlog items using the drag-and-drop interface, with clear visual indicators of priority levels. Automated notifications ensure that the Development Team is always informed of priority changes, enabling quick adjustments to their work schedule. Real-time updates and notifications ensure seamless communication among team members. The integrated chat and comment features allow team members to discuss tasks directly within the platform, reducing the need for external communication tools.

4.2 Scrum Framework Survey

To fully understand Scrum's importance for the present article, a survey was conducted so that the need for Scrum and the opinions about this framework are better underlined. By using a quantitative research method, I intend to find and analyze what IT employees think about the impact of Scrum in their daily activities and overall, about this agile software development framework.

The first question addressed the individual's current role in the company they work for. The survey was answered by 21 developers, 4 Scrum Masters, 2 Testers, 1 Agile Coach, 1 Product Owner, and 1 Business Analyst.

A second question related to the individual's experience in the software development industry

revealed that most of the respondents have been involved in this domain. This shows that the survey has been answered by people with verified experience, which makes it more stable and truthful.

The third question refers to the experience with the Scrum Framework. Most answers fall into the range of less than a year. The answers prove that using this Agile framework is relatively a new method that organizations have started to shift towards Agile approaches to provide business value and quality at a faster rate.

The next question also proves the freshness of Scrum in organizations. When asked how many projects they used the Scrum Framework, the top answer was about half of the respondents used Scrum on only one or two projects throughout their experience within the software development industry, which points out the fact that only recently, companies have started their Agile Transformation, moving on from Waterfall approaches.

Up next in the survey, there is a multiple-choice grid that represents the participant's opinion regarding the usage of the Scrum Framework, as in Figure 21. As shown in Figure 21, most of the respondents agreed with the statements, but there were also a few opposite opinions.

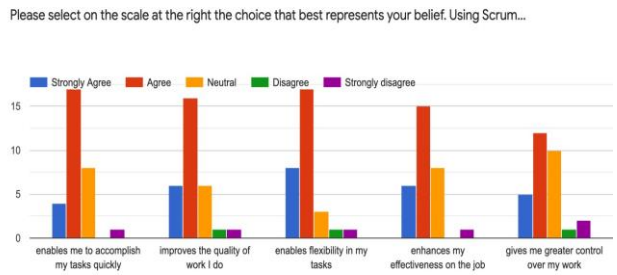

Fig. 21: "Scrummer" survey

The sixth question is also a multiple-choice grid that focuses its attention on Scrum Teams. Overall, the feedback was positive with a majority of "to some extent". The results can be seen in Figure 22. The first statement refers to the willingness of the team members to learn and change. In general, the respondent's teammates from past or present Agile projects were open to change and acquiring knowledge, which is one of the values an Agile team needs to have.

The answers to the second statement, "My team members have strong interpersonal and communication skills", denote that most of the teams the participants were part of were good when it comes to communication. This means that one of the most important Agile principles has been fulfilled. The third statement, "My team members are technically competent" had a majority outcome of "to a great extent".

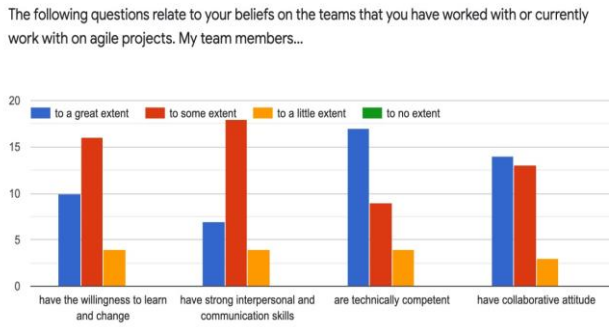

Fig. 22: Survey answers regarding team members

As a cross-functional team, individuals should have different functional expertise and implicitly be competent in their roles, in order to work with the rest of the team towards their common goal. If one member is not technically competent for their role, they automatically become a bottleneck for the whole team. The last statement, "My team members have a collaborative attitude" collected the "to some extent" answers high because, in general, some people are more pragmatic when it comes to openly collaborating with others, because of factors like introversion, reluctance, or fear of failure. It should be kept in mind that in Scrum, the whole team is responsible for the outcome of the Sprint and not just a single person.

