Table of Contents

Section 1: Agility as a Methodology

The Scrum was applied in our project during creating an environment for multiple iterations and representatives of an agile team employed the Scrum framework as well as principles. When Agile values appeared to be project management and software development rules, changes were greeted promptly, and complete functionalities delivered with excellent stakeholders feedback in small cycles.

When it comes to the advantages of the agile methodologies, it is imperative to mention that such approaches support adaptive planning, evolutional development, early and continuous delivery, and many other factors which all insist on speed and flexibility to respond to the change. In our project, these principles are not mere theories that are stated out and left to gather dust, but effective and efficient and real tools that helped a lot in enhancing the quality of the products.

Evidence and Examples

For instance, with regards to this project, stakeholders made a large change request halfway through the span of a sprint. Earlier they had planned to create only a primary user control module as the change management strategy. However, at the halfway stage of the sprint, stakeholders wanted to add biological identification to increase security. I wanted to use Agile and decided in a few minutes to have the sprint planning meeting invite the PO, reprioritize the backlog and change the goals for the current sprint in order to fit the new requests while not affecting the entire project schedule. This flexibility was important in satisfying the existing stakeholders' expectations with regard to our development schedule and rate.

Another time, our stakeholders pointed out that the real-time data analytics need to be incorporated into the so called dashboard module. Originally, management didn't consider it within the project's plan. We conducted a backlog grooming session, analyzed the effects of introducing this new practice, and incorporated such requirement into the future sprints. Thus, using Agile's releases approach, it was possible to introduce this new feature gradually, and collect feedback for each of the iterations.

Evaluation of Alternatives

We thought about the other paradigms, namely; Waterfall and Kanban methodologies. At the beginning, contracts to use Waterfall were chosen for its well-defined stages and easily distinguishable stages. Although, we observed that Waterfall was not very flexible to incorporate changes which are very crucial in our project setting since it entails constant changes based on stakeholders' feedbacks.

Kanban another contender we looked at centres on creation of a visual system, controlling the amount of work that is in progress and enhancing flow. While Kanban its flexibility and constant

delivery, it does not have the iterative nature of Scrum that defines time frames. Therefore, we were able to observe that sprint cycles of Scrum allowed for more on time and targeted key enhancement features of the projects.

Key Agile Practices

1. Sprint Planning Sessions: The sprint planning meeting s were exceptionally well-structured where distribution of work and assets was properly done. There were clear goals for each sprint cycle which increased effectiveness and worked in favor of teams' positive motivation and sense of the ownership. For example, in one of the sub-topics of the sprint backlog related to the elaboration of user interface, we pointed to a problem with the process. Thus, by reallocating resources and changing the tasks' priorities, we achieved our goals set for the sprint and optimized the design process.

2. Daily Stand-up Meetings: Such short and intensive meetings were helpful to facilitate reporting by the team members, describing challenges for them and their suggestions for overcoming these challenges. Thus, the open communication line provided us with the opportunity to detect problem areas in the course of the work and prevent possible hindrances to progress. For instance, in a stand-up meeting one of the developers gave a description of a bug encountered in the authentication module. The issues were addressed during the stand-up team meeting wherein immediate discussions and brainstorming occurred; therefore, the possibilities of delay did not arise.

3. Sprint Review Sessions: We practiced these sessions properly at the end of each of our sprints to assess what we had done, how we could interact with stakeholders, and what had to be adjusted for the next sprints. This cross-verification feedback loop was indeed crucial in guaranteeing that works produced met stakeholder expectations and in enhancing the software quality. For example, when the competing teams compiled the progress of the project in the first sprint review, the stakeholders gave their opinion on the newly developed real-time data analytics option. According to their feedback, changes were made in the subsequent sprint and the usability was greatly increased.

Tools and Technologies

We applied modern means of work with tasks in the team which included the using of Trello for collaboration, tasks tracking and visualization of the progress. Trello was our project hub that allowed for the proper tracking of the tasks, specific user stories, and the timeline of the project making it clear and responsive. Trello's Scrum boards let us be aware of the process and the state

of the project and if there are any issues that slow it down. If only the specific Trello workflows can be changed, it means that this tool can easily be tailored to fit the needs of the company and thus, facilitate the effective project management.

