



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Module Title:
Agile Development and Software Dev Ops

Module Code:
ACCA7023

Assignment Title:
Assignment

Student Name:
MD Mahadi Hasan

Student ID:
2505591

Programme:
MSc in Software Engineering and Artificial Intelligence

University Name:
University of Wales Trinity Saint David (Swansea Campus)

Lecturer:
Nathan Morgan

Submission Date:
09 January 2026

Table of Contents

Abstract

1. Introduction.....	4
2. Overview of Agile Software Development.....	4
3. Scrum Framework and Agile Principles.....	5
3.1 Multiple Short Iterations.....	6
3.2 Face-to-Face Communication and Collaboration.....	7
3.3 Adaptability to Changing Requirements.....	7
3.4 Quality Ownership Across the Lifecycle.....	8
4. User Stories in Scrum.....	8
4.1 Flow of User Stories.....	9
4.2 User Story Mapping and Iteration Planning.....	10
5. Estimation and Progress Tracking.....	11
5.1 Estimating User Stories and Assessing Difficulty.....	11
5.2 Burndown Charts: Purpose and Value.....	11
6. SwDevOps and Agile Integration.....	12
7. Web Application Case Study for SwDevOps Evaluation.....	13
7.1 Application Overview.....	13
7.2 Components Requiring Testing.....	13
7.3 Continuous Integration and Deployment Using GitHub Actions.....	13
7.4 API Testing with Postman.....	14
7.5 UI Testing with Selenium.....	15
7.6 Benefits and Limitations of SwDevOps Adoption.....	15
8. Critical Discussion.....	15
9. Conclusion.....	15
10. References.....	16

Abstract

The way software systems are changing quickly and how user needs keep updating have made old ways of building software no longer work well. Agile software development has emerged as the leading way of resolving these issues as it is entered around flexibility, group work, and the provision of software in smaller and more manageable components. Scrum is one of the most prominent Agile methodologies that offers a comparably organized yet adaptable approach to the management of a complex project. The strategy involves the application of short-term and frequent cycles that are referred to as sprints, constant feedback, and shared accountability by the team members to ensure high-quality outputs are always attained. Besides Agile, the concept of Software Development Operations, which is also called DevOps, is gaining more popularity. DevOps unites the various processes of development, testing, deployment, and operations into a smooth and automated operation that makes the whole process of software delivery efficient and dependable.

In this report, the concept behind Agile is delved and Scrum method is implemented to apply the concepts in practical scenarios. It touches upon such key subjects as how to use user stories to gather requirements, how to present and choose these stories, how to estimate work by using story points, and how to measure progress with the help of sprint burndown charts. The report also provides an in-depth examination of the use of SwDevOps in a web application project, such as the continuous integration and deployment using GitHub Actions, the testing of the API using Postman, the testing of the user interface using Selenium.

The study finds that using Scrum together with good planning and teamwork makes a big difference in how adaptable, clear, and responsible the software development process is for quality. SwDevOps also helps make software delivery faster and more reliable and improves feedback through automation and working across different teams.

However, the report also points out some challenges teams might face, such as people resisting change, the difficulty of learning new tools, and the extra work needed to keep and grow these processes. In the end, the report says that mixing Scrum with SwDevOps creates a strong and flexible way for organizations to build high-quality software quickly in fast-changing and competitive environments.

1. Introduction

Software systems are changing fast, and what users want and what's popular in the market keeps changing too. This makes it difficult for outdated software development methods to keep up. Traditional methods like the Waterfall model have problems with handling changes, finding errors too late, and taking too long to finish projects [7], [14]. Because of these issues, Agile approaches have become very popular in the software world. Agile focuses on the need to be flexible and work with teams, frequent feedback, and provide practical software segments in small bites [1], [4].

Among different Agile methods, Scrum is one of the most used. It is a structured but flexible framework, which assists with the complex software projects. Scrum enables teams to divide their tasks into short bursts known as sprints, which make them move forward to remain flexible [2], [9]. Simultaneously, the Software Development Operations or SwDevOps is an extension of Agile that introduces operations, testing, deployment, and monitoring to make a single, flow-like process [5], [6]. SwDevOps is predominantly seen to eliminate the divisions between development and operations divisions, which leads to faster, more reliable and superior quality software releases [13].