The next question was designed as multiple choice and refers to the SDLC phases Scrum is used for. As illustrated in Figure 23, Scrum is most frequently used in the development phase, as per the responses of 90% of the participants to the survey. Moreover, there was also a significant number of responses for the other choices in the list, as 53.3% of the respondents equally opted for the use of Scrum.

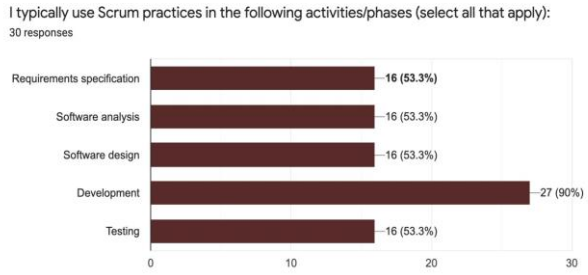

Fig. 23: Survey regarding the usage of Scrum practices

For question number 8 asked if using appropriate tools enables the effective use of Scrum,

53.3% of the participants believed that the statement is largely true, while 43.3% only to some extent.

We can therefore conclude that, with a cumulated majority of 96.6%, the importance of using an appropriate tool for Scrum is acknowledged by the respondents to the survey. Tools are essential when it comes to Scrum, to get the maximum out of every Sprint. They need to be practical, appropriate, and capable of supporting the team to deliver their best results for the project. With the right tool, the time and effort involved can be drastically reduced, while the transparency and performance can be increased.

The options for the ninth question are ranked from "Strongly agree" to "Strongly disagree". The participants had to choose the one that best related to their beliefs for each of the statements presented in the following section of the paper.

The first statement suggests that Scrum can enhance the functionalities of the applications. Out of the total of 30, 7 respondents strongly agreed with the fact that this framework increases the quality of the applications that are built with the help of Scrum, and 18 individuals agreed.

The second statement refers to how using Scrum decreases the number of issues that appear in the applications. By using time-boxed iterations, the risk is automatically minimized and the product's chances of success become greater. This statement was strongly agreed by only 2 people but still agreed by 15. Eleven respondents preferred to stay neutral, but there were also 2 that had a negative opinion regarding this matter.

The following statement was strongly agreed by 7 participants and agreed by 15 who believed that using Scrum also improves the quality of the products. The Definition of Done assures that the quality of the product is always optimal and if defined correctly, this definition can lead to a significant increase quality-wise. Seven individuals remained neutral. Also, when it comes to the quality of the product, seven people strongly agreed with the fact that Scrum makes the team more conscious of the values and implicitly pay more attention when considering the finished Increment. Another 17 agreed with the statement, which makes it accurate.

Moving to the next affirmation, a total of 22 respondents agreed/strongly agreed with the fact that Scrum speeds up the development of new applications. Scrum is known for a more rapid and more efficient delivery since its approach is incremental and iterative.

A surprising percentage of 50% of the participants preferred to remain neutral for the next statement which suggests that there is a correlation between using Scrum and a decrease in development time. Still, the other half agreed/strongly agreed, which means that the affirmation is still valid.

For the last statement, such as in Figure 24, when it comes to productivity measurement, most people (25) agreed/strongly agreed with the affirmation that Scrum helps teams to be more productive.

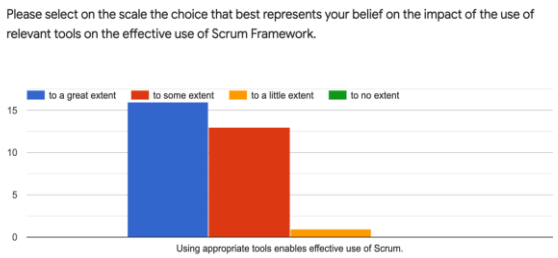

Fig. 24: Scrum Framework feedback

Working in fixed time-length Sprints and having a clear scope in mind lead to a better dynamic within the team and an increase in motivation. Moreover, transparency and risk minimization are also factors that offer the team members a more proactive attitude toward reaching their goal.