Thus, we used Trello as a bug tracker, Confluence for documentation, GitHub for version control and Slack for communication. Confluence assisted us in having project documentation, so the tasks could be coherent to the team members so that everyone stays updated. The use of GitHub made it easier to manage the codes since several developers could be working at a go, and there would be no issues of merge conflicts. It provided the features to make real-time communication and resulted in the formation of a structured collaboration where a member of a particular team could convey the details, discuss something, and sort out the problems.

Continuous Improvement

Although scrum, kanban and other elements are applied primarily for project management, agile is a way of life. Regular meetings also known as sprint review meetings that are conducted at the end of each sprint were effective we regards. In the course of these meetings, the team discussed the positives and negatives of the project and how to correct the arising faults in order to perform well in subsequent sprints.

For example, after few sprint implementations, several recognized that estimations of tasks were imprecise and resulted in overworking. As a result of the retrospectives, we found out that the root of the problem was in the low quality of the estimation and introduced planning poker instead. This change bettered the means of estimate and load sharing which improved the effectiveness and cohesiveness of the team significantly.

Section 2: Role and Contributions

Being both a PM and developer, I was a participant in a set of intricate activities that influenced the project's actions. Maintaining the management of a project and directly performing technical work demanded meaningful knowledge of Agile and practical hands-on knowledge to be able to check that important features were implemented into our banking software system. The following section includes a memo, in which several activities, as well as challenges that have been encountered and possible outcomes of those actions for the project, are described by me.

Project Management

From the description of the responsibilities of the project manager it was apparent that one of my key duties was to steer the team towards a more Agile way of functioning. This incorporated a set of meetings that would aim at planning the sprints and defining the objectives, as well as assigning responsibilities and resources systematically. The management also considered it necessary to make changes in the work environment and promote the establishment of a culture

of cooperation so as to help the personnel to enhance their efficiency as well as to keep the channels of communication open. The synchronisation between the team and project objectives ensured that all team members remained productive contributors as opposed to during iterations where productivity is assumed to be the responsibility of a certain member towards the end of the project.

For instance, in the first planning meetings of the sprint, I observed that my colleagues had a tendency to overestimate their capacities which led to delays and the aggravation of pressure. To this, I embarked on what I called ‘capacity planning,’ whereby I wrote down how many hours every member of the team was likely to spend on his work plan and depended on that, as well as his previous productivity level to estimate how much work to assign to him. In this way people could set more achievable sprint goals and increase the rate of task completion; in addition the working conditions became healthier.

Daily Stand-ups and Sprint Reviews

Another crucial component of what I was doing was to conduct leading daily stand-up meetings. These meetings allowed each member of the specific team to present their findings, describe difficulties that they encountered, and discuss them with other members to find the solution. As a result of the accountability vested on each individual and ownership of the project, the aforementioned meetings were instrumental in saving time. For instance, during one of the stand-ups, a developer mentioned a frequent problem with the user authentication module. In this case, we were able to address this practically in the meeting where without any delay, we arranged a separate meeting with the necessary participants of the team and thus stopped the problem from emerging further and becoming a cause of more delays.

Sprint review meetings conducted with a lot of stringency and at the end of each sprint helped us review our developed work and got a chance to interact with the stakeholders and set the future change requirements for the next sprints. The purpose of these is to present the functionalities that have been worked on during the sprint, ask for feedback, and talk about the necessary modifications. For instance, when explaining a recently incorporated loan management feature, the audience also proposed other forms of reporting that were essential for compliance. That feedback was integrated into the next sprint, and all regulatory standards of the final product were taken into consideration.

Hands-on Development

Besides the managerial task, I used to assist in programming of some key areas in the bank including account management, loan processing, and security. Being involved in the first-hand experience of how the company operates and what issues they encounter has been very impactful because it has allowed me to work on the functional features of the project.

For instance, I contributed to the ideas of strong authentication schemes, including the MFA. The merge of different methods of authentications like, OTP through SMS, biometric options, and quiz questions were included in this. In collaboration with backend developer, these authentication methods were made strong, friendly to the user, and easy to interlink with the security features of the banking software.

Quality Assurance and Testing

Quality verification was another major of my responsibilities during my work at Symbio. I was/is involved in the code reviewing, unit testing and the continuous integration processes as the team leader. These preventative quality assurance activities helped avoid delivering software that is subpar in function or form. For example, when it adopted a stringent code review process, this shifted the code problems earlier reducing the bugs issues and enhancing the quality of the code.