In this paper, the author explores the role of Scrum in Agile projects. It explains the construction and utilization of user stories, their arrangement, the calculation of the required amount of work, and the use of burndown charts to track the progress [2], [3]. By looking at a web application example, the research also examines DevOps approaches, such as continuous integration and deployment using GitHub Actions, testing APIs using Postman, and testing the user experience using Selenium [10], [11], [12]. Finally, it discusses the pros and cons of using Scrum and SwDevOps in today's software development environment.

2. Overview of Agile Software Development

In this paper, the author explores the role of Scrum in Agile projects. It explains the construction and utilization of user stories, their arrangement, the calculation of the required amount of work, and the use of burndown charts to track the progress. It focuses on delivering working software instead of creating lots of detailed documents, works closely with customers instead of relying on long contracts, and prefers adapting to changes rather than sticking to a fixed plan [1], [4]. These ideals are supported by twelve basic ideas that facilitate software delivery in short, frequent cycles, keeping stakeholders in touch, using sound technical practices and having a sustainable development process that can be sustained over time [7].

Agile acknowledges the changes in needs of the project are common and that change is the natural process of development. Agile methods do not attempt to plan everything

initially, instead acknowledging that things may be uncertain and develop the software gradually, modifying it, as needed, in the process [1], [3]. This is done by making frequent visits to the customers and the team members to update and refine the requirements. This approach is beneficial in avoiding issues and allowing customers to be happier with the outcome and allows the staff to deal with the emerging notions or concerns.

Even though Agile has many advantages, it can be tough to use effectively. These objectives are backed by twelve major concepts that promote software delivery in small, frequent cycles, keep the stakeholders updated, employ sound technological methods, and develop a sustainable development process that can be maintained in the long run [7], [14]. Many companies struggle because they try to use Agile without really understanding the basic ideas behind it. This is the reason why frameworks like Scrum are so helpful—they offer precise instructions and assistance to enable teams to successfully and meaningfully apply Agile [2], [9].

3. Scrum Framework and Agile Principles

Scrum is a flexible and simple way for teams to handle complex projects. It assists in splitting large tasks into small tasks, which are easy. The teams operate in brief intervals referred as sprints that allow them to review their progress and adjust where necessary. This approach keeps all the team members updated on the progress of the project, gives the team members a chance to revise their efforts periodically, and provides an opportunity to make changes in case something is not working. Scrum also possesses distinct roles, such as the Product Owner, Scrum Master and Development Team, as well as fixed meetings and tools that can be used to keep the project on track [2], [9]. These parts support the main ideas of Agile while letting teams change the approach to fit their specific situation and needs.

3.1 Multiple Short Iterations



Figure 1: Scrum Sprint Cycle Diagram

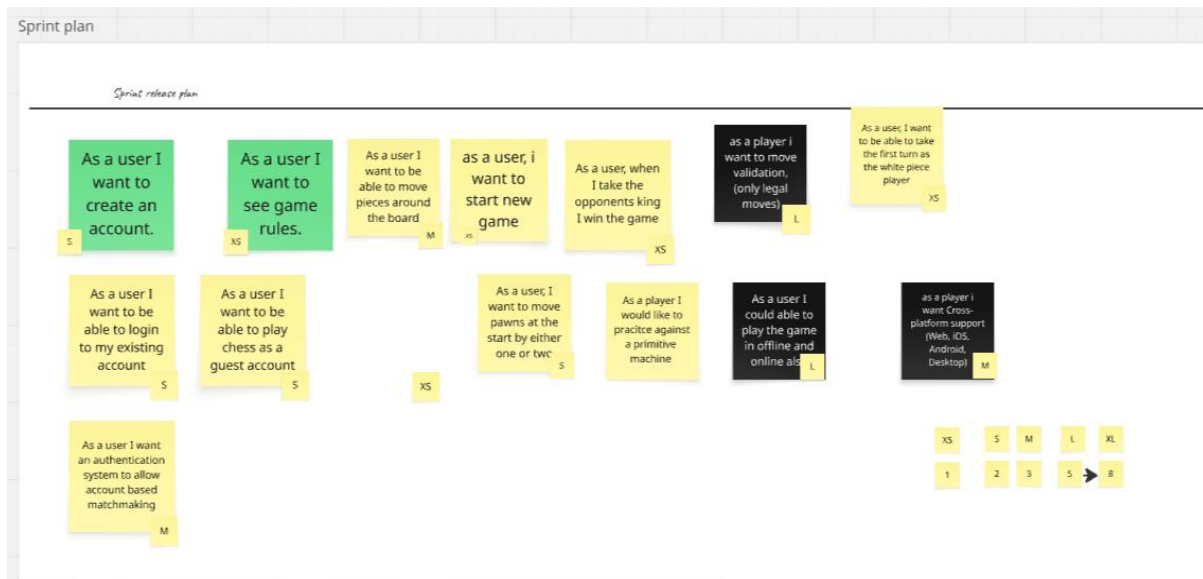


Figure 2: Sprint Plan (Classwork)