The merged results of the survey have been included within Table 2 (Appendix).

In conclusion, this survey demonstrated that the Scrum Framework has mostly positive feedback from its practitioners and that organizations are still in their transformation phase, in which they change from a fixed, Waterfall mindset, to an Agile, open mindset. Scrum indeed makes software development faster and minimizes the risk when it comes to delivery or client satisfaction, but it also makes the developers' lives easier knowing exactly what they need to focus on and constantly receiving feedback.

5 Conclusions and Future Work

This paper introduced Scrummer, a web-based Agile project management tool designed to facilitate Scrum practices through an intuitive and user-friendly interface. By focusing on the unique requirements of Scrum roles, Product Owner, Scrum Master, and Development Team, Scrummer offers tailored functionalities that simplify sprint planning, backlog prioritization, and team collaboration.

The integration of real-time updates, role-specific features, and a streamlined user interface ensures that Scrummer not only meets but exceeds

the needs of Agile teams. Case studies and measurable results illustrate how the tool enhances team efficiency, shortens project delivery timelines, and improves overall user satisfaction.

In today's business environment, web development is in constant progress and growth, as web applications offer numerous benefits to their end-users. Their cross-platform capabilities make them very reachable, as they can be accessed by users regardless of their OS. There is a wide variety of internet browsers one can access the web application through. Web applications are also easier to grow when it comes to building on top of existing architecture. New features can be added as the update of web-based software is done on the server, which is an easier task to undertake.

Software development is a big domain that is growing from day to day. The Agile Methodology helps this industry towards its growth and the frameworks under the Agile umbrella play their part as well. Scrum is the most popular framework, therefore I considered this an opportunity to further examine and analyze it, by developing a tool that would improve the effectiveness of Scrum. This tool, "Scrummer" can be easily used by Scrum teams, and it would make it possible to visualize the workflow. With an intuitive interface, "Scrummer" allows the team to use all its functionalities from the very first day. If the interface of the tool is too complex, the team requires more time to get familiar with it. Since it is a web application, "Scrummer" does not need any installation, which means that the users can access it from anywhere, if there is an internet connection, and not worry about any upgrades. This is also advantageous for team members that are spread across multiple locations since distance is overcomeable.

Future research can explore expanding Scrummer's capabilities to integrate AI for predictive analytics, such as forecasting task completion timelines or identifying potential bottlenecks. Additionally, comparing its performance against other Agile tools in diverse project environments would provide deeper insights into its adaptability and scalability. Some future work that would help improve the quality and overall flow of the "Scrummer" would be: Implement a mailing system for sending out notifications to users (for registering, forgot password, newly assigned work items, password change, etc.); Add charts for the finished Sprints (Burn-down Chart and Velocity Chart); Add avatars for the user, with the possibility of changing it; Send an invitation on the email for the user to register; Be able to start a Scrum ceremony, take notes, and save it for later use.

In conclusion, the application is suitable for Scrum teams and in the end, it provides what is required of a Scrum tool: a Scrum Board and the ability to manage work items accordingly ensuring transparency and efficiency within the team. By blending theoretical foundations with practical applications, Scrummer demonstrates its potential to transform Agile project management and drive innovation in team collaboration and productivity.

References:

[1] Dank, N., and Hellström, R., 2020. *Agile HR: Deliver value in a changing world of work.* Kogan Page Publishers, pp. 25-30.

[2] Lowell, K. R., 2023. Agile Principle 12: "At Regular Intervals, the Team Reflects on How to Become More Effective, Then Tunes and Adjusts Its Behavior Accordingly". *In Leading Modern Technology Teams in Complex Times: Applying the Principles of the Agile Manifesto,* pp. 169-174. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-36429-7_19.