Furthermore, we used other testing frameworks, for instance; Selenium for UI testing and JUnit for API testing. Due to comprehensive test suites creation and integration of test suites into CI, issues could be caught and fixed rapidly, and every new feature could be delivered to customers as stable as possible. For instance, release regression tests enabled us to check that the new code changes did not impact the existing functionalities hence ensuring that the software was not compromised at any stage of the development phase.

Monitoring and Reporting

Hence, it is paramount that monitoring and reporting where critical to guarantee project transparency apart from helping the stakeholders make sound decisions. Paid particular attention and kept track of activities within the Trello tool for agile project management and was able to use it both for tracking the status of tasks, defining challenges, and changing the plans based on those challenges. With help of Trello and its Scrum boards we were able to dictate the flow of work and manage the organization by closely observing the boards and seeing how much work was done and whether there are any problems or issues that needs to be solved in order to finish all the tasks.

For example, in one sprint, it was realized that the response time of a particular feature was extremely slow because of some technical constraints. Just with the Reports feature of Trello, it was easy to recognize that the problem had arisen due to the distribution of resources and correct the situation. This proactive behavior ensured appropriate timeliness of the work and allowed to adhere to the established deadlines.

Continuous Improvement

The other essential thing that shapes Agile frameworks is the adaptation concept which emphasizes on improvement. I assumed the daily Scrum master responsibility of conducting retrospective meetings to look back at the sprint benefits and fluctuations. These retrospectives were crucial in figuring out the problems that needed to be fixed as well as coming up with improvements that would be incorporated into the procedures.

For instance, at the beginning of the project, one of the issues we realized was that the task estimating process was flawed and misleading most of the time, which saw us promising to deliver what we could not fulfill in the required time. While in the retrospectives we pointed these problems and to improve them we planned to use the planning poker, which is a consensus agreement. I observed the change enhancing the accuracy of the estimates we used, reflecting on the equal distribution of workload to the various teams, and consequently increasing the efficiency of my team.

Section 3: Project Planning

Project planning is one of the significant aspects that determine the progress of our banking software system. The strategic approach that has been adopted in this project made sure that it was not only planned but also flexible when it comes to the changes and meeting of the stakeholders' needs. This section shows a detailed elaboration of the planning methodologies as well as tasks that were involved and how they were implemented in the actual project.

Agile Methodology and Scrum Framework

Which method and framework to apply was the next decision: we selected the Agile-methodology with the Scrum-framework for the planning of the project. They turned out to be quite effective because Agile is recognized as a flexible approach and allows me to make changes within a short time frame. These iterations were planned and structured by Scrum, which concentrates on self-organization of the team, team members' collaboration, and review of the process constantly.

Detailed Planning and Milestones

With regard to the planning process, the first step we followed involved subsystems identification and definition of projects that are manageable and comprehensible with defined activities as well as goals to be achieved. We came up with a project timeline that included most of the project's key activities and products. Every point was a measurable property; it could be a feature or user story, and was followed by the possibility to stop and gather feedback.

For instance, it could be the creation of an original system of user authentication that was initially set as one of our goals. This work was deconstructed into the following tasks; plotting the graphical user interface for the login page, coding the back end authentication process, and incorporating multi-factor authentication mechanisms. Every of the sub-task was accompanied by the output and timetable to show that the work is progressing and can be controlled.

Sprint Cycles and Planning Sessions

Our work breakdown was done using sprints, which commonly take about two weeks. In the process of carrying out sprint planning meetings, the team revisited the list of tasks to be accomplished according to the backlog, decided on which tasks should be taken up first, and generated goals for the sprints. Such sessions simply kept all the participants in tune on set goals and the expectations from each of them.

For example, in the case of one sprint planning meeting, the development of the loan management module was deemed to be of the utmost importance. The tasks also comprised of the design of application interface used in borrowing products, writing of back end logics when processing loans, and reporting processes of the loan products. If these tasks were left as a single team member's responsibility, they could easily become clustered and cause inefficiency in the workflow By dividing and distributing them, the responsibilities were appropriately shared among the team, and everyone's work was directed efficiently.

Regular Stand-up Meetings and Progress Tracking

Also, it is worth mentioning the adherent utilization of the daily stand-up meetings. These short meetings entailed each of the team members to be able to go and report on their achievement, identify challenges and determine their activities for that particular day. This practice encourage the flow of information, fast decision making, and continuous work flow.