Figure 1: Scrum Sprint Cycle Diagram

The Scrum Sprint Cycle diagram describes the process of Scrum in stages at each successive cycle. It starts with the Product Backlog that is a list of all the features, tasks, and requirements that require to be constructed. The Product Owner will be responsible to arrange such items, and to determine which ones are the most essential. In the Sprint Planning, a group of such prioritized items is chosen and placed into the Sprint Backlog by the team. The Development Team will work on these items during the sprint and strive to produce a completed and functional portion of the product, a so-called potentially shippable Increment [2]. To be in the same page and progress, the team holds short daily meetings referred as Daily Stand-ups during which they update each other what they have accomplished, what they want to accomplish and the challenges that they are encountering [9]. The Sprint Review comes at the end of the sprint, during which the team presents the work it has done to those who are interested in the project, such as the customers or the managers, and requests their feedback on it. This will be preceded by a Sprint Retrospective in which the group will review what has been successful, what has been unsuccessful and how they can perform better the next time they work on the task. This circle is used to demonstrate that Scrum is all about regular feedback, checking the progress, and change, when necessary [3], [7]. One of the fundamental elements of Agile is to provide working software at regular intervals in brief cycles. Scrum achieves this by use of time-based sprints that normally take two or three weeks. With every sprint, there is a version of the product that can be released, and it is necessary to ensure the delivery of value on a regular basis during the development.

The short iterations provide the stakeholders with additional opportunities to provide feedback, and it is possible to detect problems, such as defects, usability problems, or vague requirements at an early stage. Early detection ensures that major changes are not required in the future which may be costly and time-consuming. Management wise, short iterations are useful in forecasting the next occurrence and enabling frequent reviews on the project progress.

Conversely, short iterations should also have a good backlog management and powerful sprint planning. The failure to write user stories properly or writing them too big might lead to work remaining unfinished and demotivating the team. Due to that, regular planning and dedication to continually evolving the process should be present to achieve the beneficial effects of the iterative development to the fullest.

3.2 Face-to-Face Communication and Collaboration

Scrum is all about working together and having regular conversations. It takes the form of meetings such as daily stand-ups, sprint planning, sprint reviews, and retrospectives to ensure that all members of a team and stakeholders in the project are on the same track [2]. These meetings assist in maintaining openness and clarity of communication hence everybody is aware of what is going on. When people talk face-to-face or through video calls, they can quickly solve problems and avoid misunderstandings. This assists in making the entire team collaborate in a better manner and comprehend each other.

In teams that are spread out across different places, working together can be challenging because of time zone differences and cultural backgrounds. However, nowadays such tools as video calls, shared workspaces, and real-time messaging applications allow connecting and collaborating [1], [4], [6]. Although maintaining good records is not neglected, Scrum lays more emphasis on face-to-face talks to ensure that everyone is on track and is working effectively.

3.3 Adaptability to Changing Requirements

Agile development thrives on adaptability to enable the team to adapt fast to new information and changing requirements. This flexibility in Scrum is facilitated by update to the backlog on a regular basis and a common review of the backlog once every sprint [2], [9]. The Product Owner plays a significant role in such a process since he or she organizes and ranks the product backlog and considers such factors as business value, stakeholder feedback, and market dynamics.

This strategy allows the team to concentrate on the aspects that are the most beneficial at the present time [1], [7]. However, when the changes occur too frequently or are not managed in an appropriate manner, it may become a source of trouble to the team. To prevent this, Scrum ensures the protection is made to the sprint backlog immediately a

User stories are an important part of the Scrum framework. They help explain what the end users need from a product. These narratives are in a clear simple format that explains clearly who the user is, what he/she wants to do, and why a particular feature is necessary [2], [8]. They help the development team understand the user's requirements and make sure the product being built matches what the users need and want.