[3] Mumtaz, M., Ahmad, N., Ashraf, M. U., Alghamdi, A. M., Bahaddad, A. A., and Almarhabi, K. A., 2022. Iteration Causes, Impact, and Timing in Software Development Lifecycle: An SLR. *IEEE Access,* 10, pp. 65355-65375. https://doi.org/10.1109/ACCESS.2022.3182703.

[4] Lindskog, C., and Netz, J., 2021. Balancing between stability and change in Agile teams. *International Journal of Managing Projects in Business,* 14(7), pp. 1529-1554. https://doi.org/10.1108/IJMPB-12-2020-0366.

[5] Buturugă, O.C., Gogoi, V.M. and Prodan, I.A., 2016. Agile project management tools. *Academy of Economic Studies. Economy Informatics,* 16(1), pp.19-26.

[6] Flora, H.K. and Chande, S.V., 2014. A systematic study on agile software development methodologies and practices. *International Journal of Computer Science and Information Technologies,* 5(3), pp.3626-3637

[7] Unger-Windeler, C., Klünder, J. A. C., Reuscher, T., and Schneider, K., 2021. Are Product Owners communicators? A multi-method research approach to provide a more comprehensive picture of Product Owners in

practice. *Journal of Software: Evolution and Process,* 33(1), pp. 1-10. https://doi.org/10.1002/smr.2311.

[8] Heath, F., 2021. *The Professional Scrum Master Guide: The unofficial guide to Scrum with real-world projects.* Packt Publishing Ltd., pp. 35-40.

[9] Hron, M., and Obwegeser, N., 2022. Why and how is Scrum being adapted in practice: A systematic review. *Journal of Systems and Software,* 183, pp. 1-10. https://doi.org/10.1016/j.jss.2021.111110.

[10] Lederer, M., and Thummerer, J., 2022. Organizing a Self-organized Team: Towards a Maturity Model for Agile Business Process Management. *In International Conference on Subject-Oriented Business Process Management, S-BPM ONE,* Karlsruhe, Germany, June 29 – July 1, 2022, pp. 152-164. Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-19704-8_10.

[11] Ewim, C. P. M., Achumie, G. O., Adeleke, A. G., Okeke, I. C., and Mokogwu, C., 2024. Developing a cross-functional team coordination framework: A model for optimizing business operations. *International Journal of Frontline Research in Multidisciplinary Studies,* 4(01), pp. 15-34. https://doi.org/10.56355/ijfrms.2024.4.1.0030.

[12] Mahadik, S., Pakanati, D., Cherukuri, H., Jain, S., and Jain, S., 2024. Cross-Functional Team Management in *Product Development. Modern Dynamics: Mathematical Progressions,* 1(2), pp. 270-294.

[13] Kula, E., van Deursen, A., and Gousios, G., 2024. Context-Aware Automated Sprint Plan Generation for Agile Software Development. In Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering, ASE '24, Sacramento, USA, October 27 – November 1, 2024, pp. 1745-1756. https://doi.org/10.1145/3691620.3695540.

[14] Patrucco, A. S., Canterino, F., and Minelgaite, I., 2022. How do scrum methodologies influence the team's cultural values? A multiple case study on agile teams in Nonsoftware industries. *IEEE Transactions on Engineering Management,* 69(6), pp. 3503-3513. https://doi.org/10.1109/TEM.2022.3146717.

[15] Hron, M., and Obwegeser, N., 2018. Scrum in Practice: An Overview of Scrum Adaptations. In *Hawaii International Conference on System Sciences, HICSS 2018,* Hilton Waikoloa Village, Hawaii, USA, January 3-6, 2018, pp. 4496-4505. https://doi.org/10.24251/HICSS.2018.679.

[16] Mosser, S., Pulgar, C., and Reinhar, V., 2022. Modelling Agile Backlogs as Composable Artifacts to support Developers and Product Owners. *J. Object Technol.,* 21(3), pp. 1-3. https://doi.org/10.5381/jot.2022.21.3.a3.