For instance, during a stand-up meeting one of the members mentioned that there will be a problem with integration of third-party API's for the financial data. Focusing on this issue, we were able to address it right away, which allowed for the proper staffing and extra help needed to ensure that the integration was finished on the established timeline.

Stakeholder Involvement and Feedback

Stakeholders had to be involved in the project at all stages in order to ensure that the goals of development matched their expectations. As for post-sprint communication with the client, we held sprint review meetings where we showed the completed functionalities and their works. This round of feedback enabled us to make changes as needed and to guarantee that the developed end product was adequate to fulfil the requirements of the stakeholders.

For example, when presenting the first version of the account management feature, the stakeholders were asked to suggest other reports they need and make other changes to the interface. These suggestions were integrated into some of the subsequence sprints and they improved the usability and functionality of the software.

Use of Agile Tools

For smoother planning and tracking, the Agile tools like Trello and Trello were used. These tools enabled us to have organized and described methods of creating and having lists of tasks, tracking progress, and schedules of projects. A perfect workflow was created through the Trello Scrum boards where everyone could see the current state of the workflow, potential issues and obstacles, and the timely delivery of tasks.

For instance, Trello given capabilities allowed for the proper distribution of tasks within the sprints, monitoring of their progress, and making priority changes if necessary. Thanks to Trello for the tasks, we had all responsibilities and updates on each other, which improved the level of transparency and accountability among the team members.

Adaptability and Continuous Improvement

Both in general and particularly in the context of Agile we need to mention that planning is highly flexible and conceptualized as a continuous process. To affirm the result of the sprint we conducted a retrospective meeting to look at the outcome of the project and the success and failed aspects of it. These retrospectives were helpful for the detailed process improvement as well as for the general improvement of the team's work.

For instance, the initial reports from retrospective meetings showed that there were the systematic inaccuracy in task estimates, which continually resulted in overruns. This was tackled using the planning poker technique that enhance the accuracy of estimation and distribution of the workload. This change brought with it more achievable sprint goals and improvement in the efficiency of the developing team.

Section 4: Prototype Design and Testing

Prototyping and the actual development of samples are two important stages in the construction of banking software system. Our plan was systematic and had a clear roadmap and thus the final outcome was very easy to use, accurate, and of good quality. This section has been devoted to the description of the design and testing processes with focus on the methods applied to accomplish the goals.

User-Centric Design Approach

The design principles for our user interface were rooted in user perspectives that made the interface very user-friendly. We started with constructing sophisticated paper-based prototypes and graphical specifications since they help depict the basic structure of the program together with a picture of how its components will interoperate and respond to the user's actions. During this phase, tools which may include Adobe XD enabled the quick testing and refining of the ideas received and discussed with the stakeholders besides the users.

For example, the account management interface was first sketched and then created in three different prototypes with users' input after each phase. They will also outline how the iterative layout and navigation along with developmental modifications to the Sherah's visual scheme all contributed to a stronger design. Adobe XD also allowed the team members to work collaboratively and all the team members were always on the same page concerning the design options.

Prototyping and Usability Testing

After the first layouts were developed, it became possible to proceed to the creation of functional prototypes. These prototypes were used to model the users and to exercise some functions of a number of interfaces. Usability testing was carried out early on and iteratively, that is, observing real users in order to obtain feedback on potential problems.

Since the chosen interface is rather complex, we decided to carry out a number of iterations of usability testing where each round was concentrated on certain aspects of the given system. For example, in the first round, participants focused on examining the accessibility of the main account management functionalities. Activities from simple, like checking balance or transferring the money to more complex were performed by users and their experience while interacting with the tool was monitored and the problems observed, were recorded.

The feedback from these sessions was indeed the most useful. For instance, users succeeded in noting that the navigation menu was almost messy and thus hard to follow. Following this feedback, we un-compartmentalized the menu over the users' interface to enhance their ability to locate various features within the application. Further RRs found these changes helped enhance the usability of this application by a large extent.

Development of Robust Testing Standards

In parallel with design and prototyping, it was also necessary to construct quite tough testing protocols to guarantee that the system is functional and will perform as it was designed. As for

the testing we designed a combination of functional and non-functional testing approaches including automated and manual tests.