4.1 Flow of User Stories

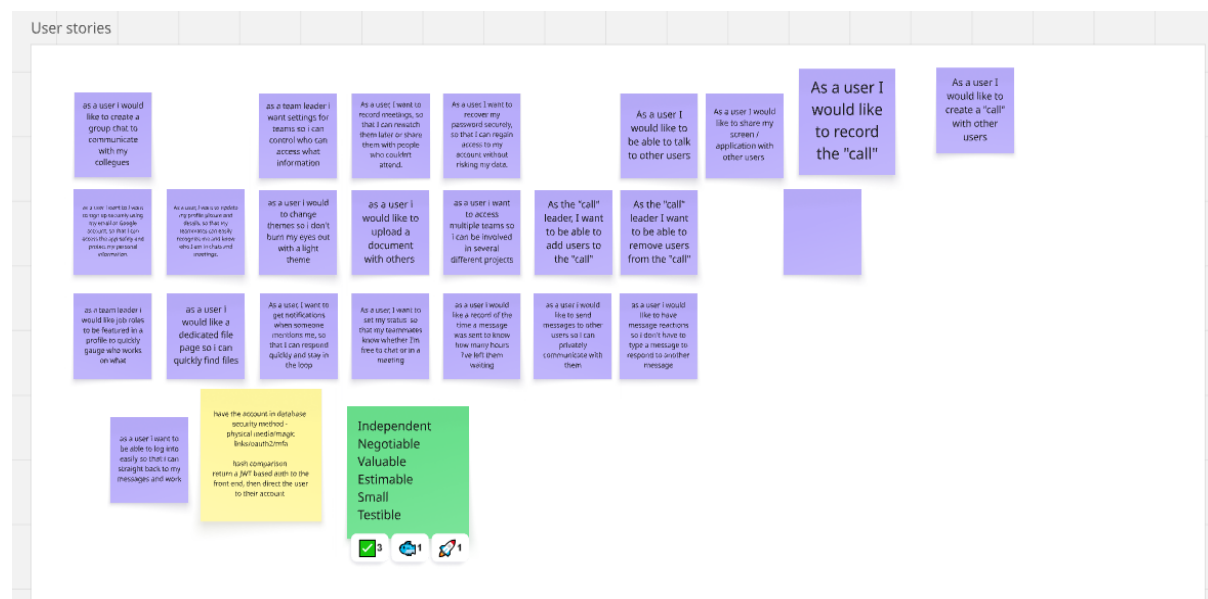


Figure 4: User Stories

The user stories in Scrum start in the product backlog, which is the list of all the features and needs that must be developed to the product, prioritized. The stories are then transferred to the sprint backlog where they are selected to the current sprint depending on their importance and the amount of work that the team can take [2], [9]. Once a sprint begins, the team tries to complete these stories to have something real and ready to be released in case of necessity, which is a part of the product. This way the stories are taken out of the product backlog into the sprint backlog and the real product, so that nothing is lost as to the original user intention.

To ensure that all people know what should be done and how to test it, it is crucial to establish acceptance criteria [8]. These are explicit guidelines that indicate the completion of a user story. When these are not clear, it may create confusion on what should be delivered which may result in errors and delays.

Effective management of user stories will assist the teams to achieve timely goals and deliver value to the users. But when the stories are not clear or are too general it may lead to issues during the sprint. This may cause confusion, lack of deadlines, and reduced efficiency and productivity of the team.