[17] Tyagi, M. K., Tyagi, D. K., Tyagi, L. K., Tyagi, N., Kumar, D., and Sikandar, A., 2021. Agile Planning and Tracking of Software Projects Using the State-Space Approach. In Advanced Computing*: 10th International Conference, IACC 2020,* Panaji, Goa, India, December 5–6, 2020, Revised Selected Papers, Part II, 10, pp. 355-371. Springer Singapore. https://doi.org/10.1007/978-981-16-0404-1_26.

[18] Bhuiyan, F. A., Sharif, M. B., and Rahman, A., 2021. Security bug report usage for software vulnerability research: a systematic mapping study. *IEEE Access,* 9, pp. 28471-28495. https://doi.org/10.1109/ACCESS.2021.3058067.

[19] Ebert, C., and Avasthi, P., 2022. Technologies for Agile Teams. *IEEE Software,* 39(5), pp. 21-27. https://doi.org/10.1109/MS.2022.3180905.

[20] Boye, T., Cheng, E., and Gan, W., 2022. IT Industry Service Management Tools for Managing Large Classes. I*n 2022 IEEE Frontiers in Education Conference (FIE),* Uppsala, Sweden, October 8-11, 2022, pp. 1-6. https://doi.org/10.1109/FIE56618.2022.9962746.

[21] Shin, H., Shim, W., Kim, S., Lee, S., Kang, Y. G., and Hwang, Y. H., 2021. # twiti: Social listening for threat intelligence. *In Proceedings of the Web Conference 2021*, Ljubljana Slovenia, April 19 - 23, 2021, pp. 92-104. https://doi.org/10.1145/3442381.3449797.

[22] Fosci, P., and Psaila, G., 2021. Towards flexible retrieval, integration and analysis of JSON data sets through fuzzy sets: a case study. *Information,* 12(7), pp. 1-8. https://doi.org/10.3390/info12070258.

[23] Phaltankar, A., Ahsan, J., Harrison, M., and Nedov, L., 2020. *MongoDB Fundamentals: A hands-on guide to using MongoDB and Atlas*

in the real world. Packt Publishing Ltd, pp. 21.

[24] Brown, E., 2019. *Web development with node and express: leveraging the JavaScript stack.* O'Reilly Media.

[25] Banks, A. and Porcello, E., 2017. *Learning React: functional web development with React and Redux. "* O'Reilly Media, Inc.".

[26] Ulrich Anders, E., 2017. *React Redux: Concept, Workflow & Cheatsheet.* Cologne Business School.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare.

APPENDIX

Table 1. Comparison between Scrummer, Jira and Tuleap

Feature	Scrummer	Jira	Tuleap
User Interface (UI)	Intuitive and user-friendly, designed to reduce cognitive load	Comprehensive but complex; often cited as overwhelming	Simplistic but lacks advanced UI elements
Task Prioritization	Easy drag-and-drop prioritization for Product Backlog	Supported but requires additional setup	Supported with predefined templates
Role-Specific Features	Distinct actions for PO, SM, and Developers	Available but requires advanced customization	Focused on Scrum roles with basic capabilities
Integration Capabilities	Real-time integration across roles and devices	Advanced integrations with third-party tools	Limited integration support
Pricing	Competitive and scalable	Tiered pricing based on user count	Free
Reporting Features	Basic but focused on actionable insights	Comprehensive, including Burndown Charts, Velocity, and Sprint Reports	Burn-down Charts with basic tracking

Table 2. Classification of answers based on the statements

1 - Strongly agree 2 - Agree 3 -Neutral
4 - Disagree 5 - Strongly disagree

Statement	1	2	3	4	5
Using Scrum has enhanced the functionality of applications that we build	7	18	4	0	1
Using Scrum has decreased the number of errors in the systems/software products we build	2	15	11	1	1
Using Scrum has improved the quality of the systems/software products we build	7	15	7	0	1
Using Scrum has made me/my team more conscious of software quality	7	17	3	2	1
Using Scrum has greatly speeded up our development of new applications	7	15	6	1	1
Using Scrum has significantly reduced the time we spend in software/systems development	3	11	15	0	1
Using Scrum has definitely made me/my team more productive	8	17	4	0	1