One of the featured components of our strategy was the focus on the use of automated testing. We used well-defined frameworks in this sector such as Selenium for the frontend testing and for the backend testing we utilized JUnit structure. These tools enabled us to execute heavy test suites that included tests with different types of parameters and cases and attest that the application's primary functions were performing correctly.

For instance, Selenium was employed to drive an automobile that tested User Interface and Interactivity, whereby the test included login, section altering, and transactions among others. This automation not only helped in terms of the ability to conduct more tests within a shorter span of time but also to run regression tests which helped in defining if the introduced code changes posed a problem.

Integration of Automated Testing

It was, therefore, necessary to incorporate automated tests in our development process to maintain the quality of the developed softwares. CI was enabled in order to check the quality of the code and run the tests, each time that new pieces of code were committed in the repository. It entailed that any problems could be spotted and corrected during the conception stage of the project.

For example, the CI pipeline of our application had unit tests, integration tests, and end-to-end tests. For testing, JUnit for writing unit tests was used and it tested segments of code one at a time. Integration tests ensured elemental interactions between the modules while the end to end tests, done with Selenium, certified the complete possible journey of a user starting from the login interface to the completion of a transaction.

Non-Functional Testing: Performance, Security, and Accessibility

Apart from the functional test, non-functional testing was also performed to establish the stability of the developed system. Load and stress testing with help of tools like Apache JMeter, identified the ability of the system to perform under overloaded conditions and certain rates of concurrency. These test gave the opportunity to expose the critical performance issues and determine the ability of the system to provide the necessary level of performance when the number of users is high.

Another activity was security testing which was also carried out. The identified OWASP ZAP tool was utilized for assessment of vulnerabilities and managing of security threats. For instance, we performed checks on potential generic and well-known issues like; SQL injections, Cross site

scripting (XSS) and Insecure authentication. Thus, addressing such vulnerabilities at this stage strengthened the security of the given system as a whole.

The same was made possible through the accessibility testing that proved our software was-friendly to the disabled. WCAG 2. 0 AA standards were used; in addition to automatic testing, methods of manual testing were employed to ensure compliance. Such elements as Axe and testing by the accessibility specialist allowed us to address the issues regarding Web software accessibility and opened the possibility for the appeal to a wider population.

Continuous Improvement and Iterative Testing

It is necessary to state here that testing was an iterative process where improvements were made based on feedback and testing results. Daily or weekly meetings enabled the analysis of the efficiency of using certain testing approaches. This way of applying the continuous improvement approach guaranteed our software to have the highest qualities of functionality and performance.

For instance, at the end of each testing cycle, we encouraged the team members to sit and evaluate wonderful experiences and possible enhancements. From such discussions we experienced improvements in the areas; test coverage, effective utilization of testing tools and integration of development and testing.

Section 5: Software Engineering Tools and Techniques Assessment

Advances in Software Processes

One major success factor for enhancing the development process was the integration of the Agile methodology, more precisely the Scrum framework. Thus, the fact that Agile is an iterative process provided ability to quickly adapt to such changes in requirements and conditions on the market. Such flexibility was critical for a project like ours, the product of which was banking software with many restrictions and regulations by authorities and user interface and safety standards which are expected by the users and clients.

In this context, Scrum became very instrumental in creating some sort of structure as to how our work was to be done. The core practices within the framework involved: sprint planning, daily stand-up, sprint review, and retrospectives that helped create discipline along with having flexibility in working. This was cascaded through sprint planning to align the objectives set to the cycle whereby everyone understood the goals to be met within the cycle and daily scrums for on-the-fly communication and problem solving. Sprint reviews also helped to present what has been done to the stakeholders, get their feedback, and address with the necessary corrections, if

needed. Reflections were important in the process improvement since they enable the identification of better practices in the subsequent iterations.

Project Management with Trello

Trello was one of our primary tools where for project and Agile management and it has plethora of features. To track assignments we adopted the Agile boards offered by Trello where the work to be done is clearly depicted. The boards were also a good way to reveal the current work and enhance the organization's sense of how the work can be best managed.

They noted that Trello offered particularly helpful implementations of some of this aspects, which are the sprint planning features. They let us organize the project into the easily discernible user stories and tasks as well as estimate the amount of effort we need to invest and distribute the workload. After developing a systematic approach of managing the workflow, everything went in order and each was focused on the intended goals.