4.2 User Story Mapping and Iteration Planning

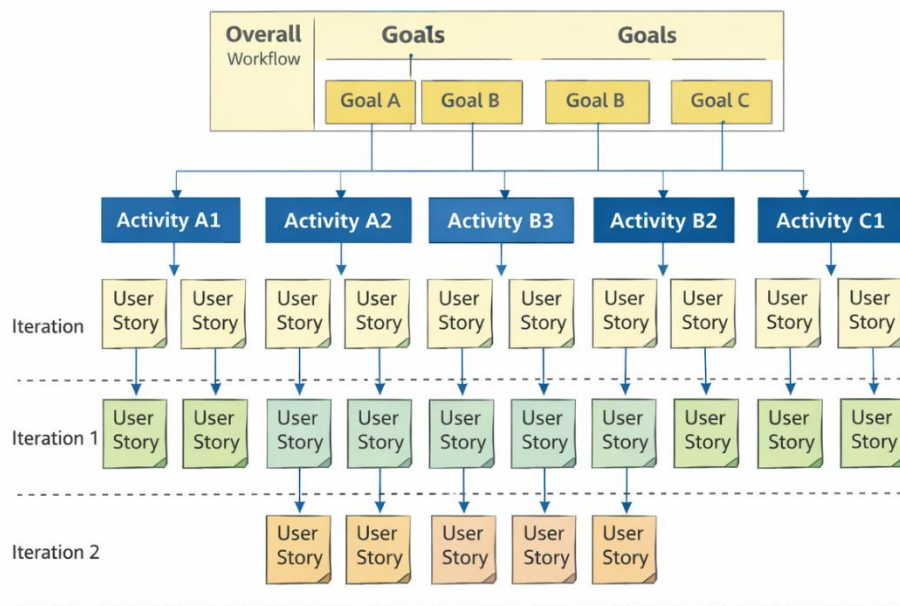


Figure 5: User Story Mapping Diagram

User Story Mapping diagram provides a clear and detailed image of the interaction of a user with a system during their experience. It splits the user experience into the steps that are displayed on the horizontal line and indicate the sequence in which users utilize various system components. The user stories are listed along the vertical line, and the order is determined by the level of importance or urgency. The most crucial ones are the top stories and constitute the Minimum Viable Product (MVP), and it is essential to ensure that the key features are created initially [8]. Other minor stories will be developed in later stages. This tool can assist teams in knowing how various features are related to each other, establishing attainable and linked objectives on a per-sprint basis, and making sure every release brings actual worth to the user, rather than new and unrelated features.

User story mapping can be useful in uniting stakeholders and the development team and make all of them aware of the scope of the project, the priorities of the project, and how the releases can be planned. It is a graphical method of organizing and prioritizing user stories within the entire journey of the user. With this approach, the teams will be able to better calculate what features are necessary in the MVP and create meaningful updates to the product that will lead to the final and functioning product.

User story mapping enhances planning and ensures that everybody is on the same page including the stakeholders by matching the development with the way users work (as illustrated in Figure 5) [3], [8]. This approach however will require that stakeholders are engaged and continue to update it to keep the mapping relevant and useful.

5. Estimation and Progress Tracking

5.1 Estimating User Stories and Assessing Difficulty

Story points are typically used by teams in Scrum to determine how much work they have. Story points do not directly measure time, but indicate the difficulty of a task, the complexity of a task, and the risk that a task may involve [3]. This assists the team to think of the difficulties rather than merely in the amount of time it will take. A typical method of teams doing this is Planning Poker, in which all people give their guesses regarding the amount of effort required on a task [2], [3]. This makes the process enjoyable and enables the opinions of all to be listened to, making the team come to an improved estimate.

The difficulty of tasks is important to know as it assists the team to identify potential issues or factors that may slow down the team. As they accomplish more tasks and learn about what they are capable of, they begin to observe a trend in the amount of work they can accomplish within a time. This trend is referred to as velocity, and it assists them in becoming better estimators in future. However, one should bear in mind that estimation is not aimed at evaluating the quality of a person, but it is used to facilitate planning. It is one of the methods to assist the team in making wiser decisions, not a hard method of measuring success.

