**Date: 9th January, 2024**

**Apple Mahmood Oni**

**Task-8**

**1. Verification vs Validation:**

* Verification is checking if we are building the system right. Validation is checking if we are building the right system.
* Verification is done before Validation.
* Verification is a static technique which is done without executing the code. Validation is a dynamic technique which is done by executing the code.
* Verification involves walkthroughs, reviews , inspections etc. Validation involves white box, black box, grey box testing etc.
* Verification target the srs document, design document etc. Validation target the actual product.

**2. V vs W model:**

* In V model testing is associated with each development stage. W model ensures that the testing activity begins from the very first day of software development process in parallel.
* The V model maintain linear structure and direct relationship between testing and development phase. W model introduces iteration.
* W model covers activities which are skipped by V model.

**3. Incremental vs Iterative:**

* Incremental development involves developing the product in small portion and then gradually integrate those portions together until the final product is complete.

Iterative development involves building the product with some basic features and functionality and then based on feedback adjusting the product again and again until the final product is achieved.

* In incremental process, customer doesn’t see the product until it finishes. In iterative process, customer gives feedback after each iteration is completed.

**4. Burndown Chart**

A **burndown chart** helps agile project management teams keep track of what’s been done, what needs to be done and how much time is left in the project. There are two burndown chart variants: a **sprint burndown** and a **product burndown**. A sprint burndown is used for work remaining in the iteration while a product burndown illustrates the work remaining for the entire [project plan](https://www.projectmanager.com/guides/project-planning).

Components of burndown chart: Axes, Ideal work remaining line, Actual work remaining line.

**5. User Story (Student Management):**

As a student, I want to view my CGPA so that I can track my current academic progress.

As a student, I want to receive teacher reviews so that I can understand my mistakes and areas of improvement.

As a student, I want to see the course list that I am enrolled in so that I can plan my study schedule accordingly.

**6. Automation Test:**

From the slide 36 of class 7, It seems DevOps was referred.

DevOps is a combination of two words. Development and Operation. This allows a single team to handle the entire software application life cycle from development to testing, deployment and operation.

Automation is the crucial need for DevOps practices, and automate everything is the fundamental principle of DevOps. Automation kick starts from the code generation on the developers machine, until the code is pushed to the code and after that to monitor the application and system in the production.

Continuous testing is an integral part of continuous integration and continuous delivery (CI/CD) practices, adopting automated testing makes CI/CD more efficient and enables teams to release software more frequently.

Quality assurance engineers should focus on developing automated integration and end-to-end tests while developers perform unit tests for each block of code they build. These tests should be executed early enough in the DevOps CI/CD pipeline to ensure each component works as expected.

**Task 9**

**1. Burndown chart for 15 days sprint**

There are 3 types of burndown chart.

* Sprint Burn Down
* Product Burn Down
* Release Burn Down

A sprint burndown chart tracks the total work remaining in a specific sprint or iteration.

The horizontal or X-axis of a burndown chart displays the iteration timeline (usually in days) for a project or sprint. For this, Day 1 to Day 15 and the last point would be “Done”.

The vertical Y-axis of a burndown chart represents the story points and team efforts. This axis indicates the remaining work that needs to be completed in a sprint or project.

The project/sprint end is the rightmost point of your burndown chart that indicates whether you completed a project/sprint on time, behind, or ahead of schedule.

An ideal work line shows how much time it would take your team to complete a project depending on your team’s recent performances, meaning it’s not always accurate or absolute.

The actual work line shows the real-life progress you and your team accomplish on a project.

**2. Spiral vs Waterfall**

|  |  |
| --- | --- |
| **Spiral Model** | **Waterfall Model** |
| Spiral model works in an evolutionary method. | Waterfall model works in a sequential method. |
| Spiral model is suitable for large complex project. | Waterfall model is suitable for small project with clear goals. |
| Testing is performed after the engineering phase | Testing is performed after the implementation phase |

**3. Spiral vs Agile**

* The spiral model requires more upfront planning and analysis, particularly in terms of risk assessment and mitigation. It follows a more structured and formal process than agile, with predefined phases and deliverables, and produces more documentation such as risk management plans, technical specifications, and evaluation reports. Additionally, it relies more on expert judgment and experience than agile which favors empirical evidence and experimentation.
* In spiral model testing is done at the end of each cycle while testing can begin at the start of the project in the agile model

**4. Spiral vs Iterative**

* The main difference between spiral and iterative methodologies is the focus on risk management versus time management. The spiral methodology prioritizes identifying and reducing the risks of the software project, while the iterative methodology prioritizes delivering and improving the software product.
* Another difference is the scope and complexity of the software project. The spiral methodology handles large, complex, and dynamic software projects that need constant changes and adaptations, while the iterative methodology handles small, simple, and stable software projects that have fixed and clear requirements.
* A third difference is the feedback and testing mechanisms. The spiral methodology incorporates user feedback and testing at the end of each cycle or iteration, while the iterative methodology incorporates user feedback and testing throughout each increment or version.