In the same manner, the flexibility that Trello provides regarding the flow allowed the customization of the resource to suit the needs of the project. For example, we established state channels to monitor the progress of tasks from the time they are assigned to a developer, till they are fully coded reviewed and tested. This level assisted us in having tight control over the development process while at the same time making sure that no process was omitted.

Prototype Design with Adobe XD

Adobe XD was used mostly in the process of creating the interfaces and the prototypes. It has a user-friendly interface and many features that enabled me to design a near perfect model that would reflect the final outcome. Adobe XD was used for creating basic wireframes and also interactive prototypes to be produced as early and as often as possible to get feedback from stakeholders and users.

There are significant collaboration features of the tool that I found very useful. These could include plans where team members could share out prototypes, collect feedback and redesign as a team. It also enhanced the possibility of the designs being user-oriented and falling within the expectations of the stakeholders. Adobe XD also supported usability testing by enabling users to carry out the interaction testing to identify any problems that would be observed by the users.

Version Control with Git and GitHub

Controlled version is required in most software development initiatives, and in this project, we used Git and GitHub. Git – the version control system that was chosen for this project, provided

multiple developers' collaborative work on the project, mechanisms for changes integration and conflicts solving.

GitHub was a tool that was used in this project as it acted as a source control and project hosting service. By using GitHub, we were able to use a Branching model that enable us work side by side while merging squared away. New functionalities were developed in the feature branches, while for the peers' review and to check if the code corresponds to our quality requirements, the pull requests were employed and the code was merged in the main branch.

As well, to make a coordination and track the works, we used applications such as issues and project boards in the Github. Tasks such as bugs, enhancements and others were created, with description and acceptable conditions well defined. Beneath the abstract organizational structure of project boards, which allowed us to focus on prioritizing tasks and their states throughout the project's development process.

Social, Ethical, and Entrepreneurial Impact

Social Impact

The principles of our banking software system were aimed at improving the level of financial education of the population. Thus, making our banking services safe and accessible, we intended to target such populations and assist them in handling their money wisely. Aspects like mobile banking, simple interfaces, and tutorial aids helped in achieving this objective; that of making banking accessible and easy to understand for all the users.

Ethical Impact

As has already been mentioned above, ethical problems remained the top priority as the product was being designed. From the beginning, use data security, privacy, and legal requirements' adherence were important for us. Multi-factor techniques used in authentications saw to it that users' information was kept secured. Security measures such as encryption for the data while receding and when idle, and lawful adherence to measures like GDPR and PCI-DSS were under stringent regulation.

We also followed the ethical recommendations when it comes to the acquisition of permission from the users and their awareness. Users were given information about usage of their data and the usage was regulated by the users. The adherence to ethical values in the design and creation of this software positively impacted the resulting automated financial system, which required the aspects of trust as well as responsibility.

Entrepreneurial Impact

In terms of the structural benefit from an entrepreneurial standpoint, our software system aligned with the development of the banking sector. Growth in the abilities of the product and focusing the experience involved creating difference in a market that seems to have a lot of similar products. Thus, with the help of increasing the companies' operational efficiency and customers' satisfaction, our software helped the financial institutions to become competitive and receive more clients.

Moreover, our approach helped to develop appropriate organizational culture within the team based on the idea of innovation and cooperation. Pertaining to the processes, the Agile method promoted the enhancement and overall adaptability of planning, and regarding the tools, they made problem-solving creative as well as project management on Trello and Adobe XD better. This was particularly important as an entrepreneurial approach allowed for maintaining the quality that was needed to fulfil the public's demand.

References

[1] Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (n.d.). *Agile Software Development Methods: Review and Analysis*. http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf.

[2] Abrahamsson, P., Warsta, J., Siponen, M. T., & Ronkainen, J. (2003). New directions on agile methods: a comparative analysis. *25th International Conference on Software Engineering, 2003. Proceedings.*, 244–254. https://doi.org/10.1109/ICSE.2003.1201204

[3] Fernandez, D. J., & Fernandez, J. D. (2008). Agile Project Management —Agilism versus Traditional Approaches. *Journal of Computer Information Systems*, *49*(2), 10–17. https://doi.org/10.1080/08874417.2009.11646044

[4] Strode, D. (n.d.). *Agile methods: a comparative analysis*. www.naccq.ac.nz