5.2 Burndown Charts: Purpose and Value

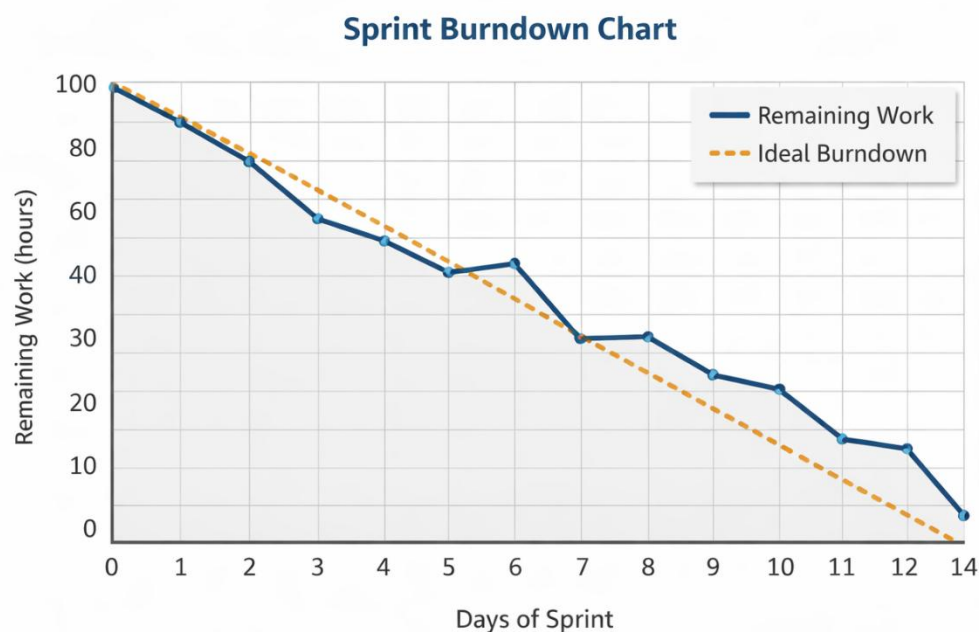


Figure 6: Sprint Burndown Chart

The Sprint Burndown Chart is an image that indicates the amount of work remaining to be accomplished within a sprint and how the amount of work varies with time. The chart has a vertical line on the left which indicates the number of story points left to be completed. On the right side, one can see a horizontal line indicating the days or weeks of the sprint. The chart contains two lines, one indicates what the team were to do and the other one indicates what they are doing. The non-alignment of these two lines may imply that not all tasks were planned properly, some issues appeared unpredictably, or the objectives of the project were changed. Such a chart will assist the team in realizing the remaining amount of work and the current situation to ensure that they detect problems early and implement necessary adjustments during the sprint [2], [3].

Burndown charts are best utilized when one wants to know what is going on in the project rather than to monitor the performance of people. In this manner, the team members will be able to collaborate and continue to improve their process. These charts indicate the extent of work remaining in a sprint and everyone is kept informed of the level of progress being achieved. Having the remaining tasks in view, they simplify the process of identifying problems at early stages and taking actions before the situation gets out of hand.

Burndown charts can be used to monitor the amount of work remaining in a project and can assist in maintaining a schedule of teams. However, one should keep in mind that these charts are not to be applied to compare the success of people. When applied in that manner, it may lead to such issues as the lack of trust in one another and the decrease in the team collaboration. The primary aim of a burndown chart is to assist the entire team to progress, rather than to strain an individual [9].

6. SwDevOps and Agile Integration

SwDevOps is founded on Agile and combines the development, testing, deployment, and operations into a single continuous mechanism [5], [6]. This technique involves automation of the whole software development process, encourages collaboration between the various sections of the company, and maintains a vigilant watch on the performance of the system to ensure the smooth and reliable delivery of software [6], [13].

Cooperation between developers and operations personnel is beneficial in that it aids in the early detection of errors and makes it simpler to obtain feedback about the systems or users. However, to achieve success with SwDevOps, it is not only necessary to change the way tasks are processed but also the way people think and behave in the company. It requires a considerable investment in the appropriate tools and technology as well as continuous efforts to continue enhancing and sustaining these processes as time goes by.

7. Web Application Case Study for SwDevOps Evaluation

7.1 Application Overview

The case study concerns an e-commerce site which provides options like user login and registration, allows the customers to view a product, purchase items in a shopping cart, pay safely, and handle orders [6], [13]. Such websites must be always on, thus always perform well, secure to store information and payments of users and scalable to allow more users and transactions as they increase without becoming slow.

7.2 Components Requiring Testing

The system has several key components that should be thoroughly inspected to ensure that nothing goes wrong.

These main areas are:

- The front-end user interfaces, the parts of the app that the users can see and that the users directly interact with.
- The backend APIs, which are the tools and services that handle the data and the rules that are operating behind the scenes.
- Database operations, which entail the way the data is stored, recalled, modified, and maintained in order in the system.
- Authentication and authorization systems, which are the means of control over access to what and security of things.
- The integrations of third-party payments, the links to third-party payment solutions that enable users to pay.

7.3 Continuous Integration and Deployment Using GitHub Actions

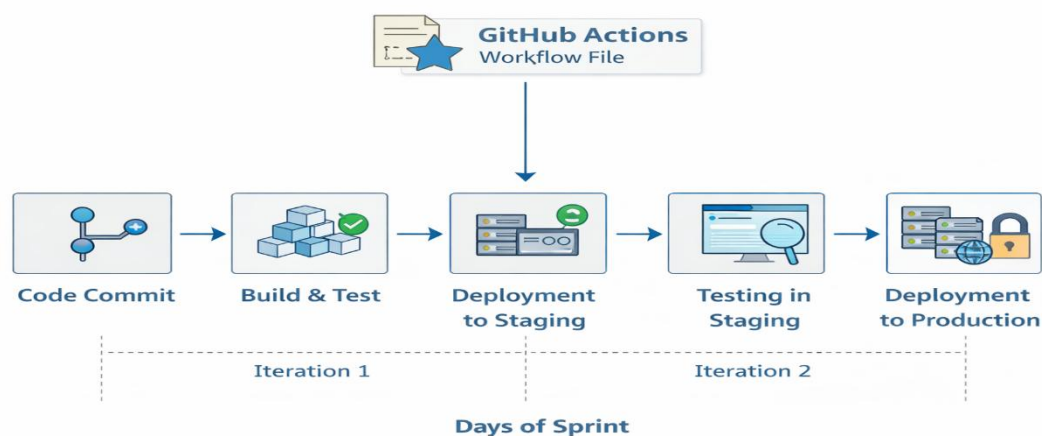


Figure 7: CI/CD Pipeline Using GitHub Actions

The CI/CD pipeline diagram depicts an entirely autonomous process that is triggered when a person adds code or makes a pull request. Some of the important steps involved in this workflow are to obtain the source code in the repository, install any required tools, run automated unit and integration tests, build it, and deploy it to either a staging or production environment [5], [6]. GitHub Actions manages all these steps in the form of custom workflows pre-configured in advance, which makes everything consistent and provides timely feedback to developers [10]. This figure illustrates the ways SwDevOps practices can be used to reduce the time spent on manual work, eliminate mistakes in the deployment process, and allow teams to release updates more often and with a higher level of reliability.

Though these pipelines are highly efficient, they must be established cautiously to take care of issues such as secrets, environment variables, and undo change plans. GitHub Actions is adopted to enable automated pipelines of CI/CD to be run when the code is modified. It allows automatically building, testing and deploying applications without human assistance and, therefore, makes the processes more consistent and quicker (see Figure 7), although GitHub Actions requires proper configuration to handle secrets, dependencies, and the cost of running the actions.

7.4 API Testing with Postman

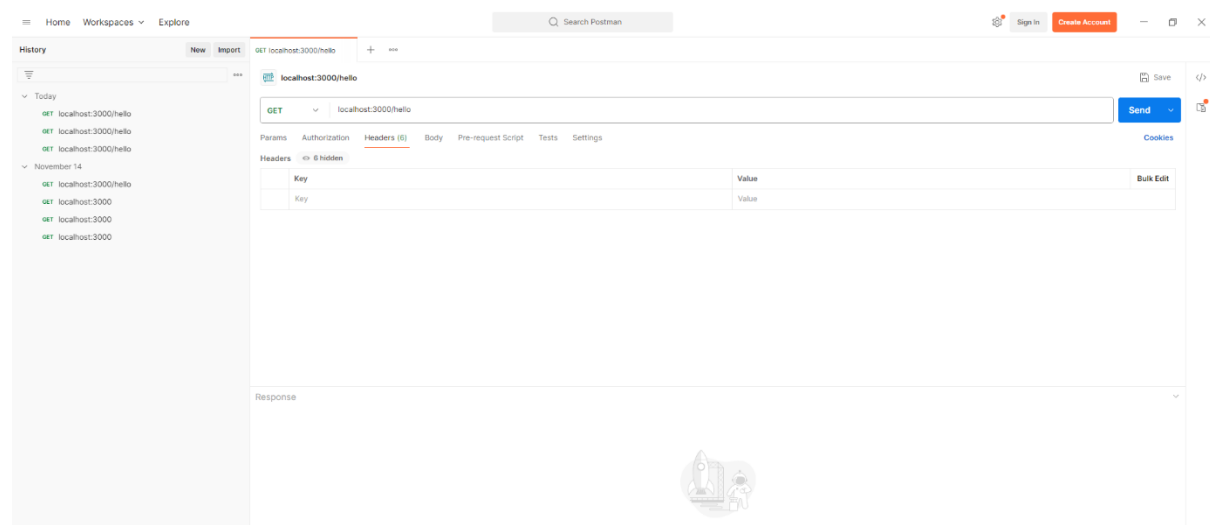


Figure 8: Postman Testing Process

Postman also lets you automatically test APIs by ensuring the endpoints are operating, ensuring that correct information is being sent in the requests, and ensuring that the back-end responses are as expected. It also may be associated with continuous integration tools, which assist in identifying the problems in the back end sooner in the development process, which makes the testing process quicker and more reliable [5], [11].

7.5 UI Testing with Selenium

Selenium is a program that can help in automated web browser testing. This also makes sure that the user experience remains the same regardless of the devices or environments in which the application is being used [12], [6]. It is a potent and multi-purpose tool which allows testers to construct comprehensive test cases. However, despite being useful, the tests that are created using Selenium may easily break down, and they must be reviewed and revised frequently to ensure that they remain stable.

7.6 Benefits and Limitations of SwDevOps Adoption

Advantages: Advantages are prompt delivery of projects, enhanced quality of the outcomes, enhanced cooperation among the team members, and reduced problems in the introduction of new features or updates.

Limitations: The limitations may include the difficulty of learning how to use the tool, time to become proficient with the tool, resistance of individuals who are still attached to the old ways, and the additional effort required to take care of the tool and keep it current.

8. Critical Discussion

Scrum and Software Development Operations (SwDevOps) may be of great help to an organization, yet their effectiveness is predetermined by several factors. They are the present state of the organization, the level of readiness of the team to employ these approaches, and the extent of encouragement and guidance provided by the leaders. When a team attempts to employ these practices without having a proper grasp of the key concepts, it may result in increased complexity and additional effort without tangible gains [6], [7], [13], [14]. It is necessary to adapt these strategies to the needs of a particular organization and ensure that all concerned are on the same level and willing to see them through.

9. Conclusion

In this report, the close consideration has been paid to Agile development and SwDevops practices with particular attention to Scrum. Scrum is a powerful model that aids in adhering to the Agile principles through the cycles of repetition, the promotion of teamwork, the ability to make changes, and the responsibility of all individuals in the quality of the provided software [2], [9]. Properly used, such tools as user stories, how to estimate work, and burndown charts allow one to see the progress of a project and contribute to more adequate planning.

SwDevOps also improves Agile by automating the important processes like integrating, testing and deploying software which enables the teams to work at a better and more

consistent speed [5], [6], [13]. Although such approaches have certain challenges, the combination of Scrum and SwDevOps provides a strong foundation of creating high-quality software in dynamic and fast-changing settings [6], [13].

10. References

- [1] K. Beck et al., “Manifesto for Agile Software Development,” Agile Alliance, 2001.
- [2] K. Schwaber and J. Sutherland, The Scrum Guide: The Definitive Guide to Scrum, Scrum.org, 2020.
- [3] M. Cohn, Agile Estimating and Planning, Upper Saddle River, NJ, USA: Prentice Hall, 2005.
- [4] M. Fowler and J. Highsmith, “The Agile Manifesto,” Software Development, vol. 9, no. 8, pp. 28–35, 2001.
- [5] J. Humble and D. Farley, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, Boston, MA, USA: Addison-Wesley, 2010.
- [6] N. Forsgren, J. Humble, and G. Kim, Accelerate: The Science of Lean Software and DevOps, Portland, OR, USA: IT Revolution Press, 2018.
- [7] A. Cockburn, Agile Software Development, 2nd ed., Boston, MA, USA: Addison-Wesley, 2007.
- [8] J. Patton, User Story Mapping: Discover the Whole Story, Build the Right Product, Sebastopol, CA, USA: O’Reilly Media, 2014.
- [9] P. Deemer, G. Benefield, C. Larman, and B. Vodde, “The Scrum Primer,” Scrum.org, 2012.
- [10] GitHub, “GitHub Actions Documentation,” GitHub Inc., 2024.
- [11] Postman Inc., “Postman API Testing Documentation,” 2024.
- [12] Selenium Project, “Selenium WebDriver Documentation,” 2024.
- [13] L. Bass, I. Weber, and L. Zhu, DevOps: A Software Architect’s Perspective, Boston, MA, USA: Addison-Wesley, 2015.
- [14] IEEE Computer Society, “IEEE Guide for Software Engineering Body of Knowledge (SWEBOK),” IEEE, 2014.